Universidade de Coimbra
Faculdade de Ciências e Tecnologia
Departamento de Engenharia Electrotécnica

CONSTRUÇÃO DE MAPAS LOCAIS EM ROBÓTICA MÓVEL

Tese de Mestrado

José Luís Silva Tavares da Cruz

Licenciado em Engenharia Electrotécnica

Coimbra
2001

Faculdade de Ciências e Tecnologia
da
Universidade de Coimbra

CONSTRUÇÃO DE MAPAS LOCAIS EM ROBÓTICA MÓVEL

José Luís Silva Tavares da Cruz
Licenciado em Engenharia Electrotécnica

Dissertação apresentada no âmbito do Mestrado em Sistemas e Automação, na especialidade em Automação Industrial, pela Faculdade de Ciências e Tecnologia da Universidade de Coimbra, sob orientação do Professor Doutor Urbano Nunes.

Coimbra
2001

Aos meus Pais, Manuel e Graciete, por tudo.

À minha mulher Cláudia pelo complemento da vida.

Agradecimentos

Quero agradecer:

Ao Professor Doutor Urbano Nunes pela orientação, apoio prestado, encorajamento e confiança que sempre manifestou em relação ao meu trabalho.

Ao Departamento de Engenharia Electrotécnica da Faculdade de Ciências e Tecnologia da Universidade de Coimbra e ao Instituto de Sistemas e Robótica de Coimbra, pelos meios disponibilizados e pelas condições de trabalho proporcionadas.

À Escola Superior de Tecnologia do Instituto Politécnico de Castelo Branco, pela compatibilização do serviço docente com o trabalho de dissertação e pelas condições de trabalho que me proporcionaram.

Ao Professor Adjunto Eurico Lopes e a todos os colegas pelo ânimo e apoio que sempre me transmitiram.

Ao Eng. Gabriel Pires por me introduzido ao simulador do Nomad, pela ajuda e apoio desde o início do Mestrado.

Ao Eng. António Paulino pela sua disponibilidade e ajuda na resolução de problemas no Linux e na comunicação com o robô Super Scout II.

Ao Rui Amaral pela sua contribuição na instalação do novo hardware no robô Super Scout II.

Ao Professor Doutor Rui Araújo e ao Eng. Rui Cortesão pelas suas sugestões e críticas.

À minha mulher Cláudia por todo o apoio e compreensão.

Aos meus Pais e irmãos pelo ânimo e incentivo que sempre demonstraram.

Aos familiares e aos amigos que de alguma forma me ajudaram na realização deste trabalho.

À FCT (Fundação para a Ciência e Tecnologia), projecto Mentor/1998, cujo financiamento proporcionou parcialmente as condições experimentais para o desenvolvimento do trabalho de investigação.

A todos muito obrigado.

Resumo

A construção de mapas é muito importante para dotar sistemas que envolvem robôs móveis com níveis crescentes de autonomia. No Instituto de Sistemas e Robótica (ISR-Coimbra) existem projectos a decorrer relacionados com a navegação de robôs móveis orientados-a-humanos, mais concretamente envolvendo uma cadeira de rodas motorizada. Os mapas locais são uma componente importante a incorporar na navegação deste tipo de robôs móveis.

O objectivo do trabalho apresentado nesta dissertação consistiu no desenvolvimento de uma arquitectura para a construção de um mapa de grelha de células que modelize dinamicamente as características locais do meio envolvente do robô móvel, em termos de regiões ocupadas e regiões desocupadas. Esse mapa pode ser construído utilizando várias fontes de informação sensorial e é independente da disposição dos sensores.

A arquitectura é constituída por módulos. Um dos módulos implementa uma rede neuronal artificial com propagação para a frente, através da qual se interpreta a informação sensorial, tendo por objectivo determinar, para cada célula, a probabilidade de estar ocupada. Depois de conhecida a referida probabilidade, as células são actualizadas através da aplicação de uma fórmula baseada no teorema de Bayes. Foram ainda implementados outros módulos para a pesquisa de todas as células e para a escolha dos sensores mais adequados à actualização de cada célula.

O método apresentado foi testado em ambientes reais e simulados. Os mapas obtidos em cada ambiente foram analisados segundo determinados parâmetros.

Abstract

Map building is very important to endow systems that involve mobile robots with growing levels of autonomy. In the Institute of Systems and Robotics (ISR-Coimbra) there are some projects related with the navigation of human-oriented mobile robots, more precisely involving a motorized wheelchair. Local maps are an important component to be incorporated in the navigation of this type of mobile robots.

The aim of the work presented in this thesis had consisted in the development of an architecture to build maps of grid cells that models dynamically the local characteristics of robot's surroundings, in terms of occupied and unoccupied regions. The map can be build using several sources of sensory information and is independent of sensors layout.

The architecture is composed by modules. One of them implements a feedfoward artificial neural network, through which sensory information is interpreted, having in view to determine the probability of a cell being occupied. After the probability is known, the cells are updated through the application of a formula based in Bayes theorem. Other modules were implemented to examine all the cells and to choose the appropriate sensors to update each cell.

The method was tested in real and simulated environments. The maps obtained in each environment were analysed according to some relevant parameters.

Conteúdo

Abreviaturas e Tradução de Termos em Inglês

Propagação para a frente. Tradução da palavra inglesa *Feedfoward*.

Redes concorrentes. Tradução de *Competitive networks*.

Redes recorrentes. Tradução de *Recurrent networks*.

Retro-propagação. Tradução da palavra inglesa *Backpropagation*.

RNA. Rede Neuronal Artificial.

LRF. '*Laser Range Finders*'. Dispositivos de medição de distância constituídos por uma câmara e uma fonte de luz estruturada.

Clustering. Algoritmo para agrupar células.

Transformada de Hough. Algoritmo utilizado para procurar segmentos de recta em conjuntos discretos.

Camada oculta. Tradução de *Hidden Layer* pode também ser referida por camada escondida.

Off-line. Termo utilizado para indicar a execução separada de determinado algoritmo.

Capítulo 1

Introdução

Para alargar a gama de aplicações de robôs, tanto na indústria como na investigação, torna-se necessário desenvolver sistemas com elevados níveis de autonomia e grande capacidade de trabalhar em ambientes não estruturados, com pouca informação a priori. Para conseguir um elevado grau de autonomia, o robô deve possuir um grande conhecimento do ambiente envolvente, adquirindo e manipulando um bom modelo do seu ambiente de trabalho ou operação. A informação acerca do ambiente envolvente é adquirida através de vários sensores, sendo necessários mecanismos para extrair a informação relevante à construção do modelo do meio envolvente. Sistemas com pouca ou nenhuma capacidade sensorial estão limitados a sequências de operação fixas, em ambientes estruturados, não podendo ter qualquer grau de autonomia ou capacidade de adaptação [Elfes 87].

1.1 Construção de Mapas em Robótica Móvel

A construção de mapas de ambientes interiores tem sido desenvolvida segundo duas vertentes: a representação geométrica (baseada em grelhas de células ou em primitivas geométricas) e a representação topológica. Um dos modelos mais utilizados, dentro da representação geométrica, é aquele que se baseia numa grelha de células. Neste modelo, o ambiente apresenta-se sob a

forma de uma matriz bidimensional de células representando uma área onde serão representadas zonas ocupadas e zonas livres (ou desocupadas). Cada célula da matriz contém um valor que traduz a confiança (ou certeza) acerca da ocupação da área real correspondente que representa. Este conceito foi desenvolvido por Alberto Elfes [Elfes 87] e desde então tem sido muito utilizado no domínio da robótica móvel. Com este trabalho de investigação e muitos outros que se seguiram ficou provado que é possível construir mapas, com erros relativamente baixos, associando as grelhas de células a sensores sonares. Estes mapas são também adequados para resolver problemas relacionados com o planeamento de caminhos e para evitar colisões com obstáculos [Borenstein-Koren 89]. No entanto, torna-se muito complexo utilizar estes mapas para a extracção de formas e para o cálculo de estimativas da posição do robô [Vandorpe et al. 96].

Em alternativa aos modelos baseados em grelhas de células, ainda no contexto da representação geométrica, existem modelos que se baseiam na descrição do ambiente através de primitivas (elementos) geométricas. As primitivas geométricas pertencem quase sempre a um grupo constituído por segmentos de recta, círculos ou cantos. Alguns métodos para a construção de mapas baseiam-se apenas num par de primitivas, sendo estas segmentos de recta e círculos ou segmentos de recta e cantos, dependendo da forma como se assume o tipo de ambiente envolvente [Leonard et al. 92]. Em ambientes interiores (salas, corredores, etc) podem-se utilizar apenas segmentos de recta e cantos, mas a detecção de um elevado número de pequenos objectos (pernas de cadeiras ou de mesas) poderá complicar o modelo do ambiente [Vandorpe et al. 96].

Apesar de alguns métodos associarem medidas de incerteza a cada primitiva, a actualização do mapa e a correspondência entre as mesmas primitivas observadas, através de sensores, de pontos de vista distintos pode tornar-se num problema muito complexo.

Existem alguns métodos híbridos que utilizam ambas as representações geométricas (grelhas de células e primitivas geométricas). Os trabalhos que abordam esta combinação [Weigl et al. 93], [Howard-Kitchen 96] e [Anousaki-Kyriakopoulos 99] começam por construir um mapa de grelhas de células e depois, sobre este, aplicam algoritmos para a construção de mapas baseados em primitivas geométricas.

A representação topológica conduz a descrições do ambiente baseadas em grafos. Nestes grafos os nodos representam locais ou marcos especiais distintos uns dos outros. Os arcos correspondem a caminhos, passagens ou ligações existentes entre locais ou marcos especiais.

Estes arcos podem ou não conter informação complementar, como por exemplo a distância da ligação e podem ser ainda interpretados como acções para se mover de um local para outro. Existem duas formas para construir mapas topológicos: a primeira é construir o mapa directamente a partir dos dados sensoriais; a segunda é construir, em primeiro lugar, um mapa geométrico e a partir deste construir o mapa topológico. Exemplos de algoritmos que fazem a construção do mapa topológico directamente podem ser encontrados em [Kuipers-Byun 91], [Choset et al. 97] e [Shatkay-Kaelbling 97]. Em relação à construção de mapas topológicos a partir de mapas geométricos pode-se referir o trabalho apresentado em [Thrun 98]. Alguns algoritmos utilizados para construir mapas topológicos utilizam diagramas de *Voronoi* [Thrun 98] e [Choset et al. 97].

A forma como são interpretadas as leituras dos sensores também foi objecto de atenção por parte dos investigadores desta área. As abordagens anteriores baseavam-se em modelos probabilísticos aliados sempre ao modelo do sensor. Em [Thrun 98] foi apresentada uma nova forma de interpretar as leituras dos sensores recorrendo a uma RNA, utilizando mapas de grelhas de células. As entradas da RNA são as coordenadas polares da célula a actualizar e quatro leituras de distância provenientes de sensores, com orientações próximas da orientação da célula. A única saída da RNA representa a probabilidade dessa célula estar ocupada. As RNAs também foram utilizadas, de forma semelhante, em [Arleo et al. 99] na construção de mapas. Os resultados obtidos foram muito positivos e motivaram a utilização da RNA no presente trabalho.

1.2 Âmbito e Objectivos da Tese

No Instituto de Sistemas e Robótica (ISR - Coimbra) existem projectos a decorrer relacionados com a navegação de robôs móveis orientados-a-humanos, mais concretamente envolvendo uma cadeira de rodas motorizada [Nunes et al. 2000a, Nunes et al. 2000b]. Os mapas locais são uma componente importante a incorporar na navegação deste tipo de robôs móveis. O conhecimento do ambiente envolvente fornecido ao controlador do robô móvel pode funcionar

como um dispositivo de segurança e pode assegurar uma navegação livre de colisões. Além disso, pode ser utilizado para facilitar manobras e ainda para planear trajectórias locais.

O objectivo da tese é construir um mapa de grelhas de células que modelize dinamicamente as características locais do meio envolvente do robô móvel, em termos de regiões ocupadas e regiões desocupadas. Esse mapa deve poder ser construído utilizando várias fontes de informação sensorial e ainda ser independente da configuração dos sensores.

Este trabalho foi inspirado nos trabalhos apresentados em [Thrun 98] e [Burgard et al. 98], onde foi utilizada uma RNA para interpretar as leituras dos sensores. Os resultados muito positivos motivaram a investigação desta técnica com o objectivo de construir mapas locais. As células do mapa local são actualizadas através de uma fórmula baseada no teorema de Bayes, integrando ao longo do tempo as novas leituras dos sensores, iterativamente e em tempo real. O mapa local é constituído por uma grelha com $n \times n$ células e representa uma visão local do ambiente envolvente que acompanha os movimentos do robô móvel.

1.3 Trabalho Realizado

O trabalho consistiu no desenvolvimento de um algoritmo para a construção de um mapa local do ambiente envolvente de um robô móvel. O modelo para o mapa local baseia-se numa grelha de células. A cada célula está associado um valor, pertencente ao intervalo $]0, 1[$, cujo significado é a probabilidade dessa célula estar ocupada. Um valor próximo de 1 significa que está ocupada enquanto que um valor próximo de zero significa que está desocupada. Se esse valor estiver muito próximo de 0.5 então existe incerteza quanto ao seu estado (ocupada ou livre). Inicialmente atribui-se a todas as células o valor 0.5.

Neste trabalho foi desenvolvida uma arquitectura para a construção de um mapa de grelha de células. Esta arquitectura é constituída por módulos. Um dos módulos implementa uma RNA com propagação para a frente (incluindo a parte relacionada com o seu treino), para determinar a probabilidade de uma célula estar ocupada. A RNA tem quatro entradas: as coordenadas polares da célula em causa (duas) e duas leituras de distância correspondentes aos dois sensores com orientações mais próximas da célula. Posteriormente desenvolveu-se um algoritmo para

visitar todas as células no campo de 'visão' de todos os sensores, aplicando a cada célula a fórmula de actualização proposta em [Thrun 98] baseada no teorema de Bayes. A visita das células implica a utilização de alguns módulos relacionados com a escolha dos sensores e dos sectores aos quais pertencem as células a actualizar. Foi também incluída a odometria para fazer o deslocamento correcto do mapa, de acordo com os movimentos do robô. Foi ainda desenvolvido um interface gráfico que permite a visualização da grelha de células num monitor.

Este método para a construção do mapa local de um robô móvel apresenta quatro características importantes:

- Independência da Disposição dos Sensores: Para a construção do mapa é apenas necessário conhecer a medida de distância e a orientação do sensor;

- Fusão Sensorial: O método é independente do tipo e quantidade de sensores existentes no robô móvel. Existe um módulo encarregue de fornecer, em cada momento, a melhor medida de distância em determinada orientação (mesmo que exista mais de um sensor para a mesma orientação);

- Ambientes Dinâmicos: Detecta rapidamente mudanças no ambiente envolvente.

- Independência do Tamanho da Célula: A dimensão das células pode ser alterada pois o método não está dependente de uma dimensão específica.

O trabalho foi desenvolvido em ambiente linux utilizando o simulador da *Nomadic Technologies*. Foi depois transferido para um robô móvel real onde também foi testado com sucesso.

1.4 Organização da Dissertação

A dissertação está organizada da seguinte maneira: no segundo capítulo são apresentados, de uma forma resumida, os métodos mais relevantes que têm sido utilizados na construção de mapas em robótica móvel; as RNAs são abordadas no capítulo três onde se apresentam as

RNAs mais utilizadas actualmente, as respectivas características e tudo o que se relaciona com a RNA utilizada neste trabalho (método de treino – retro-propagação, função de activação e algoritmos de treino e implementação); o ambiente de simulação e as características do robô real, onde foi testado o algoritmo, são descritos no quarto capítulo; no quinto capítulo é apresentada a arquitectura do método incluindo a descrição de todos os seus componentes, procedimentos e algoritmos; os resultados e respectiva discussão podem ser encontrados no sexto capítulo; no sétimo capítulo são descritos os procedimentos e as alterações necessárias para aplicação do algoritmo a outros tipos de robôs móveis; para finalizar, o oitavo capítulo contém as conclusões e propostas para futuros desenvolvimentos.

Capítulo 2

Construção de Mapas Locais em Robótica Móvel

Neste capítulo são apresentados trabalhos de investigação na área da construção de mapas em robótica móvel. Os trabalhos estão agrupados segundo duas vertentes: a construção de mapas utilizando as representações geométrica e topológica. Dentro da representação geométrica, os métodos foram ainda divididos em três grupos: os que se baseiam em grelhas de células; em primitivas geométricas; e aqueles que utilizam grelhas de células e primitivas geométricas. Os trabalhos abordados não representam o universo de trabalhos na área, simbolizam apenas as orientações principais e mais marcantes da investigação na área.

Para que um robô móvel possa desenvolver actividades específicas em ambientes interiores deve ser capaz de adquirir e manter um bom modelo desses ambientes. Adquirir bons modelos não é tarefa fácil e ainda existe um longo caminho a percorrer, uma vez que existem diversos factores que impõem limitações à aquisição e construção de modelos precisos [Thrun 98]:

- Sensores: Os sensores nem sempre podem medir directamente aquilo que se pretende. Por exemplo, as câmaras podem medir a cor e o brilho, os sensores de proximidade (sonares ou infravermelhos) medem distâncias, no entanto, para apoio à navegação haveria interesse em identificar a existência de características específicas como por exemplo a localização de uma porta;

- Limitações Perceptuais: A gama perceptual da maioria dos sensores está limitada a uma área à volta do robô. Para adquirir informações globais, o robô deve explorar todo o ambiente;

- Tempo Real: O modelo e o método de construção e manutenção do mapa deve permitir alterações em tempo real. Durante a navegação ou em movimentos de manobra, o robô deverá reagir rapidamente às eventuais alterações que o ambiente possa sofrer de modo a evitar colisões.

- Desvios e Deslizamentos: Os movimentos do robô também não são precisos. Um problema importante é que os erros da informação odométrica vão-se acumulando ao longo do tempo. Por exemplo, um erro pequeno numa rotação poderá provocar grandes efeitos nos erros em movimentos de translação seguintes;

- Ruído dos Sensores: As medidas de distância provenientes dos sensores são corrompidas por ruído e a distribuição desse ruído nem sempre é conhecida com precisão;

- Características do Ambiente: Normalmente, os ambientes reais onde o robô trabalha são complexos e dinâmicos (com pessoas a circular). Por conseguinte, é quase impossível manter modelos exactos e fazer estimativas com elevada precisão;

Como já foi referido anteriormente, a investigação em robótica móvel, no domínio da construção de mapas, conduziu a dois paradigmas fundamentais na modelação de ambientes interiores: a representação geométrica e a representação topológica. Na representação geométrica estão incluídos os modelos baseados em primitivas geométricas (segmentos de recta, cantos ou círculos) e os modelos baseados em grelhas de células onde o ambiente é dividido em pequenas áreas (a cada área está associado pelo menos um valor que traduz a ocupação ou não dessa área). A representação topológica conduz a modelos do ambiente descritos através de grafos onde os nodos correspondem a locais ou marcos específicos e os arcos simbolizam caminhos ou espaços livres entre esses locais ou marcos.

2.1 Representação Geométrica

Nesta secção serão abordados os três tipos de mapas com informação geométrica do ambiente: apenas grelhas de células; grelhas de células e primitivas geométricas; e apenas primitivas geométricas. Para cada caso serão apresentados alguns trabalhos exemplificativos da construção de mapas do ambiente.

2.1.1 Grelhas de Células

Nos modelos baseados em grelhas de células, cada célula pode representar a presença de um obstáculo na área correspondente do meio ambiente. Este método foi inicialmente proposto por Alberto Elfes [Elfes 87] utilizando duas grelhas, uma para guardar as células ocupadas e outra para as células livres. A implementação combinava a probabilidade de ocupação com a incerteza espacial e foram utilizados vários sensores sonares. As respectivas leituras de distância eram interpretadas através de distribuições de probabilidade para determinar áreas ocupadas e áreas livres. Além destas áreas eram também representadas áreas desconhecidas. O método de actualização tinha por objectivo reduzir as incertezas e os erros provenientes dos sensores. Sempre que as áreas desocupadas eram certificadas, em actualizações seguintes, estas eram reforçadas, acontecendo o mesmo para as áreas ocupadas. Borenstein e Koren [Borenstein - Koren 90,91] utilizaram o mapa de grelhas de células para efectuar o desvio de obstáculos em robôs móveis que se deslocavam rapidamente. Para isso, fizeram uma alteração que consistia apenas em actualizar as células presentes no eixo acústico, enquanto que na abordagem anterior [Elfes 87] todas as células, abrangidas pelo cone do sonar, eram actualizadas o que tornava o algoritmo mais lento.

Uma nova forma de actualizar as células baseada no teorema de Bayes foi apresentada em [Moravec 88]. Esta fórmula permitiu também a integração de leituras dos sensores obtidas em instantes de tempo diferentes. Este método veio melhorar a precisão dos mapas de grelhas de células uma vez que, as abordagens anteriores baseavam-se em métodos e modelos estatísticos heurísticos. A teoria Dempster-Shafer [Murphy 98] também pode ser utilizada na actualização das células e na fusão sensorial (de forma semelhante à teoria de Bayes). Em [Murray-Jennings

97] a visão estéreo é aplicada na navegação e construção de mapas. O algoritmo proposto utiliza apenas a visão estéreo (com dois pares de câmaras) e as grelhas de células para construir um mapa do ambiente. Neste caso o valor das células varia entre zero e 255 e um valor elevado significa que a célula está ocupada. Em [Yamauchi et al. 98] o mapa de grelhas é construído utilizando a fusão sensorial entre sonares e um LRF (emitindo um plano de luz). Se o LRF devolver uma distância menor que o sonar será assumido que o valor mais preciso provém do LRF. Os conceitos da lógica difusa também podem ser utilizados na construção de mapas, recorrendo a sensores sonares [Oriolo et al. 97], [Kang et al. 01].

2.1.2 Grelhas de Células e Primitivas Geométricas

Nesta secção serão referidos trabalhos de investigação que abordam métodos de construção de mapas compostos por grelhas de células e primitivas geométricas.

Em [Weigl et al. 93], assumindo um mundo poligonal, quando se detectam duas células ocupadas, relativamente próximas, determina-se a recta que passa por ambas e marcam-se as células atravessadas por essa recta como estando *provavelmente ocupadas*. No trabalho apresentado em [Howard-Kitchen 96] são propostos dois "agentes", um deles constrói um mapa de grelhas de células e o outro "agente" procura marcos baseando-se em conjuntos de características (segmentos de parede e passagens de portas). Este último "agente" prepara então o segundo mapa para posterior utilização em navegação autónoma. Uma abordagem diferente é proposta em [Anousaki-Kyriakopoulos 99] onde inicialmente é construído um mapa de grelhas de células e sobre este são depois aplicados dois algoritmos: o *clustering* e a transformada de Hough. Estes algoritmos têm por objectivo agrupar células de modo a definir segmentos de recta. O resultado é um mapa do ambiente construído através de vários segmentos de recta.

Um novo método para a aquisição da informação sensorial, através de sonares, é proposto em [Moita-Nunes 01]. Este método utiliza ecos múltiplos combinados com a técnica DRUEE (Disparo Rápido Ultra-sónico Eliminatório do Erro), que permite aumentar a taxa de aquisição das medidas e obter medidas simultâneas provenientes de dois sonares. Com esta técnica, e utilizando uma configuração adequada para os sensores, pode-se identificar rapidamente planos,

cantos e arestas. Com este método é possível construir simultaneamente um mapa de grelhas de células com sobreposição de informação geométrica (cantos, planos e arestas).

2.1.3 Primitivas Geométricas

Os métodos que recorrem a este tipo de representação modelizam o ambiente através de primitivas geométricas (segmentos de recta, curvas ou cantos). Neste pressuposto, a interpretação dos sensores é orientada para a procura de objectos no ambiente que possam ser modelizados à custa destas primitivas. Utilizando esta interligação é possível construir mapas do meio ambiente, no entanto, o método de construção fica sujeito a algumas restrições ou suposições (ex. ambientes quase estáticos ou com objectos de grandes dimensões e geométricos).

A construção do mapa na abordagem apresentada em [Leonard et al. 92] baseia-se na procura de figuras geométricas que surgem no ambiente. Através das leituras de sensores sonares, as regiões com distância constante são agrupadas com o objectivo de determinar se correspondem a uma parede, a um canto ou coluna. Neste algoritmo, cada figura geométrica tem associada uma matriz de covariância e uma medida de confiança que traduzem a certeza da localização de uma determinada figura geométrica.

Em [Sabatini-Benedetto 95] são utilizados três sonares dispostos em linha recta, montados numa plataforma rotativa para reconhecer zonas planas (ex. paredes), cantos, arestas e cilindros no ambiente.

Um algoritmo para a construção dinâmica de mapas é apresentado em [Vandorpe et al. 96]. Este mapa é constituído por segmentos de recta e círculos, e os parâmetros que descrevem estas primitivas geométricas têm medidas de incerteza associadas. A informação do mundo é adquirida através de um LRF 2D e o mapa pode ser utilizado para localização e planeamento de caminhos.

Uma estratégia diferente para construir mapas de primitivas geométricas é proposto em [Chong-Kleeman 97]. Nesta abordagem, os elementos planos e cantos são reconhecidos através de três sonares com uma disposição em T invertido. Durante a construção do mapa, os elementos identificados que se encontram muito próximos, ou que são adjacentes, são

agrupados de modo a construir elementos maiores. Desta forma, as lacunas existentes no mapa inicial podem ser eliminadas diminuindo assim os erros existentes no mapa.

2.2 Representação Topológica

Os mapas construídos por algoritmos baseados na representação topológica têm a aparência de grafos. Os nodos destes grafos representam locais ou marcos específicos e os arcos entre os nodos representam caminhos, passagens ou ligações entre os locais respectivos. Os mapas topológicos podem ser construídos directamente ou então a partir dum mapa geométrico. Na abordagem de Kuipers e Byun [Kuipers-Byun 91] o mapa topológico é construído directamente. Os locais distintos e caminhos são definidos em termos de estratégias de controlo e das leituras de distâncias obtidas através de sensores. Os locais topológicos são definidos através de pontos que maximizam o número de obstáculos equidistantes. A partir do mapa topológico é depois construído um mapa geométrico do ambiente utilizando as informações adquiridas durante a construção do mapa topológico. Estas informações incluem as direcções e as distâncias aos objectos mais próximos. Com estas informações, o mapa geométrico é construído através de pequenos pontos que irão dar forma aos contornos do mapa do ambiente. Em [Choset et al. 97] utilizam-se diagramas de *Voronoi* para construir mapas topológicos. Estes diagramas são constituídos por conjuntos de pontos que têm a característica de estarem equidistantes de dois obstáculos (ideia semelhante à abordagem anterior). Em [Shatkay-Kaelbling 97] recorre-se à observação de marcos específicos no ambiente. Assume-se que o mundo é composto por um conjunto finito de estados e que podem conter informações (por exemplo a orientação). A dinâmica do mundo é descrita através de distribuições de transição de estado que especificam a probabilidade de se efectuarem transições de um estado para o seguinte. A este modelo é aplicado um algoritmo recursivo para fazer estimativas de estado e construir assim o mapa topológico à medida que o robô móvel se desloca no ambiente.

Um exemplo de construção de mapas topológicos a partir de um mapa geométrico pode ser encontrado em [Thrun 98]. Nesta abordagem começa-se por construir um mapa de grelhas de células [Elfes 87] utilizando uma actualização baseada no teorema de Bayes ([Moravec-Cho 89] e

[Burgard et al. 98]). Em seguida, o grafo topológico é construído a partir do mapa de grelhas de células. Este método utiliza também o diagrama de *Voronoi*, dividindo o mapa em pequenas regiões sobre as quais se constrói o grafo topológico. Esta abordagem introduz uma novidade, que não tinha sido utilizada antes, que é a interpretação das leituras dos sensores através de uma RNA. Para obter a probabilidade de uma célula estar ocupada (necessária na fórmula de actualização das células do mapa), é utilizada uma RNA (com propagação para a frente) com seis entradas: quatro delas são as leituras de quatro sensores com as orientações mais próximas da orientação da célula e as duas restantes são as coordenadas polares da célula em causa. Os resultados obtidos foram muito positivos uma vez que a contribuição das leituras dos sensores vizinhos aumentou a precisão do mapa. As RNAs também foram utilizadas para interpretar as leituras provenientes de sensores em [Arleo et al. 99]. Nesta abordagem também se constrói o mapa topológico a partir de um mapa de grelhas de células.

2.3 Discussão

Não existe um método suficientemente genérico que funcione bem em todas as situações. Alguns métodos obtêm bons resultados em ambientes estruturados. De seguida são apresentadas as vantagens e desvantagens dos métodos baseados em mapas constituídos por grelhas de células, primitivas geométricas e mapas topológicos.

As grelhas de células têm várias vantagens: geram mapas pormenorizados; podem ser aplicadas a qualquer ambiente; o reconhecimento de locais normalmente não é ambíguo e não está tão dependente do local onde se obtém a informação sensorial; podem servir de base à construção de outros mapas para aplicações específicas; para estimar a posição e orientação do robô; efectuar o reconhecimento de marcas e navegação autónoma. Como desvantagens pode-se referir que certos algoritmos que envolvem a actualização de várias células, em cada iteração, tornam problemática a sua utilização em aplicações de tempo real, sobretudo quando estão associados à visão estéreo. O planeamento de caminhos é difícil e na construção de mapas globais a posição do robô tem que ser conhecida com precisão necessitando ainda de muito espaço de memória. Embora existam métodos que optimizam o número de células a actualizar,

também podem conduzir a mapas com falta de pormenor. Alguns algoritmos têm alguns problemas quando utilizados em ambientes dinâmicos. Estes mapas também não possuem um interface adequado para aplicar algoritmos de resolução de problemas específicos.

Os modelos baseados em primitivas geométricas, que conduzem a mapas bem definidos, podem ser mais eficientes quando aplicados directamente em navegação autónoma, mesmo em ambientes dinâmicos. O problema principal é conseguir um mapa bem definido. Existem muitas fontes de erros desde o sensor ao próprio algoritmo de actualização. Encontrar a correspondência entre o mesmo objecto observado em instantes distintos e de ângulos diferentes é um problema complexo.

Os mapas topológicos permitem um planeamento de caminhos eficiente (recorrendo a algoritmos aplicados a grafos, ex. caminho mais curto), não têm requisitos especiais de espaço de memória e a resolução está dependente da complexidade do ambiente. Estes mapas não requerem a determinação da posição do robô com precisão e são uma representação conveniente para a resolução de problemas específicos. No entanto, a construção e actualização destes mapas pode não ser fácil em ambientes muito grandes, sobretudo se a informação proveniente dos sensores for ambígua (ex. reconhecer locais distintos com características semelhantes). O reconhecimento de locais é muito sensível à posição onde é feita a aquisição sensorial, como consequência, o planeamento de caminhos pode gerar caminhos não óptimos.

O objectivo principal, de todos os métodos, é reduzir ao mínimo os erros inerentes a todas as fontes: sensores, características do ambiente, movimentos, a localização e orientação do robô. Um caminho a seguir para limitar os erros passa pela combinação de vários sensores (sonares, LRF, visão e infravermelhos), odometria e um processamento eficiente para gerir todos os processos simultaneamente, e assim conseguir comportamentos em tempo real.

Capítulo 3

Redes Neuronais Artificiais

O desenvolvimento no domínio das RNAs evidencia a inspiração biológica para a resolução de problemas em diferentes áreas científicas.

Pesquisas em muitos campos científicos levaram à utilização da inteligência artificial para resolver diversificados problemas, nomeadamente, de optimização, classificação, controlo de sistemas, reconhecimento de padrões, processamento de imagem, visão por computador e ainda em áreas tão distintas como a medicina, aplicações militares ou sistemas financeiros.

No cérebro humano existem aproximadamente dez biliões de neurónios e cada um está ligado a cerca de dez mil neurónios. Um neurónio (figura 3.1) consiste numa célula biológica formada pelo corpo celular e por dois tipos de ligações: dendrites e axónios. O neurónio recebe impulsos de outros neurónios através das dendrites e transmite impulsos gerados pelo corpo celular através dos axónios. As sinapses asseguram a transmissão dos impulsos entre as diferentes ligações, de notar que este tipo de ligação não é por contacto físico e pode ser feita entre um axónio e o núcleo da célula, entre dois axónios ou entre um axónio e uma dendrite.

Os neurónios comunicam através de cadeias de impulsos. O melhor tempo de comutação de um neurónio é cerca de um milisegundo, bastante mais lento que a velocidade de comutação dos processadores actuais que já é inferior a um nanosegundo. No entanto, o ser humano consegue tomar decisões muito complexas num curto espaço de tempo. Por exemplo, para que se possa reconhecer, visualmente, uma pessoa muito próxima leva cerca de 0,1 segundos.

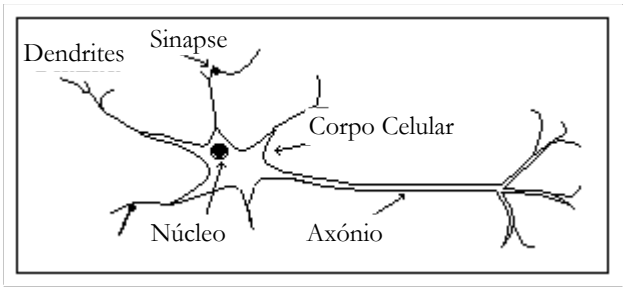

Figura 3.1. Neurónio biológico.

Mesmo com uma velocidade de comutação de cerca de um milisegundo o cérebro consegue tomar decisões rapidamente, usando milhões de neurónios. Esta característica levou muitos a considerar que o cérebro internamente apresentava um funcionamento baseado em processos paralelos, distribuídos pelos diversos neurónios [Mitchell 97].

As RNAs começaram a ser exploradas há cerca de cinquenta anos, mas o estudo e a investigação das mesmas teve um crescimento exponencial nos últimos anos motivada pela crescente capacidade e velocidade computacional.

Este capítulo está organizado da seguinte forma, começa-se por apresentar a definição de uma RNA e os marcos históricos mais significativos na evolução das RNAs. A secção 3.3 aborda as arquitecturas mais utilizadas actualmente, salientando as características específicas de cada uma e as suas aplicações mais frequentes. Na secção seguinte, descreve-se o modo de funcionamento da RNA, o processamento efectuado em cada neurónio artificial (NA) e as funções de activação mais utilizadas no modelo do NA. Na secção 3.5 são apresentados os três paradigmas de treino das RNAs, antes de descrever detalhadamente o algoritmo de treino com retro-propagação e a implementação da RNA com propagação para a frente utilizada no trabalho experimental. Finalmente, são mencionados alguns problemas que podem ocorrer com a utilização do algoritmo de treino com retro-propagação.

Na primeira parte do capítulo é feito o enquadramento da RNA, utilizada no método de construção de mapas, no contexto das RNAs mais importantes. Por isso, nas arquitecturas apresentadas são omitidos vários aspectos relacionados com o seu funcionamento, treino e utilização. Na segunda parte, são aprofundados e desenvolvidos todos os assuntos relacionados com a RNA utilizada, nomeadamente o seu funcionamento, treino e os aspectos fundamentais da sua utilização.

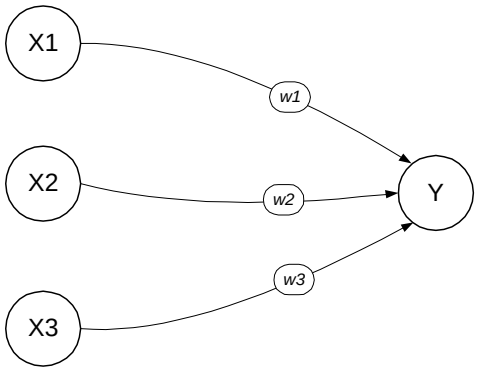

Figura 3.2. Uma rede neuronal artificial muito simples.

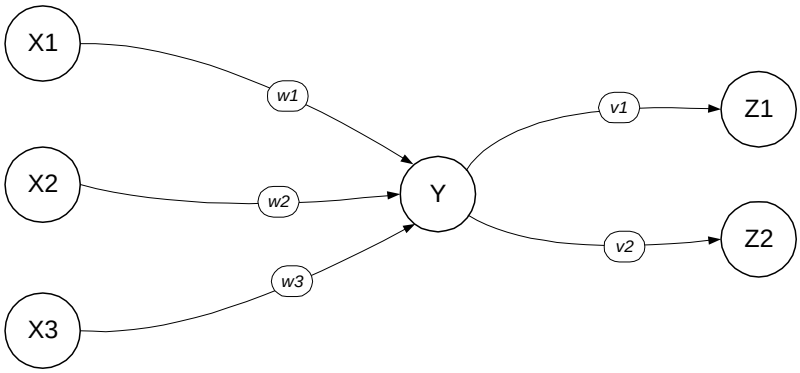

Figura 3.3. Uma rede neuronal artificial.

3.1 Definição da Rede Neuronal Artificial

Uma Rede Neuronal Artificial (RNA) é uma estrutura de processamento de informação paralela e distribuída, constituída por elementos de processamento denominados por neurónios, unidades ou células. Estes neurónios podem ter memória local e levar a cabo operações de processamento de informação local, interligados através de canais de sinal unidireccionais, denominados por ligações. Cada elemento de processamento tem uma única saída que pode ser ligada a muitas outras ligações. Esta saída pode ser de qualquer tipo dentro do contexto matemático. O processamento efectuado em cada elemento de processamento deve ser local, ou

seja, depende apenas dos valores actuais das entradas e dos valores guardados na memória local do respectivo elemento.

Normalmente, a RNA desenvolve uma funcionalidade através de uma ou mais formas de treino. Uma RNA pode não ser constituída só por uma rede, mas por uma família de redes. A funcionalidade ou função da RNA é determinada pela topologia (ou arquitectura) da rede, pelas características individuais dos neurónios, pela estratégia de treino ou aprendizagem e pelos padrões de treino. Por exemplo, o neurónio Y ilustrado na figura 3.2, recebe as entradas a partir dos neurónios X1, X2, e X3. Os sinais de saída destes neurónios são respectivamente x1, x2, e x3. Os pesos nas ligações de X1, X2, e X3 para Y são respectivamente w1, w2, e w3. A entrada do neurónio Y, *y_in* é igual à soma ponderada dos sinais provenientes de X1, X2, e X3, ou seja:

$$y\_in = w_1 \cdot x_1 + w_2 \cdot x_2 + w_3 \cdot x_3 \tag{3.1}$$

A activação do neurónio *y_out* (processamento que gera a saída do neurónio), é dada por uma função matemática aplicada ao sinal de entrada *y_in*, ou seja, *y_out* = *f*(*y_in*). Esta função pode ser do tipo:

$$f(x) = \frac{1}{1 + \exp(-\lambda x)} \tag{3.2}$$

em que λ é uma constante positiva.[1]

Considere-se a RNA da figura 3.3 onde o neurónio Y está ligado aos neurónios Z1 e Z2, com os pesos v1 e v2 nas respectivas ligações. O neurónio Y envia o seu sinal de saída para Z1 e Z2, mas os valores recebidos por Z1 e Z2, geralmente, serão diferentes uma vez que serão afectados pelos pesos v1 e v2 que dificilmente serão iguais. Normalmente, nas RNAs utilizadas na prática, a activação dos neurónios Z1 e Z2 (as saídas da rede) depende de vários ou mesmo de um elevado número de neurónios, e não de apenas um como no exemplo da figura 3.3. Em certas aplicações podem existir várias camadas (ou níveis) ocultas. No exemplo da figura 3.3 existe apenas uma camada oculta com o neurónio Y.

[1]Na secção (3.5.2) serão apresentadas as funções de activação mais utilizadas nas RNAs.

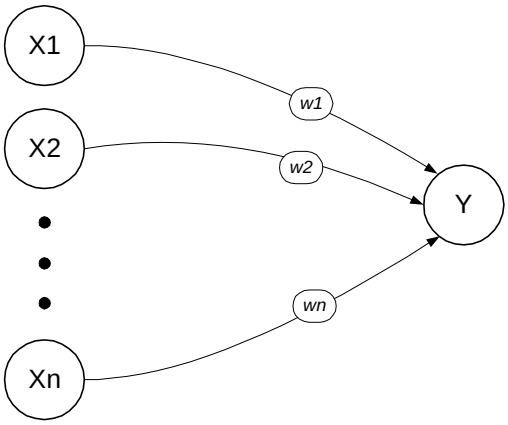

Figura 3.4. Arquitectura de um perceptrão.

3.2 História das Redes Neuronais

As primeiras redes neuronais foram apresentadas por Warren McCulloch e Walter Pitts em 1943 [McCulloch-Pitts 43]. Estes investigadores utilizaram RNAs para implementar as funções lógicas mais simples. Desta forma, podiam combinar essas RNAs em redes mais complexas para produzir uma saída, que pudesse ser representada à custa de combinações dessas funções lógicas.

Em 1949, Donald Hebb, um psicólogo da Universidade de McGill, enunciou a primeira lei para aprendizagem das RNAs: "se dois neurónios forem activados simultaneamente, então o peso da ligação entre eles deve ser aumentado". Esta lei foi refinada para a aplicação nas simulações em computador.

Nos anos cinquenta e sessenta, um grupo de investigadores Frank Rosenblatt, Blok, Minsky & Papert, introduziu e desenvolveu uma classe de RNAs denominada por perceptrão. O perceptrão mais vulgar consiste numa camada de entrada, com um número variável de neurónios, ligada à camada de saída que apenas tem um neurónio (figura 3.4). Os pesos das ligações são ajustáveis.

Bernard Widrow e Marcian Hoff, em 1960, desenvolveram uma regra de aprendizagem (regra Delta) para RNAs do tipo perceptrão. Esta regra permite o ajuste dos pesos das ligações, utilizando o erro médio quadrático, sempre que um neurónio apresenta uma resposta incorrecta e foi precursora do algoritmo de retro-propagação para RNAs com várias camadas.

