Universidade Estadual de Londrina

RAFAEL GOMES MANTOVANI

UTILIZAÇÃO DE REDES NEURAIS DE SPIKES PARA TAREFAS DE NAVEGAÇÃO DE AGENTES ROBÓTICOS AUTÔNOMOS

Londrina
2011

RAFAEL GOMES MANTOVANI

UTILIZAÇÃO DE REDES NEURAIS DE SPIKES PARA TAREFAS DE NAVEGAÇÃO DE AGENTES ROBÓTICOS AUTÔNOMOS

Trabalho de Dissertação de Mestrado apresentado á Universidade Estadual de Londrina como parte dos requisitos para obtenção do título de Mestre em Ciência da Computação.

Orientador: Prof. Dr. Pedro Paulo da Silva Ayrosa.

Londrina
2011

Catalogação elaborada pela Divisão de Processos Técnicos da Biblioteca Central da Universidade Estadual de Londrina

Dados Internacionais de Catalogação-na-Publicação (CIP)

M293u Mantovani, Rafael Gomes

Utilização de Redes Neurais de *Spikes* para tarefas de navegação de agentes robóticos autônomos / Rafael Gomes Mantovani. – Londrina, 2011.
154 f.: il.

Orientador: Pedro Paulo da Silva Ayrosa.
Dissertação (Mestrado em Ciência da Computação) - Universidade Estadual de Londrina, Centro de Ciências Exatas, Programa de Pós-Graduação em Ciência da Computação, 2011.
Inclui bibliografia.

1. Robótica – Teses. 2. Redes neurais (Computação) – Teses. 3. Navegação de robôs móveis – Teses. 4. Neurônios Biológicos – Teses. 5. Sistemas de Controle – Teses. 6. Redes neurais (Neurobiologia) – Teses I. Ayrosa, Pedro Paulo da Silva. II. Universidade Estadual de Londrina. Centro de Ciências Exatas. Programa de Pós-Graduação em Ciência da Computação. III. Título

CDU 519.685

RAFAEL GOMES MANTOVANI

UTILIZAÇÃO DE REDES NEURAIS DE SPIKES PARA TAREFAS DE NAVEGAÇÃO DE AGENTES ROBÓTICOS AUTÔNOMOS

Trabalho de Dissertação de Mestrado apresentado á Universidade Estadual de Londrina como parte dos requisitos para obtenção do título de Mestre em Ciência da Computação.

BANCA EXAMINADORA

_____
Orientador. Prof. Dr. Pedro Paulo da Silva Ayrosa
Universidade Estadual de Londrina - UEL

_____
Profa. Dra. Maria Angélica de Oliveira Camargo Brunneto
Universidade Estadual de Londrina - UEL

_____
Prof. Dr. Wesley Attrot
Universidade Estadual de Londrina - UEL

_____
Prof. Dr. Álvaro José Periotto
Universidade Estadual de Maringá - UEM

Londrina, 17 de março de 2011.

Agradecimentos

Agradeço primeiramente a Deus, pela vida que possuo e por tudo que vivencio. Aos meus pais, Ângelo e Madalena, pela paciência e incentivo, mesmo que as vezes pensassem que o mestrado não fosse cansativo e talvez não equivalesse a um trabalho.

Agradeço ao meu orientador, professor Dr. Pedro Paulo pelo apoio, dedicação, orientação, e acima de tudo pela amizade em mais de três anos de trabalhos em conjunto, iniciados na graduação.

Agradeço também ao Ernesto e à Estefânia, estagiários do laboratório de robótica, pela ajuda nas etapas do desenvolvimento do trabalho, pois sem tal auxílio creio que não teria conseguido terminar as simulações e experimentações dentro do prazo necessário.

Agradeço à Camille, que foi quem fotografou o robô e trabalhou nas imagens para que pudesse aproveitá-las no trabalho.

Aos professores e funcionários, tanto do departamento de computação bem como os vinculados ao programa de mestrado, pelos conhecimentos transmitidos, pelos auxílios e pela amizade conquistada no dia-a-dia. A todas estas pessoas, deixo meus sinceros agradecimentos. Muito Obrigado.

A todos aqueles que ...
de uma maneira ou outra,
contribuíram para a realização
deste trabalho.

MANTOVANI, Rafael Gomes. **Utilização de Redes Neurais de Spikes para tarefas de navegação de agentes robótico autônomos**. 2011. 154 f. Dissertação (Mestrado em Computação) - Universidade Estadual de Londrina, Londrina. 2011.