Nos anos setenta, investigadores como Teuvo Kohonen e James Anderson trabalharam com RNAs com a configuração de memória associativa. Gail Carpenter e Stephen Grossberg desenvolveram a teoria de ressonância adaptativa (ART) para RNAs auto-organizadas. Esta teoria tem duas variantes: ART1 para padrões de entrada com valores binários e ART2 para padrões de entrada com valores contínuos.

A seguir a uma década mais calma no domínio das RNAs, na década de oitenta deu-se a descoberta de um método geral para o treino de RNAs com várias camadas. Em 1985, Parker descobriu então um método para propagar a informação do erro da camada de saída para as camadas anteriores (as camadas ocultas), denominado de retro-propagação. Este método era similar a um outro descoberto em 1974 por Werbos mas na altura não obteve grande sucesso.

Neste pequeno caminho histórico, pelo mundo das RNAs, estão resumidas as descobertas chave que fomentaram todas as aplicações e evoluções das RNAs que hoje são conhecidas. Associadas a um poder computacional crescente certamente irão continuar a ser utilizadas em aplicações cada vez mais complexas e mais evoluídas.

3.3 Arquitecturas das RNAs

As RNAs podem ter várias configurações e dimensões. Nesta secção serão abordadas as arquitecturas mais utilizadas.

3.3.1 Redes com PROPAGAÇÃO PARA A FRENTE

Nas arquitecturas com propagação para a frente as activações dos neurónios de entrada são propagadas através da rede até que os valores dos neurónios de saída estejam determinados. Uma rede com propagação para a frente é composta por uma hierarquia de unidades de processamento (neurónios), organizadas numa série de dois ou mais conjuntos de neurónios ou camadas [Schalkoff 97]. Na figura 3.5 está representada a configuração genérica para a arquitectura com propagação para a frente. A camada de entrada é utilizada para apresentar as entradas, em cada momento, à RNA, enquanto que a última camada fornece a saída da RNA.

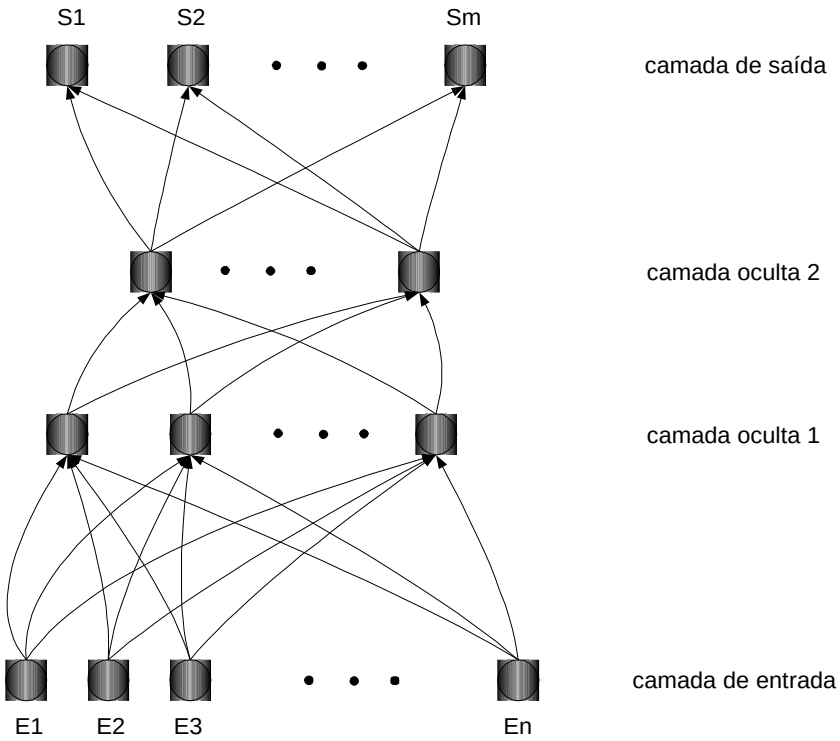

Figura 3.5. Configuração genérica para uma arquitectura com propagação para a frente. O número de neurónios em cada camada e o número de camadas ocultas não tem limite conceptual. Todas as ligações entre os neurónios têm pesos associados.

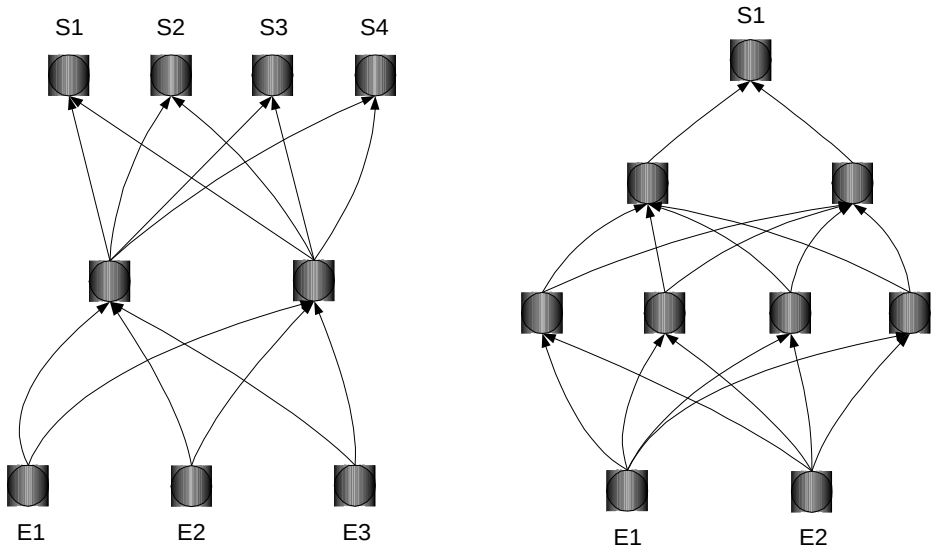

Figura 3.6. Exemplo de redes com propagação para a frente a) 3/2/4, b) 2/4/2/1. A sigla 3/2/4 indica a existência de três camadas. A camada de entrada tem três neurónios, a camada oculta tem dois neurónios e a camada de saída tem quatro neurónios.

Entre as duas camadas anteriores, podem existir desde zero a várias camadas ocultas; é nestas camadas que é realizado a maior parte do processamento da RNA. Nas ligações entre neurónios existe um factor (peso da ligação) que será multiplicado pelo valor da activação (ou saída) do neurónio origem, antes de chegar ao neurónio destino. Na figura 3.6 estão exemplificadas duas configurações concretas para uma rede com propagação para a frente.

Este tipo de RNA pode ser utilizado em todas as situações onde se conheçam os padrões de saída correspondentes aos padrões de entrada para os quais se pretende treinar a RNA. Desta maneira, cada vez que a RNA processa um padrão de entrada (na fase de treino) pode comparar o seu padrão de saída com o padrão de saída desejado e actuar nos parâmetros da RNA, de modo a minimizar a diferença.

3.3.2 Redes CONCORRENTES

Este tipo de redes é semelhante a uma rede com propagação para a frente sem camadas ocultas, a única diferença reside no facto de haver ligações entre os neurónios da camada de saída. Podem ser ligações positivas (sinal excitatório) ou ligações negativas (sinal inibitório). Estas ligações fomentam uma competição (em inglês denominam-se "*Competitive networks*") entre os neurónios de saída para representar o padrão de entrada. A figura 3.7 contém um exemplo deste tipo de redes.

A configuração da camada de saída neste tipo de RNAs pode contribuir para a resolução de ambiguidades entre neurónios daquela camada, pode ainda ser utilizada na identificação de grupos para padrões com características semelhantes.

3.3.3 Redes RECORRENTES

Redes recorrentes são RNAs que têm ciclos fechados na arquitectura da rede. Nas figuras 3.8 e 3.9 estão ilustradas algumas configurações deste tipo de arquitectura. A RNA da figura 3.8 é um exemplo de uma rede recorrente totalmente interligada e simétrica mas sem ligação de um neurónio para si próprio.

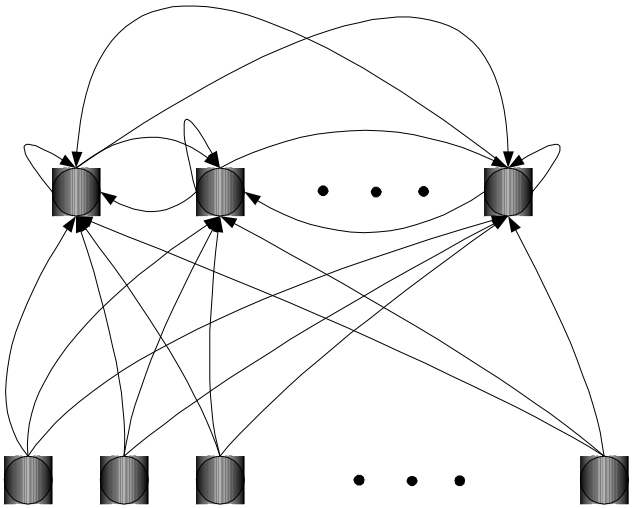

Figura 3.7. Exemplo de uma rede concorrente do tipo "*winner-take-all*".

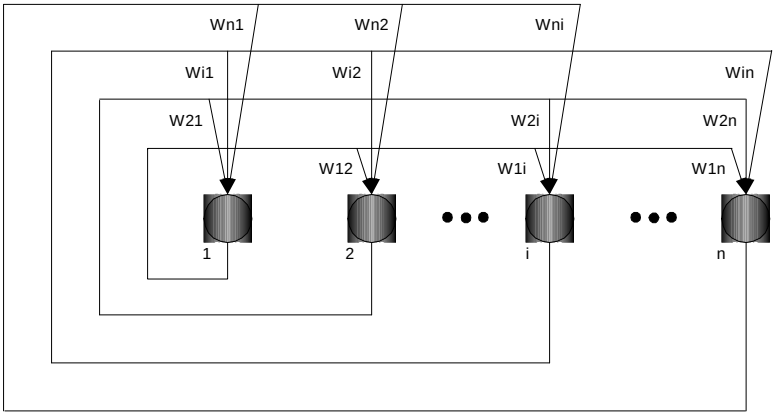

Figura 3.8. Rede recorrente do tipo *Hopfield* para n neurónios.

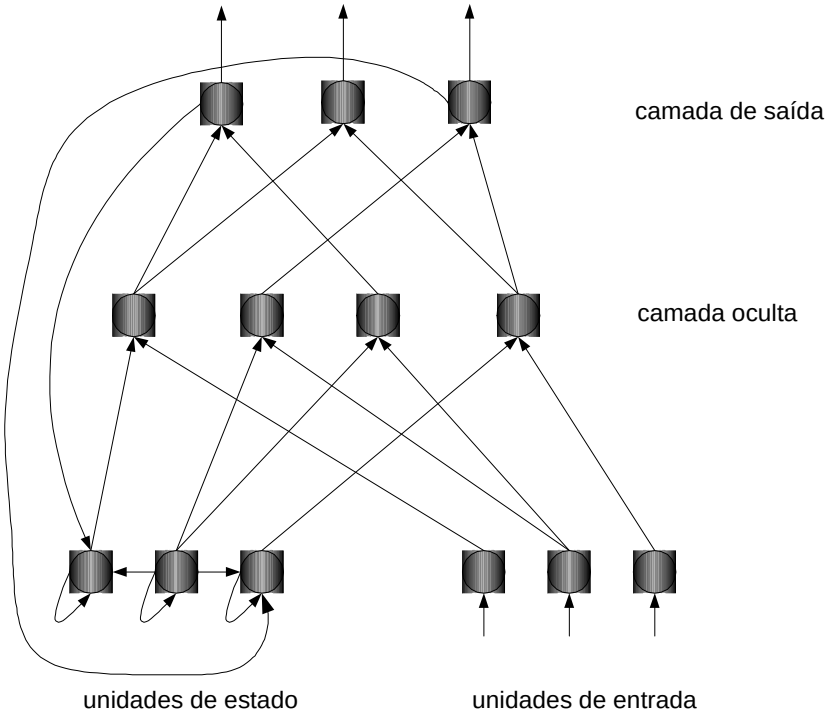

Figura 3.9. Exemplo de uma rede recorrente do tipo *Jordan*.

As redes deste tipo, quando um padrão é apresentado na camada de entrada, levam mais tempo a estabilizar do que no caso de uma rede com propagação para a frente. Isto acontece porque se desencadeia um processo circular (ou de ressonância) de actividade neuronal, dentro da rede, onde algumas (ou todas) camadas de neurónios são activadas repetidamente. Com um treino adequado, este processo circular termina quando a rede atinge um estado estável, senão pode continuar indefinidamente. O facto de possuir ligações bidireccionais introduz uma dinâmica que não existe nas RNA com propagação para a frente. No entanto, esta dinâmica pode provocar comportamentos estáveis ou instáveis e até mesmo caóticos [Bose-Liang 96].

Normalmente, este tipo de redes é utilizado em memórias associativas e em aplicações de optimização e satisfação de restrições.

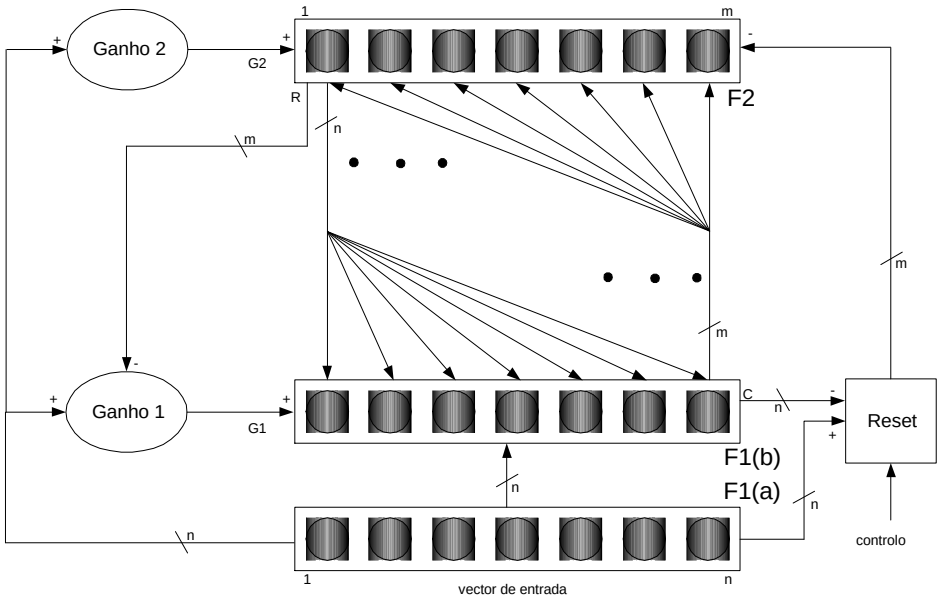

Figura 3.10. Arquitectura genérica para uma rede ART1.

3.3.4 Redes ART1

Normalmente, as redes do tipo "*Adaptative Resonance Theory*" (ART1) são utilizadas para padrões de entrada constituídos por valores binários.

A arquitectura duma rede ART consiste em duas camadas principais de neurónios denominada genericamente por (F1) e (F2). Por vezes, estas camadas têm nomes específicos, em [Pandya-Macy 96] a camada (F1) é conhecida por camada de comparação e (F2) camada de reconhecimento. Em [Fausett 94] são denominadas por interface de entrada e grupo de neurónios, respectivamente. A camada (F2) é denominada por "*winner-take-all*" em [Bose-Liang 96] pois existem ligações entre os neurónios desta camada (ver figura 3.7).

Além das camadas (F1) e (F2), existe ainda uma unidade de *RESET* para controlar o grau de semelhança dos padrões da camada (F2).

Para facilitar a compreensão desta arquitectura, a camada (F1) é expandida em duas, para que se possa representar explicitamente um camada de neurónios representando as entradas. Descrevendo muito sucintamente a arquitectura ART1, pode-se observar na figura 3.10 que a camada de entrada (F1a) tem o mesmo número de neurónios que a camada (F1b), e que existe

uma ligação unidireccional entre cada neurónio correspondente, ou seja, *n* ligações. São também *n* ligações que partem da camada de entrada e da camada (F1b) para a unidade de *RESET*. De cada neurónio da camada (F1b) parte uma ligação para cada um dos *m* neurónios da camada (F2), ou seja, partem *m* ligações de cada neurónio da camada (F1b) para a camada (F2). Na camada (F2), partem *n* ligações para a camada (F1b), uma para cada neurónio. As ligações entre a camada (F1b) e (F2) têm pesos associados. Os neurónios da camada (F2) têm ligações entre si do tipo "*winner-take-all*" (ver figura 3.7). Para controlar melhor o comportamento da rede utilizam-se duas unidades de ganho (G1) e (G2). Cada neurónio presente nas camadas (F1b) e (F2) possui três fontes de sinal. Na camada (F1b) cada neurónio recebe sinais dos neurónios da camada (F1a), dos neurónios da camada (F2) e da unidade de ganho (G1), por sua vez, os neurónios da camada (F2) recebem sinais da camada (F1b), da unidade de ganho (G2), da unidade de *RESET* e dos outros neurónios da camada (F2). Existe ainda uma ligação entre os *m* neurónios da camada (F2) e a unidade (G1).

As RNAs com arquitectura ART são orientadas para a formação de grupos com determinado significado a partir dos padrões de entrada. Uma das aplicações deste tipo de RNA é a classificação e reconhecimento de padrões, todavia existe uma particularidade que a distingue das outras arquitecturas. Estas RNAs podem ser dotadas da capacidade de "aprender" novos grupos sem "esquecer" os grupos já organizados noutro treino, ou seja, o conhecimento adquirido anteriormente. Pelo contrário, as RNAs apresentadas anteriormente se tiverem que reagir a novos padrões de entrada, muito distintos daqueles que constituíram o treino anterior, podem começar a produzir erros consideráveis. Se isto acontecer, a RNA tem que ser treinada de novo com todos os padrões.

Com a arquitectura ART termina a secção onde foram apresentadas quatro arquitecturas que representam uma parte importante das arquitecturas mais utilizadas nas RNAs. Existem muitas variantes em cada arquitectura, como por exemplo as redes ART2, que têm um comportamento semelhante às redes ART1 mas para valores contínuos e não apenas binários. Contudo, esta pequena diferença, em termos de arquitectura tem um impacto elevado no aumento da sua complexidade [Fausett 94].

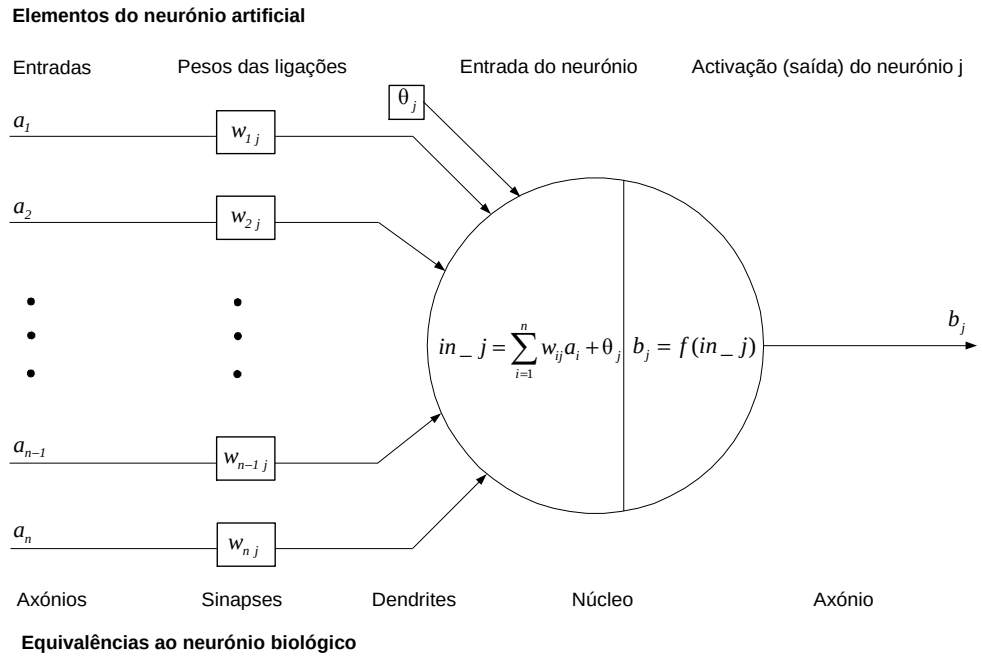

Figura 3.11. Modelo do neurónio artificial (NA). Cada a_i representa uma entrada, o θ_j representa o parâmetro *BIAS* e b_j a saída do NA.

3.4 Funcionamento das RNAs

Como já foi referido anteriormente, uma RNA é constituída por neurónios artificiais. O modelo para o NA que é utilizado praticamente em todas as RNAs é apresentado na figura 3.11. Na mesma figura estabeleceu-se um paralelo entre o NA e o neurónio biológico (NB). O NA, assim como o NB, recebe várias entradas provenientes de outros neurónios afectadas por um factor multiplicativo (denominado geralmente por peso da ligação), e produz apenas uma saída. No entanto, esta saída pode ser ligada a muitos outros neurónios. O neurónio em si (a parte do núcleo) comporta-se como uma função de transferência. Esta função de transferência é constituída por duas partes: na primeira é efectuado um somatório de todas as entradas (afectadas pelos respectivos pesos das ligações) e ainda de uma variável denominada em inglês por *BIAS* (ver parágrafo seguinte); na segunda parte, é aplicada uma função de activação (ver secção seguinte) ao resultado da primeira parte ficando então definido o valor da saída do respectivo neurónio.

O parâmetro *BIAS* é uma entrada adicional em cada neurónio. Este parâmetro é somado à soma ponderada das restantes entradas e pode ser afectado por um peso na ligação a cada neurónio. Este pequeno ajuste pode contribuir para melhorar as propriedades de convergência da RNA. De facto, o parâmetro *BIAS* fornece mais um ponto de ajuste e pode ser determinante para que o valor do somatório das entradas seja conduzido para o conjunto de valores correcto. Normalmente, a unidade *BIAS* tem o valor 1 e o peso da ligação entre esta unidade e cada neurónio também pode ser modificado durante a fase de treino da RNA. Na secção (3.7.3) será descrito o método para a sua actualização.

3.4.1 Funções de Activação

Normalmente, nos modelos dos neurónios artificiais, a primeira parte do processamento consiste num somatório das entradas, no entanto, na segunda parte existem algumas escolhas possíveis de funções de activação a utilizar para produzir a saída do neurónio.

Inicialmente, as funções de activação eram lineares e aumentavam constantemente em função do valor de entrada. Consequentemente, o comportamento destas RNAs era muito instável uma vez que a saída dos neurónios tinha tendência para aumentar sem limites. A seguir são apresentadas as funções de activação mais utilizadas.

Função Lógica

Esta função, também denominada função binária ou em degrau (figura 3.12) é utilizada para implementar neurónios binários muito simples. A sua activação é '1' se o resultado da primeira parte do processamento no neurónio excede determinado limiar e zero caso contrário. A variante bipolar (figura 3.12) tem como limites 1 e –1. Este tipo de funções foi muito utilizado nas redes Hopfield, no entanto, as RNAs com este tipo de activação apresentam algumas limitações na capacidade de aprendizagem e no seu desempenho [Boullart et al. 92].

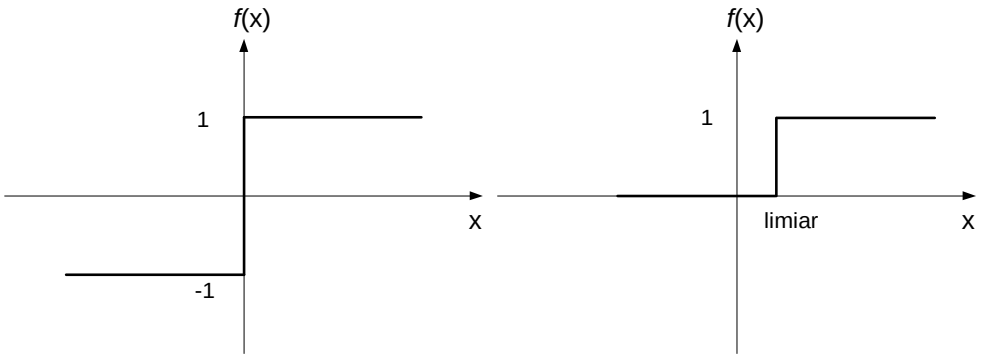

Figura 3.12. Função Lógica. A versão bipolar [-1, 1] encontra-se à esquerda.

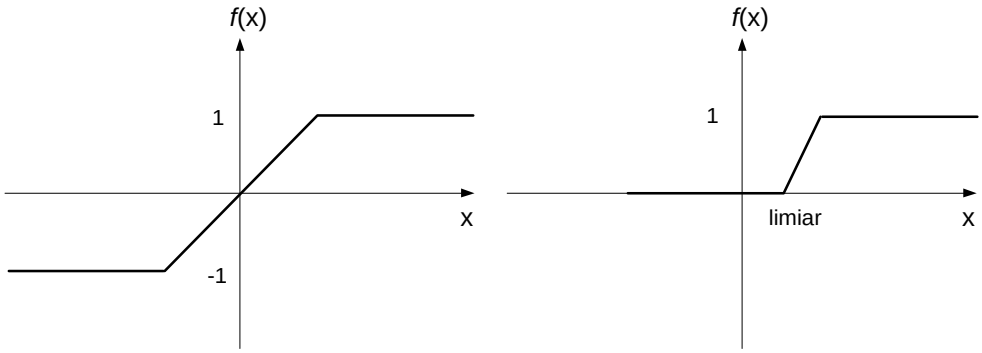

Figura 3.13. Função Semi-Linear. A versão bipolar [-1, 1] encontra-se à esquerda.

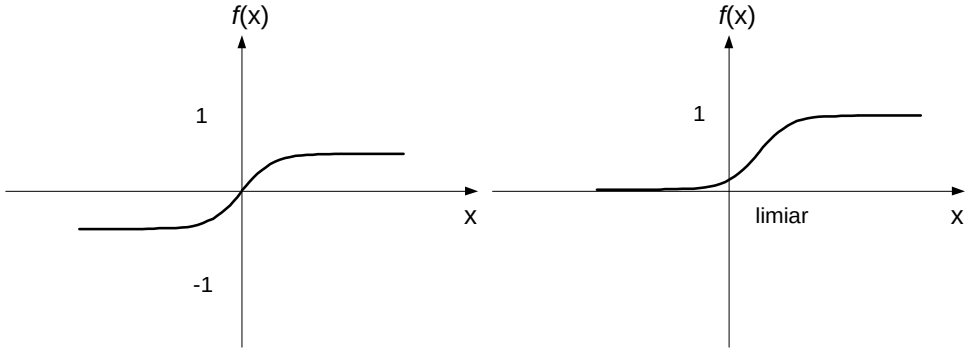

Figura 3.14. Função Sigmóide. A versão bipolar [-0.5, 0.5] encontra-se à esquerda.

Função Semi-Linear

Esta função pode ser interpretada segundo duas vertentes, na primeira, é uma função linear crescente com limites superior e inferior, na segunda, uma função lógica com uma fase de transição linear. Existem dois limiares, um superior e outro inferior. Para um valor abaixo do limiar inferior a saída é zero (ou -1), acima do limiar superior a saída é 1. Entre os dois limiares a saída é função linear da entrada (entende-se por entrada o valor resultante da primeira parte do processamento do neurónio). Assim como na função lógica não existe derivada nos pontos de transição. Esta característica pode limitar a sua aplicação.

Função Sigmóide

A utilização deste tipo de função de activação, diferenciável em todo o domínio, foi um grande avanço no domínio das RNAs. Com o aparecimento do algoritmo de retro-propagação (secção 3.7) que se baseia no cálculo de derivadas das saídas em ordem às entradas, que são depois propagadas para as camadas anteriores, foi necessário encontrar uma função de activação que fosse sempre diferenciável em todo o domínio de utilização. Paul Werbos, que apresentou o algoritmo de retro-propagação para o treino de RNAs, incluiu também a fórmula para a função de activação sigmóide (figura 3.14) cuja equação é:

$$y = \frac{1}{1 + \exp(-\lambda x)} \tag{3.3}$$

em que λ é uma constante, x representa a entrada e y a saída.

O grande sucesso atingido pelo algoritmo de retro-propagação tornou esta função sigmóide na mais utilizada nas RNAs. No entanto, existe a função tangente hiperbólica que apresenta um comportamento semelhante e que apresenta valores à saída entre –1 e 1.

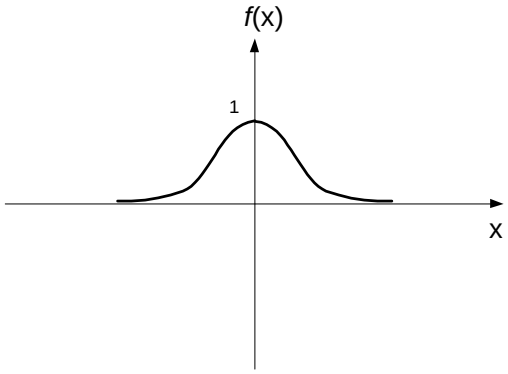

Figura 3.15. Função Gaussiana.

Função Gaussiana

Ao contrário das funções anteriores, esta gera o valor máximo para um valor de entrada perto do seu centro (figura 3.15) e apresenta um valor perto de zero para valores afastados do centro. O centro e a largura da curva são variáveis de modo a que a rede consiga diferentes tipos de comportamento. A equação genérica que traduz a função é a seguinte:

$$Yj = \exp(\beta) \tag{3.4}$$

$$\beta = -\frac{1}{2 \cdot L_j^2} \sum_{i=1}^{n} \left(X_{ij} - C_i^j \right)^2 \tag{3.5}$$

em que *Yj* representa a activação do neurónio *j*; X_{ij} denota a entrada no neurónio *j* proveniente do neurónio *i*; *Lj* representa a largura da função e C_i^j representa o centro da função do neurónio *j* para a entrada *i*.

Este tipo de função pode ser utilizada para melhorar o desempenho em tarefas de classificação, uma vez que fornece uma interpolação mais suave entre regiões de classificação, o que por vezes não acontece com a função sigmóide.

3.5 Treino da RNA

O treino (ou aprendizagem) da RNA consiste no processo de mudança dos pesos das ligações entre os neurónios e dos parâmetros associados aos neurónios (se os houver), de modo a atingir determinado comportamento. Os métodos de treino de RNAs podem ser classificados segundo três categorias: treino supervisionado, treino supervisionado com reforço e treino não supervisionado. Seguidamente serão abordadas as características dos métodos em cada categoria.

3.5.1 Treino Supervisionado

É um processo que consiste em fornecer à RNA exemplos de padrões que depois do treino deverá reconhecer. A forma como se desenvolve este tipo de treino baseia-se no fornecimento de um conjunto de pares de vectores (padrões). O primeiro vector de cada par é um exemplo e o segundo vector (vector objectivo) é o vector que a RNA deverá produzir, na sua saída, quando à entrada estiver o primeiro vector. Quando um vector é apresentado na entrada da RNA, depois de efectuado o processamento dentro da rede, o resultado da saída é comparado com o vector de saída desejado. De seguida os pesos são alterados no sentido de reduzir o erro nos vectores de saída. Este tipo de aprendizagem requer o conhecimento dos conjuntos de padrões de entrada e dos respectivos conjuntos de padrões de saída (conhecidos por conjuntos de treino ou padrões de treino), de modo a 'ensinar' correctamente a RNA. Depois da conclusão do treino, a RNA é utilizada com os pesos fixos, ou seja, com os valores obtidos na última iteração da fase de treino.

O método de treino supervisionado mais conhecido é o algoritmo de retro-propagação (secção 3.7).

3.5.2 Treino Supervisionado com Reforço

Neste tipo de treino não é fornecido o valor desejado para a(s) saída(s). Os vectores de entrada são apresentados à entrada da RNA, é feito o processamento dentro da rede, e no final existe apenas uma medida da qualidade da resposta obtida. Esta medida serve para conduzir a RNA no sentido do comportamento pretendido, denomina-se por Reforço e é redireccionada para trás (camadas anteriores da RNA) de modo a premiar os comportamentos correctos ou a penalizar os comportamentos incorrectos.

A fase de treino da RNA processa-se da seguinte forma: primeiro, a RNA efectua o processamento com as entradas e pesos actuais, a seguir, a saída é avaliada e o sinal de Reforço é gerado e injectado na RNA. Depois, os pesos são corrigidos de acordo com o sinal de Reforço, aumentando o valor dos pesos que contribuem para um bom desempenho e diminuindo os que contribuem para um mau desempenho. A RNA procura assim um conjunto de pesos que tende a reduzir sinais negativos de Reforço no futuro.

Este processo de treino é muitas vezes denominado por treino com crítica. Normalmente, é um treino mais difícil e também demora mais tempo. Apesar destas desvantagens, por vezes é a única forma de treinar a rede, uma vez que nem sempre é possível saber qual a saída óptima para determinado conjunto de entradas. Por exemplo, considere-se o caso em que se pretende utilizar uma RNA para controlar a temperatura dum processo químico, de modo a não ultrapassar determinado limite. Neste caso, pode-se desconhecer o estado dos componentes do processo num determinado momento. Mas se a temperatura ultrapassar o limite sabe-se que a RNA não teve um comportamento correcto, ou seja, pode-se fornecer informação crítica mas não "ensinar", neste caso pode-se recorrer a um algoritmo baseado no sinal de Reforço.

3.5.3 Treino Não Supervisionado

O objectivo deste tipo de treino é separar os dados de entrada em classes (ou grupos) com determinado significado. Neste tipo de treino existem apenas vectores de entrada, não existem vectores indicadores da saída desejada, também não existe um valor que traduza uma medida

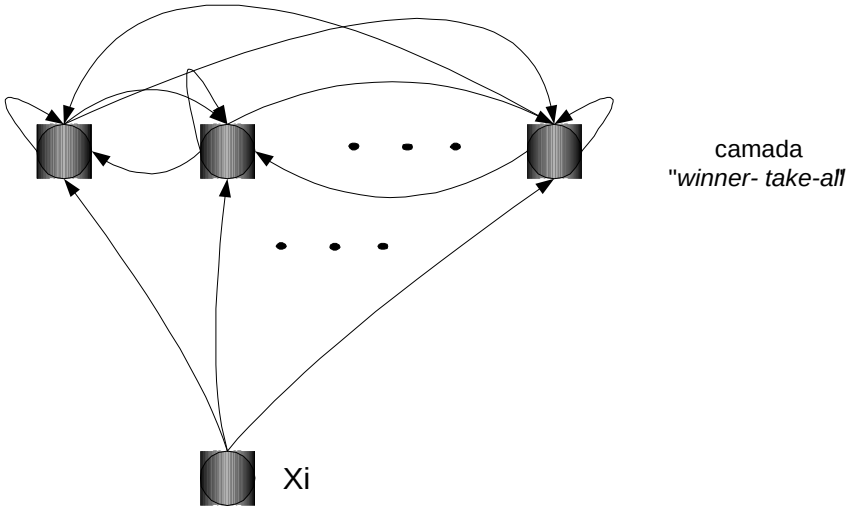

Figura 3.16. RNA muito simples para exemplificar a arquitectura da RNA (ou mapa) do tipo Kohonen. Por simplicidade apenas está representada uma entrada (Xi). Se existissem outras entradas seriam ligadas à camada superior da mesma forma. A camada superior pode ser uma matriz bidimensional de neurónios todos interligados formando assim um mapa de Kohonen.

da qualidade da saída da rede. Baseia-se apenas num critério interno de modo a conduzir a aprendizagem da RNA.

Este tipo de treino é muito utilizado nas RNAs de Kohonen. Estas RNAs têm duas camadas (figura 3.16) (onde cada neurónio da primeira camada, a de entrada, está ligado a cada neurónio da segunda camada, a de saída) sendo a segunda camada do tipo *concorrente*. Normalmente, estas redes são utilizadas para reconhecer grupos com características idênticas no conjunto de padrões de entrada, de uma forma não supervisionada.

De uma forma muito sucinta, o algoritmo funciona do seguinte modo: apenas existem vectores de entrada que são apresentados na camada de entrada. De seguida calcula-se o valor da distância (euclidiana) para cada neurónio da segunda camada. O neurónio com a menor distância euclidiana é denominado por neurónio "*ganhador*". O passo seguinte consiste em identificar uma vizinhança à volta desse neurónio, que é o conjunto constituído pelos neurónios mais próximos do neurónio ganhador. Os pesos correspondentes às ligações dos neurónios envolvidos neste grupo são os que terão mais alterações.

Neste tipo de treino pode-se ainda fazer referência a algoritmos com comportamentos dinâmicos para a escolha das classes [Pandya-Macy 96].

Foram apresentados, de uma forma muito sucinta, os três paradigmas de treino de RNAs. Dentro de cada um, existem muitas variantes e formas de fazer a actualização dos pesos das ligações de modo a conseguir treinar correctamente a RNA. No trabalho desenvolvido, foi utilizada uma RNA com propagação para a frente utilizando o treino supervisionado, mais concretamente, o algoritmo de retro-propagação que será descrito na secção seguinte.

3.6 Algoritmo de Retro-Propagação

O aparecimento deste algoritmo para treinar uma RNA com várias camadas veio potenciar e desenvolver a aplicação das RNAs para a resolução de diversos problemas em diferentes áreas científicas. Este método de treino, também conhecido por regra delta generalizada, é baseado no gradiente descendente para minimizar o erro médio quadrático total à saída da rede. O gradiente de uma função (neste caso a função é o erro e as variáveis são pesos da rede) fornece a direcção na qual a função aumenta mais rapidamente, por outro lado, a negação do gradiente indica a direcção na qual a função diminui mais rapidamente. O objectivo deste algoritmo é treinar a rede de modo a conseguir um equilíbrio entre a capacidade de responder correctamente aos padrões de entrada, que são utilizados durante o treino, e a capacidade de fornecer boas respostas a padrões de entrada semelhantes mas não idênticos, ou seja, com um erro muito baixo.

O treino da RNA com o algoritmo de retro-propagação envolve três passos [Fausett 94], [Rumelhart et al. 86]: o processamento da RNA face ao padrão de treino na entrada da rede; o cálculo do erro associado e a sua propagação para as camadas anteriores; e o ajuste dos pesos. Depois do treino, a aplicação da RNA consiste apenas na execução do primeiro passo do treino, ou seja, o processamento da informação de entrada e a utilização dos resultados produzidos pela RNA. Existem algumas variantes deste algoritmo, mas na secção 3.7.3 será apresentado o algoritmo standard aplicado a uma RNA com propagação para a frente, apenas com uma camada oculta, isto é, com as características idênticas à RNA utilizada no trabalho de

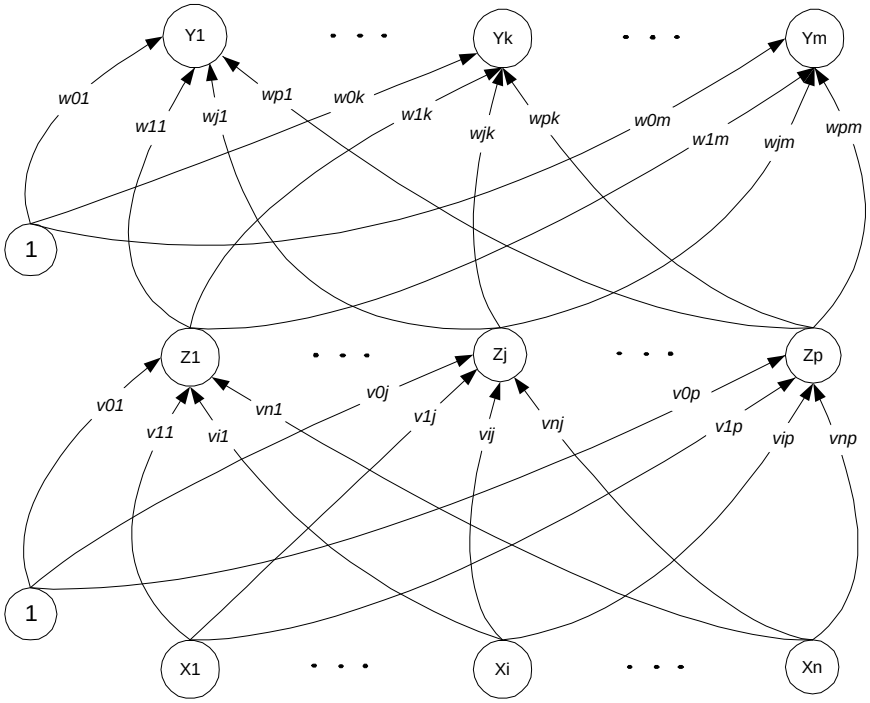

Figura 3.17. RNA com propagação para a frente com a nomenclatura utilizada no algoritmo de retro-propagação.

construção de mapas, mais concretamente no cálculo da probabilidade de uma célula específica estar ocupada.

3.6.1 Arquitectura

A figura 3.17 representa a arquitectura de uma RNA com propagação para a frente com a nomenclatura que servirá de base à apresentação do algoritmo. Neste caso, os neurónios da camada oculta e da camada de saída têm o parâmetro *BIAS* associado. Apenas está representado o fluxo de informação com propagação para a frente. Durante o treino, na fase de retro-propagação a informação é enviada no sentido contrário.

3.6.2 Função de Activação

A função de activação a utilizar no algoritmo de retro-propagação deve ter as seguintes características: ser contínua, diferenciável e monotonamente crescente. Além disso, para uma boa eficiência computacional deverá ser fácil de implementar. Normalmente, espera-se que a função sature, isto é, que se aproxime assimptoticamente de um valor máximo e de um valor mínimo finitos. Uma função que satisfaz todos estes requisitos é a função sigmóide. Esta foi a função seleccionada para ser a função de activação na RNA utilizada no trabalho experimental. A equação matemática que traduz a função sigmóide para $\lambda=1$ (secção 3.5.1) é:

$$f(x) = \frac{1}{1 + \exp(-x)} \tag{3.6}$$

cuja derivada é:

$$f'(x) = f(x)[1 - f(x)] \tag{3.7}$$

3.6.3 Algoritmo

Como já foi referido, o treino efectuado com este algoritmo envolve três passos: no primeiro (a parte de propagação para a frente), cada entrada (X_i) recebe a informação de entrada difundindo-a depois para cada neurónio da camada oculta $Z_1....Z_p$. De seguida, cada unidade oculta processa a sua activação (z_i) e envia-a para cada neurónio da camada de saída. Cada neurónio da camada de saída (Y_k) processa a sua activação (y_k) de modo a construir a resposta da RNA para o respectivo padrão de entrada.

Durante o treino, cada saída (y_k) é comparada com o valor objectivo (t_k) para determinar o erro associado ao padrão de treino em causa. Este erro é depois utilizado para calcular o factor (δ_k) (k=1,....,m) que será usado para difundir o erro para a camada anterior (os neurónios da camada

oculta que estão ligados aos neurónios de saída). É ainda utilizado (mais tarde) para actualizar os pesos das ligações entre a camada de saída e a camada oculta. De forma semelhante, calcula-se o factor (δ_j) (j=1,....,p) para cada neurónio oculto (Z_j), mas neste caso não é necessário propagar o erro para a camada de entrada, no entanto, é utilizado para actualizar os pesos entre a camada oculta e a camada de entrada. Depois de determinar todos os factores δ, todos os pesos na RNA são ajustados simultaneamente. O ajuste do peso w_{jk} (do neurónio oculto Z_j para a saída Y_k), baseia-se no factor δ_k e na activação z_j do neurónio oculto Z_j. De forma semelhante, o ajuste do peso v_{ij} (da entrada X_i para o neurónio oculto Z_j), baseia-se no factor δ_j e na activação x_i do neurónio de entrada X_i.

Nomenclatura

A nomenclatura utilizada no algoritmo de treino com retro-propagação é a seguinte [Fausett 94]:

x Vector de entrada a utilizar durante o treino: $\mathbf{x} = (x_1, \ldots, x_i, \ldots, x_n)$.

t Vector de saída a utilizar durante o treino: $\mathbf{t} = (t_1, \ldots, t_i, \ldots, t_n)$.

δ_k Termo que conterá a informação do erro, obtido a partir do erro no neurónio de saída Y_k, para o posterior ajuste do peso w_{jk}; é ainda utilizado para propagar a informação do erro na saída Y_k para as camadas anteriores (ocultas).

δ_j Termo que conterá a informação do erro, para o ajuste correcto do peso v_{ij}, obtido a partir da retro-propagação da informação do erro da camada de saída para o neurónio oculto Zj.

α Taxa de aprendizagem.

X_i Neurónio de entrada i:
Para um neurónio de entrada, o sinal de entrada e de saída é o mesmo: x_i.

v_{0j} *BIAS* para o neurónio oculto j.

Z_j Neurónio oculto j:
A entrada da rede em Z_j é:

$$z\_in_j = v_{0j} + \sum_i x_i \cdot v_{ij}. \qquad (3.8)$$

O sinal de saída (activação) de Z_j é:

$$z_j = f(z\_in_j). \tag{3.9}$$

w_{0k} *BIAS* para o neurónio de saída k.

Y_k Neurónio de saída k:
A entrada da rede em Y_k é:

$$y\_in_k = w_{0k} + \sum_j z_j \cdot w_{jk}. \tag{3.10}$$

O sinal de saída (activação) de Y_k é:

$$y_k = f(y\_in_k). \tag{3.11}$$

Algoritmo de Treino

O algoritmo de treino utilizado para implementar o treino da RNA é o seguinte [Fausett 94]:

Passo 0. Atribuir os pesos iniciais (utilizando pequenos valores aleatórios)

Passo 1. Enquanto a condição de paragem for falsa, fazer os Passos de 2 a 9.

Passo 2. Para cada padrão de treino, executar os Passos de 3 a 8.

PROPAGAÇÃO PARA A FRENTE

Passo 3. Cada neurónio da camada de entrada (X_i, i = 1,..., n)recebe o sinal de entrada x_i e envia este sinal para todos os neurónios da camada seguinte (neurónios ocultos).

Passo 4. Em cada neurónio oculto (Z_j,j = 1,..., p) é feita a *soma* dos seus sinais de entrada (contendo já o efeito dos pesos respectivos das ligações),

$$z\_in_j = v_{0j} + \sum_{i=1}^{n} x_i \cdot v_{ij},$$

depois é aplicada a função de activação para obter o sinal de saída,

$$z_j = f(z\_in_j),$$

e o sinal é enviado a todos os neurónios da camada seguinte (neurónios de saída).

Passo 5. Em cada neurónio de saída (Y_k, $k = 1,\ldots, m$) é feita a soma dos seus sinais de entrada (contendo já o efeito dos pesos respectivos das ligações),

$$y\_in_k = w_{0k} + \sum_{j=1}^{p} z_j \cdot w_{jk},$$

depois aplica-se a função de activação para obter o sinal de saída

$$y_k = f(y\_in_k),$$

sinal que é enviado a todos os neurónios da camada seguinte (neurónios de saída).

***RETRO-PROPAGAÇÃO DO ERRO*:**

Passo 6. Cada neurónio de saída (Y_k, $k = 1,\ldots, m$), recebe o padrão objectivo correspondente ao padrão de treino apresentado na entrada da RNA, e calcula-se o termo (δ_k, $k = 1,\ldots, m$) que ficará com a informação do erro,

$$\delta_k = (t_k - y_k) \cdot f\,'(y\_in_k),$$

calcula-se também a correcção a efectuar (mais tarde) nos pesos w_{jk},

$$\Delta w_{jk} = \alpha \cdot \delta_k \cdot z_j,$$

e ainda a correcção do *BIAS* para actualizar w_{0k},

$$\Delta w_{0k} = \alpha \cdot \delta_k,$$

os δ_k são enviados para os neurónios da camada anterior (oculta).

Passo 7. Em cada neurónio oculto (Z_j, j = 1,..., p), calcula-se a soma das entradas delta (provenientes da camada acima – a de saída),

$$\delta\_in_j = \sum_{k=1}^{m} \delta_k \cdot w_{jk},$$

multiplica-se depois pela derivada da função de activação para calcular o termo que conterá a informação do erro (δ_j, j = 1,..., p),

$$\delta_j = \delta\_in_j \cdot f\,'(z\_in_j),$$

calcula-se a correcção a efectuar (mais tarde) nos pesos v_{ij},

$$\Delta v_{ij} = \alpha \cdot \delta_j \cdot x_i,$$

e determina-se a correcção do *BIAS* para actualizar v_{0j},

$$\Delta v_{0j} = \alpha \cdot \delta_j.$$

Ajuste dos pesos e dos BIAS:

Passo 8. Em cada neurónio de saída (Y_k, k = 1,..., m), o ajuste dos seus *BIAS* e dos pesos (w_{jk}, j = 0, ..., p) é feito da mesma forma:

$$w_{jk}\ (\text{novo}) = w_{jk}\ (\text{anterior}) + \Delta w_{jk},$$

Em cada neurónio da camada oculta (Z_j, j = 1,..., p), o ajuste dos seus BIAS e pesos (v_{ij}, i = 0, ..., n) é feito da seguinte forma:

$$v_{ij}\ (\text{novo}) = v_{ij}\ (\text{anterior}) + \Delta v_{ij},$$

Passo 9. Teste da condição de paragem.

Um ciclo significa um percurso através de todo o conjunto de vectores com padrões de entrada. Normalmente, são necessários vários ciclos para efectuar o treino da RNA. O algoritmo

de treino, apresentado anteriormente, faz o ajuste dos pesos a seguir ao processamento de cada padrão de entrada. No entanto, o ajuste dos pesos pode ser efectuado por lotes, ou seja, o valor para o ajuste dos pesos em cada padrão de entrada, é acumulado durante um ciclo, ou durante um número determinado de padrões de entrada, antes de ser efectivamente aplicado. A seguir ao treino, a RNA é aplicada utilizando apenas a parte de propagação para a frente do algoritmo de treino.

3.6.4 Problemas da Retro-Propagação

Embora este algoritmo seja muito utilizado e tenha conseguido muito sucesso, existem alguns aspectos que limitam o uso generalizado do algoritmo. Os problemas na fase de treino podem surgir de duas situações: mínimos locais e "paralisia" da rede [Krose-Smagt 96]:

- Paralisia da Rede. Durante o treino, os pesos podem assumir valores muito elevados. Consequentemente, as entradas dos neurónios da camada oculta podem atingir valores elevados (positivos ou negativos), com o efeito da activação, da função sigmóide, a saída dos neurónios fica muito perto de 1 ou muito perto de zero. Se isto acontecer, como se pode concluir através das equações do algoritmo de treino, o ajuste dos pesos pode ser muito próximo de zero, e o processo de treino pode parecer que está parado, uma vez que não existem evoluções no erro.

- Mínimos locais. A superfície (espaço) do erro numa RNA complexa tem muitas zonas elevadas e muitas depressões. Com o gradiente descendente, a RNA pode ficar "presa" num mínimo local quando existe ainda um mínimo inferior. Podem-se utilizar métodos probabilísticos para evitar este problema mas tendem a aumentar o tempo de treino. Outra forma de contornar o problema é aumentar o número de neurónios da camada oculta. Com o aumento da dimensão do espaço de erros, a probabilidade de ficar "preso" diminui, no entanto, existe um limite para o número

de neurónios da camada oculta a partir do qual esta probabilidade aumenta novamente.

Capítulo 4

Robôs Móveis e Ambiente de Simulação

Neste capítulo descrevem-se os meios que permitiram o desenvolvimento do método para a construção de mapas locais. Os meios utilizados envolveram um robô móvel real e um ambiente de simulação. Foi no ambiente de simulação que se desenvolveu toda a aplicação de software que suporta o método de construção de mapas, utilizando um robô móvel simulado, diferente do robô real. O ambiente de simulação foi desenvolvido pela empresa *Nomadic Technologies, Inc.* fabricante dos robôs Nomad 200 e Super Scout II. No ambiente de simulação podem-se construir mapas, verificar e visualizar a posição do robô, obter leituras dos sensores e controlar os movimentos do robô com um joystick utilizando apenas software em ambiente linux. Além do ambiente de simulação, serão ainda referidas as características dos robôs envolvidos no desenvolvimento do método de construção de mapas locais. Enquanto que o robô Nomad 200 foi utilizado apenas na parte de simulação, o robô Super Scout II foi utilizado para testar o método em ambiente real. Embora este último disponha apenas de sensores sonares, as características de mobilidade e disposição dos sensores são idênticas.

No final do capítulo é feita uma breve referência ao modo de programação dos robôs.

4.1 Robô Móvel Nomad 200

Este robô é constituído por um sistema integrado com vários módulos sensoriais: infravermelhos, sonares, visão simples, visão com luz estruturada (LRF), detecção de contacto, bússola e odometria (figura 4.1). Este robô possui um computador de bordo para controlar os motores, os sensores e ainda para efectuar comunicações como cliente.

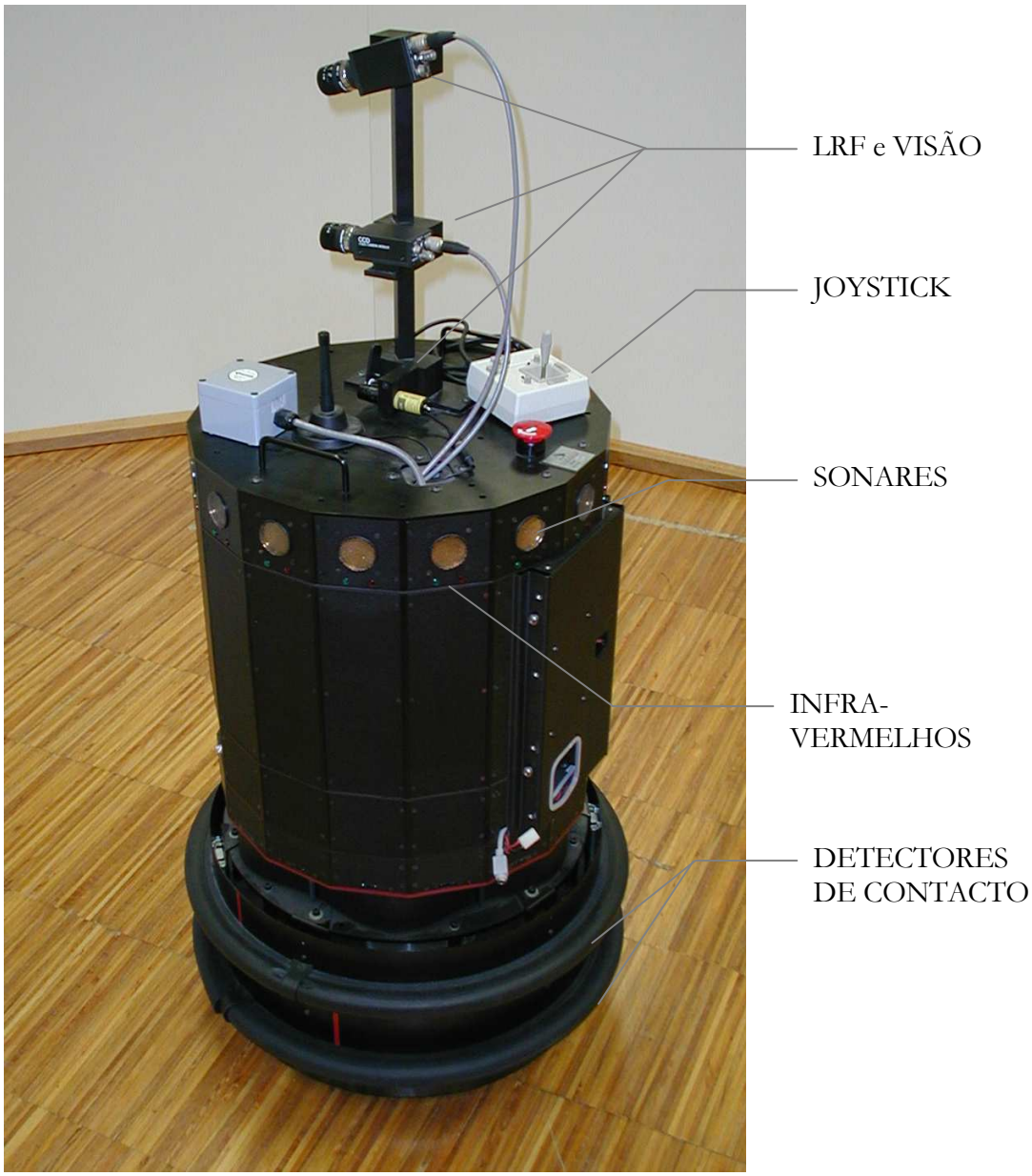

Figura 4.1. Robô móvel Nomad 200.

A base do robô está assente em três rodas. Pode efectuar movimentos de translação em qualquer direcção com a particularidade de poder rodar sobre o seu centro. A base deste robô é independente da torre (parte superior do robô onde estão as plataformas horizontais que contêm os diversos elementos do robô). Através de um motor é possível controlar a posição angular da torre. A velocidade máxima de translação do robô é de 0.61 metros/segundo e a velocidade máxima de rotação é de 60º por segundo. A estrutura do Nomad 200 é constituída por uma série de plataformas horizontais onde está todo o equipamento necessário para o funcionamento de todos os componentes do robô. Em relação às dimensões, o robô tem uma forma aproximadamente cilíndrica com raio de 22.86 centímetros (26.67 centímetros com o pára-choques), a altura pode variar entre 76.2 centímetros e 88.9 centímetros pesando cerca de 50 quilos.

O robô Nomad 200 utiliza um sistema multiprocessador com memória partilhada. Para controlar os sensores, o programador realiza previamente a configuração do sistema sensorial. Depois de configurar os sensores, a informação sensorial é escrita na memória do computador principal. Por conseguinte, o acesso à informação sensorial resume-se a efectuar acessos a zonas de memória específicas correspondentes a cada sensor.

4.1.1 Sistemas Sensoriais

Os sistemas sensoriais do robô Nomad 200 são constituídos por sonares, detectores de contacto, infravermelhos, visão simples, visão com luz estruturada, bússola e odometria. De seguida serão abordadas as características dos sistemas sensoriais utilizados no método.

Sistema de Detecção de Contacto (Sensus 100)

O sistema Sensus 100 é um sistema de detecção de contacto de 20 canais constituído por 20 sensores de pressão independentes (interruptores). O robô utiliza este sistema para detectar o contacto com objectos. Os 20 interruptores estão dispostos em dois anéis, cada um contendo 10 interruptores. A disposição dos sensores nos dois anéis foi feita de modo a fornecer uma

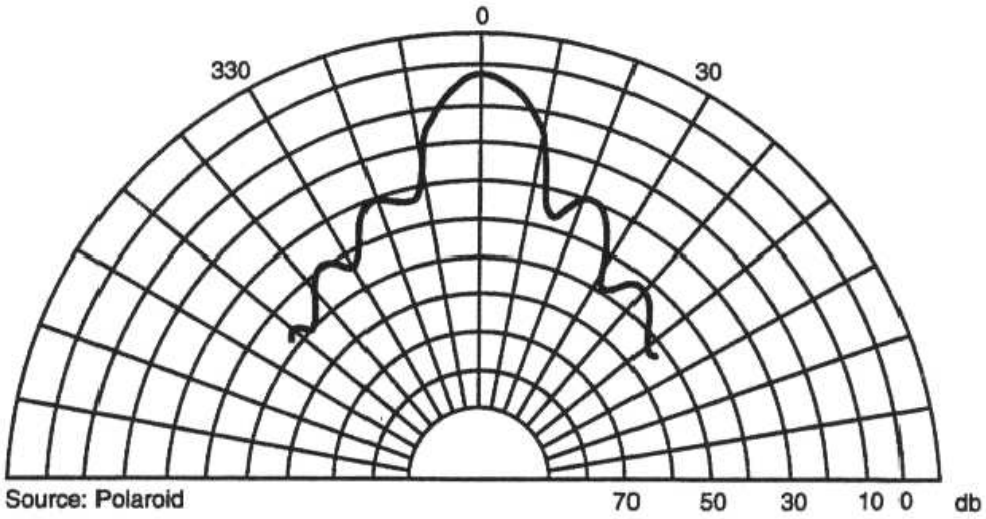

Figura 4.2. Padrão radial do transdutor em db.

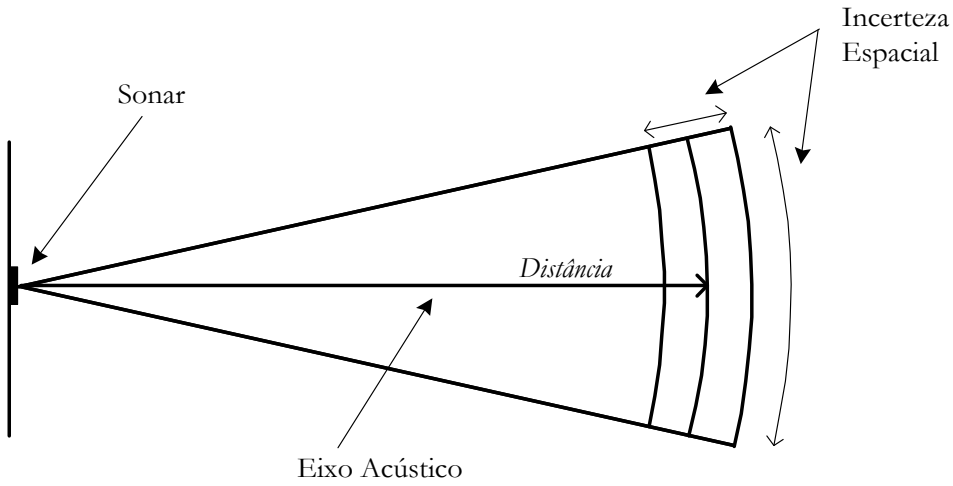

Figura 4.3. Modelo espacial do sonar.

cobertura de 360º com uma resolução de 18º. Cada sensor tem, aproximadamente, uma sensibilidade de 227 gramas. Este sistema sensorial é muito adequado para realizar tarefas que envolvam a detecção de objectos ou onde seja necessário empurrar objectos. Foi ainda desenhado para absorver a energia de impacto protegendo assim o sistema Nomad 200 em colisões acidentais.

Sistema de Medição de Distâncias através de Sonares (Sensus 200)

O sistema Sensus 200 é um sistema sonar com 16 canais. Este sistema pode fornecer distâncias desde 15 centímetros até 10.67 metros com uma precisão de 1% em toda a gama. Os sensores medem a distância através da medição do tempo de voo do sinal acústico desde que é emitido até ser recebido o eco. Os sensores utilizados no sistema são transdutores Polaroid com uma abertura do feixe de 25º (figura 4.2).

Os 16 sensores estão dispostos à volta do robô, separados por 22.5º, de modo a obter uma abrangência de 360º.

Como já foi referido, este tipo de sensor é utilizado para medir distâncias, no entanto, existe uma incerteza espacial associada que pode ser compreendida através do seu modelo espacial (figura 4.3). No modelo do sonar, as ondas acústicas são propagadas segundo um cone com uma abertura de 25 graus (este valor não é fixo). Na figura 4.3 este cone está projectado num plano 2D. Deste modo, não é possível saber com precisão a posição de um objecto detectado, uma vez que existem alguns locais, à mesma distância dentro do cone, susceptíveis de conter o objecto.

As distâncias medidas por este tipo de sensor apresentam um erro aproximado de 1% em toda a gama, mesmo no caso em que o eixo acústico seja normal às superfícies [Nomadic 4].

Estudos completos sobre os modelos experimentais dos sonares podem ser encontrados em [Lee 96] e em [Moita-Nunes 94].

Sistema de Medição de Distâncias por Infravermelhos (Sensus 300)

O sistema Sensus 300 é um sistema, com 16 canais, de medição de distâncias baseado na intensidade do sinal de infravermelhos refletido, podendo medir distâncias até 92 centímetros. Cada um dos 16 sensores, utilizados neste sistema, é constituído por dois Leds emissores e um fotodíodo detector que estão dispostos por forma a cobrir 360º à volta do robô. Os emissores

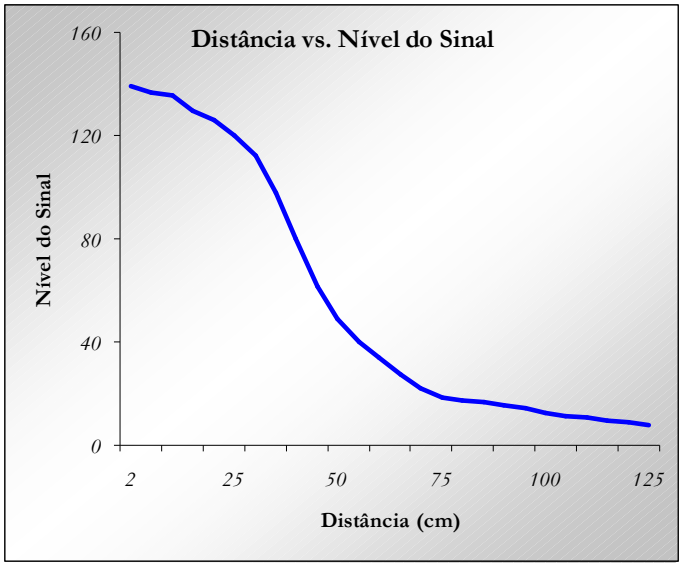

Figura 4.4. Distância vs. Nível de Sinal para o sistema Sensus 300.

emitem luz infravermelha e o receptor mede a quantidade de luz reflectida utilizando um detector de luz (fotodíodo). A distância é medida em função da intensidade de luz reflectida. O gráfico da figura 4.4 representa a relação entre a distância e o nível do sinal devolvido.

Quando estes sensores são utilizados em robôs reais é preciso ter atenção à luz ambiente. Se esta possuir muita energia na gama dos infravermelhos pode afectar as leituras destes sensores [Curran-Kyriakopoulos 93].

4.2 Robô Móvel Super Scout II

Este robô também é constituído por um sistema integrado como o robô anterior mas com menores capacidades (figura 4.5). Apenas dispõe dos seguintes sistemas sensoriais: detectores de contacto, sonares e odometria. O robô está assente em três rodas, duas rodas motrizes e uma pequena para manter o equilíbrio. A velocidade máxima do robô é de 1 metro/segundo. Este robô apresenta a mesma forma que o robô Nomad 200, aproximadamente cilíndrica e com raios semelhantes (21 centímetros e 23 centímetros com pára-choques). A altura é de 35 centímetros e o peso é de 25 quilos (com as baterias). Estes valores são muito inferiores aos do robô Nomad 200 devido às várias plataformas horizontais existentes neste último.

Figura 4.5. Robô Super Scout II.

O sistema de processamento e aquisição da informação sensorial é idêntico ao do robô Nomad 200 [Nomadic 4].

O Super Scout II foi o robô utilizado para experimentar o método de construção de mapas em condições reais.

4.2.1 Sistemas Sensoriais

Como já foi referido o robô Super Scout II possui três sistemas sensoriais. A plataforma dos sensores sonares é idêntica à do Nomad 200, tendo o mesmo número de sensores sonares, a mesma configuração e disposição. Como a plataforma dos sonares está mais perto do solo a gama de medidas é menor, variando entre 15 centímetros e 6.5 metros.

A detecção de contacto é feita de uma forma diferente da do Nomad 200. O Super Scout II possui apenas um anel que contém 6 interruptores, em contraste com os 20 interruptores

dispostos em dois anéis que equipam o Nomad 200. Apesar de serem em menor número não significa que sejam menos eficientes neste robô, uma vez que cada interruptor pode ser actuado por uma zona de contacto maior. Embora não tenha uma resolução tão boa como a do Nomad 200 a detecção de contacto é bastante eficiente. Como o Super Scout II é mais pequeno e tem um centro de gravidade mais perto do solo, a configuração deste sistema é bastante adequada à protecção do robô contra colisões acidentais. Pode-se salientar ainda a adequada protecção dos sensores de contacto que é feita através de uma camada de borracha.

4.2.2 Comando do Robô utilizando um Joystick

O robô dispõe de um joystick de dois eixos (figura 4.6). O manípulo do joystick deve ser utilizado conjuntamente com os dois botões de movimento presentes no canto superior esquerdo (figura 4.6). O robô Super Scout II e o robô Nomad 200 têm o mesmo joystick. Para mover o robô é necessário, em primeiro lugar, pressionar um dos botões de controlo de movimento, só depois se pode mover o manípulo do joystick.

A diferença de funcionamento do joystick nos dois robôs é que no caso do Super Scout II os dois botões de controlo de movimento têm a mesma função, o que não acontece no Nomad 200. Nos movimentos de translação o comportamento do robô com ambos os botões é idêntico. A diferença reside nas rotações, se apenas o botão da esquerda for pressionado a rotação terá efeito apenas na base do robô onde estão as rodas, ou seja a torre não roda. Se apenas o botão de cima estiver pressionado, as rotações terão apenas efeito na torre, mantendo a base fixa. No entanto, pressionando ambos os botões a base e a torre terão rotações solidárias com a mesma velocidade. Os movimentos compostos do manípulo permitem combinar movimentos de translação e de rotação.

O joystick foi referido nesta secção porque foi o robô Super Scout II que foi utilizado para experimentar o método de construção de mapas em condições reais. Durante as simulações foi utilizado o joystick disponibilizado pelo ambiente de simulação que será descrito na secção seguinte.

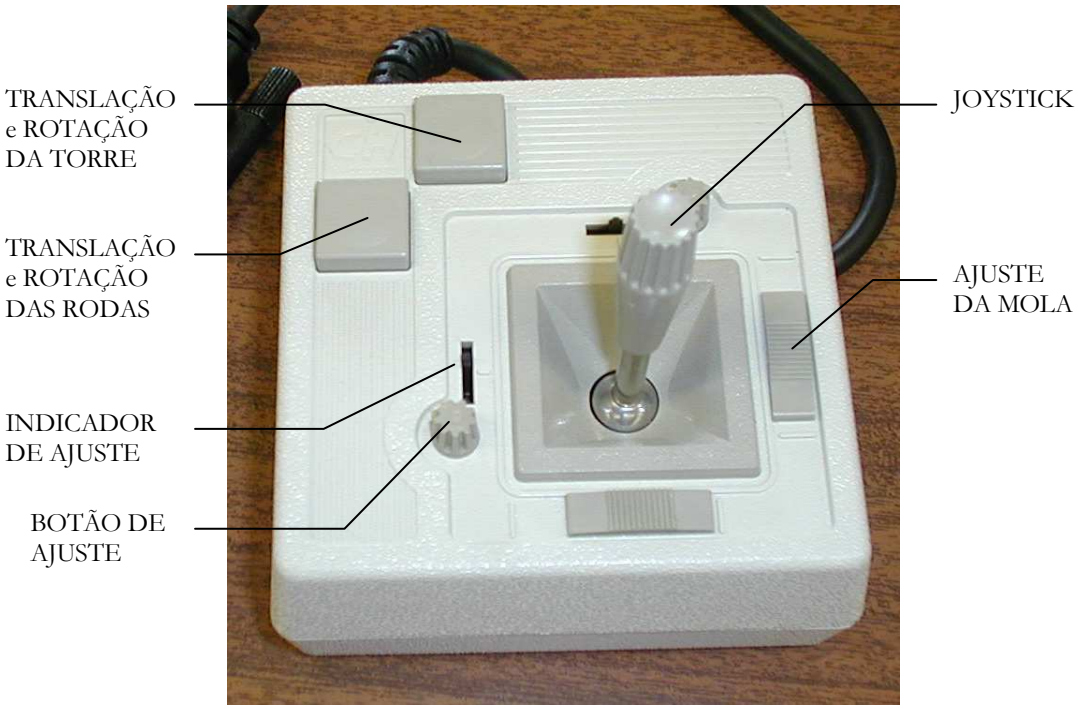

Figura 4.6. O joystick.

4.3 Ambiente de Desenvolvimento

O ambiente de desenvolvimento que inclui um interface gráfico (GUI), permite o acesso a robôs reais ou a robôs simulados e permite representar o mundo, em simulação, com medidas reais (figura 4.7). Através do GUI, o utilizador pode enviar comandos para os robôs, monitorizar a execução dos comandos visualizando os movimentos do robô no ecrã e visualizar os dados, instantâneos e acumulados, provenientes dos sensores. O utilizador pode ainda criar e modificar o ambiente de simulação e utilizá-lo para testar os programas de controlo dos robôs.

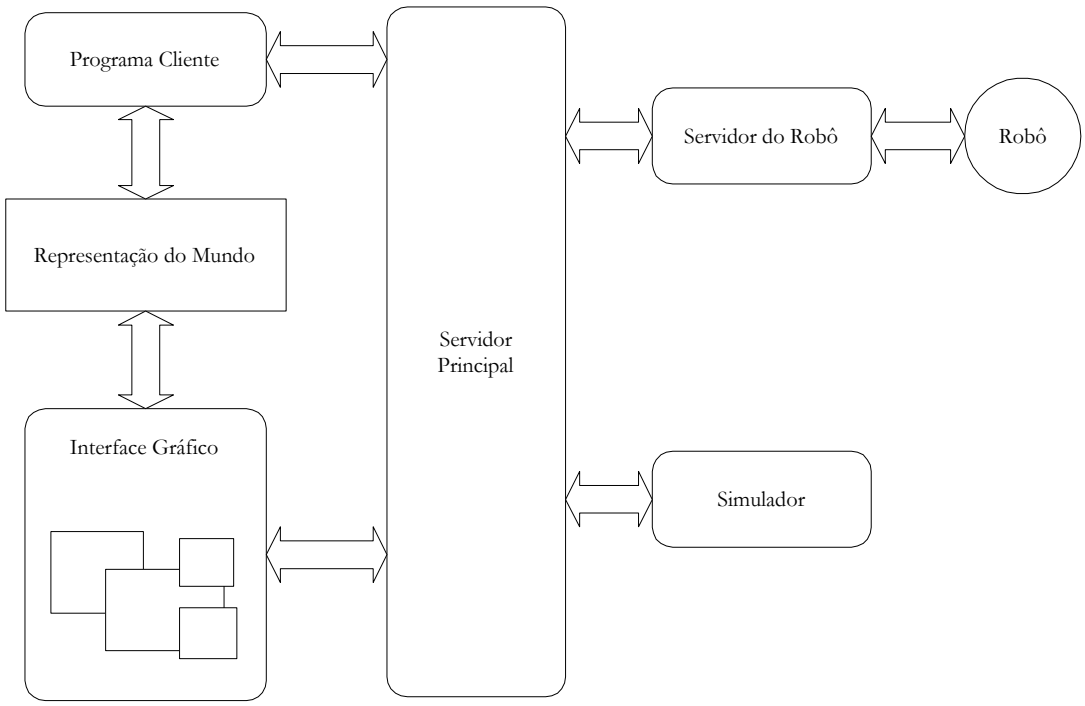

Figura 4.7. Ambiente de desenvolvimento.

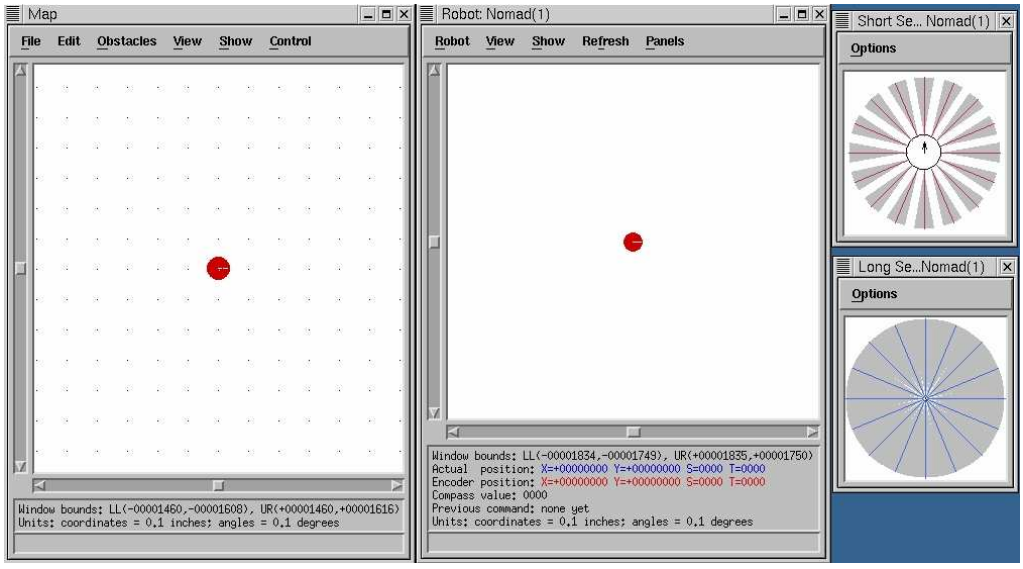

Figura 4.8. Janelas do ambiente gráfico (GUI). À esquerda está a janela do mapa, ao centro a janela do robô e à direita (em cima) a janela dos sensores sonares e (em baixo) a janela dos sensores de infravermelhos.

4.3.1 Interface Gráfico

Para ter acesso ao GUI é necessário executar o programa *Nserver*, em ambiente linux, com a biblioteca *Xwindows* instalada. Ao executar o programa, o ecrã fica com um conjunto de janelas com o aspecto da figura 4.8. Este ambiente gráfico é constituído por quatro janelas: a janela do mapa (Map); a janela do robô (robot); a janela dos sensores sonares (Long Sensors) e a janela dos sensores de infravermelhos (Short Sensors).

Janela do Mapa

É nesta janela que se constroem os mapas e onde se pode visualizar todo o mapa e todos os robôs, nas suas posições relativas, no mundo simulado. Os limites do ambiente onde se podem construir mapas podem ser alterados e os valores actuais estão presentes no rodapé da janela. Os menus desta janela permitem criar, ler ou gravar mapas como ficheiros, mover, copiar e eliminar obstáculos no mapa. No menu *obstacles* pode-se adicionar objectos de forma rectangular ou polígonos com forma variável, alterar e eliminar os objectos já existentes. É possível mudar o zoom sobre o mapa. Desta forma, pode ver-se o mapa através de vários níveis de pormenor. Esta janela permite ainda adicionar robôs ao ambiente de simulação e aumentar ou diminuir a velocidade de simulação.

Janela do Robô

É esta janela que contém a informação correspondente a cada robô individualmente. A janela mostra a parte do mapa onde se encontra o robô em causa (durante o desenvolvimento do método de construção de mapas foi utilizado apenas um robô), a sua posição e orientação e os comandos que estão a ser executados no simulador, em cada momento. É possível gravar sequências de movimentos e as leituras dos sensores do robô. Pode-se também activar uma janela (figura 4.9) que contém um joystick (foi através deste joystick que se conduziu os movimentos do robô durante as simulações).

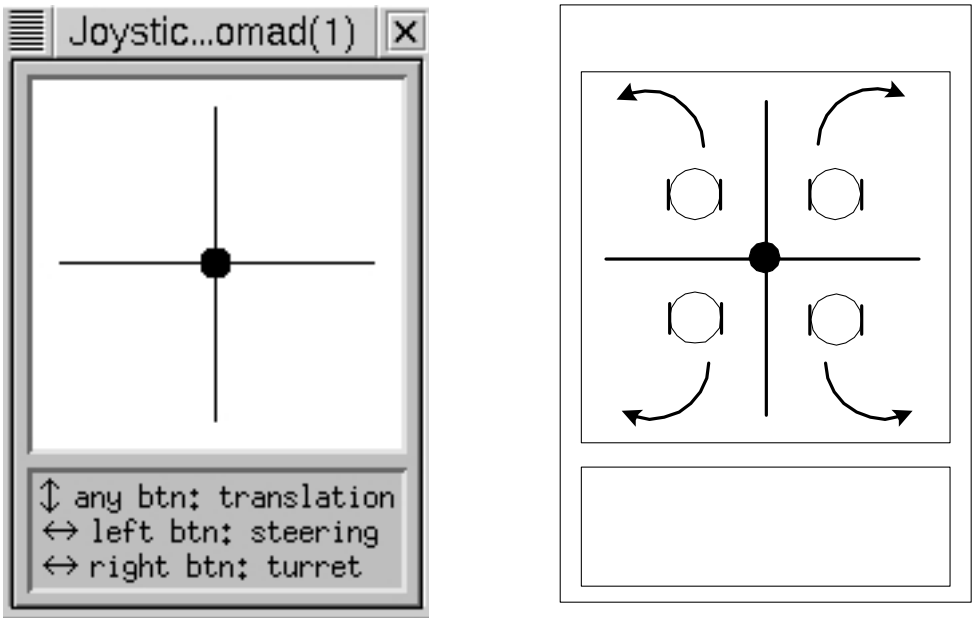

Figura 4.9. Joystick (por software) no ecrã do monitor. Á direita estão esquematizados os movimentos compostos do joystick

Janela dos Sonares (Long Sensors) e janela dos sensores de infravermelhos (Short Sensors)

Estas janelas permitem visualizar os feixes dos sensores. Se algum feixe está interrompido pode-se identificar o sensor e mediante o tamanho do feixe pode-se ficar com uma ideia da proximidade do obstáculo em relação a esse sensor.

4.3.2 O Simulador

No simulador é possível criar muitos ambientes diferentes, movimentar o robô e obter leituras dos sensores de uma forma muito próxima das condições reais. Isto permite ao programador do robô o desenvolvimento de programas num ambiente controlado. Uma das vantagens é poder

criar situações específicas para testes específicos. Por exemplo, a introdução de alguma dinâmica no ambiente (colocar ou retirar um objecto nas proximidades do robô).

No interface gráfico não existe diferença entre os modos simulado e real. A única diferença é que no modo real a informação dos sensores provém dos sensores que estão no robô real e os movimentos, não sendo consequência do programa do utilizador, são efectuados através de um joystick real accionado por um utilizador.

O ambiente de simulação é um mundo bidimensional, visto de cima, onde podem existir vários robôs e vários obstáculos. A informação proveniente dos sensores, durante a simulação, é dada pelos módulos simuladores correspondentes a cada tipo de sensor. Estes módulos fornecem a resposta dos sensores de acordo com o ambiente e com a posição do robô em cada momento.

Na janela do robô existem algumas informações no rodapé fornecidas pelo simulador, como por exemplo a posição do robô (AP- Actual Position) e a posição dada pelo modelo dos codificadores ópticos (EP - Encoder Position). Estas duas posições seriam idênticas se os movimentos do robô fossem perfeitos e nunca houvesse deslizamentos nas rodas. A diferença reside no facto da EP ser obtida a partir do modelo dos codificadores ópticos e a AP proceder do modelo dos codificadores ópticos afectado por um erro (este erro é determinado pelo simulador). A partir do instante inicial (quando ainda são idênticas), as duas posições começam a ser diferentes de acordo com o modelo do erro. Este erro vai aumentando enquanto o robô estiver em movimento. Durante as simulações deve-se ter em consideração o seguinte: quando se faz a aquisição da informação sensorial, esta é calculada a partir da posição actual do robô, enquanto que a informação odométrica é obtida directamente através dos codificadores ópticos. Consequentemente, se não houver atenção aos erros de odometria, pode-se estar a obter leituras de distância em locais diferentes daqueles que a odometria indica.