RESUMO

Detectar e prevenir possíveis colisões é um dos aspectos mais importantes na robótica móvel. Esta tarefa, embora aparente facilidade quando executada por seres vivos, mantêm sua dificuldade quando modelada e executada por agentes robóticos autônomos. Além disso, roboticistas e pesquisadores têm sempre encontrado na natureza uma fonte inesgotável de inspiração. Muitas vezes a robótica é utilizada para investigar questões abertas da neurociência e ciência cognitiva, pois ela é capaz de sujeitar hipóteses a rigorosos testes no mundo real, aperfeiçoando novos mecanismos que poderão ser utilizados em futuros sistemas de navegação. Com o aumento do foco sobre a neurociência computacional nos últimos anos, uma nova variedade de modelos e algoritmos bioinspirados têm surgido na literatura especifica, como é o caso das Redes Neurais de Spikes (Spiking Neural Networks - SNN). Tais modelos neurais incrementam o realismo biológico de suas unidades computacionais utilizando de spikes individuais, permitindo incorporar informações espaço-temporais nos processos de comunicação e computação, como neurônios reais fazem. O objetivo do presente trabalho _e empregar uma SNN no tratamento (e prevenção) de colisões com obstáculos em um ambiente desconhecido (obsctacle avoidance). O algoritmo de aprendizado bioinspirado de plasticidade sináptica dependente de tempo de spike (spike-timing-dependent plasticity - STDP) _e incluído na SNN para fazer com que o sistema seja capaz de aprender com as respostas dos estímulos externos e guiar a navegação do robô através do ambiente. Baseado no conhecimento adquirido ao longo do tempo durante as simulações, o robô aprendeu a utilizar as informações externas captadas e descreveu trajetórias livre de colisões. A distância em que os movimentos evasivos são realizados também cresce consideravelmente com o tempo, indicando que o robô executa uma navegação segura. Durante as experimentações utilizou-se o kit de robótica Lego Mindstorms NXT. A prototipação e implementação do modelo foi feita utilizando-se o framework para robótica Microsoft Robotics Developer Studio (MRDS), que alem de proporcionar um ambiente virtual para simulações, fornece recursos para a programação de diversos tipos de hardware, incluso o NXT.

Palavras-Chave: Robótica móvel. Modelos bio-inspirados. Redes neurais de Spikes. Prevenção de colisões. Lego Mindstorms NXT. Microsoft robotics developer studio.

MANTOVANI, Rafael Gomes. **Use of Spiking Neural Networks for navigation tasks in autonomous robotic agents**. 2011. 154 p. Dissertation (Master's degree) - State University of Londrina, Londrina. 2011.

ABSTRACT

Detect and avoid possible collisions is one of the most important aspects in mobile robotics. Although this task seems easy when performed by animals (and humans), presents difficulty when modeled and executed by autonomous robotic agents. In addition, roboticists and researchers have always found in nature an inexhaustible source of inspiration. Robotics is often used to investigate open questions of neuroscience and cognitive science, because it is capable of subjecting hyphtoeses to rigorous testing in real world, improving new mechanisms that could be used in the future navigation systems. With increasing focus on computacional neuroscience in recent years, a new range of bioinspired models and algorithms have emerged in the specific literature, such as the Spiking Neural Networks (SNN). These models enhanced the biological realism of their computational units using 'individual spikes', allowing the incorporation of spatial and temporal information in the processes of communication and computing, like real neurons do. The aim of the current work is to use a SNN to prevent collisions with obstacles in an unknown environment (obstacle avoidance). The bioinspired learning algorithm spike-timing-dependent plasticity (STDP) is inserted in the SNN to make the system able to learn from the responses of external stimuli and to guide the robot navigation through the environment. Based on the knowledge acquired over time during simulations, the robot has learned to use external information perceived and described collision-free trajectories. The distance at which the preventive movements are performed grows considerably over time, indicating that the robot performs a safe navigation. During the experiments it was used the robotics kit Lego Mindstorms NXT. The prototyping and implementation of the model was done using the robotics framework Microsof Robotics Developer Studio (MRDS), which besides providing a virtual environment for simulation, provides resources for the programming of variety of robotics hardware, included the NXT.

Keywords: Mobile robotics. Bioinspired models. Spiking neural networks. Obstacle avoidance. Lego Mindstorms NXT. Microsoft robotics developer studio.

Lista de Figuras

Lista de Tabelas

Lista de Algoritmos

Lista de Siglas e Abreviaturas

SNN Redes Neurais de *Spikes* - *Spiking Neural Networks*

STDP Plasticidade sináptica dependente do tempo de *spike* - *Spike timing-dependent plasticity*