4.3.3 Programação do Robô

O robô pode ser controlado através de um programa de aplicação escrito em linguagem C. Este controlo pode ser exercido de duas formas (figura 4.10):

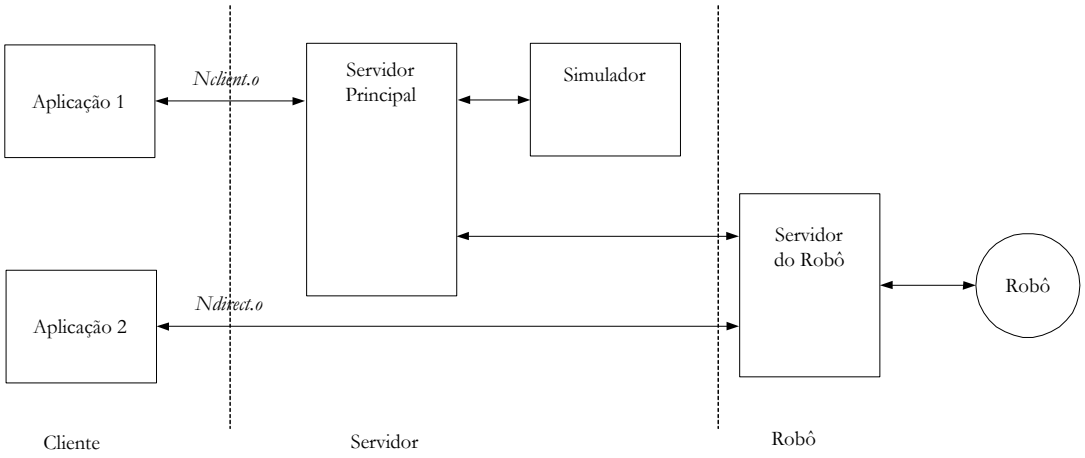

Figura 4.10. Programação em modo directo e modo cliente.

- Modo Cliente - A aplicação comunica com o servidor principal que por sua vez comunica com o servidor do robô ou com o simulador;
- Modo Directo - A aplicação comunica directamente com o servidor do robô.

No início do desenvolvimento da aplicação é aconselhável a utilização do modo cliente com o simulador por razões de segurança. Desta forma é mais fácil evitar acidentes devido a erros de programação e testar passo a passo toda a aplicação. Quando a aplicação já atingiu um patamar de estabilidade avançado então já se poderá executar no modo cliente com o robô real ou no modo directo, ou seja, sem passar pelo servidor. Comutar entre os dois modos é simples. Se o robô real for igual ao utilizado no simulador apenas será necessário mudar o ficheiro *Nclient.o* (utilizado para o modo cliente) para *Ndirect.o* (utilizado no modo directo) e recompilar a aplicação. Como neste caso o robô real é diferente do robô de simulação, além da operação anterior (a utilização do ficheiro *Ndirect.o* correspondente ao Super Scout II) é ainda necessário mudar os parâmetros da instrução *connect* (instrução que permite ligar o programa de aplicação ao robô real em causa).

Além da linguagem C pode ainda ser utilizada a linguagem C++ e ainda aplicações desenvolvidas em *LISP*.

Posição	Nome	Designação
1	STATE_IR_0	Informação do sensor infravermelhos #0
2	STATE_IR_1	Informação do sensor infravermelhos #1
3	STATE_IR_2	Informação do sensor infravermelhos #2
.....	.....	.....
16	STATE_IR_15	Informação do sensor infravermelhos #15
17	STATE_SONAR_0	Informação do sensor sonar #0
18	STATE_SONAR_1	Informação do sensor sonar #1
19	STATE_SONAR_2	Informação do sensor sonar #2
.....	.....	.....
32	STATE_SONAR_15	Informação do sensor sonar #15
33	STATE_BUMPER	Informação do detector de contacto
34	STATE_CONF_X	Informação da posição em x
35	STATE_CONF_Y	Informação da posição em y
36	STATE_CONF_STEER	Informação do ângulo da base do robô
37	STATE_CONF_TURRET	Informação do ângulo da torre do Nomad
38	STATE_VEL_TRANS	Informação da velocidade de translação
39	STATE_VEL_STEER	Informação da velocidade de rotação da base
40	STATE_VEL_TURRET	Informação da velocidade de translação da torre

Tabela 4.1. Vector de Estado com alguns índices e respectivas designações.

Como já foi referido, o computador principal obtém informações acerca dos sensores através de memória partilhada. Essas informações podem ser acedidas por um programa de aplicação através de um vector global denominado por vector de estado (*state vector*). Desta forma, para aceder a uma leitura de distância proveniente de um sensor basta apenas consultar o valor numa zona específica de memória. Para facilitar a identificação de cada sensor existem nomes específicos para cada zona de memória correspondente a cada sensor. Por exemplo, o nome *STATE_SONAR_12* corresponde ao índice do vector onde está a leitura de distância do sonar número 12 (figura 5.3). Podem-se ainda referir outros exemplos: o nome *STATE_BUMPER* indica se algum detector de contacto foi accionado, o nome *STATE_IR_2* permite consultar a leitura de distância proveniente do sensor de infravermelhos número 2, e o nome *STATE_CONF_STEER* permite conhecer a orientação do robô em cada instante. Na tabela 4.1 estão resumidas alguns índices do vector de estado acompanhadas pelas respectivas designações.

Capítulo 5

Arquitectura do Método de Construção de Mapas

Neste capítulo serão descritos todos os componentes da arquitectura do método de construção de mapas. Em cada componente serão abordadas todas as características inerentes ao seu funcionamento incluindo os respectivos algoritmos. Serão também referidos alguns pormenores relacionados com a implementação do método incluindo algumas soluções adoptadas nalgumas situações. Com o objectivo de adaptar este método a outros robôs móveis, será descrito o procedimento utilizado para aplicação do método num robô real, com algumas diferenças em relação àquele que foi utilizado durante as simulações.

O mapa local é constituído por uma grelha de células. O procedimento para a actualização de cada célula envolve a utilização de uma RNA e uma fórmula baseada no teorema de Bayes. A função da RNA consiste na interpretação da informação sensorial de modo a determinar a probabilidade de cada célula estar ocupada. A informação odométrica é utilizada para efectuar o deslocamento correcto do mapa, de acordo com os movimentos do robô.

5.1 Método de Construção do Mapa Local

A arquitectura do método de construção do mapa está representada na figura 5.1. Para a actualização das células, em primeiro lugar, o módulo *selector de sector* identifica o sector ao qual pertence a célula a actualizar (através do conhecimento da posição e da orientação da célula).

Em segundo lugar, o módulo *selector de sensores* identifica as duas informações de distância com as orientações mais próximas da orientação da célula em causa. Estas informações e as coordenadas polares da célula em causa (relativamente ao sistema de coordenadas do robô – figura 5.3) são fornecidas à RNA. A RNA tem quatro entradas e uma saída (figura 5.2). Esta saída, através de um treino adequado, traduz a probabilidade da célula em causa estar ocupada. O valor desta probabilidade é depois utilizado no método de actualização da célula baseada no teorema de Bayes. Este procedimento repete-se para todas as células do mapa. Nas secções seguintes este procedimento será desenvolvido com mais detalhe.

5.1.1 Rede Neuronal Artificial

A RNA utilizada no método de construção de mapas é uma RNA com propagação para a frente (figura 5.2). Para cada célula (x,y), a camada de entrada da RNA consiste em:

1. A observação o=(ss1, ss2) dos dois sensores orientados na direcção da célula (x,y);

2. O módulo R das coordenadas polares (R, θ), do centro da célula (x,y), em relação ao sistema de coordenadas localizado no centro do robô (figura 5.3).

3. O parâmetro desvio que é determinado através da orientação dos sensores (ss1, ss2) e do ângulo θ das coordenadas polares da célula (x,y). Este parâmetro será explicado na subsecção seguinte.

A camada de saída tem apenas um neurónio. Este neurónio fornece $P(C_{xy}|o)$, que mede a probabilidade da célula (x,y) estar ocupada dada a observação sensorial actual o=(ss1, ss2).

O treino da RNA foi efectuado "*off-line*" através do algoritmo de retro-propagação [Fausett 94] e [Rumelhart et al. 86]. Só depois do treino conduzir a um erro muito baixo (0.02) é que os parâmetros da RNA foram guardados para posterior utilização no método. Os exemplos de treino foram gerados no ambiente de simulação. Colocando o robô num ambiente conhecido, obteve-se um conjunto de exemplos de medidas dos sensores, em várias situações e posições, e

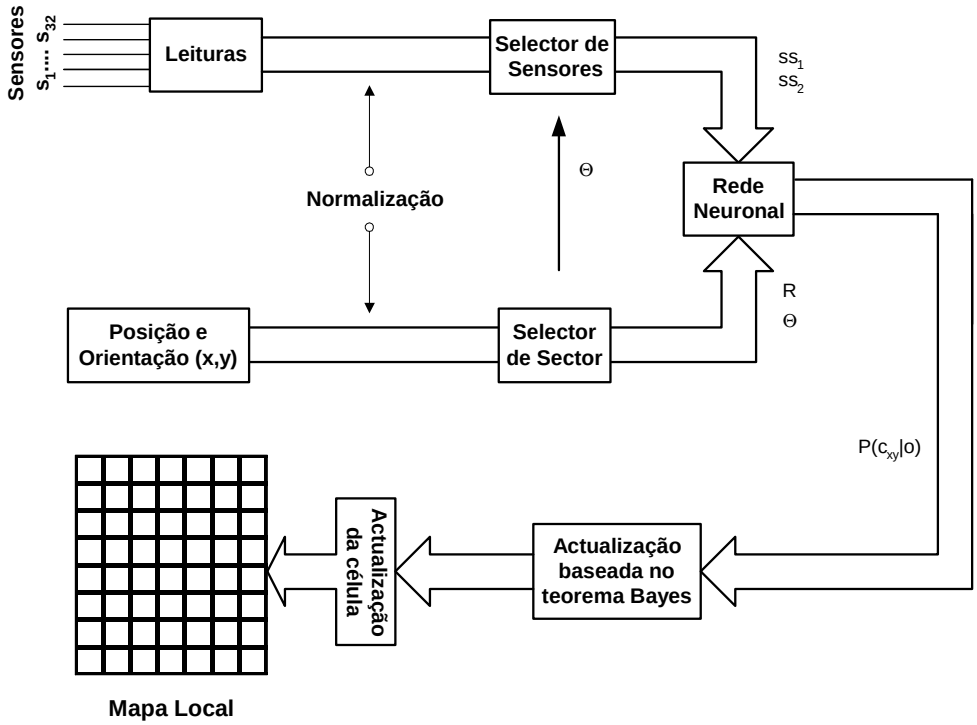

Figura 5.1. Arquitectura do método de construção de mapas locais. ss1 e ss2 (sensor seleccionado 1 e 2) são os dois sensores com direcções mais próximas da direcção da célula. (R, θ) representa as coordenadas polares da célula (x,y) a ser actualizada, em relação ao sistema de coordenadas do robô (ver figura 5.3). O número máximo de sensores utilizado foi de 32. No entanto, este número pode ser superior, uma vez que não existe um limite conceptual.

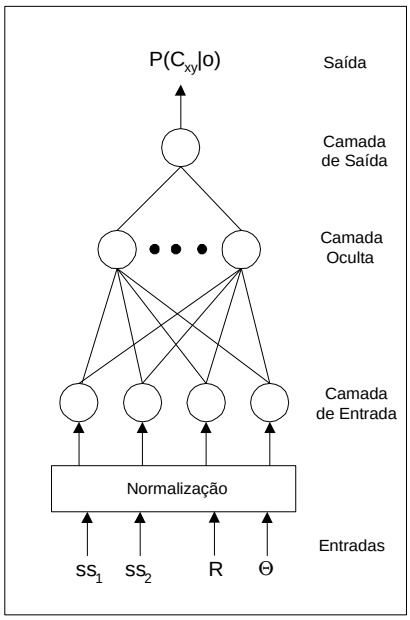

Figura 5.2. Rede Neuronal Artificial (RNA) com propagação para a frente.

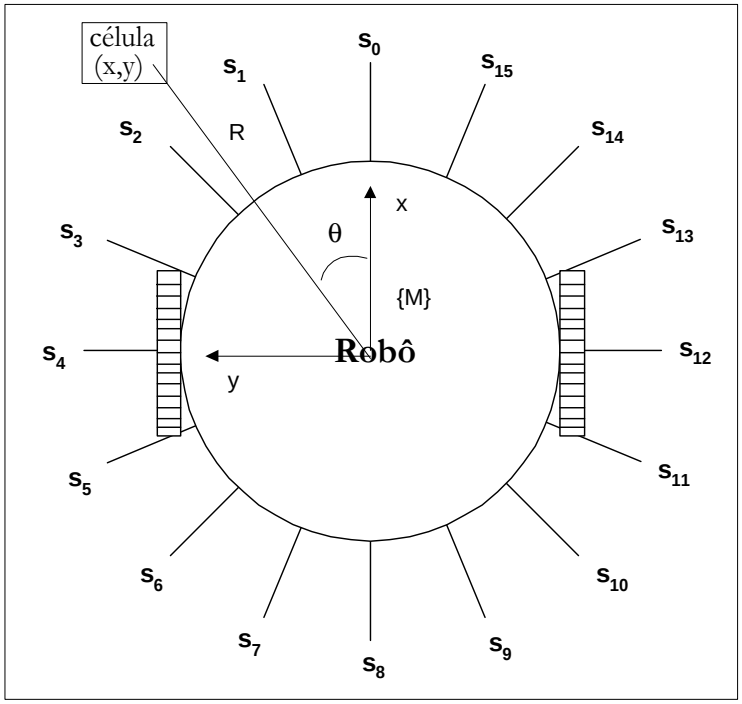

Figura 5.3. O robô e a orientação dos sensores. No robô Super Scout II cada S_i representa a orientação de um sonar. No robô Nomad 200 cada S_i representa a orientação de um par de sensores, um sonar e um sensor de infravermelhos. {M} representa o sistema de coordenadas do robô. (R, θ) são as coordenadas polares da célula (x,y) relativamente a {M}.

para a gama de medidas adequada, ou seja, tendo em consideração o tamanho do mapa de grelhas e a dimensão de cada célula. Depois do treino, a RNA fornece valores no intervalo [0, 1] que podem ser interpretados como probabilidades das células estarem ocupadas. Uma vez que o treino da RNA é baseado em exemplos, pode facilmente ser adaptado a novas situações. Outra vantagem reside na capacidade de interpretar duas leituras de sensores simultaneamente (podem ser mais). Normalmente, interpretando as leituras dos sensores no contexto dos seus vizinhos conduz a melhores resultados [Thrun 98].

Aspectos Relacionados com o Treino da RNA

A RNA tem uma camada oculta. Esta camada tem três neurónios porque foi esta a topologia que conduziu aos resultados esperados com uma boa estabilidade. Durante a fase de desenvolvimento e treino da RNA também foram testadas as topologias com quatro e dois neurónios na camada oculta. No primeiro caso, a RNA rapidamente deixou de convergir para um erro admissível, no segundo caso, a RNA apresentava um comportamento bastante instável embora tenha convergido para um erro admissível durante muitas sessões de treino.

A RNA foi treinada para medidas de distância até 3.5 metros, com um erro admissível de 0.02 (na saída), utilizando 84 exemplos de treino durante cerca de 24000 ciclos (um ciclo representa um percurso pelos 84 exemplos diferentes de treino). Na tabela 5.1 estão nove (dos 84) exemplos de treino. As três colunas correspondentes ao Sensor1 (ss1), Sensor 2 (ss2) e à distância do centro da célula ao centro do sistema de coordenadas do robô (R) têm os valores em metros. Os valores da coluna da Probabilidade têm apenas três valores: 0 (célula não ocupada), 0.5 (incerteza) e 1 (célula ocupada). A coluna do desvio pode apresentar os valores 0.4 e 0.5. O significado destes valores pode ser compreendido através da figura 5.4. Nesta figura, ss1 e ss2 representam leituras de distância dos sensores cujas orientações foram escolhidas para fornecer as distâncias para actualização das células A, B, e C. Para as células A e C, os segmentos de recta entre o centro das células e a origem de {M} formam os ângulos α e δ, com ss1 e ss2, respectivamente. Se esses ângulos forem superiores a 6 graus então o desvio será 0.4, caso contrário será de 0.5. Portanto, para o exemplo da figura 5.4, α e δ não são superiores a 6 graus logo o desvio será 0.5. Para o caso da célula B, cujo ângulo β é superior a 6 graus o desvio será 0.4. A determinação do desvio é feita no bloco de normalização (figura 5.2), imediatamente antes das entradas da RNA. Os três valores correspondentes às outras três entradas da RNA, no módulo de normalização, são apenas convertidos de centímetros para metros (dividindo pelo valor 100), uma vez que a unidade utilizada é o centímetro (excepto na RNA).

Podem-se colocar duas questões: porque não se utiliza o valor do ângulo das coordenadas polares da célula directamente ou afectado por um factor multiplicativo? Se não é utilizado o valor do ângulo, porque é que se utilizam os valores 0.5 e 0.4 para o desvio e não outros?

Em relação à primeira questão, apesar de terem sido feitas tentativas nesse sentido, não foi possível treinar a RNA nem com os valores do ângulo afectados de um factor multiplicativo,

Sensor 1	Sensor 2	R	Desvio	Probabilidade
1.00	1.00	0.95	0.5	1
1.00	1.00	0.50	0.5	0
1.00	1.00	1.05	0.5	0.5
1.70	1.10	1.05	0.4	1
1.70	1.10	0.50	0.4	0
1.70	1.10	1.90	0.4	0.5
2.60	1.80	0.80	0.4	0
2.60	1.80	1.75	0.4	1
2.60	1.80	2.20	0.4	0.5

Tabela 5.1. Exemplos de treino.

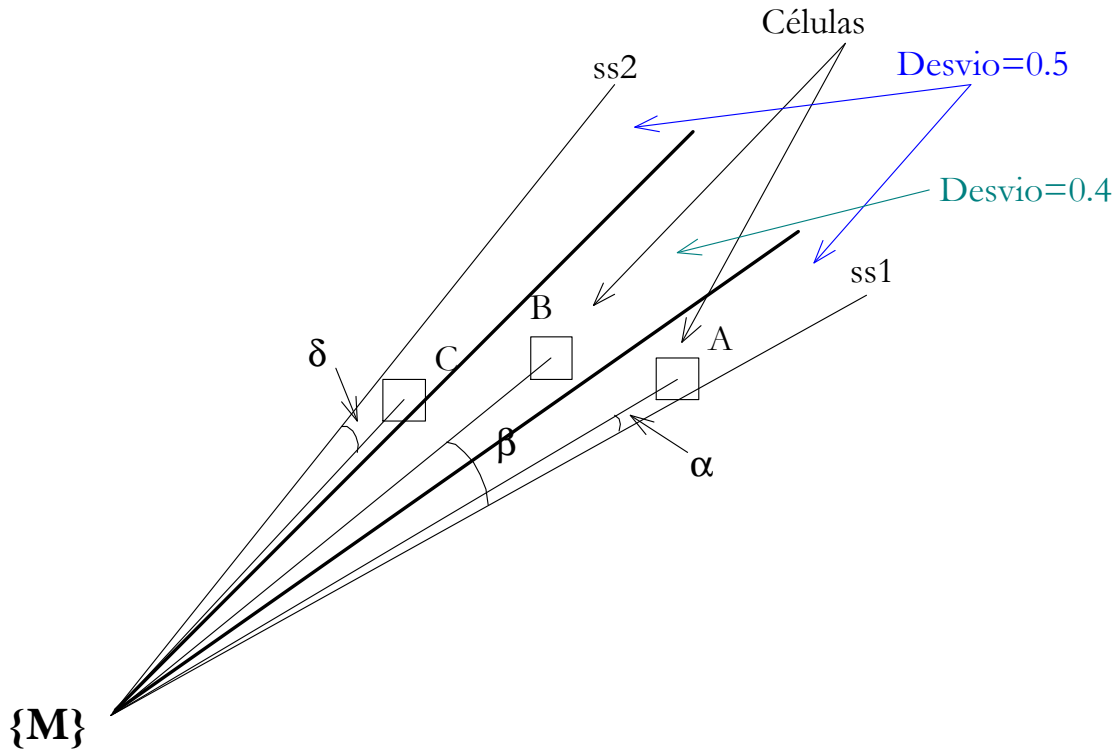

Figura 5.4. Significado do parâmetro Desvio nos exemplos de treino da RNA.

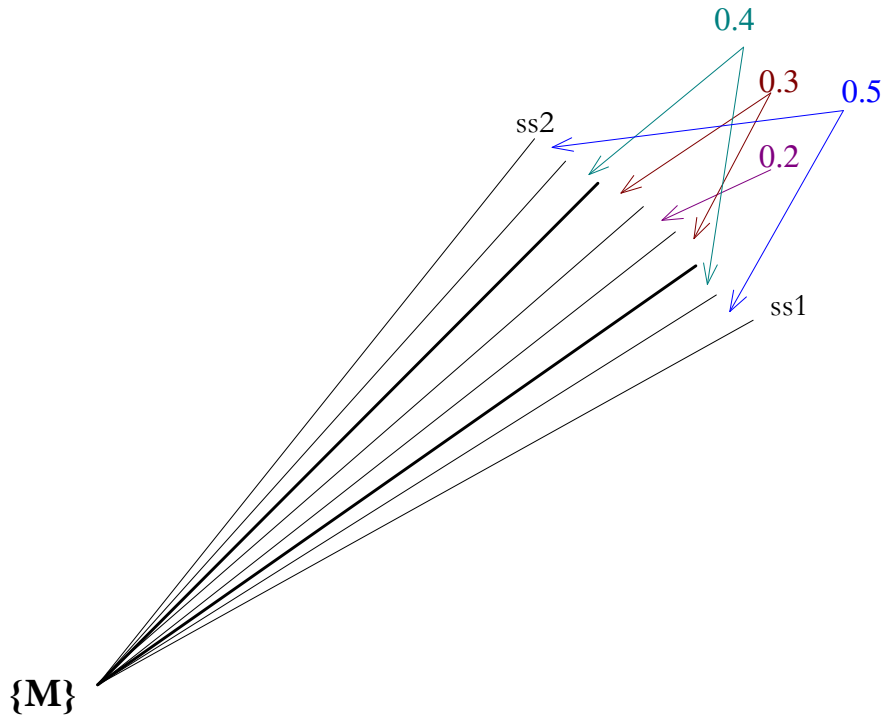

Figura 5.5. Outros Desvios utilizados no desenvolvimento do método.

nem convertendo o valor dos ângulos para o intervalo [0, 1]. De facto, esta RNA apresenta melhores comportamentos para valores normalizados. Não sendo possível conseguir resultados positivos abordou-se o problema segundo uma perspectiva diferente. O que está em causa nesta fase já não é o ângulo em relação ao sistema de coordenadas do robô, uma vez que a importância fundamental do ângulo é encontrar os sensores com orientações próximas da orientação da célula. Depois da escolha dos dois sensores, durante a actualização das células, o que está em causa é se o valor relativo da orientação da célula está ou não próximo da orientação dos sensores. Foi segundo esta abordagem que se abandonou a ideia de utilizar directamente o valor do ângulo da célula (x,y), em coordenadas polares, e se optou por traduzir a proximidade das orientações dos dois sensores através de um valor distinto. Respondendo agora à segunda questão e como já foi referido anteriormente, esta RNA apresenta um comportamento mais estável para valores normalizados. Com este pressuposto, efectuaram-se alguns testes

considerando o valor 0.5 para o caso da célula estar muito próxima de uma das orientações dos dois sensores e à medida que se afastava dessas orientações esse valor diminuía décima a décima (figura 5.5). No início foram definidas apenas duas áreas, depois estas áreas foram subdivididas em duas. No entanto, depois de efectuar vários testes chegou-se à conclusão que a utilização de quatro áreas não melhorava os resultados obtidos apenas com duas áreas. Por esta razão foram utilizadas apenas duas áreas (figura 5.4). Os valores utilizados (0.5 e 0.4) foram definidos de forma heurística. Apesar disso, nos primeiros treinos efectuados esta abordagem produziu resultados muito positivos. Como consequência, todos os exemplos de treino foram feitos segundo esta abordagem.

5.1.2 Actualização das Células baseada no Teorema de Bayes

O teorema de Bayes permite determinar a probabilidade de uma hipótese se verificar baseando-se na probabilidade anterior ou a probabilidade de observar determinada informação dada uma hipótese [Mitchel 97].

Teorema de Bayes

A fórmula que traduz o teorema de Bayes é a seguinte:

$$P(h \mid D) = \frac{P(D \mid h)P(h)}{P(D)} \tag{5.1}$$

em que $P(h)$ é a probabilidade inicial de se verificar a hipótese h (também denominada por probabilidade anterior); $P(D)$ é a probabilidade de observar D sem nenhum conhecimento acerca da hipótese; $P(D \mid h)$ é a probabilidade de observar D supondo que a hipótese h se verifica e $P(h \mid D)$ é a probabilidade de se verificar a hipótese h dada a observação D.

Actualização da Informação das Células

A construção de um mapa local consiste numa estimativa de ocupação de uma área específica à volta do robô móvel e que se move com ele. Considerando que C_{xy} denota "*célula (x,y) ocupada*", C_{xy} denota uma variável aleatória discreta com eventos no universo {0, 1}, isto é, C_{xy}=1 significa que a célula (x,y) está ocupada, enquanto que C_{xy}=0 significa que a célula (x,y) está livre (não ocupada). A construção do mapa pode ser vista como sendo o problema de estimar a probabilidade condicionada:

$$P(C_{xy} \mid o^{(1)}, \ldots, o^{(N)})$$

onde $o^{(1)}$ denota a primeira observação e $o^{(N)}$ a última observação. Utilizando o teorema de Bayes, a probabilidade condicionada da célula (x,y) estar ocupada, dada uma sequência de observações, pode ser calculada da seguinte forma [Burgard et al. 98]:

$$\begin{aligned} P(C_{xy} \mid o^{(1)},\ldots,o^{(N)}) &= \frac{P(o^{(N)} \mid C_{xy}, o^{(1)},\ldots,o^{(N-1)})P(C_{xy} \mid o^{(1)},\ldots,o^{(N-1)})}{P(o^{(N)} \mid o^{(1)},\ldots,o^{(N-1)})} \\ &= \frac{P(o^{(N)} \mid C_{xy})P(C_{xy} \mid o^{(1)},\ldots,o^{(N-1)})}{P(o^{(N)} \mid o^{(1)},\ldots,o^{(N-1)})} \end{aligned} \tag{5.2}$$

Na equação (5.2) a probabilidade condicionada $P(o^{(N)} \mid C_{xy}, o^{(1)},\ldots,o^{(N-1)})$ representa a probabilidade da observação actual dada a probabilidade anterior da célula (x,y) e todas as observações anteriores. Como a probabilidade anterior da célula já pressupõe a contribuição das observações anteriores, pode-se simplificar e substituir por $P(o^{(N)} \mid C_{xy})$. A probabilidade condicionada $P(C_{xy} \mid o^{(1)},\ldots,o^{(N-1)})$ representa a probabilidade na célula antes da nova actualização, dadas todas as observações anteriores. $P(o^{(N)} \mid o^{(1)},\ldots,o^{(N-1)})$ é a probabilidade de se verificar a observação actual, dadas todas as observações anteriores.

Substituindo a equação de Bayes,

$$P(o^{(N)} \mid C_{xy}) = \frac{P(C_{xy} \mid o^{(N)})P(o^{(N)})}{P(C_{xy})} \tag{5.3}$$

na equação (5.2) obtém-se:

$$P(C_{xy} \mid o^{(1)},...,o^{(N)}) = \frac{P(C_{xy} \mid o^{(N)})P(o^{(N)})P(C_{xy} \mid o^{(1)},...,o^{(N-1)})}{P(C_{xy})P(o^{(N)} \mid o^{(1)},...,o^{(N-1)})} \tag{5.4}$$

Este resultado é possível assumindo a independência estatística das leituras dos sensores efectuadas em instantes de tempo diferentes, para a mesma célula.

De forma semelhante à equação (5.4), a probabilidade de uma célula estar livre (não ocupada - $\overline{C_{xy}}$) é dada por:

$$P(\overline{C_{xy}} \mid o^{(1)},...,o^{(N)}) = \frac{P(\overline{C_{xy}} \mid o^{(N)})P(o^{(N)})P(\overline{C_{xy}} \mid o^{(1)},...,o^{(N-1)})}{P(\overline{C_{xy}})P(o^{(N)} \mid o^{(1)},...,o^{(N-1)})} \tag{5.5}$$

Dividindo a equação (5.4) pela equação (5.5) conduz à expressão seguinte que é denominada por *odds*[2] (probabilidades) da célula (x,y) estar ocupada [Moravec 88]:

$$\frac{P(C_{xy} \mid o^{(1)},...,o^{(N)})}{P(\overline{C_{xy}} \mid o^{(1)},...,o^{(N)})} = \frac{P(C_{xy} \mid o^{(N)})P(\overline{C_{xy}})P(C_{xy} \mid o^{(1)},...,o^{(N-1)})}{P(\overline{C_{xy}} \mid o^{(N)})P(C_{xy})P(\overline{C_{xy}} \mid o^{(1)},...,o^{(N-1)})} \tag{5.6}$$

Sabendo que $P(\overline{X}) = 1 - P(X)$ e substituindo esta igualdade na equação (5.6), e após algumas manipulações matemáticas, chega-se à fórmula, para a actualização das células, que calcula a probabilidade condicionada dada a última observação $o^{(N)}$,

$$P(C_{xy} \mid o^{(1)},...,o^{(N)}) = 1 - (1+b)^{-1} \tag{5.7}$$

$$b = \frac{P(C_{xy} \mid o^{(N)})}{1 - P(C_{xy} \mid o^{(N)})} \cdot \frac{1 - P(C_{xy})}{P(C_{xy})} \frac{P(C_{xy} \mid o^{(1)},...,o^{(N-1)})}{1 - P(C_{xy} \mid o^{(1)},...,o^{(N-1)})} \tag{5.8}$$

Na equação (5.8) $P(C_{xy} \mid o^{(N)})$ é fornecida pela RNA, $P(C_{xy} \mid o^{(1)},...,o^{(N-1)})$ representa a probabilidade na célula antes da actualização da mesma e $P(C_{xy})$ representa a probabilidade

[2] Expressão utilizada em [Moravec 88] para designar a divisão da equação 5.4 pela equação 5.5.

inicial da célula (x,y) estar ocupada (igual a 0.5). Utilizando a equação (5.7) pode-se calcular iterativamente a probabilidade da célula estar ocupada, o que significa que apenas é necessário guardar um valor por célula no mapa local. O valor da célula (x,y) no mapa representa a probabilidade do espaço correspondente estar ocupado (perto de 1) ou livre (perto de zero). Inicialmente todos os valores das células são iguais a 0.5, ou seja, desconhece-se se a célula está ocupada ou livre. Sempre que a célula parece estar ocupada o seu valor aumenta, caso contrário o seu valor diminui. Devido às características matemáticas da equação (5.7), se o valor da célula chegar a 1 ou zero, nas iterações seguintes o resultado permanecerá sempre 1 ou zero respectivamente, independentemente do valor de $P(C_{xy} \mid o^{(N)})$. Para obviar este facto, a gama de valores para as células foi limitada ao intervalo [0.01, 0.99].

5.1.3 Selector de Sector e Selector de Sensores

Neste método de construção de mapas, as funções do *selector de sector* são: identificar o sector a que pertence a célula (x,y), fornecer o ângulo das coordenadas polares ao *selector de sensores* e fornecer as coordenadas polares (R, θ) à RNA (secção 5.1.1). Por sua vez, o *selector de sensores*, através do ângulo fornecido pelo *selector de sector*, selecciona os dois sensores mais adequados para fornecer as distâncias à RNA, entre todos os sensores disponíveis.

O ambiente envolvente do robô é interpretado através de dezasseis sectores (figura 5.6) que cobrem os 360º à volta do robô. Um sector representa uma fracção do espaço que rodeia o robô. A divisão do espaço em sectores (figura 5.6) tem por objectivo flexibilizar a utilização do método no sentido de permitir a escolha das áreas a cartografar. A escolha da disposição adoptada teve em consideração a divisão do espaço em partes iguais com uma dimensão intermédia entre sectores muito grandes e demasiado pequenos.

Os selectores podem executar outro tipo de funções mais complexas, como por exemplo funções de custo mínimo ou uma função objectivo. Podem ainda ser modificados para apoiar o planeamento de caminhos, o desvio de obstáculos, o estudo do ambiente envolvente (um mapa global) e para apoiar a execução de tarefas específicas como passar através de passagens estreitas ou passagens de portas. O facto de estarem colocados em módulos distintos na

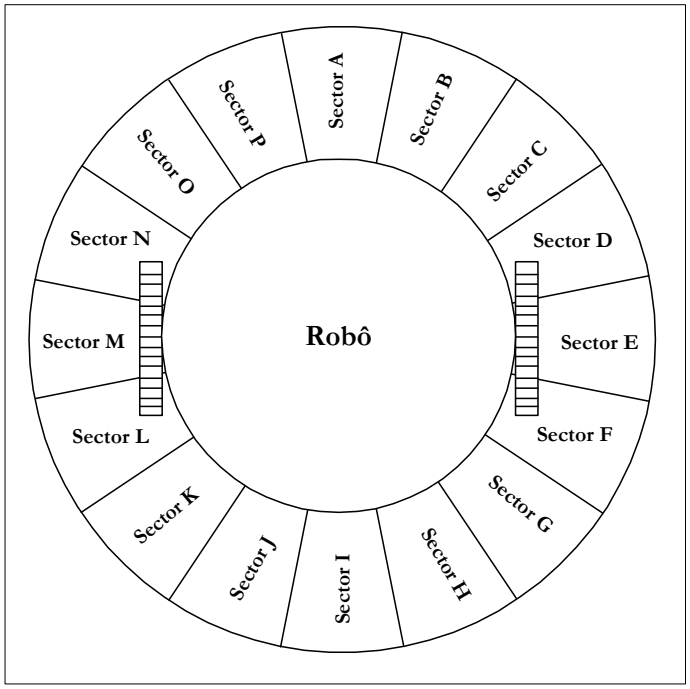

Figura 5.6. Disposição dos Sectores.

arquitectura tem por objectivo separar a interpretação das leituras dos sensores da pesquisa de células das áreas a cartografar. A vantagem principal desta configuração é permitir a aplicação deste método noutro robô móvel sem ser necessário alterar todo o método de actualização das células do mapa. Isto é válido para um robô móvel qualquer, mesmo que tenha dimensões diferentes e a disposição dos sensores não seja idêntica à dos robôs móveis utilizados no desenvolvimento do método. Outra vantagem importante é que esta arquitectura é independente do tipo e número de sensores (e da sua disposição) presentes no robô. Claro que para utilizar este método noutro robô móvel diferente o módulo *selector de sensores* terá que ser alterado e eventualmente o módulo *selector de sector*. No entanto, as alterações a efectuar poderão ser mínimas, dependendo das características do robô, nomeadamente das dimensões e da disposição dos sensores. Mais concretamente, e supondo que não existe necessidade de desenvolver software especial para aquisição das leituras dos sensores, as alterações resumem-se a modificar o valor que contém o número de sensores e a preencher o vector dos sensores com a leitura de cada sensor e o respectivo ângulo da orientação. Depois destas alterações poderá

haver situações onde seja necessário fazer alguns ajustes. Como por exemplo a situação em que existe sobreposição da gama perceptual entre sensores do mesmo tipo ou de tipos diferentes. Neste caso é preciso desenvolver um algoritmo para decidir, em cada orientação (onde acontece a sobreposição), qual o sensor a adoptar. Pode acontecer que um sensor seja mais adequado para distâncias pequenas e outro para distâncias maiores.

No caso concreto do desenvolvimento deste método, foram utilizados dois robôs da mesma família mas com configurações distintas. No robô Nomad 200 (em simulação) os sonares e os sensores de infravermelhos têm a mesma orientação. Os sensores de infravermelhos têm uma gama perceptual para distâncias pequenas (até 50 cm). Por este motivo, sempre que as leituras de distância são menores que 50 cm, nos sensores de infravermelhos e as leituras de distância nos sonares são superiores, o valor desta distância prevalece em relação à distância fornecida pelos sonares, caso contrário, prevalece a distância fornecida pelos sonares. Quando se mudou do ambiente de simulação para o robô Super Scout II apenas foram utilizados sensores sonares (com a mesma disposição do robô Nomad 200).[3]

A função dos selectores neste método é apoiar a exploração do ambiente envolvente, nas proximidades do robô, para a construção de um mapa local. Na secção 5.2.2 será apresentado um exemplo concreto da actuação destes selectores.

5.2 Algoritmo para a Construção do Mapa Local

A grelha de células foi escolhida como base para a construção do mapa local porque conduz a mapas de interpretação directa, robustos e que podem adaptar-se a vários ambientes. Neste método, o ambiente é modelizado através de uma grelha bidimensional.

[3] As alterações no software foram mínimas. Em primeiro lugar, retiraram-se as linhas de código que permitiam fazer aquisição das leituras provenientes dos sensores de infravermelhos, em segundo lugar, no vector onde eram colocadas as leituras daqueles sensores foram colocados valores muito elevados (acima da gama útil de distâncias utilizadas no método). Em terceiro lugar, foram modificados os parâmetros de duas instruções de interligação com o robô (pois são diferentes do Nomad 200 utilizado nas simulações). Como se pode ver as alterações foram poucas e o método funcionou muito bem no robô real.

$P(C_{xy} \mid o^{(N)})$	$P(C_{xy})$	Actualização nº.						
		1	2	3	4	5	10	15
0,55	0,5	0,55	0,599	0,646	0,691	0,732	0,881	0,953
0,45	0,5	0,45	0,401	0,354	0,309	0,268	0,119	0,047
0,51	0,5	0,51	0,52	0,53	0,54	0,55	0,599	0,646
0,49	0,5	0,49	0,48	0,47	0,46	0,45	0,401	0,354
0,5	0,7	0,7	0,7	0,7	0,7	0,7	0,7	0,7
0,3	0,5	0,3	0,155	0,073	0,033	0,014	0	0
0,7	0,5	0,7	0,845	0,927	0,967	0,986	1	1

Tabela 5.2. Comportamento da equação (5.7).

Como foi referido na secção 5.1.2, o valor das células está limitado ao intervalo]0, 1[. Utilizando a equação (5.7) o valor da célula tende a ser 0 ou 1 depois de algumas actualizações, mesmo que $P(C_{xy} \mid o^{(N)})$ (dada pela RNA) seja 0.45 ou 0.55 respectivamente. A tabela 5.2 mostra o comportamento da equação (5.7) para o caso de uma célula ser actualizada várias vezes com o mesmo valor de $P(C_{xy} \mid o^{(N)})$. Como se pode observar, quando o valor de $P(C_{xy} \mid o^{(N)})$ é muito próximo de 0.5 (mas não igual), o valor da célula tende a aumentar para 1 ou diminuir para zero consoante $P(C_{xy} \mid o^{(N)})$ for ligeiramente superior a 0.5 ou inferior a 0.5 respectivamente. Para tornar o método de construção do mapa dinâmico e com alguma capacidade de aprendizagem e de adaptação a ambientes dinâmicos, quando os valores de $P(C_{xy} \mid o^{(N)})$ são muito próximos de 0.5, o método deverá considerar que as células em causa não traduzem uma tendência para a ocupação ou não do espaço real que representam. Por conseguinte, o valor dessas células deverá indicar essa incerteza. Por outro lado, a fórmula de actualização (equação 5.7) não altera o valor da célula se $P(C_{xy} \mid o^{(N)})$ for 0.5, ou seja, a $P(C_{xy} \mid o^{(N)})$ =0.5 funciona como elemento neutro. Para obviar esta situação foi introduzido um procedimento heurístico no algoritmo: se $P(C_{xy} \mid o^{(N)})$ pertencer ao intervalo [0, 0.4]∪[0.6, 1], aplica-se a equação (5.7), caso contrário calcula-se a média entre o valor de $P(C_{xy} \mid o^{(N)})$ e o valor probabilístico presente na célula em causa (consultar secção seguinte).

5.2.1 Actualização das Células do Mapa Local

1. Inicialmente, para todas as células do mapa: $P(C_{xy}) = 0.5$;
2. Para cada célula (x,y) e para cada nova observação $o^{(N)}$, a RNA fornece como saída $P(C_{xy} \mid o^{(N)})$;
3. Cálculo do novo valor para a célula:

   Se $P(C_{xy} \mid o^{(N)}) \in [0\ 0.4] \cup [0.6\ 1]$ *então*

$$P(C_{xy})_{[k]} = 1 - \left(\frac{P(C_{xy} \mid o^{(N)})}{1 - P(C_{xy} \mid o^{(N)})} \cdot \frac{P(C_{xy})_{[k-1]}}{1 - P(C_{xy})_{[k-1]}} \right)^{-1} \tag{5.9}$$

   senão

$$P(C_{xy})_{[k]} = \frac{P(C_{xy} \mid o^{(N)}) + P(C_{xy})_{[k-1]}}{2} \tag{5.10}$$

4. Actualização do valor da célula limitando-o ao intervalo [0.01, 0,99]:

   Se $P(C_{xy})_{[k]}$ maior que 0.99 *então*

   $P(C_{xy})_{[k]} = 0.99$

   Se $P(C_{xy})_{[k]}$ menor que 0.01 *então*

   $P(C_{xy})_{[k]} = 0.01$

onde $P(C_{xy})_{[k-1]}$ representa o valor anterior da célula e $P(C_{xy})_{[k]}$ denota o novo valor para a célula, ou seja, $P(C_{xy})_{[k]} = P(C_{xy} \mid o^{(1)},...,o^{(N)})$. A equação (5.9) é a equação (5.7) depois de efectuada a operação seguinte. Como na equação (5.8) $P(C_{xy})$ denota o valor da probabilidade inicial na célula, que é 0.5, então:

$$\frac{P(C_{xy})}{1 - P(C_{xy})} = 1 \tag{5.11}$$

5.2.2 Selecção dos Sectores e das Leituras dos Sensores

Quando uma célula (x,y) é proposta para ser actualizada, em primeiro lugar, é necessário determinar a que sector pertence. A seguir, é necessário identificar os dois sensores mais adequados para fornecer as informações de distância na direcção da célula. Por exemplo, uma célula com as coordenadas polares (R,θ)=(65, 36.5º) (ver figura 5.3) pertence ao sector O, e sabe-se que s1 e s2 estão orientados a 22.5º e 45º respectivamente. Por conseguinte, os sensores das orientações s1 e s2 irão fornecer as leituras de distância necessárias para determinar $P(C_{xy} \mid o^{(N)})$ através da RNA.