MRDS *Microsoft Robotics Developer Studio*

3D Tridimensional

RC Resistor-Capacitor

DNP Dinitrofenol

ATP Trifosfato de adenosina

ADP Difosfato de adenosina

EPSP Potencial pós-sináptico excitatório

IPSP Potencial pós-sináptico inibitório

EPSC Corrente pós-sinática excitatória

IPSC Corrente pós-sináptica inibitória

PSP Potencial pós-sináptico

LTP Potencial de longa duração

LTD Depressão de londa duração

2D Bidimensional

H-H Hodgkin e Huxley

pSNM Modelo probabilístico de neurônio de *spike*

PFM Método do campo potencial

VFH Histograma de campo vetorial

ORM Método de restição ao obstáculo

DWA Aproximação por janela dinâmica

CCR *Concurrency and Coordination Runtime*

DSS *Decentralized Software Services*

XML *eXtensible Markup Language*

SDK *Kit* de desenvolvimento de *software*

RAM Memória de acesso aleatório

GB Gigabyte

R2 *Release* 2

S6 Simulação 06

S7 Simulação 07

S8 Simulação 08

Sumario

1 INTRODUÇÃO

Prevenir possíveis colisões é um dos aspectos mais importantes na robótica móvel. Para um robô autônomo que executa uma tarefa de navegação, detectar e evitar obstáculos é uma questão crucial para a segurança própria do robô bem como da continuidade da tarefa em si (SOUMARE; OHYA; YUTA, 2002). Embora a previsão de colisões seja uma tarefa de aparente facilidade para seres vivos, é uma tarefa um pouco difícil para robôs móveis autônomos e ainda é um tema bem pesquisado na robótica. A questão de como os sistemas biológicos transformam toda a "tempestade" de informações sensitivas em movimentos destinados a execução de múltiplos objetivos é uma busca constante para muitos especialistas em robótica e neurocientistas (HORIUCHI, 2009).

Tais especialistas (e inventores, engenheiros, etc.) têm sempre encontrado na natureza uma fonte inesgotável de inspiração, qualidade necessária para áreas de pesquisa em pleno desenvolvimento. De acordo com (SICILIANO; KHATIB, 2008) tem-se observado um crescimento no número de aplicações da robótica atual que buscam soluções bioinspiradas. Pode-se dizer que a utilização da inspiração 'natural' é a grande tendência da robótica, e tal acontecimento se deve principalmente aos recentes avanços da biologia e tecnologia.

Uma área de pesquisa onde o realismo biológico tem sido frequentemente explorado é a modelagem de redes neurais artificiais. Os modelos neurais existentes em geral variam em complexidade, partindo do neurônio mais simples (modelo de McCulloch-Pitts (MCCULOCH; PITTS, 1943)) até modelos multicomportamentais que descrevem precisamente o comportamento dos canais iônicos através de uma única célula. Izhikevich (IZHIKEVICH, 2004; IZHIKEVICH, 2007a) descreve um crescimento de interesse sobre uma nova vertente de modelos neurais bioinspirados, as Redes Neurais de *Spikes* (*Spiking Neural Networks - SNN*). Tais modelos neurais focam na formalização matemática das propriedades computacionais dos neurônios biológicos. Estes modelos neurais capturam a natureza dos *spikes* dos neurônios e retêm o essencial do comportamento artificial a ser modelado durante a tentativa de simplificar a descrição deste processo (IZHIKEVICH, 2004). Uma característica interessante é que variáveis analógicas podem ser codificadas através da **diferença de tempo** entre os *spikes* emitidos

por um ou vários neurônios. Um exemplo, como abordado pelo presente trabalho, seria a codificação da distância de um agente robótico a um objeto em rota de colisão. Conforme esta distância diminui os neurônios responsáveis pela percepção deste objeto emitem cadeias com um número maior de *spikes*, indicando a aproximação do robô ao objeto.

1.1 A dissertação

O objetivo do presente trabalho é empregar uma rede neural de *spikes* no tratamento (e prevenção) de colisões com obstáculos em um ambiente desconhecido. Por usar o conceito de *spikes* na estrutura do modelo neural, existe a incorporação de plausibilidade biológica, tanto para representação como para manipulação da informação na rede.

A tarefa de navegação explorada é a navegação livre de colisões em um ambiente desconhecido ao robô (*obstacle avoidance*). Os ambientes, onde as experimentações e simulações são executadas, são povoados por obstáculos espaçados randomicamente, onde a detecção dos mesmos é feita pelo robô durante a execução de sua movimentação.

A topologia e estrutura das conexões entre neurônios da rede são inspiradas nos trabalhos de (BRAITENBERG, 1984) e (ARENA et al., 2009). Os estímulos captados pelo robô podem ser condicionados (sensores de toque) ou incondicionados (sensores ultra-sônicos). Existe um par de sensores de cada tipo, posicionados à direita e à esquerda do robô. Com base nos estímulos adquiridos, é feita a computação da ação do robô e o movimento é gerado. O comportamento esperado é que com o passar do tempo o robô aprenda a utilizar os sensores ultra-sônicos, reduzindo o uso dos sensores de toque. A vantagem de se utilizar um modelo reativo desta natureza é que através da computação das informações locais (captadas pelos sensores), o movimento pode ser adaptado a qualquer situação imprevista com os planos iniciais de navegação.