Nas simulações do capítulo 6, e como já foi referido na secção 5.1.3, foi utilizada uma regra heurística muito simples para escolher entre as leituras dos sonares e dos sensores de infravermelhos. Para leituras de distância inferiores a 50 centímetros é dada preponderância à informação proveniente dos sensores de infravermelhos (desde que seja menor que a informação proveniente dos sonares), senão adopta-se a informação proveniente dos sonares.

A descrição anterior é apenas um tipo de funcionalidade que esta arquitectura permite. Os módulos *selector de sector* e *selector de sensores* podem ser utilizados para construir um módulo de decisão com o objectivo de dar mais atenção a sectores prioritários, em função da direcção do robô. A selecção das áreas a cartografar com mais atenção pode ser conduzida heuristicamente ou com determinado propósito. Neste último caso podem-se incluir as selecções feitas através de uma função de custo mínimo, de uma função objectivo ou através de tarefas específicas. Por exemplo, durante a navegação local de um robô móvel deve-se ter mais atenção às áreas que estão na direcção dos movimentos do robô.

5.2.3 Actualização com o Robô Móvel em Movimento

Quando o robô está em movimento, a janela (a grelha de células) move-se solidariamente com ele. Por conseguinte, torna-se necessário ter uma estimativa relativa da posição de modo a actualizar correctamente as células do mapa. A posição do robô relativamente à grelha de células

é fixa (no centro). Durante os movimentos de translação, os movimentos do robô são levados em consideração na actualização das células. A grelha de células é deslocada segundo os dois eixos (x e y) com base na informação odométrica. Quando o robô executa movimentos de rotação torna-se necessária uma abordagem diferente. Neste caso, a orientação do sistema de coordenadas do robô {M} move-se solidariamente com as rotações do robô. A consequência disto é que as orientações dos sensores também irão ser alteradas, por isso, as respectivas orientações são actualizadas para manter o contexto do mapa. A informação odométrica desempenha um papel muito importante nesta questão. Esta informação permite o deslocamento correcto da janela de células durante todo o tipo de movimentos executados pelo robô. A informação odométrica fornecida pelo simulador e pelo robô real consiste na posição do robô (x,y) e ainda no ângulo, em relação a {M}, em cada instante.

Capítulo 6

Resultados

Neste capítulo são apresentados os mapas construídos através do método proposto nesta dissertação. Os resultados presentes neste capítulo estão organizados por situações específicas. Estas situações envolvem ambientes simulados e ambientes reais. No ambiente de simulação o robô utilizado é o Nomad 200 e no ambiente real é o Super Scout II. Como já foi referido anteriormente, a principal diferença entre eles (no contexto dos mapas) é que o robô Super Scout II apenas tem sonares para medir distâncias enquanto que o robô Nomad 200 além dos sonares tem ainda sensores de infravermelhos.

Os mapas apresentados são analisados segundo vários parâmetros: precisão, robustez, adaptabilidade e impacto das rotações do robô na configuração do mapa de células.

- A precisão é analisada através da comparação entre o mapa construído com dados sensoriais e o mapa do ambiente real ou do ambiente simulado. A diferença entre as medidas origina um erro, quanto menor for o erro melhor será a precisão. As orientações segundo as quais são efectuadas as medidas são idênticas às orientações indicadas pela rosa-dos-ventos, nomeadamente, os pontos cardeais e colaterais. A orientação Norte aponta na direcção perpendicular ao cabeçalho da folha de papel. A razão da escolha destas orientações prende-se com o facto de ser fácil medir a altura, largura e a diagonal das células;

Probabilidade de Ocupação	Cor	
[0.0 0.2[	Branco	
[0.2 0.4[	Amarelo	
[0.4 0.6[	Magenta	
[0.6 0.8[	Azul	
[0.8 1.0]	Preto	

Tabela 6.1. Equivalência entre as cores das células e a gama de valores das probabilidades de ocupação.

- O impacto das rotações do robô é observado mostrando mapas depois do robô efectuar rotações, acompanhadas por vezes de movimentos de translação. Nalgumas situações o ângulo de rotação é múltiplo de 22.5 graus, noutras foram utilizados valores diferentes para evidenciar determinadas particularidades. O robô tem 16 sonares dispostos à sua volta igualmente espaçados por 22.5 graus. Sempre que o ângulo de rotação é múltiplo daquele valor, o mapa é sempre idêntico, uma vez que, embora seja outro sensor, haverá sempre um sensor com a mesma orientação do anterior. Quando o ângulo não é múltiplo de 22.5 graus o mapa poderá sofrer alterações. As alterações que surgem no mapa devem-se ao facto dos sensores mudarem de posição. Estas mudanças de posição alteram o ângulo de incidência dos feixes acústicos nas superfícies à volta do robô, alterando também as medidas dos sensores. Consequentemente, os objectos que eram detectados podem deixar de o ser e outros que não eram detectados podem aparecer no mapa.

- A adaptabilidade é analisada do seguinte modo: primeiro coloca-se um objecto no ambiente (nas imediações do robô), de seguida apresenta-se a reacção do mapa depois da primeira actualização e após 15 actualizações. O procedimento repete-se ao ser retirado o objecto. Se o mapa se adaptar rapidamente às mudanças do ambiente então o método possui uma boa adaptabilidade;

- A robustez é analisada observando o mapa depois da primeira actualização (inicialmente todas as células têm o valor 0.5) e o mapa após várias actualizações. Se o mapa não sofrer grandes alterações do primeiro mapa para o segundo então o método apresenta uma robustez elevada;

Para todos os casos apresentados são ainda mostrados mapas específicos para analisar o efeito da utilização de limites na informação sensorial. Foi estabelecido apenas um limite superior de 150 centímetros por ser um valor intermédio entre a distância do robô aos extremos do mapa de grelha de células na perpendicular (130 centímetros) e na diagonal (180 centímetros) Nos padrões de treino o valor máximo utilizado foi de 350 centímetros e a diferença mais elevada entre as duas distâncias em cada padrão foi de 80 centímetros.

Os mapas de grelha de células são apresentados a cores. A equivalência entre o valor da probabilidade de ocupação de cada célula e a cor respectiva no mapa varia da seguinte forma: cores mais claras indicam que a probabilidade da célula estar ocupada é reduzida, pelo contrário, as cores mais escuras indicam que a probabilidade de ocupação é elevada. No mapa, as células podem ter 5 cores diferentes e a equivalência pode ser observada na tabela 6.1.

Nas tabelas onde são feitas as comparações entre as medidas do mapa e as medidas obtidas através do simulador ou do ambiente real, o erro é determinado pela diferença entre as referidas medidas. Se o valor for positivo o erro é por defeito, caso contrário será por excesso.

Neste capítulo, por simplicidade, o mapa de grelha de células poderá ser referenciado apenas por mapa.

O valor dos ângulos tem como referência (0 graus) a direcção Este. A indicação da orientação do robô, no mapa, é feita através de um quadrado amarelo dentro da zona ocupada pelo mesmo. No mapa métrico dos ambientes, a indicação da orientação do robô é feita através de um pequeno triângulo que está colocado na frente do robô.

As células utilizadas nos mapas são quadradas com 5 centímetros de lado. Esta dimensão também condiciona o valor dos erros, ou seja, células com uma dimensão inferior certamente contribuiriam para erros menores mas teria como consequência o aumento do número de células para a mesma área a cartografar. A dimensão da célula deve ser escolhida ponderando alguns aspectos: a precisão, a tolerância, a utilização do mapa, as características dinâmicas e as

dimensões do robô móvel e ainda o tipo de sensores disponíveis no robô. Os mapas são constituídos por 60x60 células representando uma área de 3x3 metros quadrados.

As experiências realizadas neste capítulo foram efectuadas em ambientes de simulação (três) e em ambientes reais (três). Estes ambientes recriam situações específicas que serviram para estudar o comportamento do método de construção de mapas. Na primeira simulação, o robô encontra-se numa sala fechada, com uma área aproximada de quatro metros quadrados. Na segunda simulação, o robô encontra-se num corredor estreito, aberto numa das extremidades, com uma área aproximada de dois metros quadrados. Na terceira simulação, o robô é colocado num ambiente misto, com uma zona de espaço aberto e outra onde existe um pequeno corredor. Em relação aos ambientes reais, no primeiro, o robô está num ambiente misto, com uma parte estruturada e outra com um espaço aberto. No entanto, no espaço aberto foi colocada uma cadeira cujas pernas têm uma dimensão reduzida. O segundo ambiente real é semelhante ao anterior. Neste ambiente, a cadeira foi retirada e foram colocados dois objectos de madeira perto do robô. Estes objectos estão colocados na vertical e têm uma altura de 50 centímetros e uma área de 10 centímetros quadrados. No terceiro ambiente real, o robô é colocado num corredor estreito, aberto numa das extremidades (semelhante ao ambiente da segunda simulação).

6.1 Sala Fechada

Nesta simulação, o robô encontra-se numa sala com uma área aproximada de quatro metros quadrados. A figura 6.1 mostra o mapa construído no ambiente de simulação e a figura 6.2 mostra o mapa da sala com as respectivas dimensões.

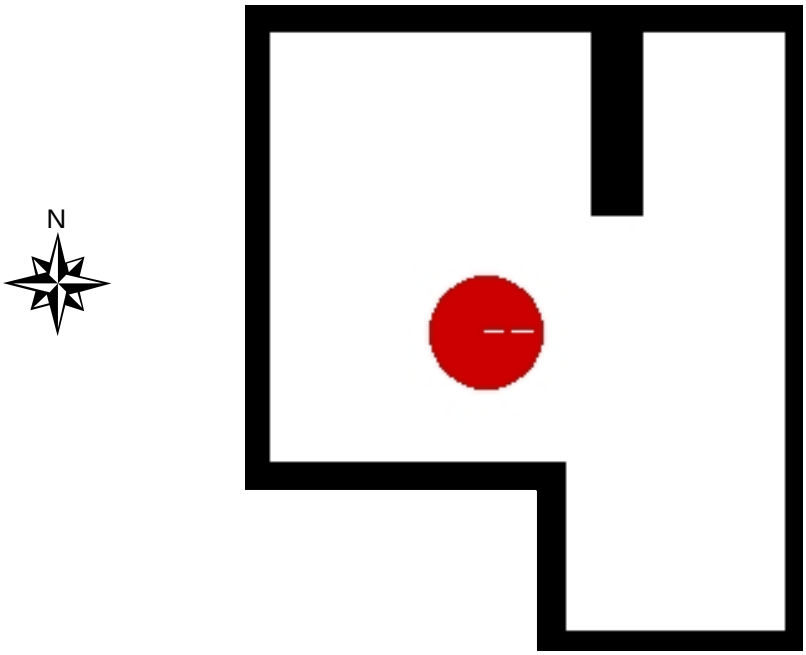

Figura 6.1. Mapa construído no ambiente de simulação.

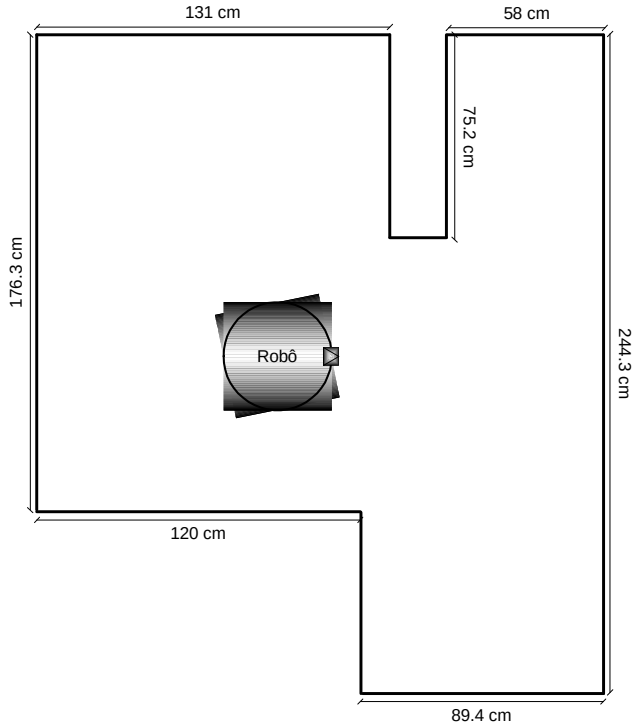

Figura 6.2. Mapa da sala.

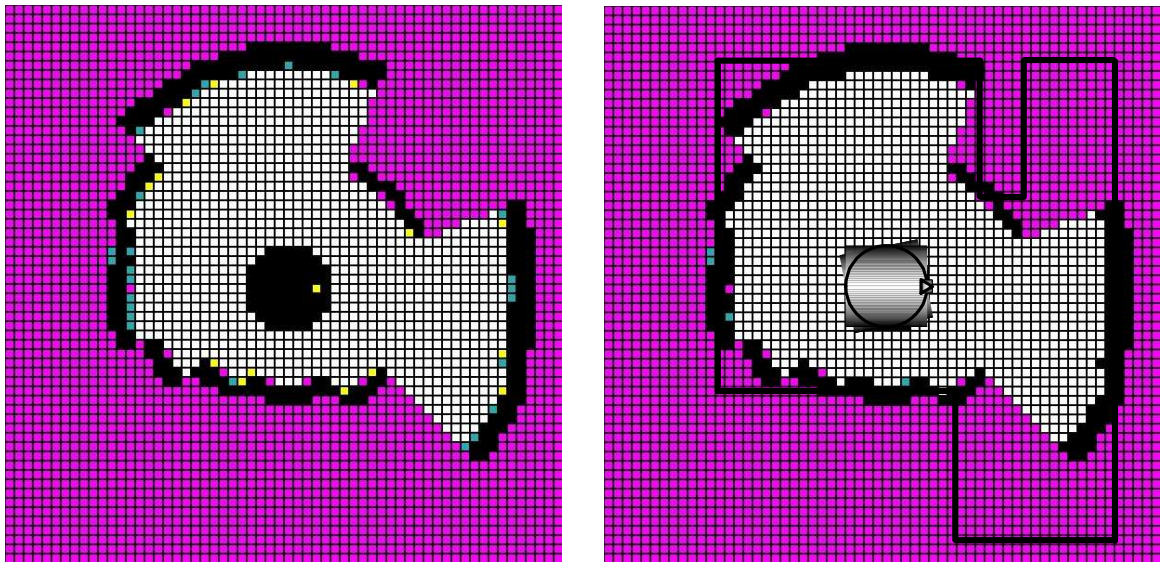

Figura 6.3. a) À esquerda, o mapa relativo à primeira actualização. b) À direita o mapa após várias actualizações com sobreposição do mapa do simulador.

		Norte		
		95		
	Noroeste	99,1	**Nordeste**	
	91,9	4,1	42,4	
	101,9		43,1	
Oeste	10,0	Mapa	0,7	**Este**
60		Simulador		95
65,1		Erro		98,2
5,1	**Sudoeste**		**Sudeste**	3,2
	35,4		42,4	
	52,6	**Sul**	45,7	
	17,2	30	3,3	
		30,5		
		0,5		

Tabela 6.2. Comparação entre as medidas obtidas através do simulador e as medidas dadas pelo mapa. Os valores estão em centímetros. Exemplificando a leitura da tabela na direcção Norte, o mapa de grelha de células indica uma distância de 95 centímetros, no simulador a medida é de 99.1 centímetros e o erro é de 4.1 centímetros.

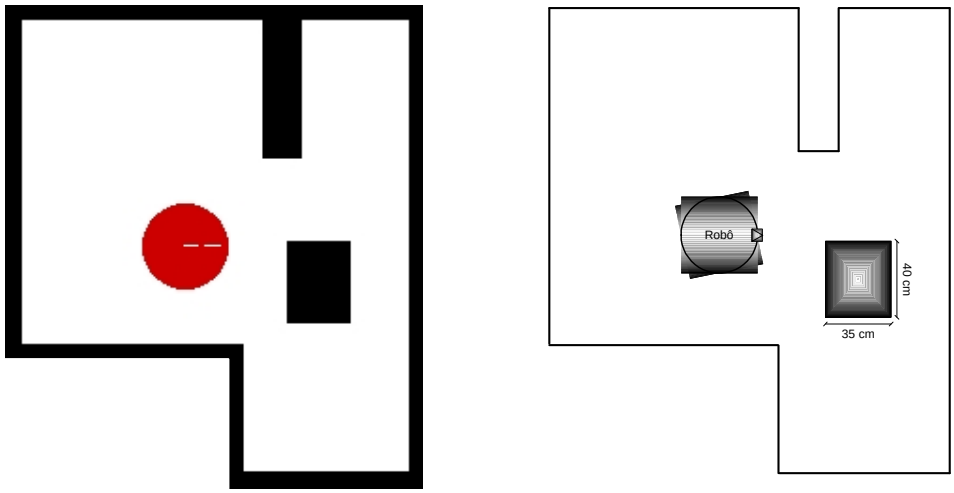

Figura 6.4. Imagem e mapa da sala fechada com introdução de um objecto.

6.1.1 Robustez e Precisão

A figura 6.3a) mostra o mapa depois da primeira actualização e a figura 6.3b) o mapa após várias actualizações. A diferença entre os dois mapas reside apenas em algumas células cuja tendência se veio a confirmar depois de algumas actualizações. Nomeadamente, as células da figura 6.3a) cujo valor da probabilidade estaria entre 0.6 e 0.8 (células azuis) ficaram com um valor entre 0.8 e 1 (células pretas) na figura 6.3b). O mesmo sucedendo com as células amarelas que se tornaram brancas. Pode-se salientar que o mapa, após as primeiras actualizações, se manteve inalterável com o passar do tempo.

Na tabela 6.2 é feita a comparação entre os valores das medidas do mapa e as medidas obtidas através do simulador. De um modo geral, o erro é inferior à dimensão de uma célula. Nos casos Noroeste e Sudoeste, o erro é superior porque existem sensores vizinhos (nessas orientações) com leituras de distância inferiores à distância correcta devido à presença de cantos nessas orientações.

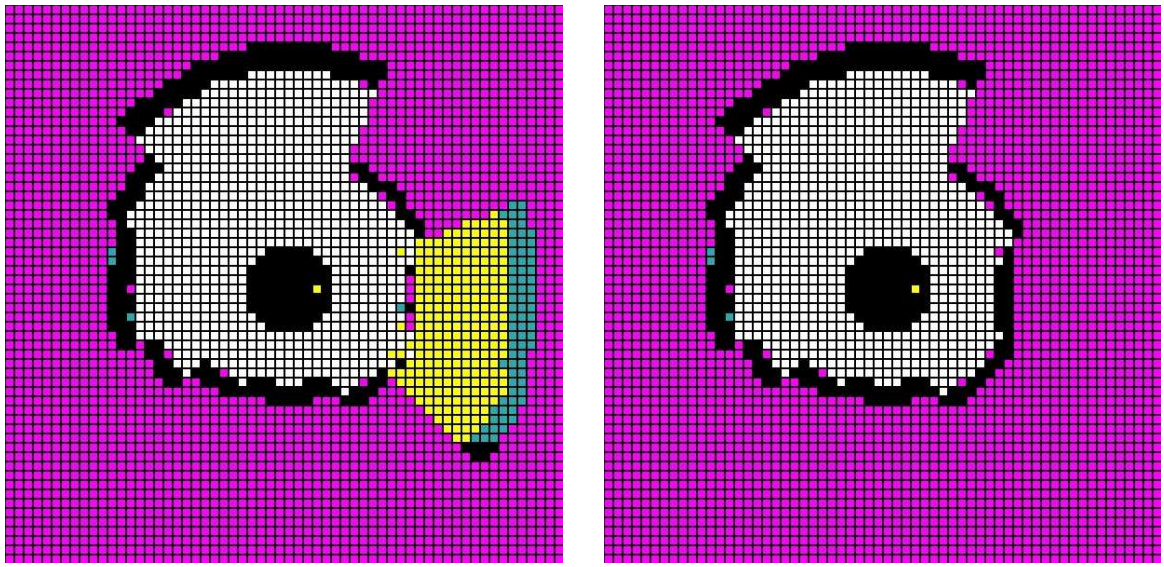

Figura 6.5. a) À esquerda: o mapa após a primeira actualização depois da introdução de um novo objecto. b) À direita: o mapa após 15 actualizações.

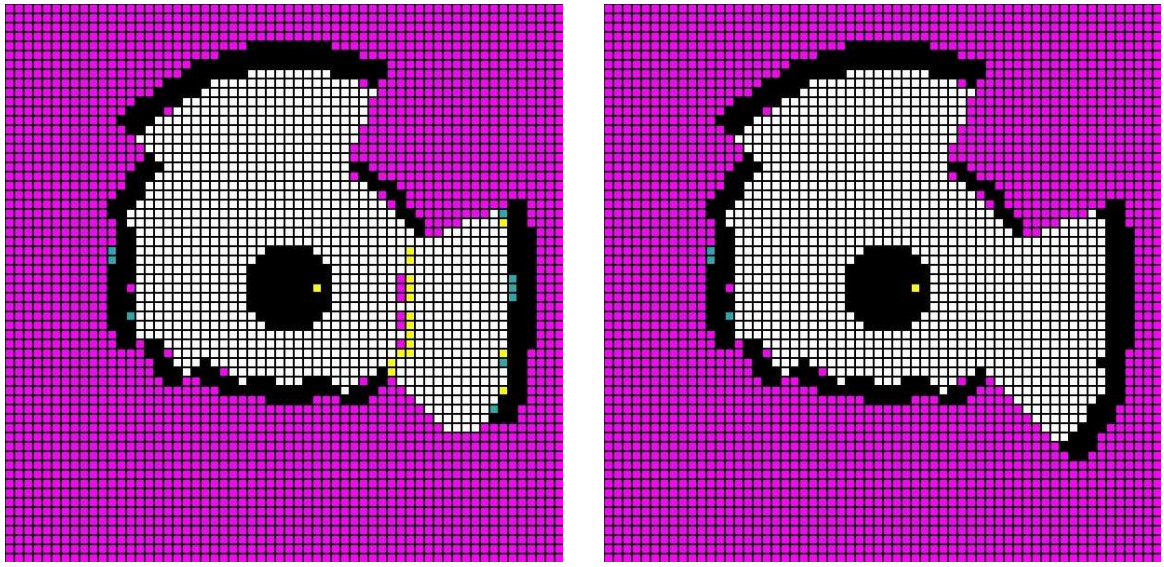

Figura 6.6. a) À esquerda: o mapa após a primeira actualização ao ser retirado o objecto. b) À direita está o mapa após 15 actualizações.

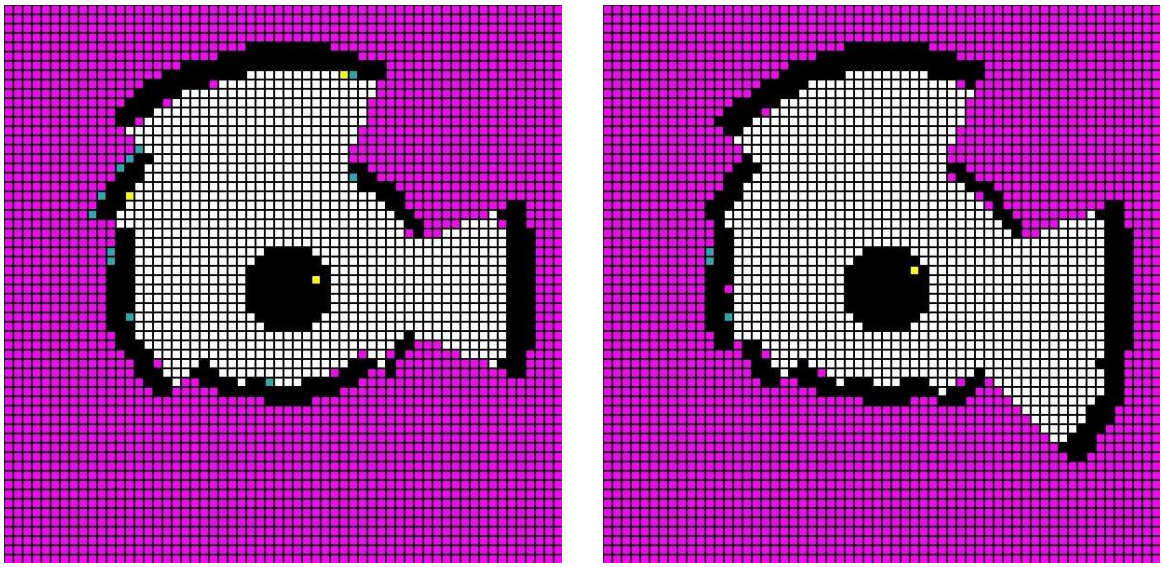

Figura 6.7. a) À esquerda está o mapa depois de uma rotação de 15 graus no sentido anti-horário. b) À direita está o mapa após uma rotação adicional de 7.5 graus no mesmo sentido.

6.1.2 Adaptabilidade

A figura 6.4 mostra onde foi colocado o objecto para observar o comportamento do método perante a alteração do ambiente. Na figura 6.5 pode-se verificar a rápida reacção do método à chegada de um objecto. Da mesma forma, quando o objecto é retirado, o mapa volta à configuração que tinha antes da chegada do objecto logo nas primeiras actualizações.

6.1.3 Impacto das Rotações do Robô na Configuração do Mapa

Com uma rotação de 22.5 graus no sentido anti-horário (figura 6.7b), o mapa mantém a configuração do mapa inicial. Se a rotação parar nos 15 graus o mapa fica ligeiramente diferente. A maior diferença reside na orientação Sudeste onde se detecta uma superfície a menor distância.

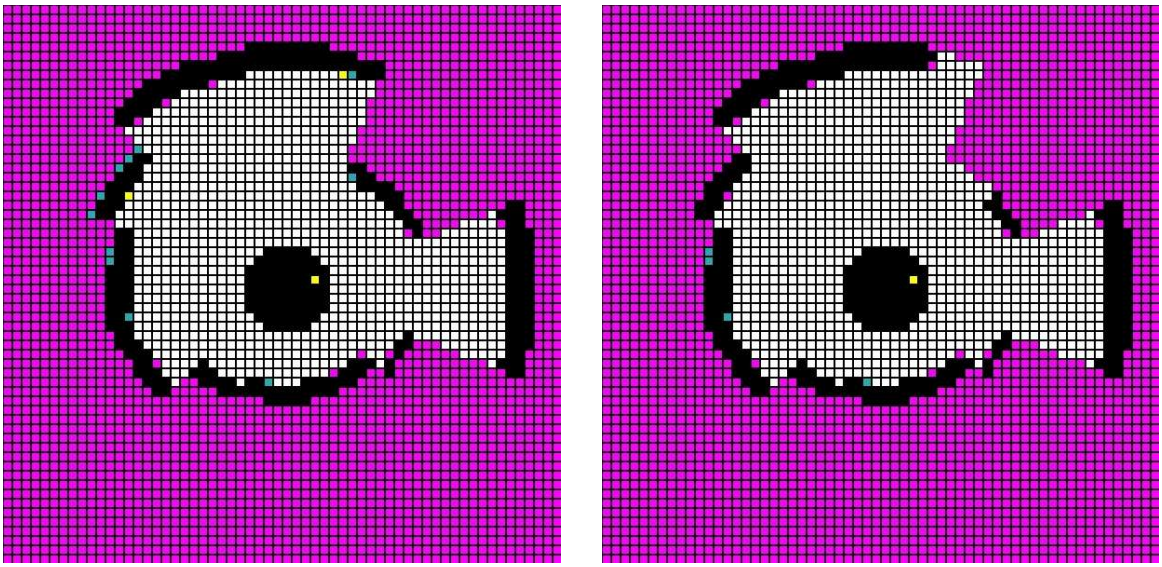

Figura 6.8. Mapas de grelhas de células depois de uma rotação de 15 graus com informação sensorial a) limitada b) não limitada.

6.1.4 Limites na Informação Sensorial

Na figura 6.8a) o mapa é idêntico ao da figura 6.7a). A figura 6.8b) mostra o mapa, nas mesmas condições do mapa à esquerda, onde a única diferença é a utilização da informação sensorial não limitada. Pode-se observar a diferença existente entre os dois mapas a Nordeste. A razão desta diferença é a leitura de um dos sensores, orientado nesse quadrante, que é de 650 centímetros enquanto que os sensores vizinhos fornecem leituras de 99 e 45 centímetros. A consequência deste facto é que a RNA não irá fornecer a $P(C_{xy} \mid o^{(N)})$ mais indicada para a actualização de algumas células daquela área, uma vez que os padrões utilizados no treino da RNA não continham valores tão díspares (a diferença máxima foi de 80 centímetros) como os que apareceram nesta situação.

6.2 Corredor no Simulador

Nesta simulação, o robô está num corredor aberto numa das extremidades.

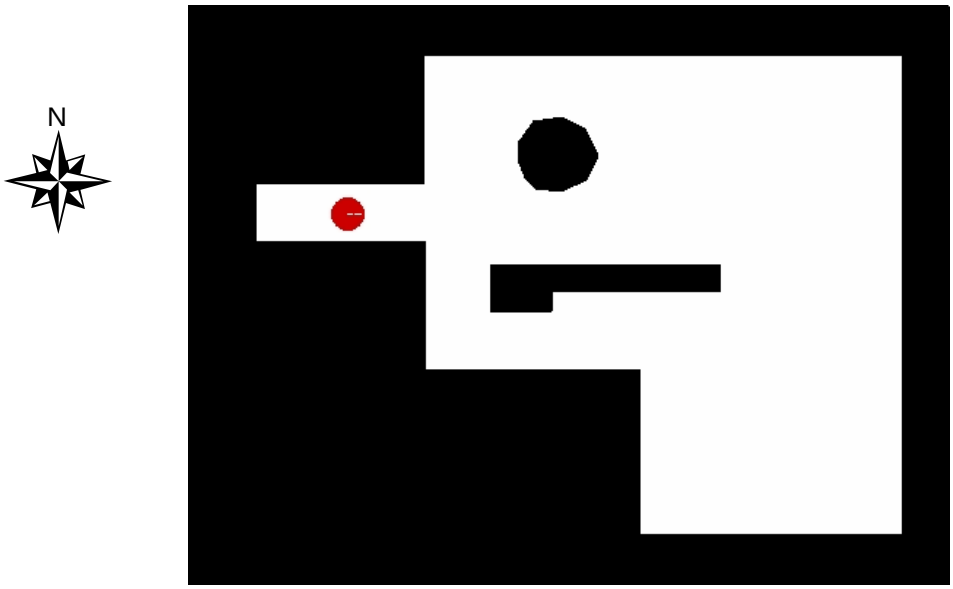

Figura 6.9. Mapa construído no ambiente de simulação.

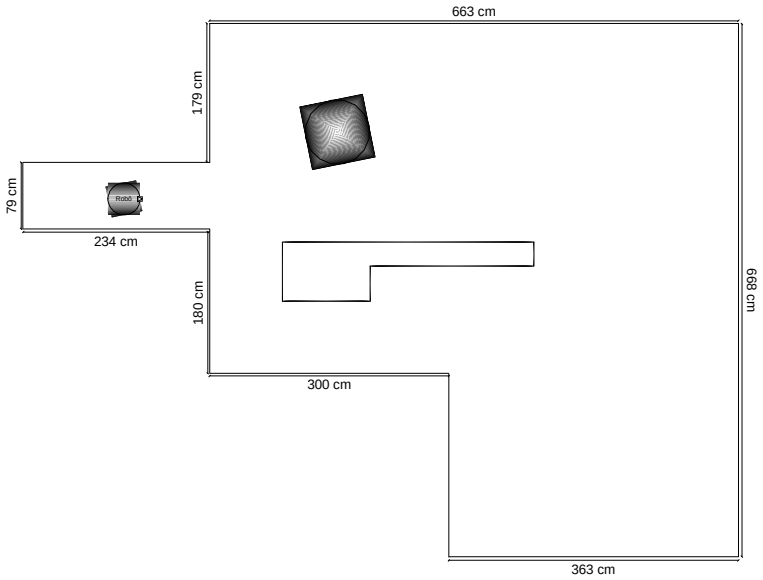

Figura 6.10. Mapa do ambiente de simulação.

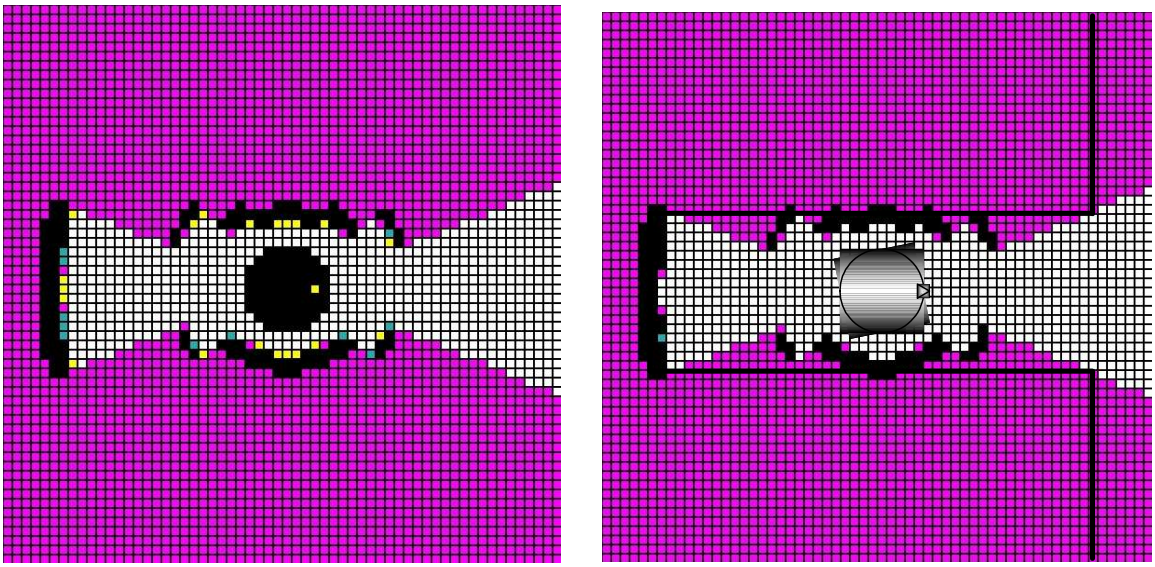

Figura 6.11. a) À esquerda está o mapa relativo à primeira actualização. b) À direita está o mapa após várias actualizações com sobreposição do mapa do simulador.

	Norte			
	15			
	Noroeste	18,2	**Nordeste**	
	14,1	3,2	14,1	
	35,3		35,3	
Oeste	21,2	Mapa	21,2	**Este**
100		Simulador		> 130
104		Erro		> 150
4,0	**Sudoeste**		**Sudeste**	
	14,1		14,1	
	31,3	**Sul**	31,3	
	17,2	15	17,2	
		15,7		
		0,7		

Tabela 6.3. Comparação entre as medidas obtidas através do simulador e as medidas dadas pelo mapa. Os valores estão em centímetros.

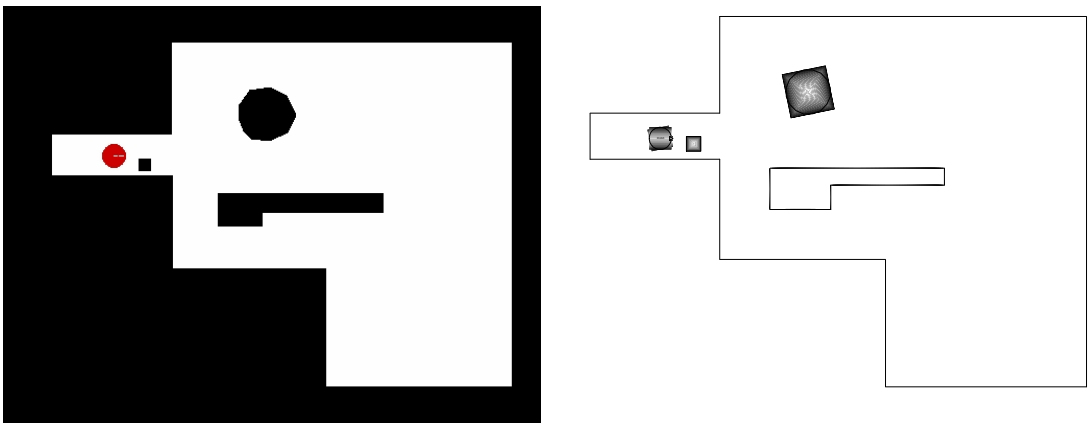

Figura 6.12. Imagem e mapa do corredor com introdução de um objecto quadrado com 26,5 centímetros de lado.

6.2.1 Robustez e Precisão

A figura 6.11 mostra dois mapas, um relativo à primeira actualização (figura 6.11a) e outro após várias actualizações (figura 6.11b). Como se pode observar, o mapa inicial é idêntico ao mapa após várias actualizações mostrando uma robustez elevada.

Na tabela 6.3 é feita a comparação entre as medidas obtidas através do simulador e as medidas dadas pelo mapa. Nas direcções Norte, Oeste e Sul, o erro é inferior à dimensão da célula. Nas restantes direcções (excepto a Este) todas as medidas apresentam um erro mais elevado, no entanto, o erro é sempre por defeito o que significa que a medida do mapa é inferior à dada pelo simulador. Os erros devem-se ao facto dos sensores vizinhos entrarem em consideração, uma vez que uma distância menor prevalece perante as distâncias maiores.

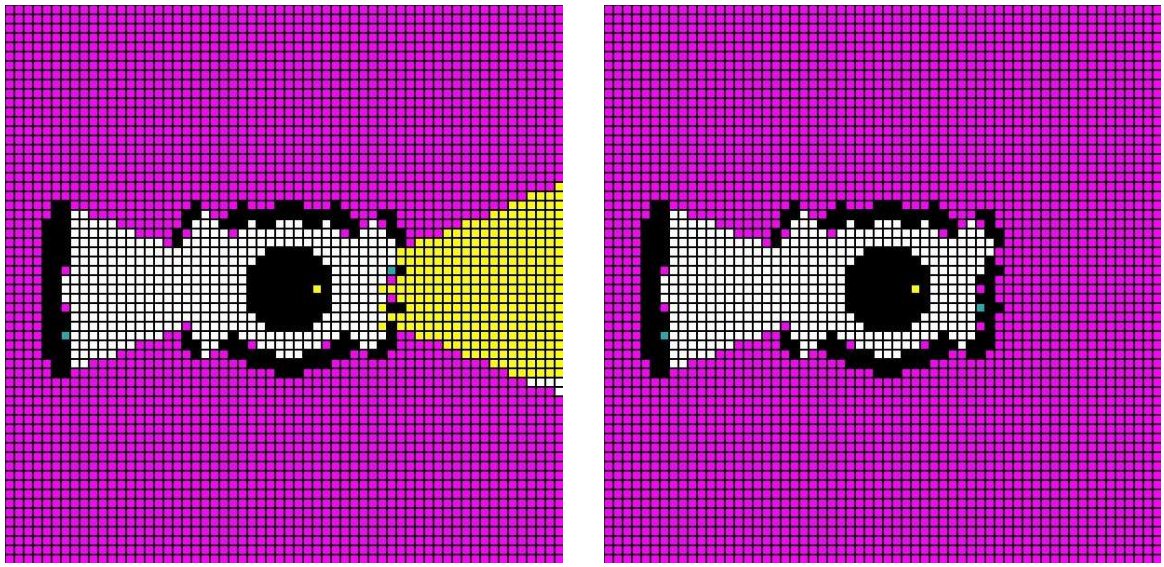

Figura 6.13. a) À esquerda: o mapa após a primeira actualização depois da introdução de um novo objecto. b) À direita: o mapa após 15 actualizações.

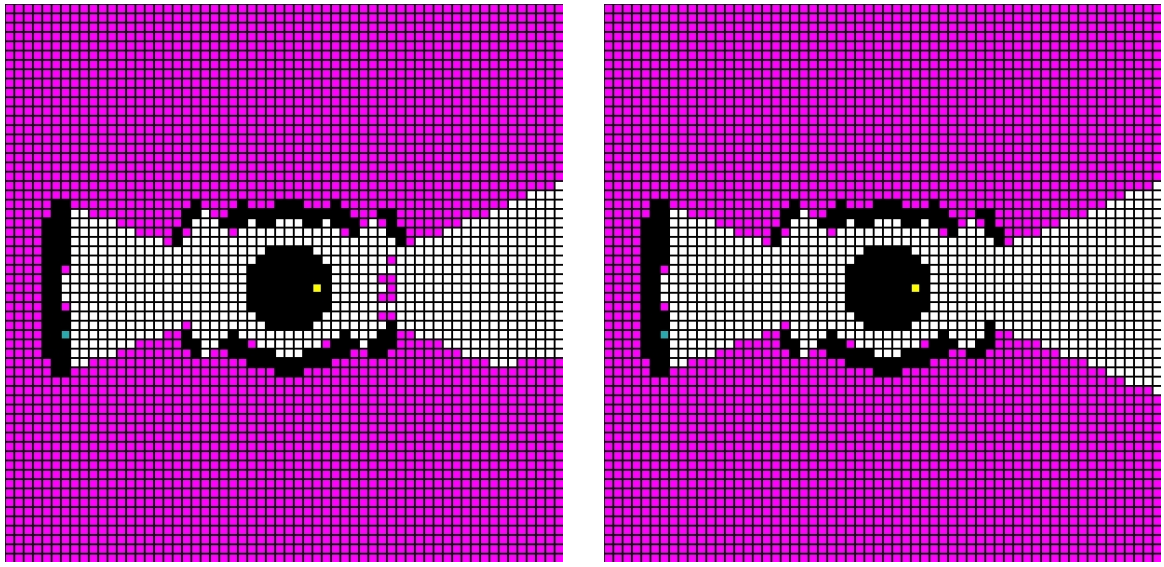

Figura 6.14. a) À esquerda: o mapa após a primeira actualização ao ser retirado o objecto. b) À direita está o mapa após 15 actualizações.