A modelagem inicial da rede descrita foi realizada com auxílio da ferramenta matemática (e computacional) Matlab. Além de permitir um maior controle e ajuste dos parâmetros, foi possível reforçar o aprendizado da 'mecânica' do modelo por meio de simulações numéricas providas de gráficos informativos (comportamento dos pesos sinápticos, potencial de membrana de um neurônio ao longo do tempo, etc).

Para facilitar a manipulação das situações errôneas durante a execução, e ganhar tempo no desenvolvimento do trabalho, foi utilizado o ambiente *Microsoft Robotics Developer Studio (MRDS)*(HUNG; LIU; KANG, 2008). Além de permitir programação e controle sobre o *kit* de robótica utilizado (*Lego Mindstorms NXT*), este *framework* da *Microsoft* possui

uma *engine* física para simulações em ambientes virtuais. Tais ferramentas são fundamentais para a realização da prototipação do modelo, pois reduz o tempo necessário para testes e ajustes de dados, simulando diversos cenários que levariam muito tempo para serem validados fisicamente.

No Apêdice A do trabalho encontra-se uma figura com todas as teorias e ferramentas discutidas durante o desenvolvimento do trabalho, e suas relações (fig.A.1). Nesta figura, as ferramentas/modelos selecionados e utilizados na metodologia do trabalho estão caracterizados pela cor "verde escura" (Matlab, Modelo de Izhikevich, Microsoft Robotics Developer Studio, etc). Além disso, através deste diagrama podemos ver como todas os conceitos descritos ao longo do desenvolvimento do trabalho se relacionam e são aplicados no problema de navegação livre de colisões.

1.2 Justificativa

Muitas vezes a robótica também é utilizada para investigar questões abertas na neurociência e ciência cognitiva, pois oferece o ponto de vista de um sistema comportamental que interage com o próprio ambiente. É comum encontrar pesquisadores que afirmam que as pesquisas com modelos bioinspirados ainda estão no início de seu desenvolvimento, já que a maioria dos trabalhos visa testar modelos ou mecanismos biológicos ao invés de encontrar a solução ótima para um dado problema.

De acordo com Franz e Mallot (2000) existem basicamente dois interesses que movimentam as pesquisas com modelos bioinspirados na robótica: (biológico) sujeitar hipóteses biológicas a um rigoroso teste no mundo real; e (biônico) de encontrar novos mecanismos que possam ser utilizados nos futuros sistemas de navegação. A tarefa de testar estas teorias biológicas de navegação em robôs móveis reais origina uma gama de testes mais rigorosos para modelos comportamentais do que as simulações computadorizadas.

Um ponto importante a ser ressaltado é que as capacidades de autonomia e adaptação demonstradas por criaturas vivas excedem em muito às dos robôs atuais, que raramente são confrontados com a necessidade de lidar com mudanças permanentes em seu ambiente externo ou necessidades internas. Talvez a principal razão da superioridade dos animais sobre os robôs encontra-se em seu maior grau de integração.

Uma das propriedades dos agentes inteligentes é a capacidade de aprender e se adaptar às novas condições ambientais. Estas características são o resultado da interação contínua e intensa do cérebro com o mundo externo, mediado pelo corpo (NOVELLINO et

al., 2007). Diferentes mecanismos bioinspirados são utilizados atualmente em trabalhos descritos pela literatura, tais mecanismo podem implementar esquemas associativos, por reforço ou de imitação. Mecanismos associativos são usados em aplicações que se aproveitam das características das células encontradas no hipocampo e região para-hipocampal do cérebro para implementar localização, e capacidades de navegação em agentes biológicos (ARENA et al., 2009). Por meio de pesquisas no campo da neurociência constatou-se que o hipocampo desempenha um papel-chave no controle da navegação em animais, codificando posições espaciais e agindo como mecanismo de memória associativa (BURGESS; O'KEEFE, 1996), (BURGESS et al., 2001) e (JENSEN; LISMAN, 2005). O que se observou foi que a comunicação entre neurônios poderia ser feita utilizando o *timing* dos *spikes* emitidos pelos neurônios de uma sinapse, processo formalizado posteriormente e nomeado de *plasticidade sináptica dependente de tempo de spike*, ou STDP (GERSTNER; KISTLER, 2002).

Com isso, pode-se dizer que o uso de neurônios de *spikes* para a construção de sistemas de navegação biologicamente inspirados apresenta novos problemas e oferece vários aspectos inovadores: acrescentam precisão ao modelo bioinspirado; requerem um novo paradigma (biológico) de aprendizado (o STDP); além de ser baseado em sistemas dinâmicos ao invés de uma relação estática entre valores de entrada e saída (ARENA et al., 2009).