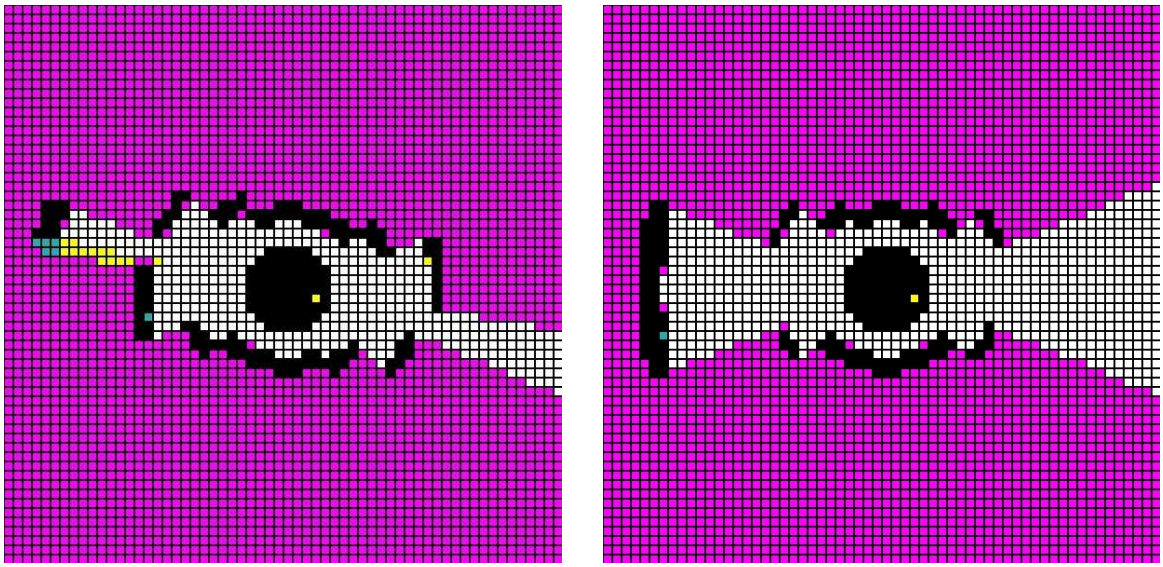

Figura 6.15. a) À esquerda está o mapa depois de uma rotação de 345 graus no sentido anti-horário. b) À direita está o mapa após uma rotação de 7.5 graus no sentido horário.

6.2.2 Adaptabilidade

Na figura 6.12 estão representados a imagem e o mapa do corredor com a introdução de um novo objecto. Como se pode verificar na figura 6.13, o método reage rapidamente à chegada do objecto. Pode-se salientar ainda que na primeira actualização, o mapa ficou logo com os contornos do mapa final (figura 6.13b). Na figura 6.14 pode-se observar o comportamento do mapa ao ser retirado o objecto. Também neste caso o método foi rápido na adaptação à mudança do ambiente.

6.2.3 Impacto das Rotações do Robô na Configuração do Mapa

Na figura 6.15 pode-se observar o impacto de uma rotação de 345 graus no sentido anti-horário (figura 6.15a) e de uma rotação de 337.5 graus no mesmo sentido (em relação à origem - Este). Como o valor de 337.5 graus é múltiplo de 22.5 graus o mapa fica idêntico ao mapa

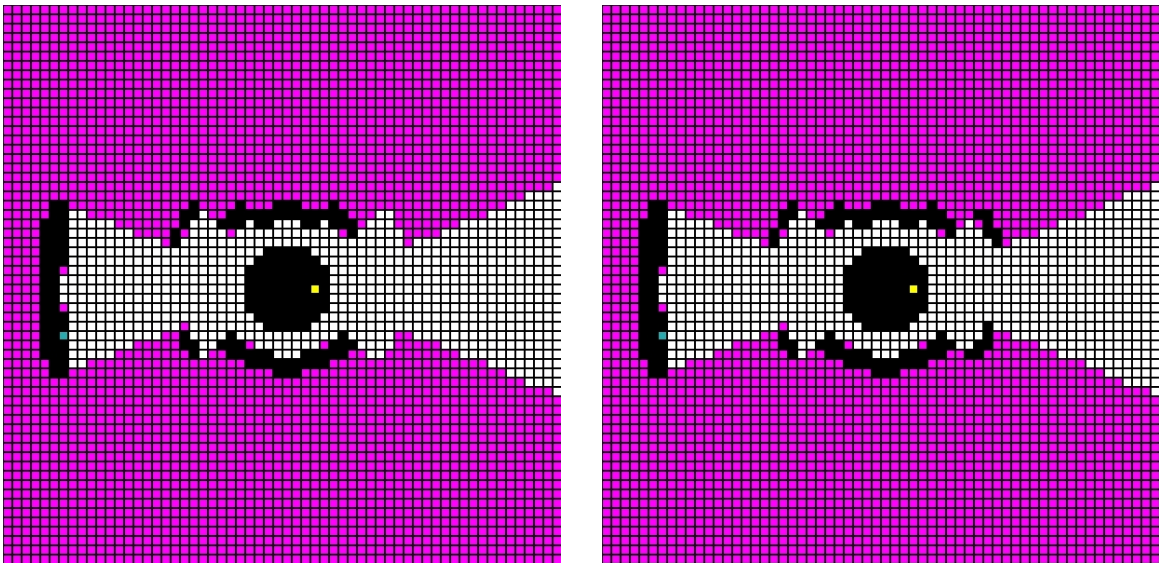

Figura 6.16. Mapa correspondente à situação inicial (figura 6.11b) com informação sensorial a) não limitada b) limitada.

inicial. Pelo contrário, o mapa da figura 6.15a) fica significativamente alterado na parte esquerda e direita em relação ao mapa da figura 6.15b). As diferenças devem-se ao facto de alguns sensores, orientados para os quadrantes Este e Oeste, estarem a fornecer leituras diferentes daquelas que se verificaram na situação da figura 6.15b).

6.2.4 Limites na Informação Sensorial

Na figura 6.16 pode-se observar a diferença entre a utilização da informação sensorial limitada (figura 6.16b) e não limitada (figura 6.16a). As diferenças encontram-se no quadrante Este, uma vez que é neste quadrante que os sensores estão a indicar distâncias de 650 centímetros. Estes valores levarão ao aparecimento de pares de distância (a fornecer à RNA) com valores muito díspares. A consequência deste facto é o aparecimento de zonas livres limitadas por células de cor magenta.

6.3 Ambiente Misto no Simulador

Nesta simulação, o robô é colocado num ambiente misto, com uma zona de espaço aberto e outra onde existe um pequeno corredor.

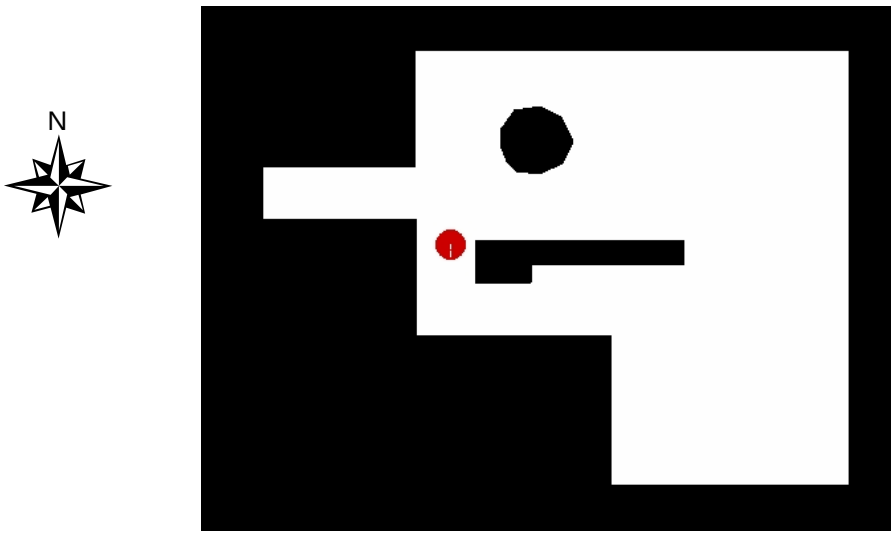

Figura 6.17. Mapa construído no ambiente de simulação.

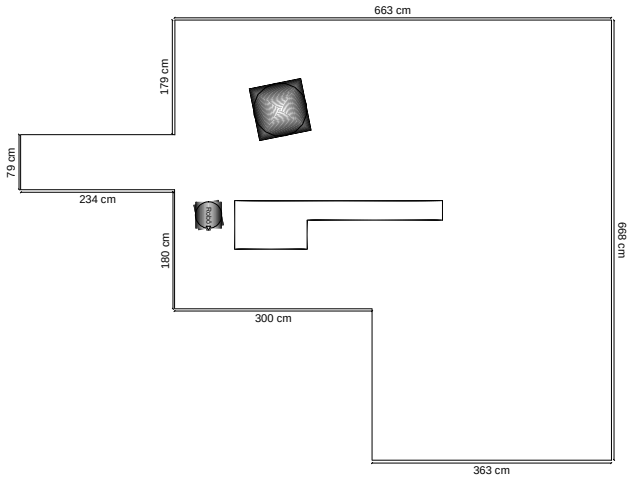

Figura 6.18. Mapa do ambiente de simulação.

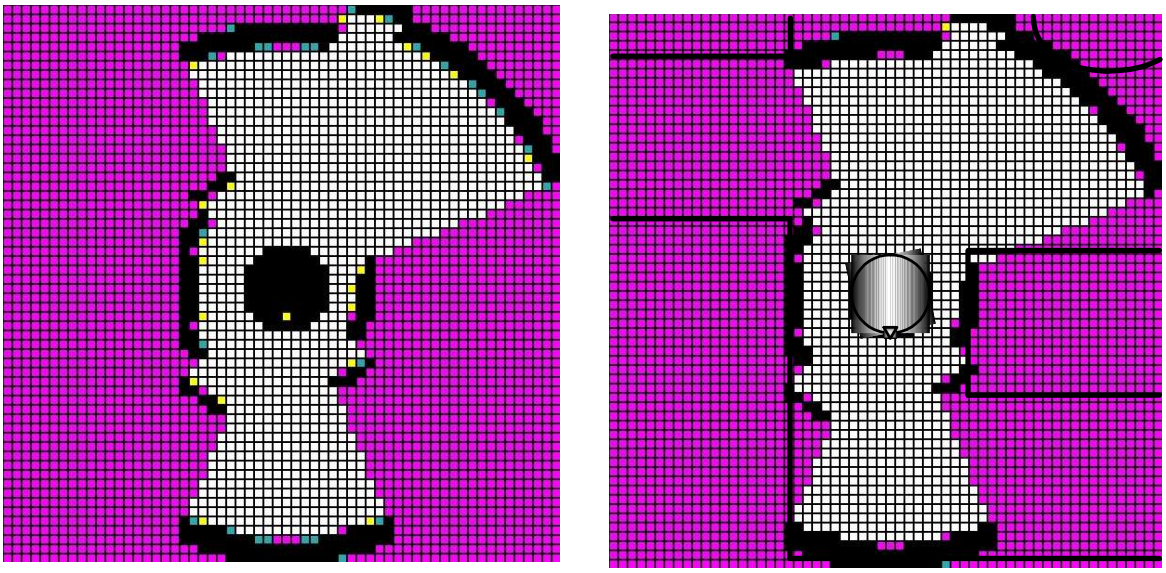

Figura 6.19. a) À esquerda está o mapa na primeira actualização. b) À direita está o mapa após várias actualizações com sobreposição do mapa do simulador.

	Norte			
	105			
	Noroeste	> 150	**Nordeste**	
	35,4		120,2	
	118,9		129,4	
Oeste	83,5	Mapa	9,2	**Este**
25		Simulador		15
27,4		Erro		15,7
2,4	**Sudoeste**		**Sudeste**	0,7
	49,5		28,3	
	49,6	**Sul**	31	
	0,1	110	2,7	
		117,6		
		7,6		

Tabela 6.4. Comparação entre as medidas obtidas através do simulador e as medidas dadas pelo mapa. Os valores estão em centímetros.

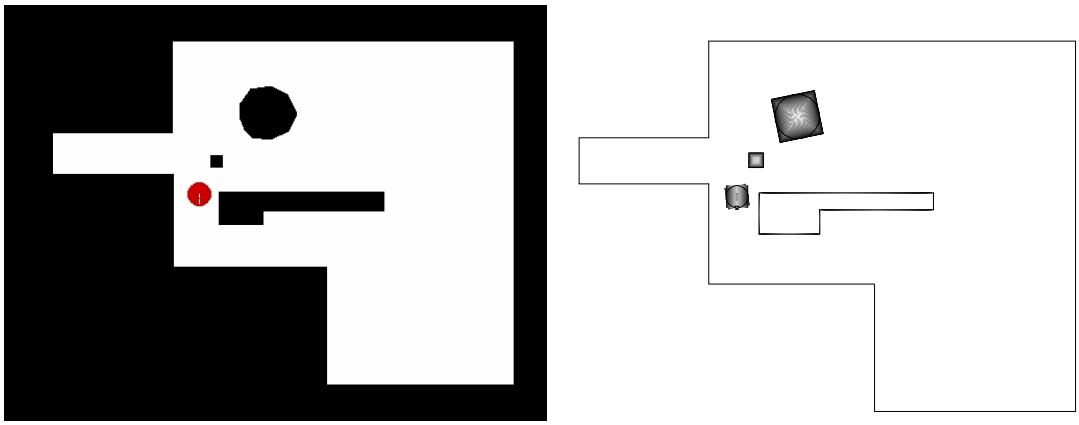

Figura 6.20. Imagem e mapa do ambiente misto em simulação com introdução de um objecto quadrado com 26,5 centímetros de lado.

6.3.1 Robustez e Precisão

Na figura 6.19 pode-se observar o mapa na primeira actualização e após várias actualizações. Neste caso, o mapa sofreu poucas alterações em termos de contornos, todavia um número razoável de células mudou de cor até ao mapa da figura 6.19b).

A tabela 6.4 mostra as medidas obtidas através do simulador e as medidas fornecidas pelo mapa. Da observação da tabela pode-se concluir que os erros são reduzidos na maior parte das orientações. As excepções são as orientações Norte e Noroeste. Também nesta situação os erros devem-se ao facto de existirem sensores com medidas inferiores nessas orientações.

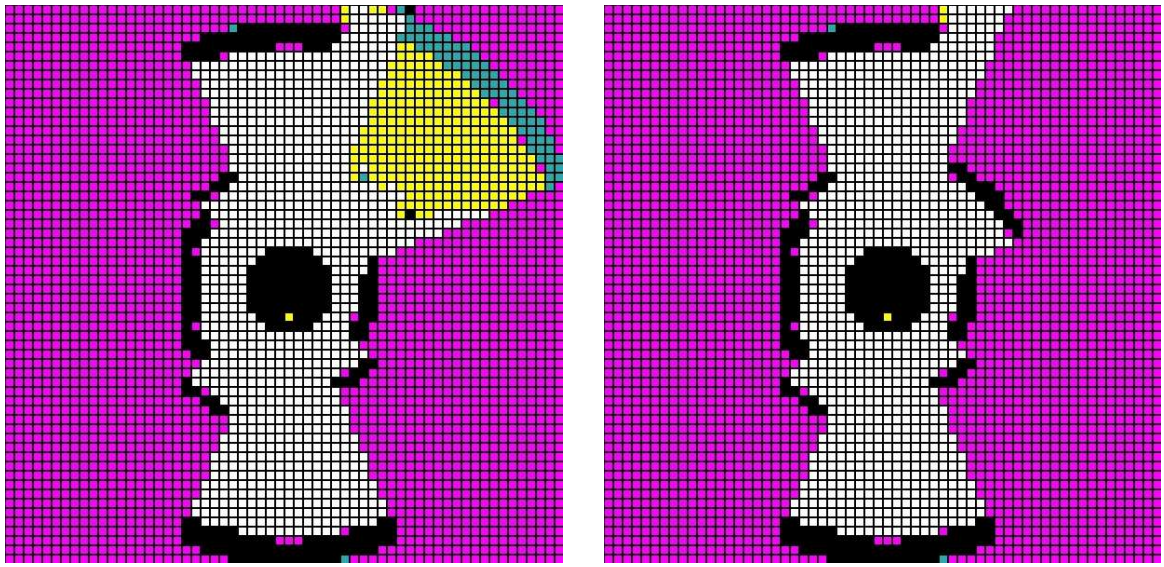

Figura 6.21. a) À esquerda: o mapa após a primeira actualização depois da introdução de um novo objecto. b) À direita: o mapa após 15 actualizações.

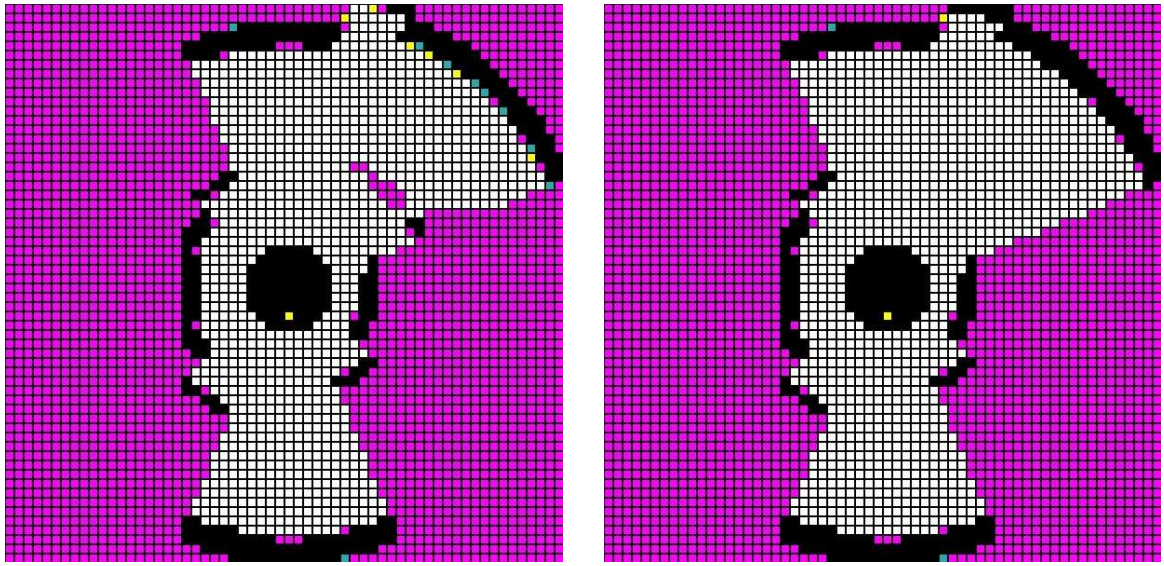

Figura 6.22. a) À esquerda: o mapa após a primeira actualização ao ser retirado o objecto. b) À direita está o mapa após 15 actualizações.

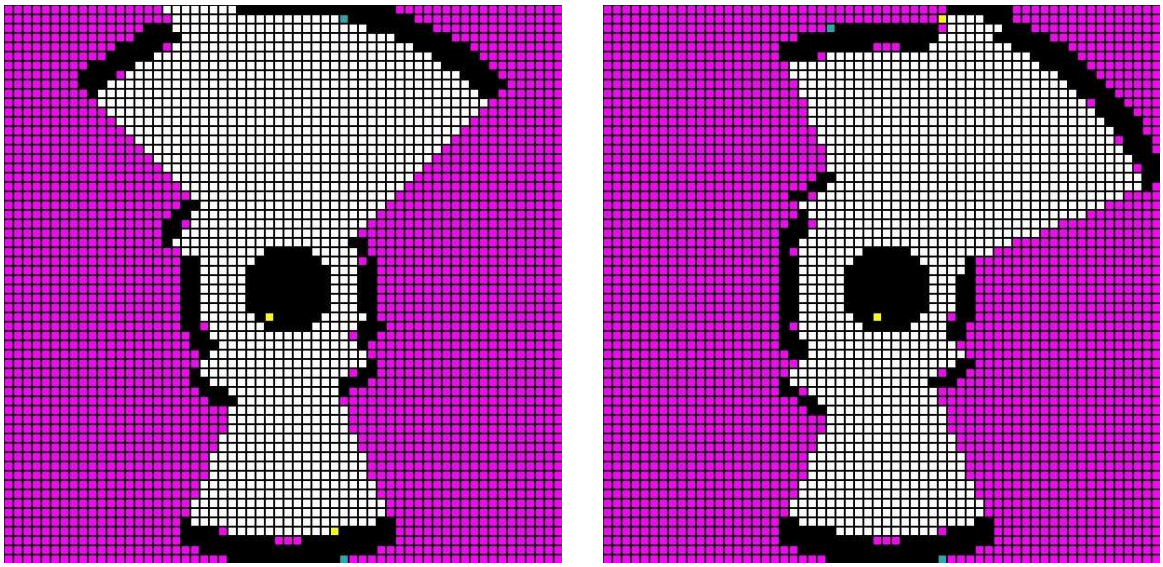

Figura 6.23. a) À esquerda está o mapa depois de uma rotação de 240 graus no sentido anti-horário. b) À direita está o mapa após uma rotação adicional de 7.5 graus no mesmo sentido.

6.3.2 Adaptabilidade

Nesta simulação, o objecto foi colocado na posição indicada na figura 6.20. Na figura 6.21 pode-se observar a reacção do método à chegada do objecto e na figura 6.22 a reacção quando este é retirado. Também nestas situações o método reagiu rápida e correctamente às mudanças do ambiente.

6.3.3 Impacto das Rotações do Robô na Configuração do Mapa

Na figura 6.23 podem-se observar as alterações no mapa com uma diferença de rotação de apenas 7.5 graus. De facto, nesta simulação, uma rotação pequena é suficiente para alterar as leituras dos sensores de uma forma brusca. O impacto desta alteração é particularmente notório no quadrante Norte. No entanto, pode-se assinalar que as distâncias indicadas pelos dois mapas às superfícies mais próximas se mantiveram constantes.

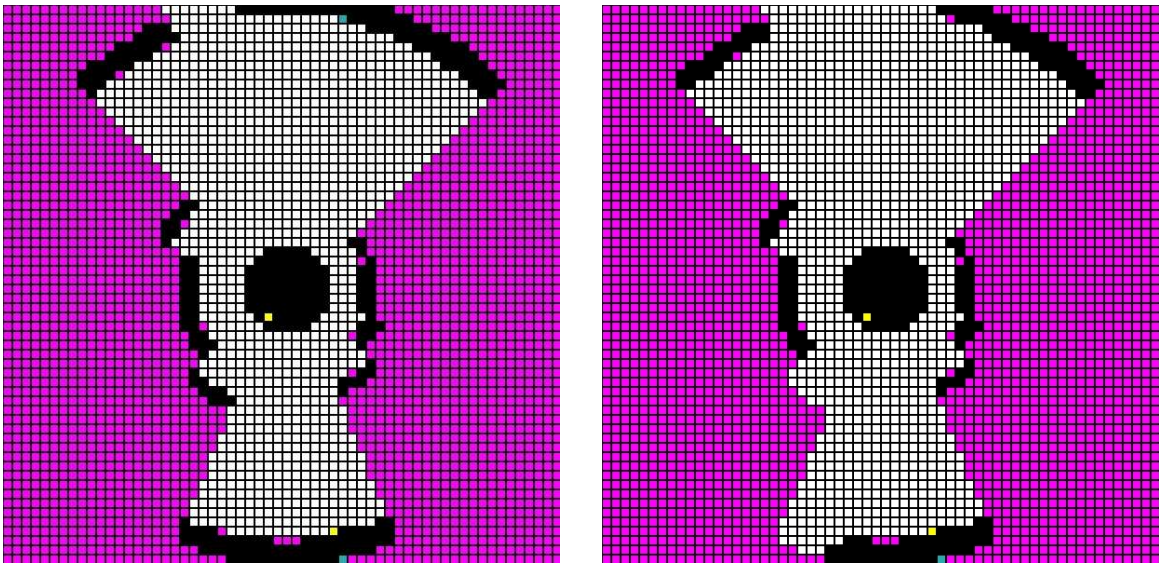

Figura 6.24. a) Mapa depois de uma rotação de 240 graus no sentido anti-horário com informação sensorial a) limitada b) não limitada.

6.3.4 Limites na Informação Sensorial

Na figura 6.24 podem-se observar as diferenças entre a utilização da informação sensorial limitada e não limitada. Neste caso existem três sensores com leituras de distância muito elevadas. Este facto originou o aparecimento de pares de distância com valores muito díspares. É esta a razão do aparecimento de zonas livres limitadas por células de cor magenta (Sudoeste).

6.4 Ambiente Misto Real

No primeiro ambiente real, o robô está num ambiente misto, com uma parte estruturada e outra com um espaço aberto. No espaço aberto foi colocada uma cadeira cujas pernas têm uma dimensão reduzida. A figura 6.25 mostra duas fotografias do ambiente real e a figura 6.26 mostra o mapa do ambiente real.

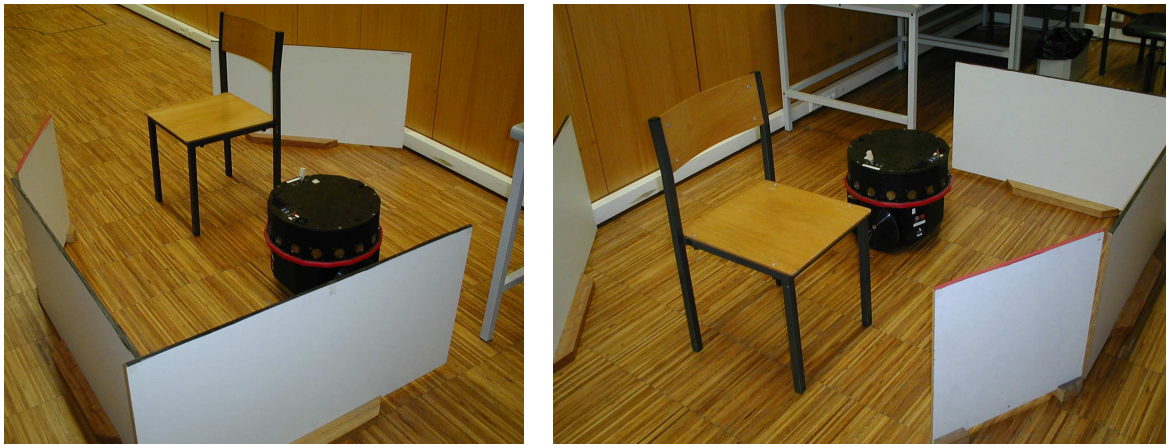

Figura 6.25. Fotografias do ambiente real.

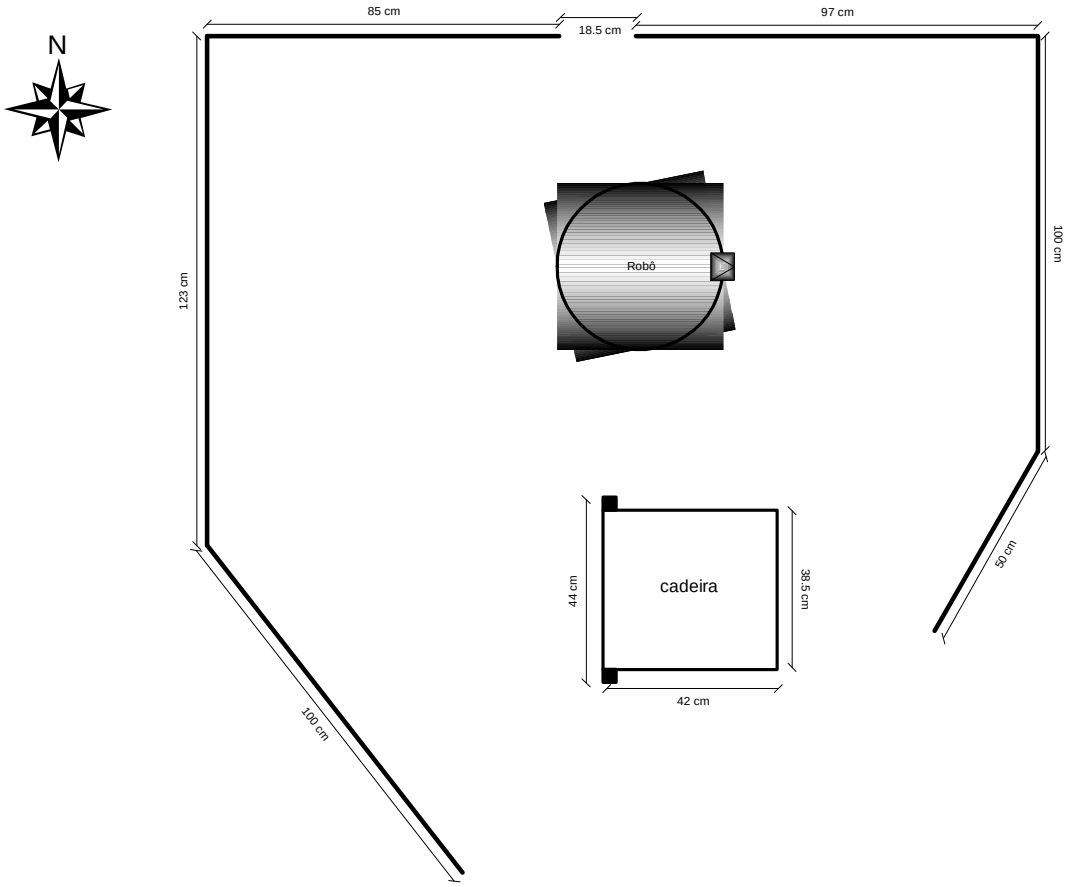

Figura 6.26. Mapa do ambiente real.

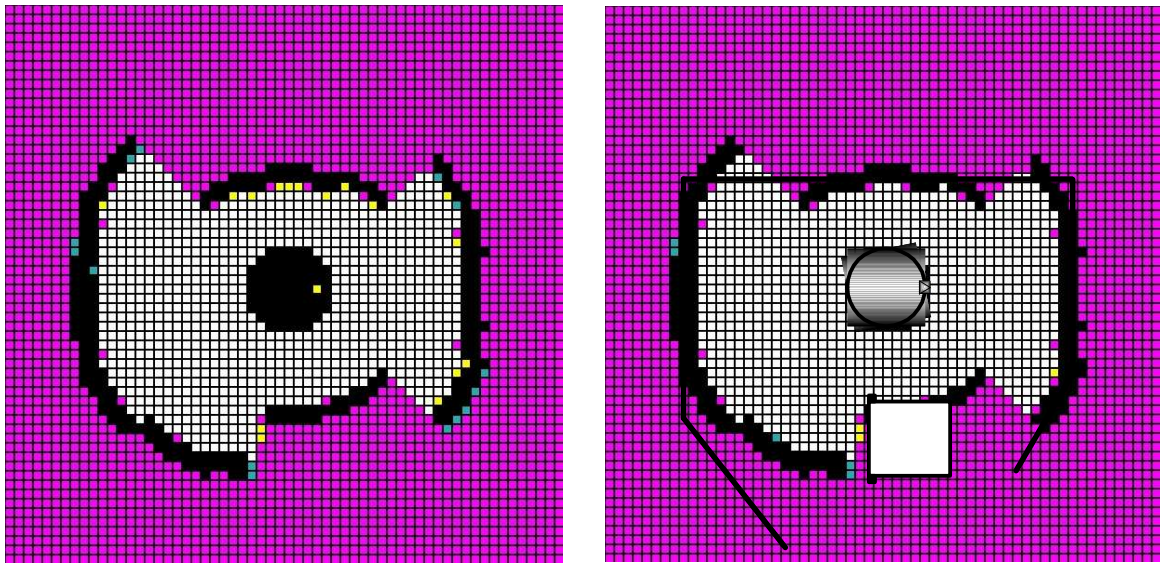

Figura 6.27. À esquerda: mapa relativo à primeira actualização. À direita: mapa após várias actualizações com sobreposição do mapa real.

		Norte		
		35		
	Noroeste	36	**Nordeste**	
	35,4	1,0	42,4	
	51		58	
Oeste	15,6	Mapa	15,6	**Este**
80		Real		70
87		Erro		73,5
7,0	**Sudoeste**		**Sudeste**	3,5
	77,8		42,4	
	103	**Sul**	89	
	25,2	40	46,6	
		38		
		-2,0		

Tabela 6.5. Comparação entre as medidas reais e as medidas dadas pelo mapa. Os valores estão em centímetros.

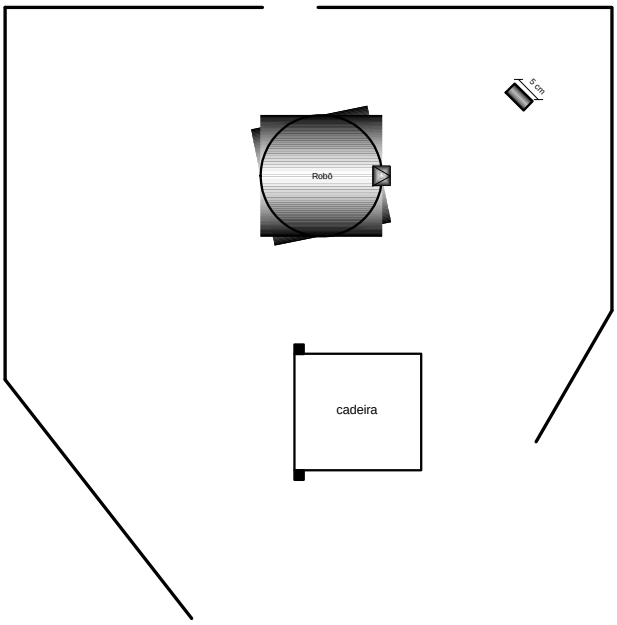

Figura 6.28. Mapa real com um novo objecto.

6.4.1 Robustez e Precisão

Na figura 6.27 podem-se observar os mapas obtidos no robô Super Scout II na primeira actualização e após várias actualizações. Algumas células mudaram de cor mas os contornos do mapa permaneceram inalteráveis.

Na tabela 6.5 são apresentados os erros entre as medidas reais e as medidas dadas pelo mapa. Pode-se observar que as pernas da mesa são detectadas na orientação Noroeste. Em relação à cadeira, a sua detecção deve-se à contribuição de um sensor. Dos três sensores com orientações para o quadrante Sul, apenas um forneceu a distância correcta, os outros não detectaram as pernas da cadeira que estão mais próximas indicando distâncias maiores. Nesta situação, a contribuição dos sensores vizinhos foi determinante para a detecção da distância correcta entre o robô e a cadeira. De um modo geral, os erros não são elevados e os maiores devem-se ao facto de existirem sensores vizinhos, nessas orientações, com leituras de distância inferiores. Note-se que os erros verificados, excepto a Sul, são todos por defeito. O erro por excesso é muito pequeno e deve-se ao facto da cadeira ter características que os sensores do robô, em certas posições e orientações, podem não detectar.

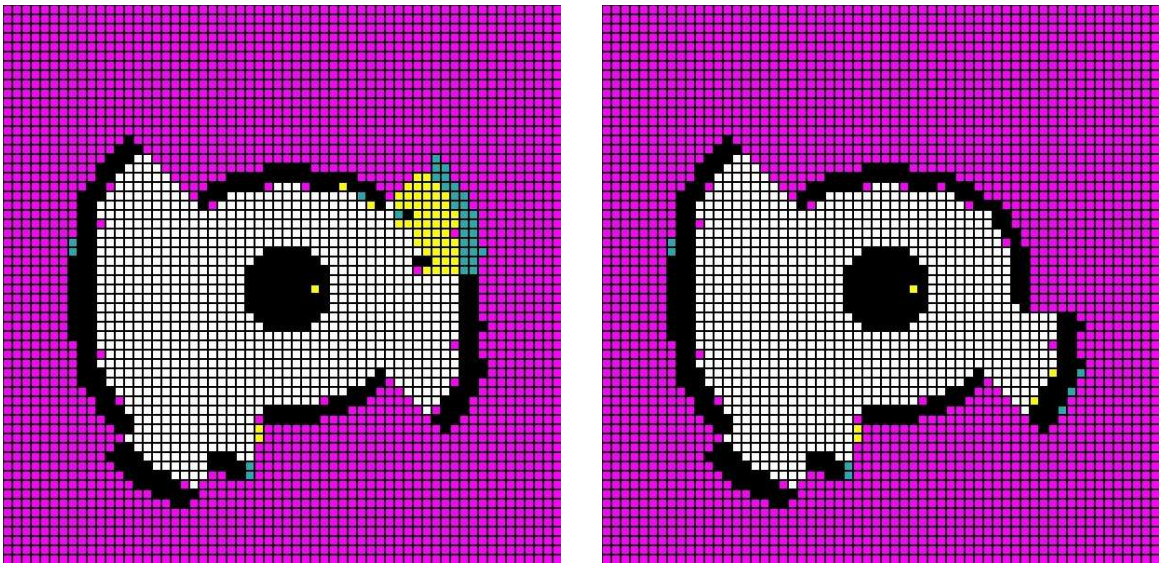

Figura 6.29. a) À esquerda: o mapa após a primeira actualização depois da introdução de um novo objecto. b) À direita: o mapa após 15 actualizações.

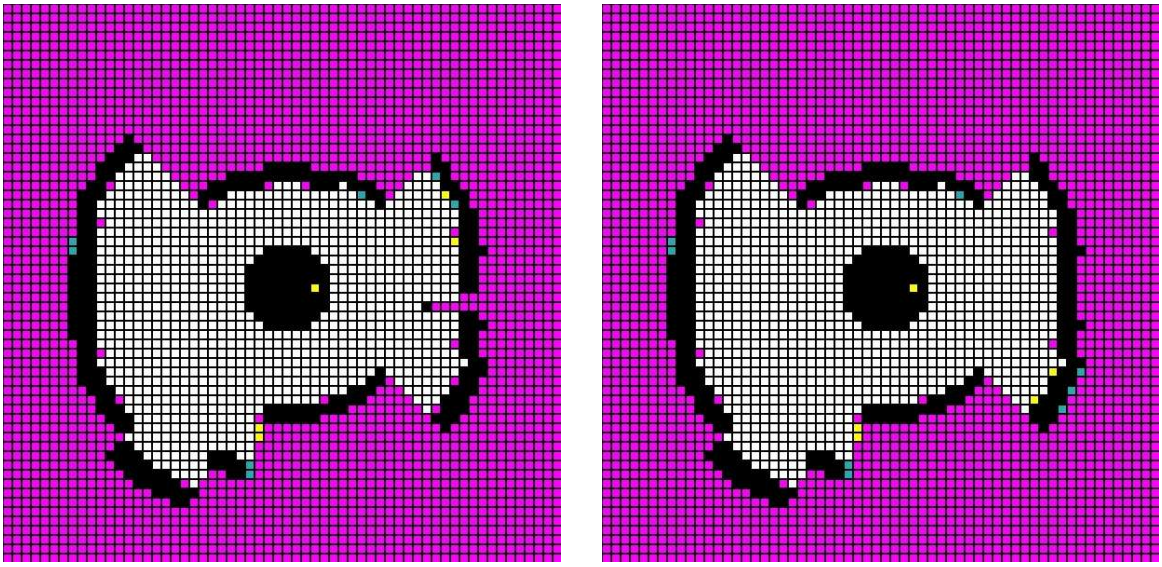

Figura 6.30. a) À esquerda: o mapa após a primeira actualização ao ser retirado o objecto. b) À direita está o mapa após 15 actualizações.

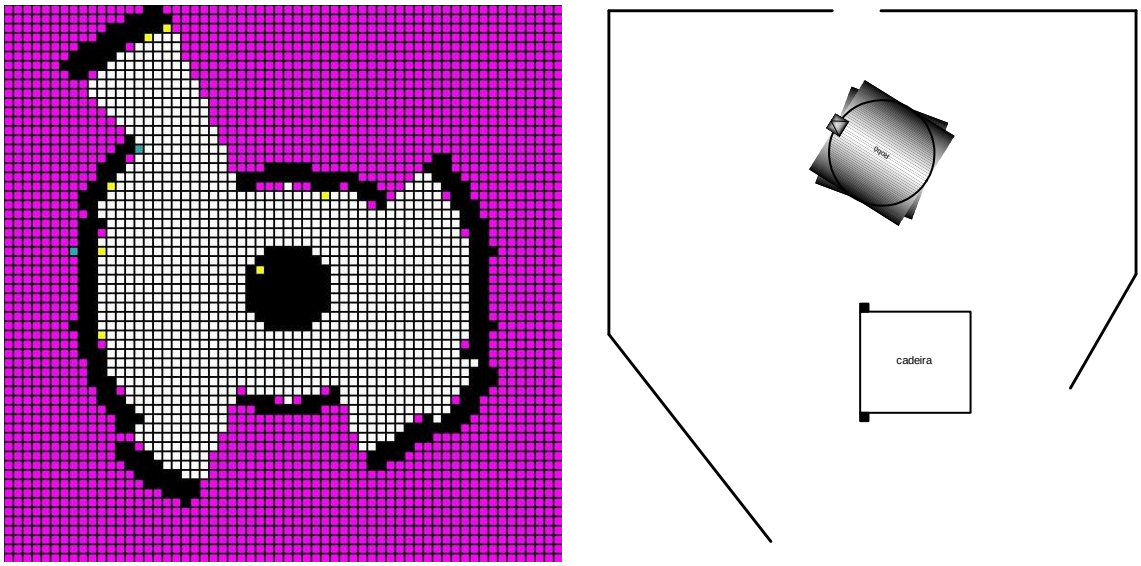

Figura 6.31. O mapa de grelha de células e o mapa real depois do robô efectuar uma rotação de 148 graus no sentido anti-horário, um deslocamento de 3 centímetros para Oeste e 2 centímetros para Norte, em relação à posição no mapa real da figura 6.26.

6.4.2 Adaptabilidade

Em relação à adaptabilidade, o comportamento do método no robô real é semelhante ao comportamento verificado durante as simulações. Consequentemente, a colocação do objecto, como indica a figura 6.28, foi rapidamente detectado pelo método (figura 6.29a). A reacção do método ao ser retirado o objecto foi muito boa estabilizando o mapa logo nas primeiras actualizações (figura 6.30).

6.4.3 Impacto das Rotações do Robô na Configuração do Mapa

Nesta experiência, o robô além de efectuar rotações efectuou também movimentos de translação. Uma vez que existe uma cadeira nas imediações do robô, as suas pernas ou as pernas da mesa, a Noroeste, podem não ser sempre detectadas pelos sonares. Este aspecto é particularmente notório no mapa da figura 6.31 onde as pernas da mesa (Noroeste) deixaram de ser detectadas. Note-se que o robô apenas se deslocou alguns centímetros. Outra diferença

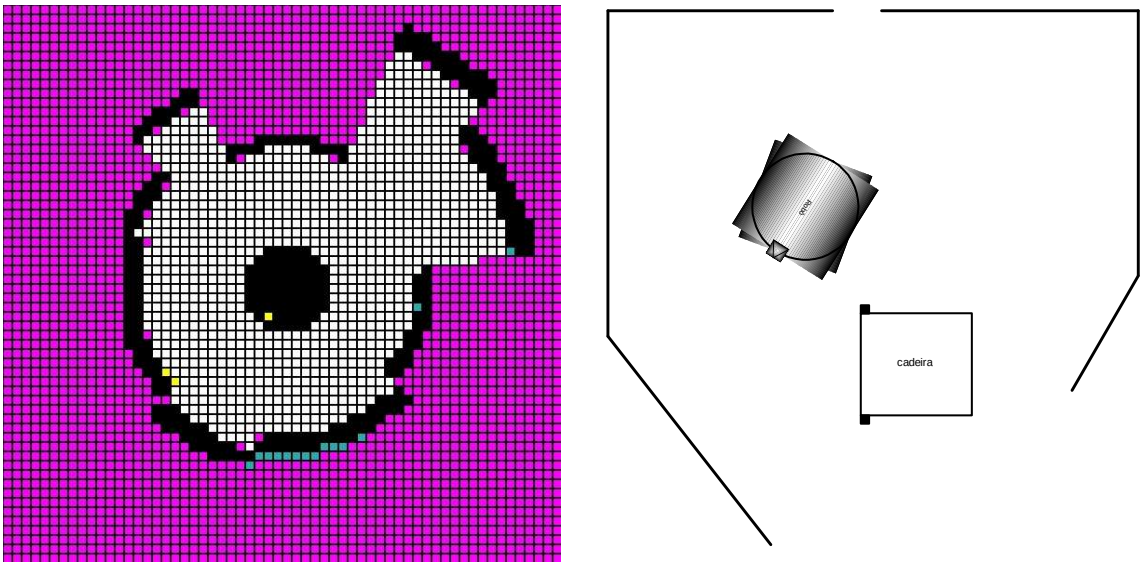

Figura 6.32. O mapa de grelha de células e o mapa real depois do robô ter efectuado uma rotação de 90 graus no sentido anti-horário, um deslocamento de 29 centímetros para Oeste e 19 centímetros para Sul a partir da posição da figura 6.31.

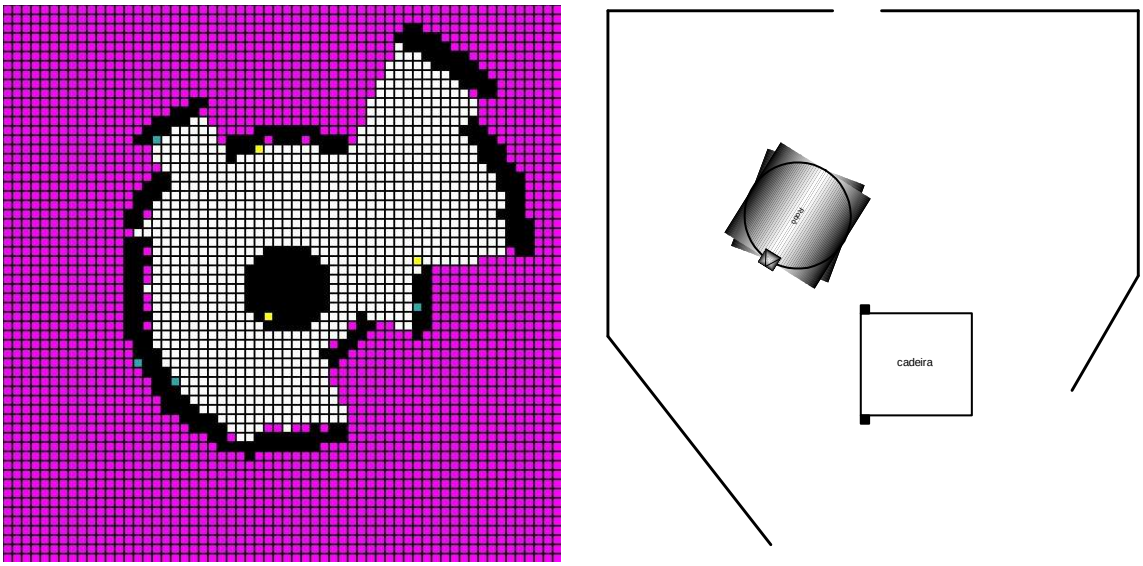

Figura 6.33. O mapa de grelha de células e o mapa real depois do robô ter efectuado uma rotação de 2 graus no sentido anti-horário, um deslocamento de 1 centímetro para Oeste e 1 centímetro para Sul a partir da posição da figura 6.32.

que se pode observar é o aumento de precisão na direcção Sudeste.

Com o objectivo de estudar o comportamento do mapa na detecção das pernas da cadeira aproximou-se o robô da mesma, como indica a figura 6.32. Nesta situação nenhum sensor detecta a perna da cadeira mais próxima que se encontra a cerca de 15 centímetros do robô. Porém, ao avançar ligeiramente o robô (figura 6.33) a referida perna da cadeira já é detectada. Em relação às restantes orientações onde existem superfícies verticais não se verificaram grandes diferenças, uma vez que possuem características de um ambiente com uma estrutura mais detectável pelos sonares.

Pode-se referir a título de exemplo as mudanças nas medidas na direcção Oeste. Desde a situação inicial cuja distância dada pelo mapa era de 80 centímetros e a medida real era de 87 centímetros (tabela 6.5), o robô efectuou, no total, um deslocamento para Oeste de 33 centímetros. Na última situação (figura 6.33), a medida dada pelo mapa é de 55 centímetros, coerente portanto com o deslocamento efectuado (87-33 = 54 centímetros). A medida real na situação da figura 6.33, a Oeste, é de 54 centímetros o que certifica a validade do mapa.

6.5 Ambiente Misto Real com Objectos Estreitos

O segundo ambiente real é semelhante ao anterior. Neste ambiente, a cadeira foi retirada e foram colocados dois objectos de madeira perto do robô. Estes objectos estão colocados na vertical e têm uma altura de 50 centímetros e uma área de 10 centímetros quadrados. A figura 6.34 mostra duas fotografias do ambiente real e a figura 6.35 mostra o mapa do ambiente real.

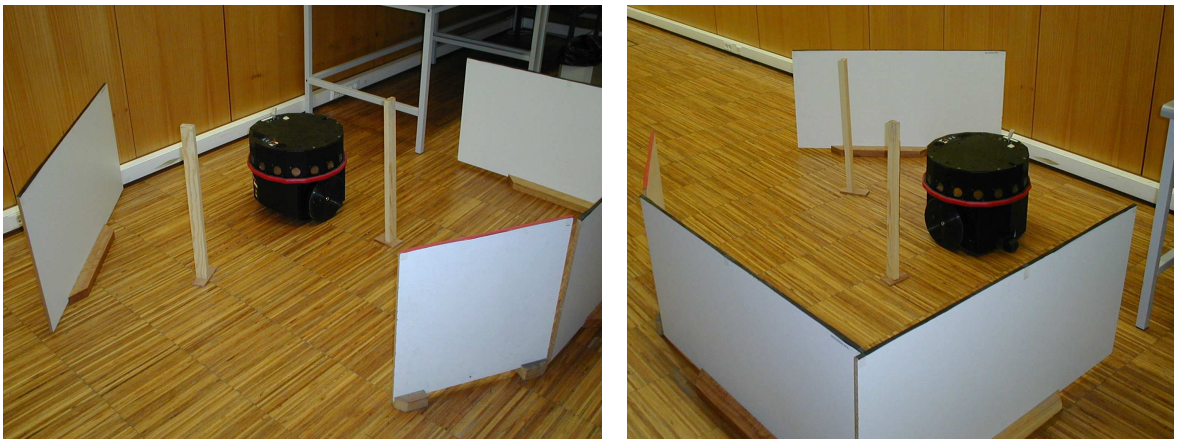

Figura 6.34. Fotografias do ambiente real.

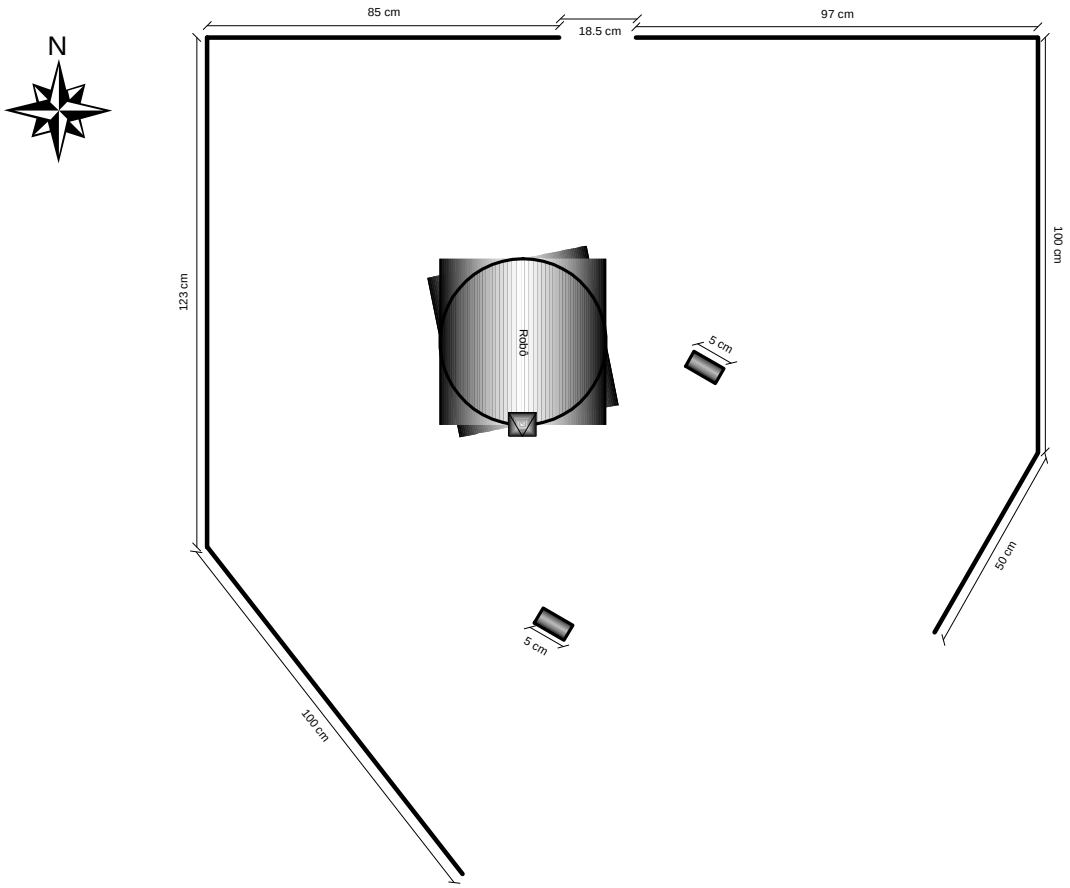

Figura 6.35. Mapa do ambiente real.

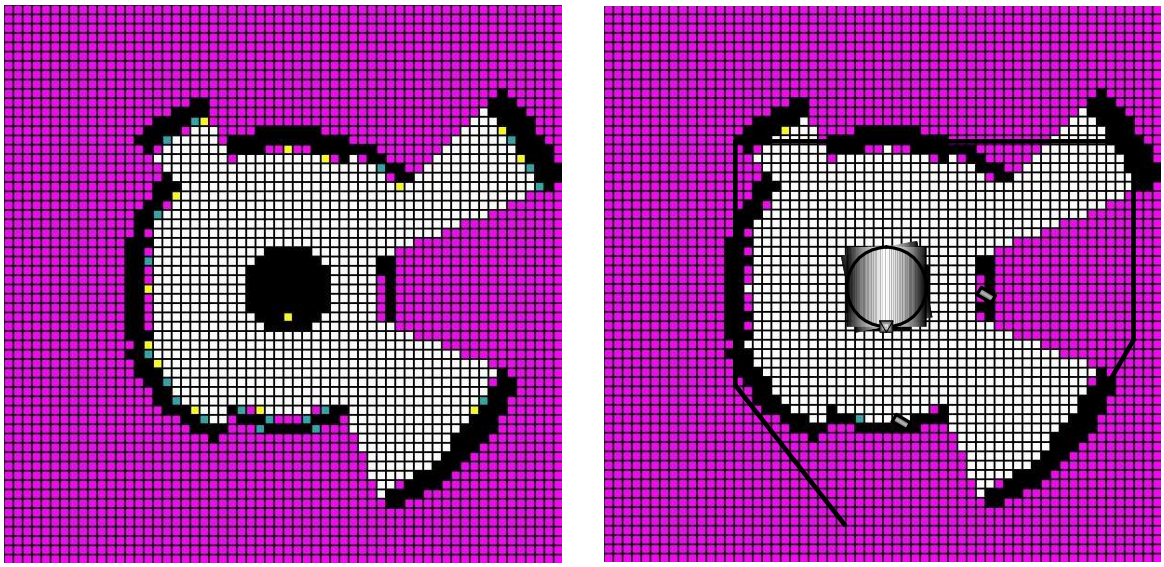

Figura 6.36. À esquerda: mapa na primeira actualização. À direita: mapa após várias actualizações com sobreposição do mapa real.

		Norte		
		55		
	Noroeste	55	**Nordeste**	
	77,8	0,0	56,6	
	82		82	
Oeste	4,2	Mapa	25,4	**Este**
55		Real		25
56		Erro		23,5
1,0	**Sudoeste**		**Sudeste**	-1,5
	56,6		99	
	61	**Sul**	> 150	
	4,4	50		
		46		
		-4,0		

Tabela 6.6. Comparação entre as medidas reais e as medidas dadas pelo mapa. Os valores estão em centímetros.

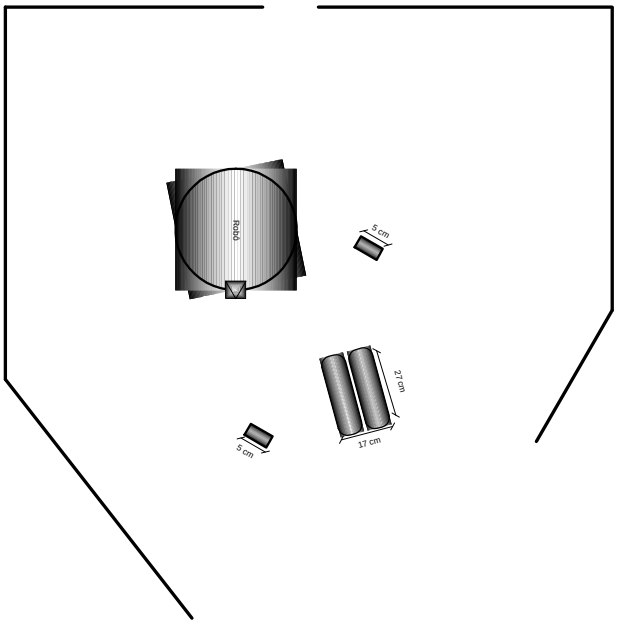

Figura 6.37. Mapa real indicando a presença de uma pessoa.

6.5.1 Robustez e Precisão

Na figura 6.36 podem-se observar os mapas obtidos no robô Super Scout II na primeira actualização e após várias actualizações. Apesar de algumas células mudarem de cor os contornos do mapa permaneceram inalteráveis.

Na tabela 6.6 são apresentados os erros entre as medidas reais e as medidas dadas pelo mapa. Como se pode observar os erros são reduzidos. A Nordeste o erro é maior porque existem sensores vizinhos com leituras de distância menores que influenciam o mapa nessa direcção. Os erros por excesso (Este e Sul) são provocados pela diferença dos valores fornecidos pelos sensores nessas direcções que é elevada. Isto acontece porque os objectos são estreitos e apenas um dos sensores os detecta. No entanto, os valores dos erros são muito baixos.

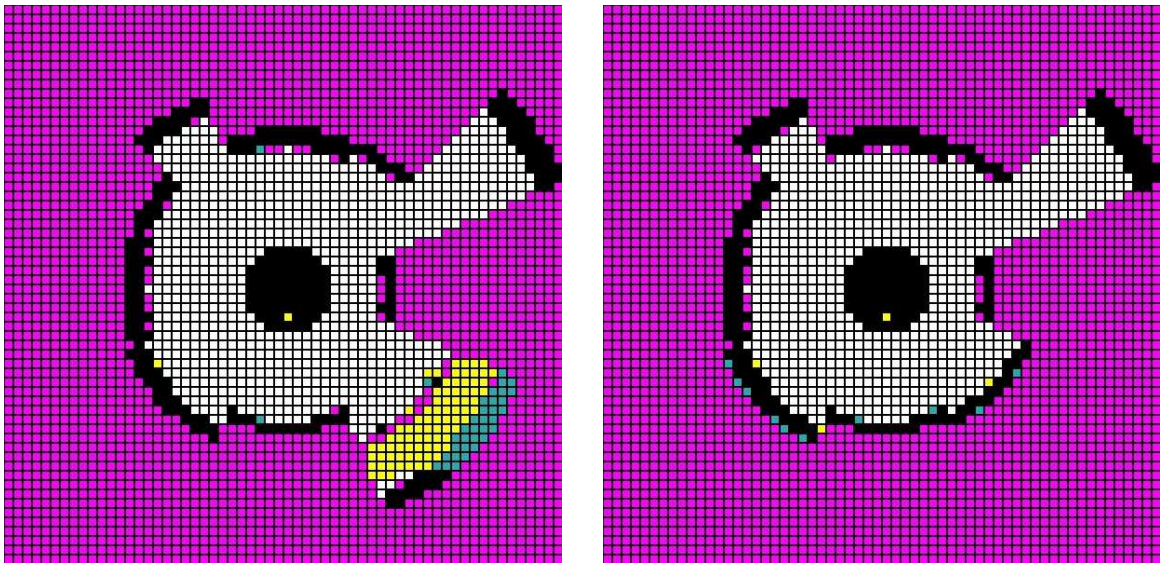

Figura 6.38. a) À esquerda está o mapa após a primeira actualização depois de chegar uma pessoa na orientação Sudeste. b) À direita está o mapa após 15 actualizações.

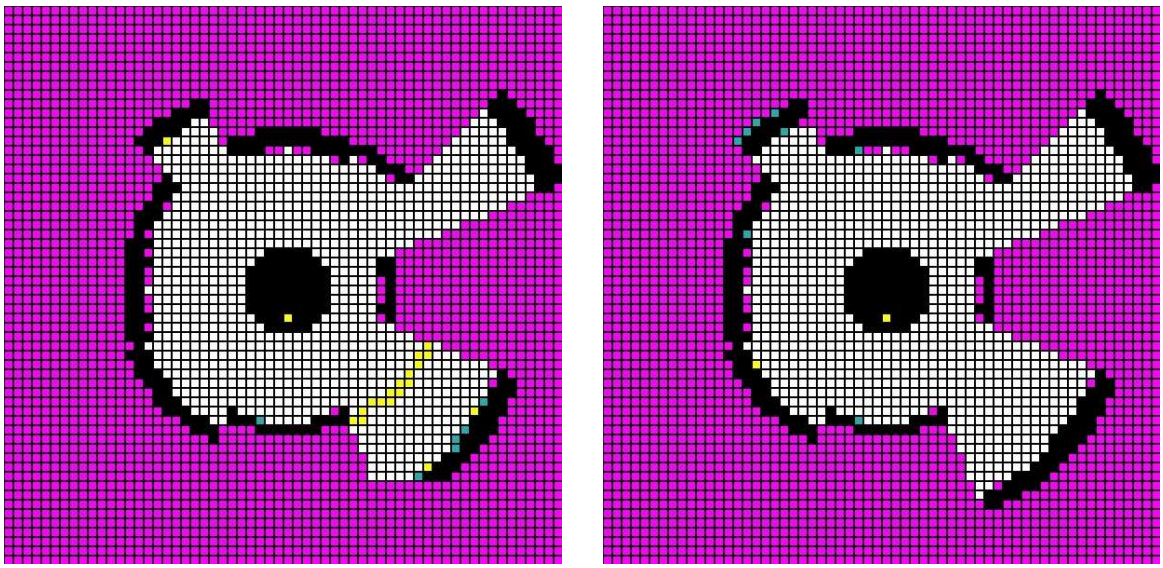

Figura 6.39. a) À esquerda está o mapa após a primeira actualização depois da pessoa se ter retirado. b) À direita está o mapa após 15 actualizações.

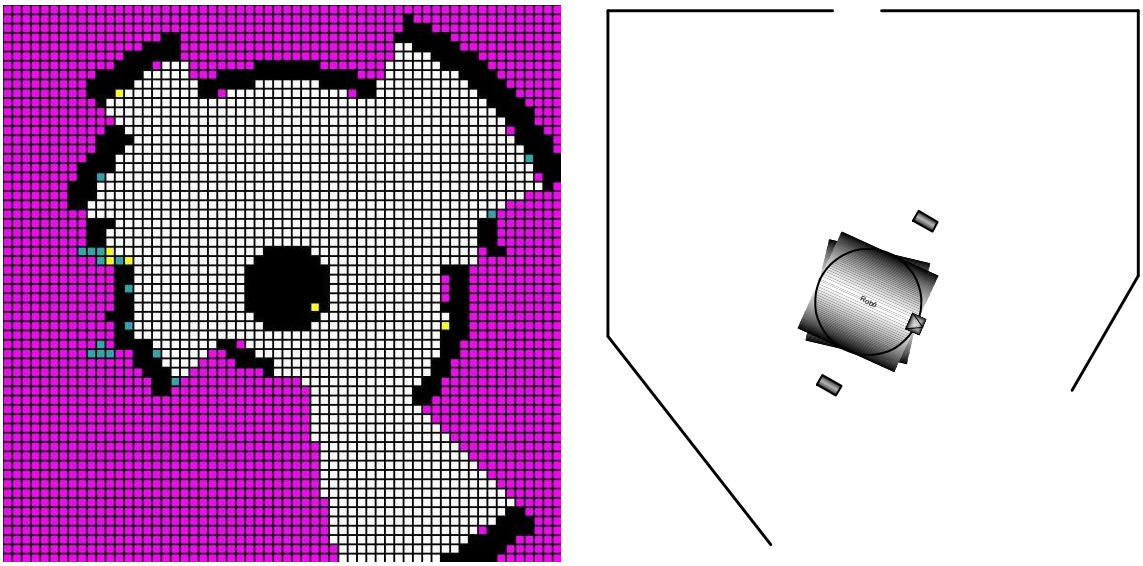

Figura 6.40. O mapa de grelha de células e o mapa real depois do robô ter efectuado uma rotação de 66 graus no sentido anti-horário, um deslocamento de 25 centímetros para Este e 5 centímetros para Sul, em relação à posição no mapa real da figura 6.35.

6.5.2 Adaptabilidade

Para observar o comportamento do método perante alterações do ambiente, uma pessoa aproximou-se do robô (figura 6.37). Na figura 6.38 pode-se observar a rápida reacção do método à chegada da pessoa. Da mesma forma, quando a pessoa se retirou, o mapa volta à configuração que tinha antes da chegada da pessoa, logo nas primeiras actualizações.

6.5.3 Impacto das Rotações do Robô na Configuração do Mapa

Nesta experiência, além das rotações também foram efectuados movimentos de translação. Na figura 6.40 pode-se observar uma situação em que, apesar do robô estar entre os objectos, no mapa não aparece a localização de um deles, nomeadamente aquele que se encontra a Nordeste. Pode-se salientar ainda, apesar da utilização da informação sensorial limitada, a existência de uma zona livre limitada por células de cor magenta no quadrante Sul. Este facto surge como consequência da disparidade das leituras nessa orientação.

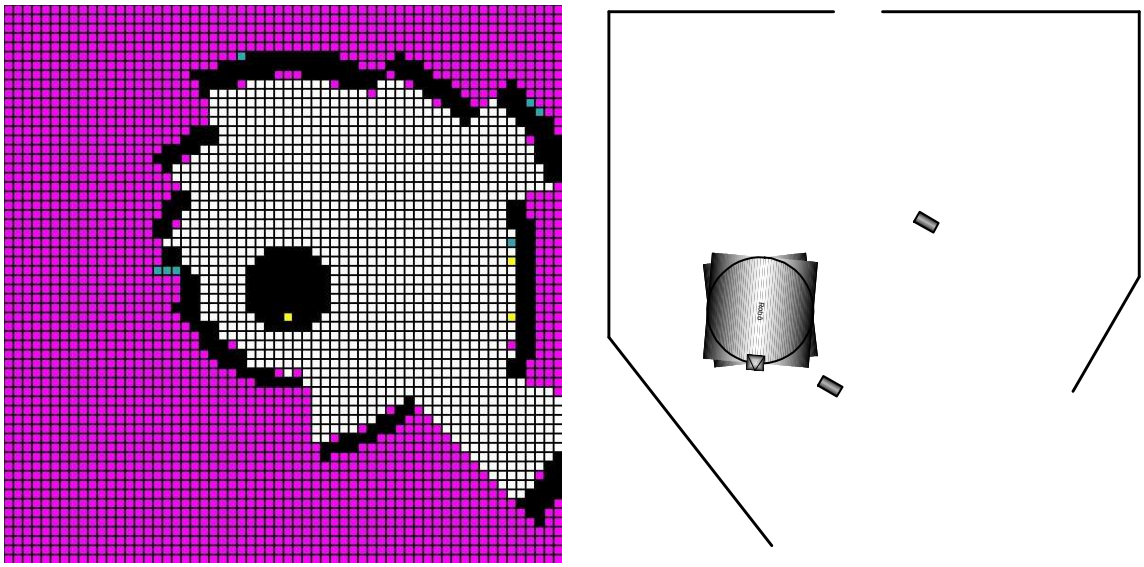

Figura 6.41. O mapa de grelha de células e o mapa real depois do robô ter efectuado uma rotação de 71 graus no sentido horário, um deslocamento de 43 centímetros para Oeste e 5 centímetros para Sul, em relação à posição da figura 6.40. Neste caso, o objecto mais próximo do robô não é detectado.

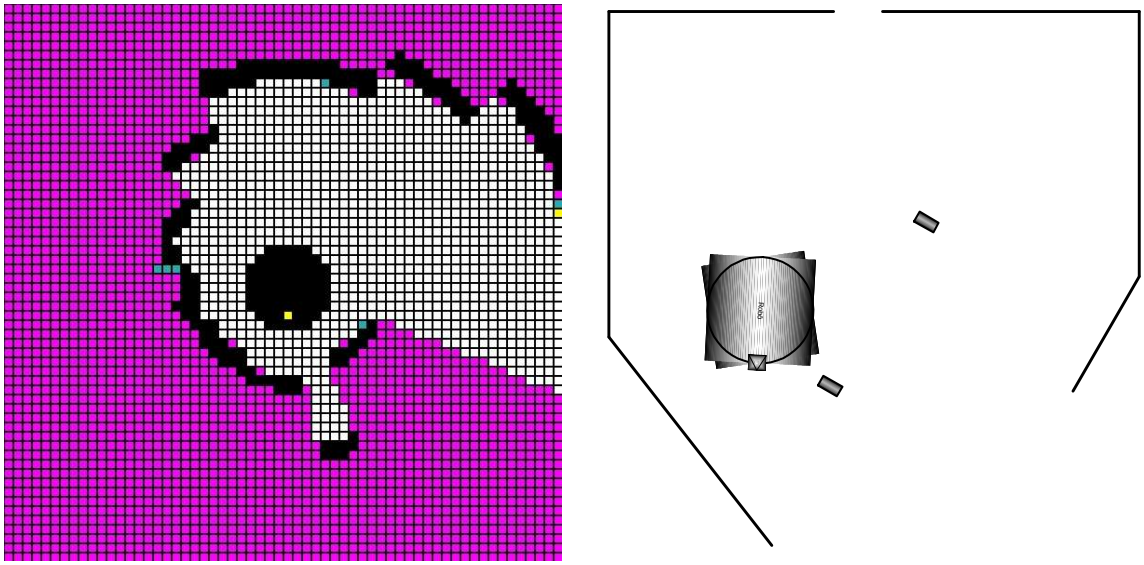

Figura 6.42. O mapa de grelha de células e o mapa real depois do robô ter efectuado uma rotação de 2 graus no sentido anti-horário, em relação à posição da figura 6.41. Agora o objecto já é detectado.

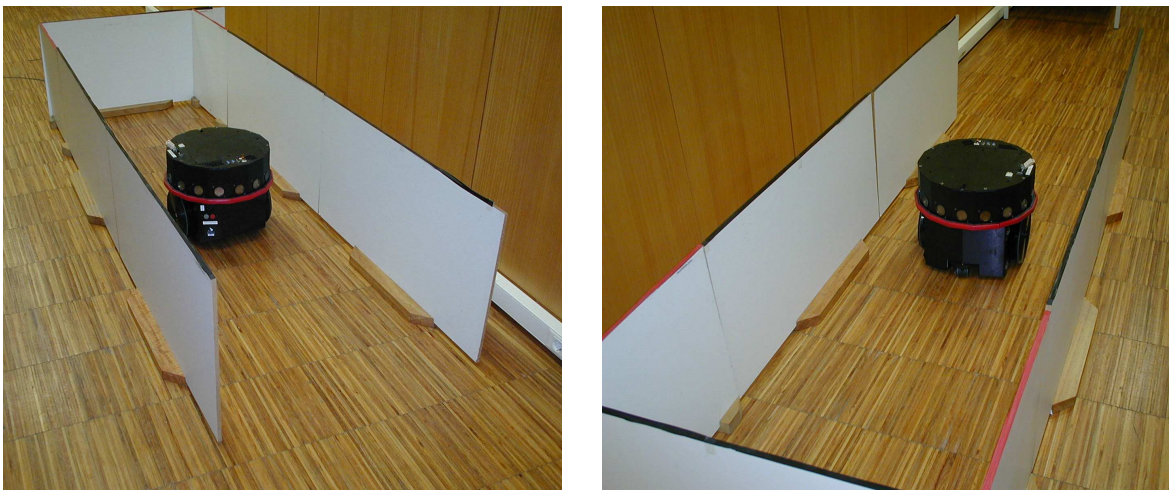

Figura 6.43. Fotografias do ambiente real.

As figuras 6.41 e 6.42 mostram um exemplo de uma experiência onde um dos objectos deixou de ser detectado rodando o robô apenas dois graus.

Também nesta experiência se pode observar a precisão do mapa na direcção Norte desde o início até ao fim da experiência. No início, o mapa indicava uma medida de 55 centímetros e a medida real era de 55 centímetros (tabela 6.6). A seguir ao último movimento, a distância real a Norte é de 92 centímetros e o mapa indica 90 centímetros, ou seja com um erro muito reduzido.

6.6 Corredor Real

Neste ambiente real, o robô é colocado num corredor estreito, aberto numa das extremidades (semelhante ao ambiente da segunda simulação). A figura 6.43 mostra duas fotografias do corredor e a figura 6.44 mostra o mapa do ambiente real.

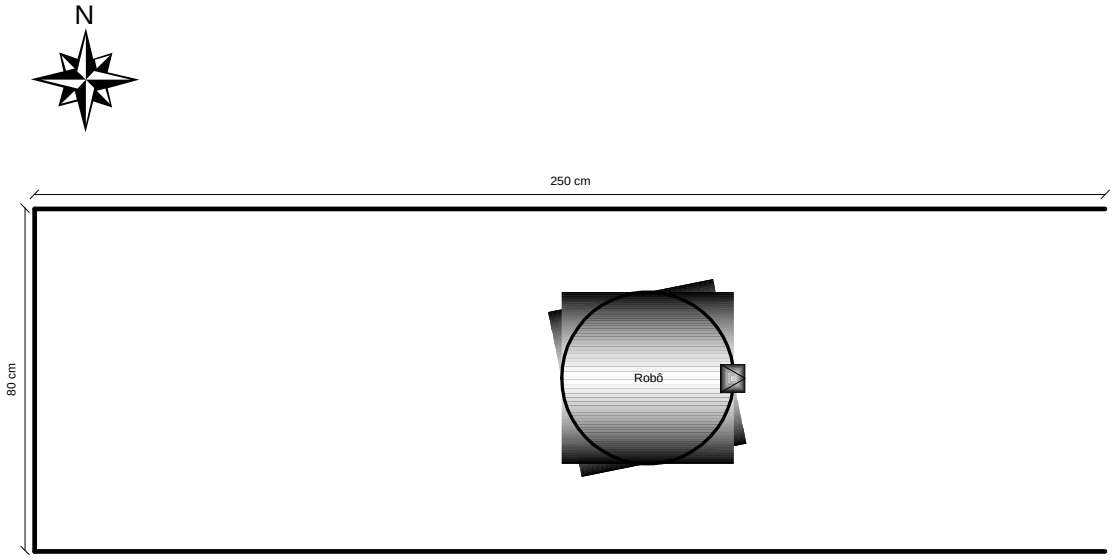

Figura 6.44. Mapa do ambiente real.

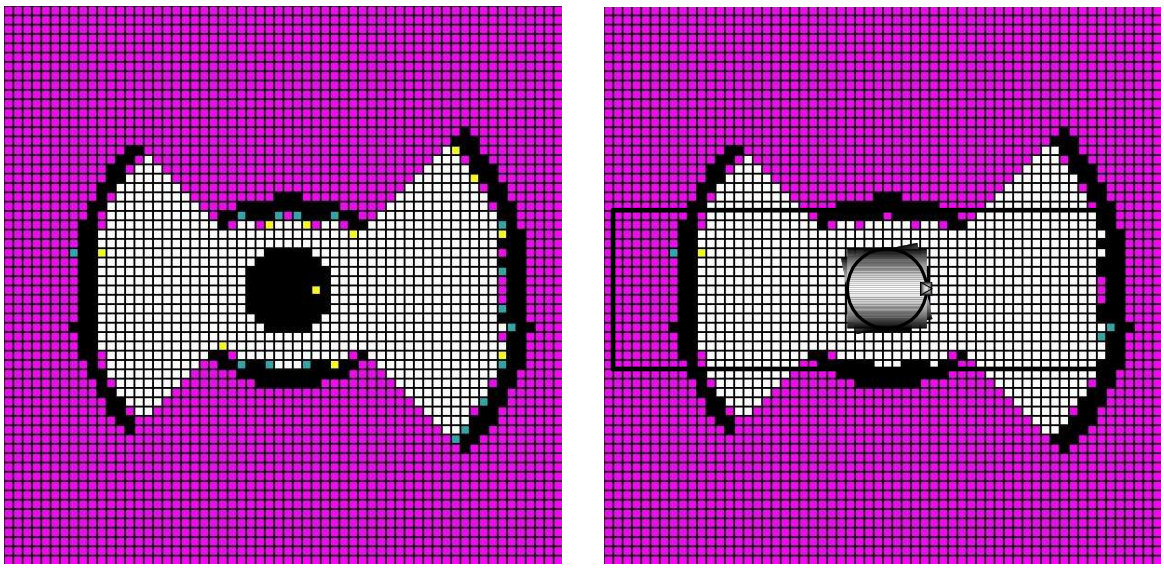

Figura 6.45. À esquerda: mapa relativo à primeira actualização. À direita: mapa após várias actualizações com sobreposição do mapa real.

		Norte		
		15		
	Noroeste	19	**Nordeste**	
	21,2	4,0	21,2	
	36		38	
Oeste	14,8	Mapa	16,8	**Este**
80		Real		90
123		Erro		> 150
43,0	**Sudoeste**		**Sudeste**	
	21,2		21,2	
	37	**Sul**	38	
	15,8	20	16,8	
		22		
		2,0		

Tabela 6.7. Comparação entre as medidas reais e as medidas dadas pelo mapa. Os valores estão em centímetros.

6.6.1 Robustez e Precisão

A figura 6.45 mostra os mapas na primeira actualização e após várias actualizações. Como se pode observar, o mapa sofreu poucas alterações.

Na tabela 6.8 são apresentados os erros entre as medidas reais e as medidas dadas pelo mapa. As características deste ambiente levam ao aparecimento, em certas orientações, de erros consideráveis, uma vez que existem leituras de distância muito diferentes entre sensores vizinhos, como por exemplo nas orientações Este e Oeste. A Norte e a Sul os erros são reduzidos.

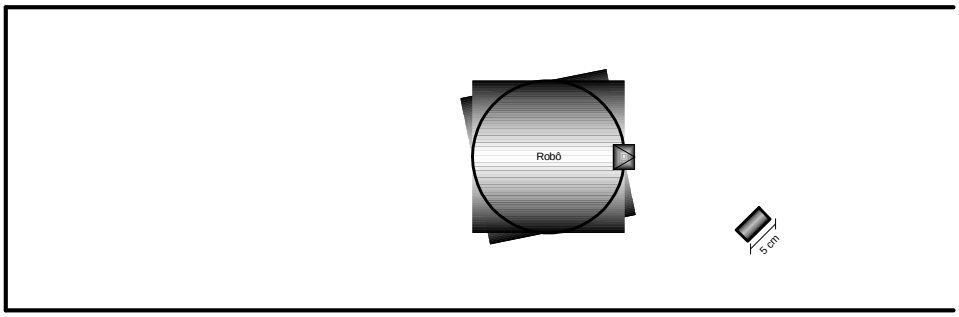

Figura 6.46. Mapa real com um novo objecto.

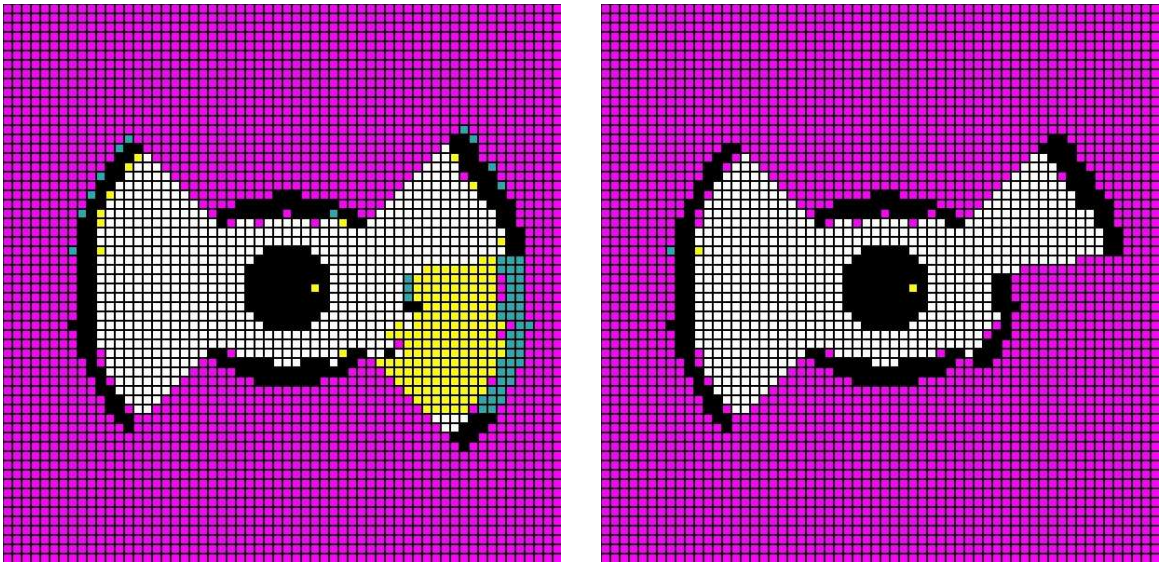

Figura 6.47. a) À esquerda: o mapa após a primeira actualização depois da introdução de um novo objecto. b) À direita: o mapa após 15 actualizações.