1.3 Trabalhos correlatos

Impulsionados pelos avanços da tecnologia e neurociência, muitos trabalhos que empregam SNN podem ser encontrados na literatura. Além de *reviews*, tem-se encontrado muitas aplicações de sucesso abrangendo áreas como a visão computacional (CHRISTODOULOU; BUGMANN; CLARKSON, 2002) (ESCOBAR; AL, 2009); segmentação de imagens (ROWCLIFFE; FENG; BUXTON, 2002) (BUHMANN; LANGE; RAMACHER, 2005); previsão de séries temporais (BURGSTEINER et al., 2007); reconhecimento de fala (MAASS; NATSHLAGER; MARKRAM, 2002) (VERSTRAETEN et al., 2005); controle de braço robótico (JOSHI; MAASS, 2005) (ROWCLIFFE; FENG, 2008); reconhecimento e lateralização de fontes sonoras (VOUTSAS; ADAMY, 2007) até à navegação autônoma de robôs (ARENA et al., 2009).

Em relação a modelos bioinspirados, o trabalho de (FLOREANO et al., 2006) fornece uma revisão das arquiteturas neurais empregadas para navegação. Alguns modelos bioinspirados empregados para navegação de robôs são descritos nos trabalhos de (BARRERA; WEITZENFELD, 2008), (MOIOLI, 2008), (SCHMICKL et al., 2009) e (LIU et al., 2009). Sistemas de navegação autônoma de robôs móveis que utilizam de informações sensoriais codificadas por

meio de *spikes* podem ser encontrados em (WANG et al., 2008), (HORIUCHI, 2009) e (ARENA et al., 2009).

1.4 A organização do texto

No capítulo 2 são apresentados os conceitos sobre membranas excitáveis. No intuito de contextualizar a modelagem da membrana de um neurônio biológico são descritas propriedades e características fundamentais para a formalização da equação da membrana, como o potencial de Nernst, o fluxo de íons entre a estrutura membranal, bombas de sódio, potencial de repouso, etc. Além disso é descrita a definição de um potencial de ação de uma célula nervosa (também conhecido como *spike*), e algumas das mudanças biofísicas que ocorrem durante este processo, como a variação da condutância elétrica da membrana e o fluxo de íons (cátions e ânions). No final do capítulo discute-se como tais *spikes* podem ser arranjados em uma rede neural e quais tipos de aplicações estão utilizando tal tipo de modelagem.

No capítulo 3 são apresentados alguns dos modelos matemáticos de neurônios biológicos presentes na literatura. Tais modelos vão de formalismos complexos, como o modelo de Hodgkin e Huxley; até modelos simples como o modelo *integrate-and-fire*. Aqui, destaca-se o modelo desenvolvido por Izhikevich (IZHIKEVICH, 2003), que por meio de uma equação de segundo grau e determinados valores de parâmetros consegue reproduzir o comportamento de uma vasta gama de tipos de neurônios. Outro modelo que vale a pena destacar é o modelo proposto por Kasabov (KASABOV, 2010) que realiza a modelagem do comportamento do neurônio utilizando de teoria probabilística.

O capítulo 4 traz o problema e conceitos básicos sobre navegação. Nas subseções são apresentadas hierarquias das técnicas de navegação, seguida da descrição do problema de prevenção de colisões. Algumas das técnicas já existentes na literatura são descritas, como o método do campo potencial e o método de aproximação por janela dinâmica.

No capítulo 5 é descrita a metodologia do trabalho. Após introduzir o problema, segue uma modelagem conceitual da rede de *spikes* utilizada. Neste capítulo também existe uma subseção dedicada exclusivamente para o algoritmo de plasticidade sináptica empregado no modelo (STDP), pois uma vez que se utilizam *spikes* como unidades computacionais, consequentemente deve-se utilizar em uma nova abordagem apropriada para o aprendizado da rede. Além disso, é descrito o processo de prototipação em simulador 3D e as etapas predecessoras e sucessoras à experimentação física do problema.

O capítulo 6 traz a descrição das simulações realizadas com o protótipo virtual e os resultados coletados durante tais experimentações, organizados por diagramas de trajetórias e gráficos de desempenho que descrevem o comportamento do modelo simulado.

Por fim, são apresentadas as conclusões do trabalho e sugestões de melhorias a serem realizadas em trabalhos futuros.

2 MEMBRANAS EXCITÁVEIS E SPIKES

As membranas biológicas exercem papeis fundamentais em muitos dos processos vitais. Muitas dessas atividades são elétricas e o **potencial de membrana** - a diferença de potencial presente em uma membrana - é um dos estados físicos dos neurônios que podem ser medidos *in vitro*. Tentativas de descrever o comportamento elétrico de células nervosas têm sido baseadas em analogia a circuitos e seus modelos matemáticos (HOPPENSTEADT, 1986).