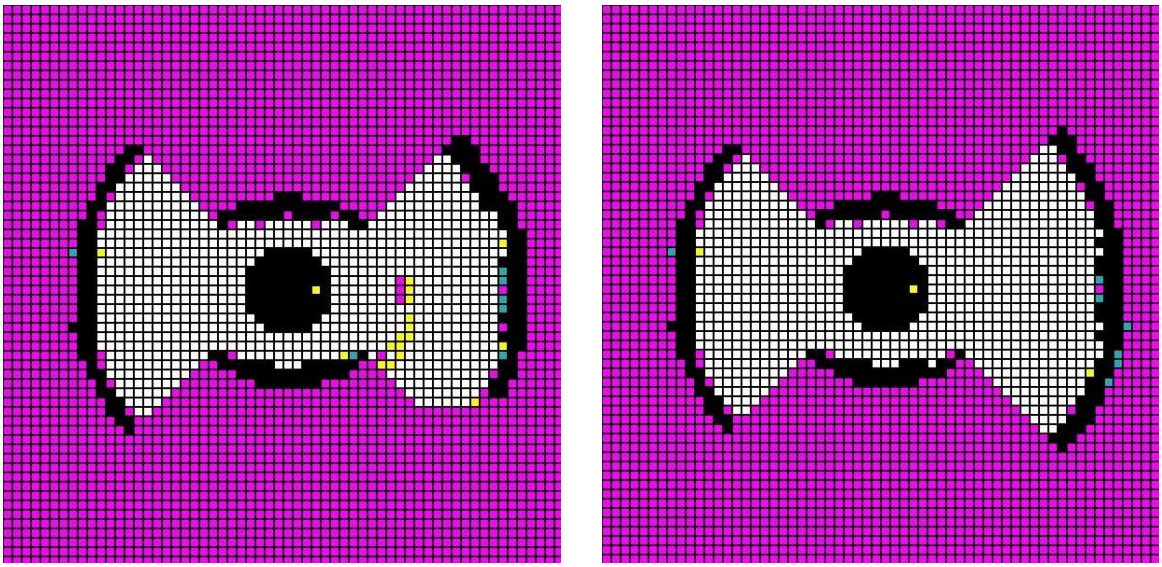

Figura 6.48. a) À esquerda: o mapa após a primeira actualização ao ser retirado o objecto. b) À direita está o mapa após 15 actualizações.

6.6.2 Adaptabilidade

Nesta experiência foi colocado um objecto perto do robô (figura 6.46). Como se pode observar pelas figuras 6.47 e 6.48, o mapa adaptou-se rapidamente às mudanças verificadas no ambiente.

6.6.3 Impacto das Rotações do Robô na Configuração do Mapa

Nesta experiência (figura 6.49), as rotações podem ter um impacto elevado na configuração do mapa. Nas situações da figura 6.49 (em ambos os mapas), nove dos dezasseis sensores do robô (mais de 50% dos sensores) apresentam leituras de distância acima dos 122 centímetros, por conseguinte aparecem zonas livres de maior dimensão. No entanto, as distâncias dadas pelo mapa às superfícies verticais mais próximas do robô (Norte e Sul) mantêm-se constantes.

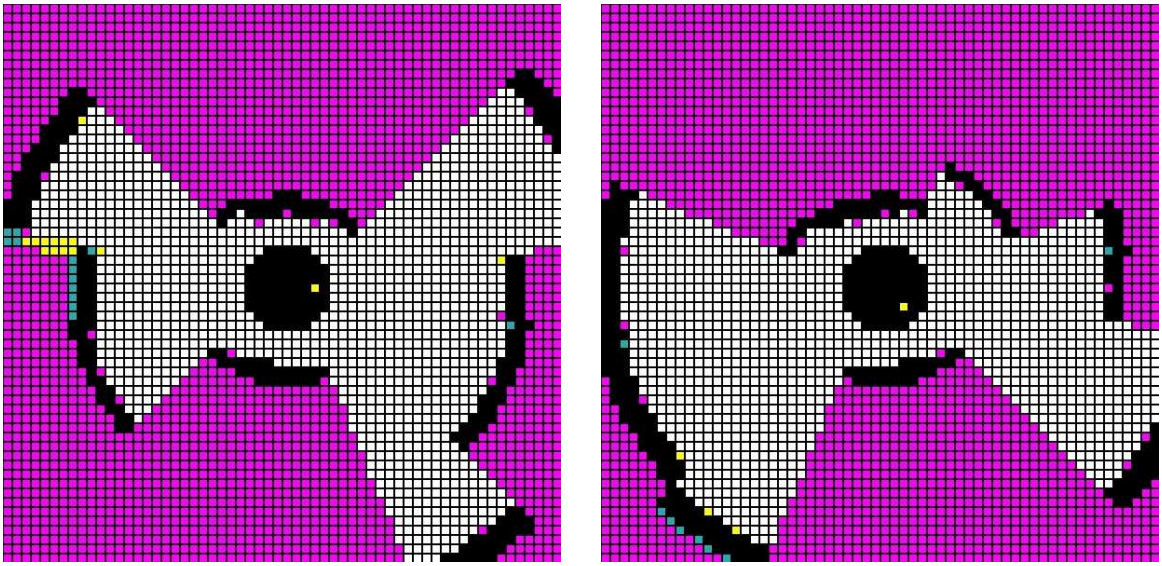

Figura 6.49. O mapa depois de uma rotação de a) 6 graus b) 308 graus relação à origem do sistema de referência.

Figura 6.50. O mapa de grelha de células após o robô efectuar uma rotação de 308 graus (figura 6.49) com informação sensorial a) limitada b) não limitada.

6.6.4 Limites na Informação Sensorial

Para exemplificar o efeito da utilização de limites na informação sensorial obtiveram-se os dois mapas da figura 6.50. Sem a utilização de limites surge uma zona livre, no quadrante Este, limitada por células de cor magenta, mais uma vez, este facto está relacionado com a existência de leituras de distância muito díspares entre sensores vizinhos.

6.7 Conclusões

Neste capítulo foram apresentados mapas construídos pelo método proposto nesta dissertação. Estes exemplos foram obtidos em simulação e em condições reais contemplando algumas situações específicas. Os mapas foram analisados segundo os seguintes parâmetros: precisão, robustez, adaptabilidade, impacto das rotações do robô na configuração do mapa e o efeito da utilização de limites na informação sensorial.

Os valores das medidas do mapa e as medidas obtidas em simulação ou com uma fita métrica, no ambiente real, foram comparados e, em cada caso, as diferenças foram alvo de comentários.

Na primeira simulação avaliou-se o comportamento do método colocando o robô numa sala fechada. Nesta experiência concluiu-se que na presença de superfícies normais em relação aos sensores, os erros são reduzidos e que neste ambiente estruturado, o impacto das rotações na configuração do mapa também é reduzido.

Na segunda experiência que consistiu num ambiente simulado do tipo corredor observou-se que existem diferenças em relação à experiência anterior. Neste caso, o impacto das rotações na configuração do mapa pode ser maior e a utilização da informação sensorial não limitada poderá também tem efeitos maiores no mapa.

Na terceira simulação foi utilizado um ambiente misto, ou seja, foi incluída uma parte de espaço mais aberto. A parte do mapa correspondente ao espaço aberto apresentou erros consideráveis, o que se deveu à contribuição dos sensores vizinhos. Os sensores que apresentarem valores menores condicionam as zonas livres no mapa levando ao aparecimento

destes erros. Por conseguinte, o impacto das rotações poderá ser maior na zona do mapa correspondente ao espaço aberto.

As duas experiências seguintes foram realizadas com o robô num ambiente misto real. Na primeira (quarta na sequência das anteriores), o ambiente tem uma parte estruturada e outra com espaço mais aberto onde foi colocada uma cadeira. Nesta experiência o impacto das rotações e dos movimentos de translação pode ser elevado de tal forma que pode não ocorrer a detecção das pernas da cadeira.

Na quinta experiência foi utilizado o ambiente da experiência anterior ao qual foi retirada a cadeira e foram colocados dois objectos estreitos de madeira perto do robô. A adaptabilidade foi analisada aproximando uma pessoa do robô e não com a introdução de um objecto. À semelhança da experiência anterior, o robô também foi colocado em posições onde os objectos deixaram de ser detectados, mesmo com o robô muito próximo deles. Pode-se ainda assinalar o aparecimento de erros por excesso em duas orientações embora sendo reduzidos.

O ambiente real da última experiência é constituído por um corredor estreito, aberto numa das extremidades. A particularidade desta experiência residiu no impacto das rotações na configuração do mapa. Neste ambiente real (corredor), com o robô em determinadas posições e orientações, podem surgir leituras de distância muito elevadas o que torna o impacto das rotações, por vezes, elevado.

De uma maneira geral, os resultados são muito positivos e com uma boa precisão. Uma vez que, na grande maioria das situações, os erros verificados são por defeito, pode-se concluir que este método fornece garantias de segurança para o robô. Nalgumas situações, por causa do ângulo de incidência dos sensores nas superfícies (ex. cantos) ser muito desfavorável, surgiram alguns erros consideráveis. A forma de reduzir os erros e ganhar alguma imunidade ao ângulo de incidência, passa pelo aumento do número de sensores e pela disposição destes de outra forma (não apenas de forma regular) de modo a aumentar a resolução do sistema sensorial. Adicionalmente, a inclusão de outros tipos de sensores pode compensar as limitações de cada tipo de sensor quando integrados num único sistema sensorial.

O problema da detecção/não detecção de objectos estreitos pode ser resolvida dotando o robô de uma estratégia activa. Por exemplo, se o robô deixou de detectar algum objecto pode efectuar rotações no sentido de certificar a existência do referido objecto. Esta análise seria prioritária se isto acontecesse no caminho do robô. Mudando a configuração sensorial do robô, podia-se incluir um sonar rotativo para explorar áreas de forma activa, desta forma, seria

possível explorar áreas onde houvesse dúvidas em relação à presença de objectos, sem alterar a orientação ou posição do robô.

Capítulo 7

Aplicação do Método a outros Robôs Móveis

Neste capítulo descrevem-se os módulos funcionais, do método de construção de mapas, por forma a facilitar a adaptação a outros robôs móveis. As intervenções a realizar para efectuar a adaptação serão abordadas de forma sucinta e apenas do ponto de vista conceptual.

Na figura 7.1 é apresentado o diagrama de fluxo do funcionamento do método de construção de mapas locais. Neste diagrama de fluxo pode-se observar a sequência de operação e a ordem pela qual estão orquestrados os módulos funcionais do método.

A figura 7.2 mostra o fluxo de dados entre os diversos módulos funcionais.

O módulo *Estabelecer_Comunicação_com_Robô* é constituído por todas as operações necessárias para estabelecer a comunicação com o robô móvel e para as configurações necessárias ao funcionamento correcto dos sistemas sensoriais.

No módulo *Aquisição_da_Informação_Sensorial* estão incluídas as funções inerentes à aquisição da informação proveniente de todos os sensores, incluindo a odometria, e as funções onde são efectuadas as respectivas conversões para as unidades adequadas ao funcionamento do método.

A escolha da informação sensorial a utilizar na RNA é feita pelo módulo homónimo. Esta escolha está dependente da orientação da célula, em relação ao sistema de coordenadas do robô, e dos sensores disponíveis com orientações próximas da orientação da célula.

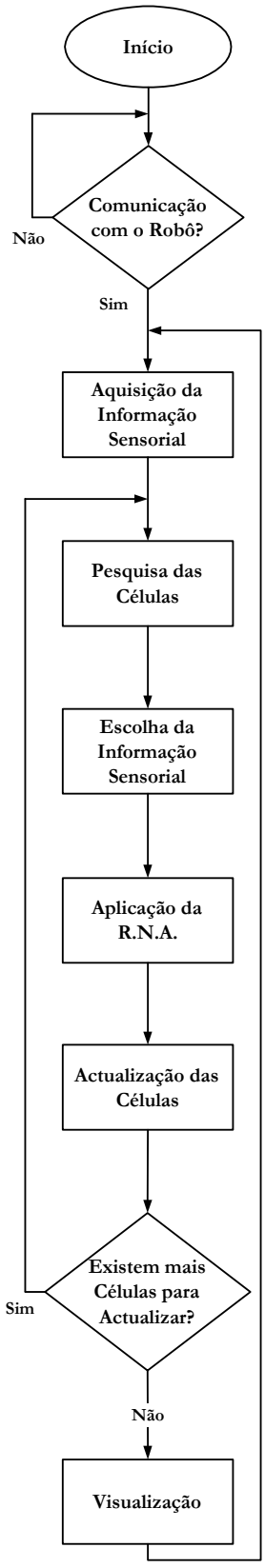

Figura 7.1. Diagrama de fluxo do método de construção de mapas.

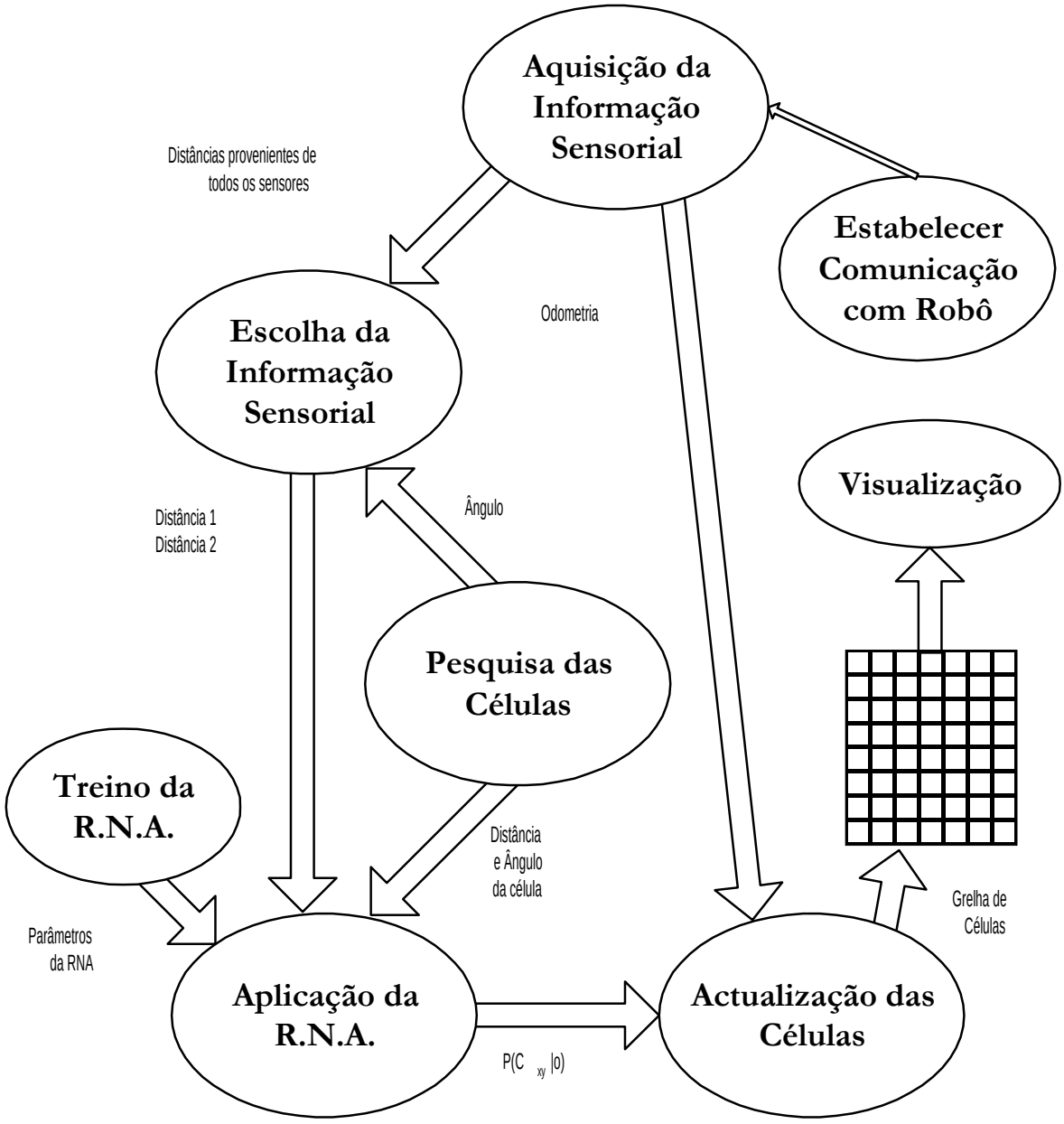

Figura 7.2. Módulos funcionais e fluxo de dados do método de construção de mapas.

O módulo onde é feita a procura das células a actualizar denomina-se por *Pesquisa_das_Células*. Além disso, fornece a distância e a orientação da célula ao módulo onde é aplicada a RNA. A RNA utiliza ainda as duas medidas de distância fornecidas pelo módulo *Escolha_da_Informação_Sensorial* para produzir a $P(C_{xy} \mid o^{(N)})$ (probabilidade da célula estar ocupada dada a observação sensorial actual). Com a $P(C_{xy} \mid o^{(N)})$, as células em causa já podem ser actualizadas.

Paralelamente ao funcionamento do método de construção de mapas existe um módulo muito importante que se encarrega do treino da RNA. Este módulo não é executado durante o método mas é através dele que a RNA é configurada para fornecer a $P(C_{xy} \mid o^{(N)})$ que é imprescindível à actualização correcta das células do mapa.

Existe ainda um módulo *Visualização* que permite mostrar o mapa de grelha de células no ecrã.

7.1 Adaptação dos Módulos Funcionais

Não sendo considerado um módulo funcional, o método necessita ainda de um módulo para gerir a memória dinâmica onde será guardada a grelha de células e de uma estrutura de dados adequada para facilitar a gestão da informação sensorial. Esta estrutura de dados deverá guardar as leituras de distância, em centímetros, provenientes dos sensores e a respectiva orientação (o ângulo em graus), em relação à origem do sistema (orientação Este). É ainda necessário uma estrutura de dados para gerir a informação odométrica, ou outro tipo de informação importante que esteja disponível no robô móvel em causa, susceptível de ser útil para o método de construção de mapas. Depois de definida a estrutura de dados pode-se começar a adaptação de cada módulo.

A primeira intervenção será no módulo *Estabelecer_Comunicação_com_Robô*. É neste módulo que se inicia a comunicação com o robô móvel em causa e se efectuam todas as configurações necessárias à aquisição da informação sensorial.

No módulo *Aquisição_da_Informação_Sensorial* é necessário adaptar a parte relacionada com a aquisição das medidas provenientes dos diversos sensores e da informação odométrica.

O objectivo do módulo *Escolha_da_Informação_Sensorial* é fornecer duas medidas de distância na orientação da célula a actualizar. Este módulo é muito importante uma vez que está dependente do número de sensores (e suas características) e da sua disposição. Depende ainda do módulo *Aquisição_da_Informação_Sensorial* uma vez que a orientação dos sensores deve estar sempre coerente com a orientação do robô.

O módulo onde é feita a pesquisa de todas as células a actualizar (à volta do robô) não necessita de ser alterado se o objectivo se mantiver. Este módulo fornece as coordenadas polares (R, θ) das células, ao módulo onde é aplicada a RNA.

O módulo onde é aplicada a RNA utiliza duas distâncias fornecidas pelo módulo *Escolha_da_Informação_Sensorial* e as coordenadas polares (R, θ) fornecidas pelo módulo *Pesquisa_das_Células* para produzir a probabilidade da célula em causa estar ocupada. O funcionamento correcto deste módulo pressupõe que a RNA tenha sido treinada para a gama de distâncias de operação do método. O módulo de treino da RNA é executado *off-line* de modo a encontrar os parâmetros que conduzem a RNA a apresentar um erro reduzido em resposta aos padrões de treino. Quando os parâmetros correctos estiverem determinados são gravados num ficheiro. No método, o módulo *Aplicação_da_RNA* apenas lê o ficheiro com os parâmetros e a RNA é utilizada de forma estática. O bloco de normalização pode necessitar de algumas alterações (secção 5.1.1).

Na parte final do método, depois de conhecer a $P(C_{xy} \mid o^{(N)})$ já se pode actualizar a célula em causa, no módulo homónimo. Este módulo é ainda responsável pelo deslocamento do mapa de grelha de células de acordo com a informação odométrica.

Quando todas as células estiverem actualizadas, o mapa de grelha de células pode ser apresentado no ecrã recorrendo a um interface gráfico.

Capítulo 8

Conclusões e Futuros Desenvolvimentos

8.1 Conclusões

De uma forma geral, os resultados obtidos pelo método de construção de mapas são positivos. A segurança do robô está salvaguardada nas diversas situações apresentadas, uma vez que, na maioria dos casos, o erro é sempre por defeito. Nos casos onde o erro foi por excesso apresentou sempre um valor reduzido.

O método evidencia uma robustez elevada, como se pode observar nos resultados apresentados no capítulo 6. Nesse capítulo, nas diversas experiências realizadas, o mapa após a primeira actualização (considerando o mapa inicial desconhecido - todas as células com o valor 0.5) ficou logo com os principais contornos do mapa obtido após múltiplas actualizações.

Outro parâmetro importante na avaliação dos métodos de construção de mapas é a adaptabilidade. Em relação a este aspecto, o método provou que rapidamente adapta o mapa às mudanças do ambiente.

O impacto de pequenas rotações do robô na configuração do mapa pode ser elevado. Perante algumas situações apresentadas no capítulo 6, os mapas ficavam com uma configuração, por vezes, bastante diferente depois do robô efectuar pequenas rotações. A consequência disto é que tornou imperceptível que ambos os mapas (obtidos no mesmo local mas rodando ligeiramente o

robô) se referiam ao mesmo local. Porém, é importante salientar que estes erros não se devem a erros introduzidos pelo método. O que acontece na realidade é que ao rodar o robô, os sensores ficam com orientações diferentes e ficando também com leituras de distância diferentes. Por conseguinte, o método, ao ler os novos valores provenientes dos sensores, irá actuar no mapa de acordo com esses valores. É esta a explicação para o aparecimento das diferenças entre os mapas. Pode-se diminuir o impacto das rotações na configuração do mapa aumentando o número de sensores à volta do robô e introduzindo outro tipo de sensores com disposições adequadas, de modo a aumentar a resolução sensorial.

Depois de analisar todos os resultados pode-se concluir que o método constrói mapas do ambiente com poucos erros, robustos e que salvaguardam a segurança do robô móvel. Além disso, pode constituir uma base de trabalho para uma evolução ou então para apoio a outros trabalhos envolvendo robôs móveis.

Na secção seguinte serão apresentadas algumas propostas para futuros desenvolvimentos deste método.

8.2 Futuros Desenvolvimentos

De seguida são apresentados alguns tópicos para uma evolução do método de construção de mapas ou então para servir como uma ferramenta de apoio a trabalhos que envolvam robôs móveis.

- **Mapa Global** - Utilizando os mapas construídos por este método é possível desenvolver um método para os integrar de modo a construir um mapa global do ambiente [Thrun 98], [Yamauchi et al. 98];

- **Grelhas Tridimensionais** - O método apresentado apenas funciona com grelhas bidimensionais, no entanto, pode-se evoluir o método por forma a introduzir a terceira dimensão. Seria interessante construir um mapa de um túnel natural;

- **Localização** - O mapa de grelha de células pode ser utilizado por um módulo de localização do robô;

- **Trabalho Cooperativo** - A utilização deste método em vários robôs, dentro do mesmo ambiente, pode potenciar o desenvolvimento de métodos para construir um mapa global ou efectuar planeamento de caminhos;

- **Sensores com Maior Precisão** - Quanto melhor for a precisão dos sensores utilizados nos robôs móveis melhor será a qualidade dos mapas;

- **Comportamento** - que faça a navegação com base no mapa construído pelo robô.

Estes são alguns tópicos para futuras evoluções deste método.

Apêndice A

Adaptação do Software a outros Robôs Móveis

Neste capítulo são descritas as alterações a efectuar no software, que implementa o método de construção de mapas por forma a adaptá-lo a outros robôs móveis. As alterações serão localizadas especificamente nas diversas funções do software. As intervenções a realizar para efectuar a adaptação serão abordadas de forma sucinta e apenas do ponto de vista conceptual.

O software que implementa o método de construção de mapas está dividido nos ficheiros seguintes:

- *cerne.cpp*
- *robo.cpp*
- *formula.cpp*
- *feedfowr.cpp*
- *vermapa.cpp*

O ficheiro *cerne.cpp* coordena todo o desenvolvimento do método. É neste ficheiro que é criada a grelha de células (uma matriz de tamanho configurável uma vez que utiliza memória dinâmica) e as respectivas funções inerentes à gestão de memória. No ficheiro *robo.cpp* são implementadas todas as funções de comunicação com o robô, de aquisição das leituras provenientes dos sensores e odometria. Estão ainda implementadas as funções de conversão de medidas e as funções que utilizam a odometria para efectuar o deslocamento correcto do mapa, para fazer a

rotação coerente dos sensores na estrutura de dados, por forma a utilizar os sensores correctos na actualização das células.

O ficheiro *feedfowr.cpp* contém todas as funções utilizadas para implementar a RNA.

A implementação da arquitectura do método (figura 5.1) é feita no ficheiro *formula.cpp*.

O ficheiro *vermapa.cpp* é executado num processo diferente uma vez que apenas é utilizado para mostrar no ecrã o mapa de grelha de células a cores. Para isso, lê os valores da grelha de células que são gravados pelo software num ficheiro, depois de cada actualização. O interface gráfico que suporta o mapa de grelhas a cores recorre às funções disponibilizadas pela biblioteca *XForms* [Zhao-Overmars 97].

Seguidamente será feito o levantamento de todas as funções susceptíveis de serem alteradas, quando o método de construção de mapas, apresentado nesta dissertação, for adaptado para outros tipos de robôs móveis. Para cada função será feita uma breve descrição da sua funcionalidade para que seja mais fácil a sua conversão para o robô móvel em causa.

A.1 Funções a Alterar para a Adaptação do Software

Em primeiro lugar, antes de começar a adaptação do software é necessário definir à partida a estrutura de dados que servirá de base à aplicação do método. Esta estrutura de dados tem que ser organizada por forma a conter as leituras de distância (em centímetros) provenientes dos sensores e a respectiva orientação (o ângulo em graus), em relação à origem do sistema (orientação Este). Adicionalmente, pode ser necessário guardar outro valor para eventuais ajustes, uma vez que à distância lida pelo sensor é necessário adicionar a distância do sensor ao centro de coordenadas do robô. Não esquecer a estrutura de dados para gerir a informação odométrica, ou outro tipo de informação importante que esteja disponível no robô móvel em causa, susceptível de ser útil para o método de construção de mapas. Depois de definida a estrutura de dados pode-se começar a adaptação do método.

Para começar, a função "*main*" está no ficheiro *cerne.cpp*. A primeira tarefa a desenvolver é estabelecer a comunicação com o robô móvel e iniciar todas as configurações necessárias à aquisição da informação sensorial. Esta tarefa pode ser efectuada adaptando a função "*estabelecer_comunicacao_robo*" (*robo.cpp*).

A função "*robo_sensores*" (*robo.cpp*) é utilizada para fazer a aquisição das medidas provenientes dos diversos sensores e da informação odométrica. Seguidamente, na função "*sensores_odometria*" (*robo.cpp*) é feita a transferência dos valores dos sensores e da odometria, presentes na memória do robô, para a estrutura de dados do software. Nesta função, os sensores são reorganizados, segundo a informação odométrica, para manter a coerência entre a orientação dos sensores e o mapa de grelha de células. Ainda na mesma função, utilizando a informação odométrica, são calculadas as diferenças entre os valores (x,y) actuais e os anteriores. Se forem diferentes, significa que o robô efectuou movimentos de translação, neste caso, o mapa é deslocado, segundo os eixos x e y, de acordo com o valor da diferença convertido em número de células.

Depois das operações efectuadas directamente com o robô, o método continua de modo a efectuar a actualização das células do mapa. A operação seguinte é efectuada pela função "*converte_vector_para_cm*", como o próprio nome indica, o objectivo é converter os valores, devolvidos pelos sensores, para centímetros. A seguir, os ângulos dos sensores são organizados de forma coerente com o valor da orientação actual do robô "*actualiza_vector_angulos_sensores*". Este vector de ângulos é muito importante, uma vez que a escolha dos sensores, para fornecer as distâncias para a actualização das células, baseia-se na análise deste vector. É importante salientar que as funções descritas até agora podem ser alteradas (aglutinadas ou subdivididas), de acordo com a estrutura de dados de base, com a forma de interacção com o robô móvel e com a estrutura de programação.

Ainda no ficheiro *cerne.cpp*, é necessário alterar a função "*coloca_robo*". O objectivo desta função é preencher uma área de células pretas, no centro do mapa, correspondente à área ocupada pelo robô móvel. Pode ainda ser incluída uma célula amarela (na área do robô) para indicar a orientação do robô.

Passando agora ao ficheiro *formula.cpp* onde é realizada a actualização das células propriamente dita, as alterações a efectuar estão directamente relacionadas com o tipo de sensores utilizados e a respectiva orientação. Para escolher os sensores (de forma a obter a probabilidade de ocupação através da RNA) existe a função "*Quadrantes*". É nesta função que se pode implementar o algoritmo para a escolha, entre todas as medidas de distância disponíveis, de duas medidas de distância necessárias para a actualização das células.

Existe ainda uma função secundária "*Distancia_Ponderada*" que tem de ser modificada porque está dependente da informação dos sensores. Esta função é utilizada apenas para estabelecer limites no algoritmo de pesquisa das células a actualizar.

O ficheiro *vermapa.cpp* é exterior ao método, ou seja, é executado num processo diferente do processo principal onde é executado o método de construção de mapas. A sua função é apresentar o mapa de grelha de células a cores no ecrã. Este processo comunica com o processo principal através de um ficheiro com os valores da grelha de células. Este ficheiro (*vermapa.cpp*) poderá ser alvo de alterações se o número de células do mapa for alterado, sobretudo se esse número aumentar. O facto da apresentação gráfica do método ser independente do método principal tem como principal vantagem a flexibilidade de poder utilizar outras bibliotecas gráficas. A biblioteca gráfica utilizada foi a XFORMS para linux. A grelha de células é gravada num ficheiro, portanto, para utilizar outra biblioteca gráfica basta apenas ler o ficheiro com a sintaxe correcta, tomando como exemplo o ficheiro *vermapa.cpp*.

Apêndice B

Algoritmos

B.1 Treino da RNA

```
  Obter base de dados com informações para o treino da
  RNA (padrões de treino)
  Criar a RNA
  Atribuir parâmetros de forma aleatória
  Enquanto o Erro maior que o erro objectivo
        Efectuar ciclo por todos os padrões de treino
            Calcular o erro para cada padrão
            Ajustar parâmetros da RNA
        Se o erro não diminui então
            {Verificar ***}
  Fim Enquanto
  Gravar parâmetros da RNA


***
Se o erro não tende a diminuir:
  Verificar a coerência dos padrões de treino
  Verificar a sequência dos padrões de treino
  Verificar a possibilidade de alterar a topologia da
  RNA, nomeadamente:

       1º  Aumentar o número de neurónios da camada
           oculta;
       2º  Aumentar o número de camadas ocultas.
```

B.2 Funcionamento do Método

```
Estabelecer a comunicação com o robô
Criar a Rede Neuronal Artificial (RNA)
Ler parâmetros da RNA
Criar matriz para o mapa de grelha de células
(1) Enquanto o método funcionar fazer:
      Aquisição da informação sensorial
      Se o robô mudou de posição então
            Calcular os deslocamentos em horizontais e
            verticais
            Fazer o deslocamento do mapa
      Conversão   da   informação   sensorial   para
      centímetros
      Se o robô mudou a orientação
            Actualizar orientação dos sensores
      Para todos os sectores
      (2) Enquanto houver células a actualizar fazer:
             Pesquisa das células
             Escolha da informação sensorial
             Determinar o parâmetro desvio
             Determinar P(C_xy | o^(N))
             Actualizar as células
       Fim Enquanto (2)
       Colocação do robô no centro do mapa
       Gravação do mapa em ficheiro
Fim Enquanto (1)
```

Bibliografia

[Anousaki-Kyriakopoulos 99] G. C. Anousaki and K. J. Kyriakopoulos, "Simultaneous Localization and Map Building for Mobile Robot Navigation", *IEEE Robotics and Automation Magazine*, September 1999, pp. 42-53.

[Arleo et al. 99] Angelo Arleo, José del R. Millán and Dario Floreano, "Efficient Learning of Variable-Resolution Cognitive Maps for Autonomous Indoor Navigation", *IEEE Transactions on Robotics and Automation December 1999*, Vol. 15, No. 6.

[Borenstein-Koren 89] Johann Borenstein and Yorem Koren, "Real-Time Obstacle Avoidance for Fast Mobile Robots", *IEEE Transactions on Systems, Man, and Cybernetics* September/October 1989, Vol. 19, No. 5.

[Borenstein-Koren 90] Johann Borenstein and Yorem Koren, "Real-Time Obstacle Avoidance for Fast Mobile Robots in Cluttered Environments", *IEEE International Conference on Robotics and Automation,* Ohio, May 13-18,1990, pp 572-577.

[Borenstein-Koren 91] Johann Borenstein and Yorem Koren, "The Vector Field Histogram - Fast Obstacle Avoidance for Mobile Robots", *IEEE Transactions on Robotics and Automation,* June 1991, Vol. 7, No. 3.

[Bose-Liang 96] N. K. Bose and P. Liang, *Neural networks fundamentals with graphs, algorithms and applications*, McGraw Hill, 1996.

[Boullart et al. 92] L. Boullart, A. Krijgsman, R.A. Vingerhoeds, *Application of artificial intelligence in process control*, Pergamon Press 1992.

[Bozma-Kuc 91] Omur Bozma and Roman Kuc, "Building a Sonar Map in a Specular Environment using a Single Mobile Sensor", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, December 1991, vol. 13 No.12.

[Burgard et al. 98] Burgard, W., A. B. Cremers, D. Fox, D. Hahnel, G. Lakemeyer, D. Schulz, W. Steiner and S. Thrun, *Experiences with an Interactive Museum Tour-Guide Robot*. In: T. R. CMU-CS-98-139, June.1998.

[Chong-Kleeman 97] Kok Seng Chong, Lindsay Kleeman, "*Sonar Based Map Building for Mobile Robot*", Proceedings of the 1997 *IEEE International Conference on Robotics and Automation*, Albuquerque, New Mexico - April 1997

[Choset et al. 97] Howie Choset, Ilhan Komuksven and Alfred Rizzi, "Sensor based Planning: A Control Law for Generating the Generalized Voronoi Graph", *In Proceedings of IEEE International Conference on Autonomous Robots,* CA. 1997.

[Curran-Kyriakopoulos 93] A. Curran and K. J. Kyriakopoulos, "Sensor-based Self Localization for Wheeled Mobile Robots", *Proceedings of the 1993 IEEE International Conference on Robotics and Automation*, 1993, Vol. 1, pp. 8-13.

[Dennis-McAuley 97] Simon Dennis and Devin McAuley, *Introduction to Neural Networks and the brain wave simulator*, 1997.
http: www2.psy.uq.edu.au/~brainwav/Manual/Whatis.html

[Elfes 87] Alberto Elfes, "Sonar-based Real World Mapping and Navigation", *IEEE J. Robotics and Automation*, Vol. RA-3, No. 3, June 1987

[Fausett 94] Laurene Fausett, *Fundamentals of neural networks -architectures, algorithms and applications*, Prentice Hall International Editions, 1994.

[Howard-Kitchen 96] Andrew Howard and Les Kitchen, "Vision based Navigation using Natural Landmarks", *Proceedings of the Fourth International Conference on Control, Automation, Robotics and Vision* - December 1996

[Kang et al. 01] Dong-OH. Kang, Sung-Hun Kim, Heyoung Lee and Zeungnam Bien, "Multiobjective Navigation of a Guide Mobile Robot for the Visually Impaired Based on Intention Inference of Obstacles", *Journal of Autonomous Robots* 10, pp. 213-230, 2001 Kluwer Academic Publishers.

[Krose-Smagt 96] Ben Krose and Patrick Van der Smagt, *An introduction to neural networks*, University of Amsterdam, 1996.

[Kuipers-Byun 91] Benjamin J. Kuipers and Yung T. Byun, "A Robot Exploration and Mapping Strategy based on a Semantic Hierarchy of Spatial Representations", *Journal of Robotics and Autonomous Systems* 8:47-63, 1991.

[Lee 96] David Lee, *The Map-Building and Exploration Strategies of a Simple Sonar- Equipped Robot*, Cambridge-University Press-1996.

[Leonard et al. 92] Leonard, J. J., H. F. Durrant-Whyte and I. J. Cox, "Dynamic Map Building for an Autonomous Mobile Robot". *In The International Journal of Robotics Research*, Vol. 11, No. 4, August 1992.

[Maren et al. 90] Aliana Maren, Craig Harston and Robert Pop, *Handbook of Neural Computing Application*, Academic Press, 1990.

[McCulloch-Pitts 43] Warren McCulloch and Walter Pitts, "A Logical Calculus of the Ideas Immanent in nervous Activity", *Bulletin of Mathematical Biophysics*, 5:115-133. *Reprinted in Anderson* & *Rosenfeld* 1988, pp. 18-28.

[Mitchell 97] Tom M. Mitchell. *Machine learning*, McGraw Hill, 1997.

[Moita-Nunes 94] F. Moita, A. Feijão, U. Nunes and A. T. Almeida, "Modelling and Calibration of an Ultrasonic Ranging System of a Mobile Robot", *Controlo 94, IST*, Setembro 1994.

[Moita-Nunes 01] Fernando Moita and Urbano Nunes, "Multi-Echo Technique for Feature Detection and Idenfication using Simple Sonar Configurations", *IEEE/ASME International Conference on Advanced Intelligent Mechatronics*, Italy, 8-11 July, 2001.

[Moravec 88] Hans P. Moravec, "Certainty Grids for Sensor Fusion in Mobile Robots", *A.I. Magazine*, Summer 1988, pp 61-77.

[Moravec-Cho 89] Hans P. Moravec and Dong Woo Cho, "A Bayesian Method for Certainty Grids", *Working notes of AAAI 1989 Spring Symposium on Robot Navigation*, Stanford, CA pp. 57-60 1989.

[Murphy 98] Robin R. Murphy, "Dempster-Shafer Theory for Sensor Fusion in Autonomous Mobile Robots", *IEEE Transactions on Robotics and Automation*, vol14, nº 2, April 1998.

[Murray-Jennings 97] Don Murray and Cullen Jennings, "Stereo Vision based Mapping and Navigation for Mobile Robots". *In Proceedings of the IEEE ICRA* 1997, Albuquerque, New Mexico – April.

[Nomadic 2] Nomadic Technologies, Inc. *Language Reference Manual*. Part Number: DOC00002, March 11, 1997.

[Nomadic 4] Nomadic Technologies, Inc. *User's Manual*. Part Number: DOC00004, January 23, 1996.

[Nomadic 5] Nomadic Technologies, Inc. *Nomad 200 Hardware Manual*. Part Number: DOC00005, March 15, 1996.

[Nunes et al. 2000a] Urbano Nunes, R. Araújo and L. Marques, "Towards Intelligent Machines: Theory, Technology, and Experiments". In Int. J. of Systems Analysis Modelling and Simulation, Vol. 38 issue 2, pp. 157-173, July 2000.

[Nunes et al. 2000b] Urbano Nunes, R. Cortesão, José L. Cruz and P. Coelho, "Shared-Control Architecture: Concepts and Experiments". *In Service Robots: Applications and safety issues in an Emerging Market. Workshop Notes of European Conference on Artificial Intelligence.* (ECAI'00) pp. 21-26, 2000.

[Oriolo et al. 97] Giuseppe Oriolo, Giovanni Ulivi and Marilena Venditteli, "Fuzzy Maps: A New Tool for Mobile Robot Perception and Planning", *IEEE Transactions on Systems, Man and Cybernetics, part B: Cybernetics*, vol. 28, No. 3, pp. 316-333, 1998

[Pandya-Macy 96] Abhijit S. Pandya, Robert B. Macy. *Pattern recognition with neural networks in C++*, CRC Press/IEEE Press, 1996.

[Petersen 92] Richard Petersen. *Introductory* C - *Pointers, Functions and Files*, Academic Press, Inc 1992.

[Rumelhart et al. 86] D. E. Rumelhart, G. E. Hinton and R. J. Williams, *Learning Internal Representations by Error Propagation*. In: Parallel distributed processing. (D. E. Rumelhart & J. L. McClelland (Eds.)), Cambridge, MA: MIT Press 1986.

[Sabatini-Benedetto 95] Angelo M. Sabatini and Orazio D Benedetto, "Spatial Localization using as a Control Point Matching Task". *Proceedings of the 3rd International Symposium on Intelligent Robotic Systems '95* Pisa, Italy, July 10-14, 1995.

[Shatkay-Kaelbling 97] Hagit Shatkay and Leslie Pack Kaelbling, "Learning Topological Maps with weak Local Odometric Information", *Proceedings of the IJCAI* 1997.

[Schalkoff 97] Robert J. Schalkoff. *Artificial neural networks*, McGraw Hill, 1997.

[Stroustrup 91] Bjarne Stroustrup. *C++ Programming Language*, Addison-Wesley Publishing Company 1991.

[Vandorpe et al. 96] J. Vandorpe, H. Van Brussel and H. Xu, "Exact Dynamic Map Building for a Mobile Robot using Geometrical Primitives Produced by a 2D Range Finder", *Proceedings of the 1996 IEEE International Conference on Robotics and Automation*, Minneapolis, Minnesota, April 1996

[Thrun 98] Sebastian Thrun, "Learning Metric-Topological Maps for indoor Mobile Robots Navigation". *In: Artificial Intelligence*, 99(1): 21-71,1998.

[Weigl et al. 93] M. Weigl, B. Siemiatkowska, K.A. Sikorski and A. Borkowski, "Grid based Mapping for Autonomous Mobile Robot", *Elsevier Science Publishers B.V.*, 13-21, - 1993

[Yamauchi et al. 98] Brian Yamauchi, Alan Schultz and William Adams, Mobile Robots Exploration and Map Building with Continuous Localization. *Proceedings of the 1998 IEEE International Conference on Robotics and Automation*, May 1998, Leuven, Belgium pp. 3715-3720.

[Zhao-Overmars 97] T. C. Zhao and Mark Overmars. *Forms library, a graphical user interface toolkit for X*, V0.86 March 1997.