De acordo com (IZHIKEVICH, 2007a), o conceito mais importante nos estudos do cérebro é o conceito de neurônio. Os *neurônios* são as unidades básicas do sistema nervoso humano. São também as células especializadas em receber informações elétricas do próprio organismo ou do ambiente externo. Eles integram as informações e as retransmitem para as outras células. Ainda que não existam dois neurônios iguais, estas células possuem uma série de características comuns importantes no seu estudo. Da mesma forma que outras células, o neurônio possui uma *membrana celular* que o reveste e o separa do meio externo, além de um *corpo celular* ou *soma* dentro do qual se encontra a maioria dos orgânulos responsáveis pela manutenção de vida da célula. Do soma partem inúmeros prolongamentos que, devido ao seu aspecto funcional, são classificados em *dendrito* e *axônio*. O axônio, geralmente único, liga o soma de sua célula a outros neurônios, enquanto os dendritos, que podem ser milhares, recebem as terminações dos axônios das outras células (fig.2.1). O axônio é constituído por uma membrana isolante e um fluido condutor intracelular, e apesar de único, ramifica-se em *colaterais* de comprimentos variados, permitindo assim a união de sua célula a um grande número de neurônios dispostos em diferentes regiões do sistema nervoso (CARVALHO, 1989).

A região de contato entre uma terminação axônica e o dendrito ou soma, de outra célula é denominado de *sinapse*. As sinapses são importantes, pois além de permitirem a união de várias células nervosas formando redes neuronais, funcionam como "válvulas" regulando o fluxo da informação transmitida entre os neurônios (CARVALHO, 1989).

Entre o interior e o exterior do neurônio existe uma diferença de potencial V_M denominada *potencial de membrana*. Para um neurônio desempenhar suas funções, o

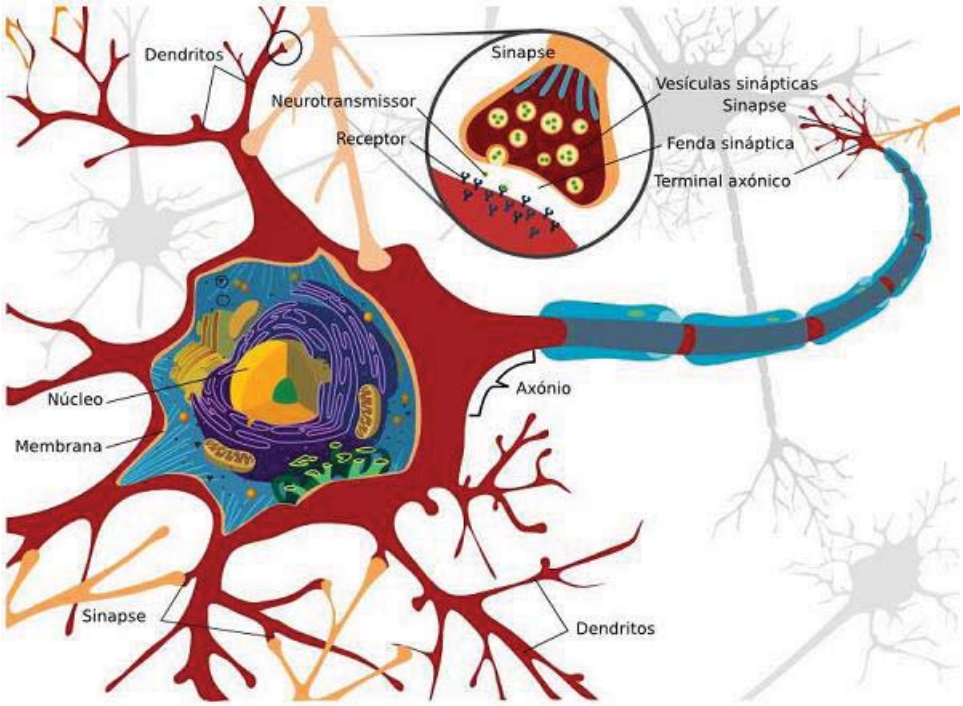

Figura 2.1: Modelo abstrato do neurônio biológico [Adaptado de (CHAVES, 2007)].

corpo celular, os dendritos, o axônio e as sinapses devem possuir determinadas características relacionadas com a maneira de gerar, conservar e transmitir impulsos elétricos, fenômeno conhecido como o *potencial de ativação* de uma célula nervosa. Os dendritos têm como função receber estes impulsos nervosos de outros neurônios e conduzi-los até o corpo celular. Uma vez situados no corpo celular, novos impulsos elétricos são gerados e repassados pelo axônio para outro neurônio, com o qual mantém contato por meio de um de seus dendritos (DURAN, 2003).

O potencial de membrana de um neurônio é uma variável física dentro do sistema nervoso que pode modificar-se rapidamente ao longo de grandes distâncias (um disparo de ativação pode variar o potencial da célula em 100 mV em um intervalo de tempo de 1 ms). Este potencial também é responsável por controlar um vasto número de canais - canais iônicos - que fornecem um rico substrato para a realização de operações neurais (KOCH, 1999). Nos seres humanos e animais, uma grande quantidade de energia metabólica (cerca de 20% da taxa metabólica basal) é constantemente despendida para manter esse processo, o que indica sua importância (OKUNO; CALDAS; CHOW, 1982).

2.1 Equação de Nernst-Plank

A maioria dos fenômenos elétricos está relacionada às cargas elétricas em movimento. Todo movimento de partículas carregadas ou portadoras de carga elétrica constitui uma *corrente elétrica*, elemento importante na preparação de potenciais elétricos ao longo de fibras nervosas (DURAN, 2003).

2.1.1 Corrente elétrica

Uma *corrente elétrica* é um fluxo de cargas elétricas. Para que seja mantida uma corrente elétrica num certo meio condutor é necessário que haja uma diferença de potencial, isto é, um campo elétrico nesse meio (IRWIN, 2000). De acordo com (OKUNO; CALDAS; CHOW, 1982), em um condutor ou solução eletrolítica, define-se a intensidade média de corrente elétrica I através de uma área A como

$$I = \frac{\Delta Q}{\Delta t} \tag{2.1}$$

onde ΔQ é a carga elétrica total que atravessa a área A durante o intervalo de tempo Δt. As unidades de I e ΔQ são, respectivamente, ampère (A) e coulomb (C), sendo que

$$1\,A = \frac{1\,C}{1\,s}. \tag{2.2}$$

Se houver uma diferença de potencial ΔV entre dois pontos de um condutor, consequentemente haverá uma corrente elétrica I. Define-se a *resistência elétrica* R como a razão

$$R = \frac{\Delta V}{I}. \tag{2.3}$$

A corrente I será positiva se as cargas elétricas positivas se deslocarem no sentido das linhas de forças do campo elétrico $\vec{E}$ (ou cargas negativas no sentido contrário); nesse caso a diferença de potencial ΔV será negativa. Assim, pela definição pode-se concluir que $R > 0$ (OKUNO; CALDAS; CHOW, 1982). A unidade de resistência elétrica é o ohm (Ω):

$$1\,\Omega = 1\frac{V}{A}. \tag{2.4}$$

Quando R for constante, a corrente elétrica I será proporcional à diferença de potencial ΔV. Nesse caso, a equação 2.3 corresponde à lei de Ohm (IRWIN, 2000).

2.1.2 Densidade de corrente

Além disso, em cada instante, pode-se definir *a densidade de corrente elétrica* por unidade de área como

$$j = \frac{I}{A}. \tag{2.5}$$

A unidade de j é A/m^2 (OKUNO; CALDAS; CHOW, 1982).

A corrente elétrica, em soluções eletrolíticas, é originada pelo deslocamento dos íons. Além do movimento de agitação térmica, essas partículas possuem um movimento devido ao campo elétrico $\vec{E}$ que produz a corrente elétrica. Esse movimento ordenado de cargas elétricas é que constitui a corrente elétrica. A relação entre a densidade de corrente j^E e a intensidade do campo elétrico E é

$$j^E = \frac{1}{\rho}E \;\; ou \;\; \rho = \frac{E}{j^E} \tag{2.6}$$

onde ρ é a *resistividade elétrica* do meio considerado. Por exemplo, o líquido (axoplasma) no interior de uma célula nervosa de uma lula é um líquido condutor com resistividade elétrica $\rho \cong 0.6\,\Omega \cdot m$ (DURAN, 2003).

Em uma solução eletrolítica, a resistividade ρ_i é diferente para cada tipo de íon i. Pode-se verificar experimentalmente que ρ_i é inversamente proporcional ao quadrado da carga elétrica q_i e à concentração C_i (número de íons por unidade de volume) desse íon, ou seja,

$$\frac{j^E}{E} = \frac{1}{\rho_1} = \mu_i q_i^2 C_i. \tag{2.7}$$

A constante μ_i é denominada de constante de mobilidade. A partir das fórmulas 2.6 e 2.7 pode-se escrever

$$j_i^E = \mu_i q_i^2 C_i E \tag{2.8}$$

(OKUNO; CALDAS; CHOW, 1982). A equação 2.8 descreve a densidade de corrente elétrica para um dado íon i.

2.1.3 Difusão iônica e equação de Nernst-Plank

Nem sempre as concentrações iônicas são uniformes e homogêneas. Quando isso for verificado significa que haverá um processo denominado *difusão* que poderá uniformizar essas concentrações. O processo de difusão, em uma solução iônica a uma temperatura T (em K), está relacionado ao movimento de agitação térmica dos íons. Os íons colidem frequentemente com as moléculas do solvente, dando origem a um movimento aleatório sem nenhuma direção preferencial (OKUNO; CALDAS; CHOW, 1982). A distribuição de um grande número de íons caracteriza-se por sua concentração C_i, correspondente ao número desses íons por unidade de volume. Se essa concentração C_i não for uniforme, a agitação térmica dos íons fará a concentração se uniformizar. Isso corresponde a um fluxo de cargas elétricas, ou seja, a uma densidade de corrente elétrica (TUCKWELL, 1988b).

Se para os íons i, o gradiente da concentração iônica $\frac{\Delta C_i}{\Delta x}$ for uniforme na direção x, haverá uma densidade de corrente elétrica j_i^D, devida à difusão, proporcional a esse gradiente

$$j_i^D = -q_i D_i \frac{\Delta C_i}{\Delta x} \tag{2.9}$$

onde D_i é a constante de difusão para os íons i. O sinal (-) indica que a densidade de corrente elétrica j_i é no sentido da diminuição da concentração C_i. A equação 2.9 é conhecida como *Lei de Flick*.

A constante de mobilidade μ_i e a constante de difusão D_i estão relacionadas por

$$\mu_i = \frac{D_i}{kT} \tag{2.10}$$

onde $k = 1.38 \times 10^{-23} J/K$ é a constante de Boltzmann. Essa relação expressa a influência

da temperatura T da solução na difusão iônica. Assim, a densidade j_i^D devida à difusão iônica pode ser escrita como

$$j_i^D = -q_i \mu_i kT \frac{\Delta C_i}{\Delta x}. \tag{2.11}$$

Em uma solução com concentrações iônicas não uniformes, e na presença de um campo elétrico $\vec{E}$, a densidade de corrente para cada tipo de íon é

$$j_i = j_i^D + j_i^E \tag{2.12}$$

onde j_i^D é a densidade de corrente devida à difusão iônica, e j_i^E é a densidade de corrente induzida pelo campo elétrico $\vec{E}$. Considerando também que todas as grandezas (j, C, E etc) dependam apenas de uma direção - x, podemos escrever o campo elétrico como sendo

$$E = -\frac{dV}{dx}. \tag{2.13}$$

Assim, a densidade de corrente j_i, para íons de carga elétrica q_i pode ser definida como

$$j_i = -q_i \mu_i \left(kT \frac{\Delta C_i}{\Delta x} + q_i C_i \frac{dV}{dx} \right). \tag{2.14}$$

Esta é a *Equação de Nernst-Plank*. Ela é fundamental para a compreensão do potencial de uma célula (TUCKWELL, 1988b).

2.2 O potencial de repouso

Entre o líquido no interior de uma célula e o fluido extracelular há uma diferença de potencial elétrico, denominada *potencial de membrana*. Este potencial pode ser medido ligando-se simultaneamente, por meio de microeletrodos, os pólos de um medidor de tensão no interior de uma célula e ao líquido extracelular. Estes eletrodos são em geral capilares de vidro, com uma ponta com menos de $1\mu m$ de diâmetro, contendo uma solução condutora de KCl (OKUNO; CALDAS; CHOW, 1982). Alguns experimentos realizados nas décadas de 1940 e 1950, como os trabalhos de Curtis e Cole (CURTIS; COLE, 1940; CURTIS; COLE, 1942) e de Hodgkin e Huxley (HODGKIN; HUXLEY, 1952), demonstraram que quase sempre, o potencial de

membrana de uma célula é negativo, apresentando um valor constante e característico (KOCH, 1999).

O potencial de membrana é obtido calculando-se a diferença entre os potenciais elétricos dos meios intra e extracelular, formalmente expresso como (KOCH, 1999):

$$V_m(t) = V_i(t) - V_e(t). \tag{2.15}$$

Quando as pontas dos dois eletrodos estão no meio externo, a diferença de potencial medida ΔV é nula, indicando que o potencial elétrico é o mesmo em qualquer ponto deste meio. O mesmo aconteceria se os dois eletrodos estivessem inseridos no interior da célula, pois ambos os meios são condutores. Por convenção, o potencial elétrico do fluido extracelular é considerado nulo, e o potencial elétrico no interior de membrana possui um valor V. Assim, a diferença de potencial ΔV entre os dois meios é

$$\Delta V = V - 0 = V \tag{2.16}$$

Na maioria das células, o potencial de membrana V permanece inalterado, desde que não haja influências externas. Quando a célula se encontra nessa condição, dá-se ao potencial de membrana V a designação de **potencial de repouso** representado por V_{rest}. Dizer que uma célula encontra-se em repouso na verdade significa dizer que ela permanece em um estado de equilíbrio dinâmico (KOCH, 1999). Nas fibras nervosas dos animais de sangue quente, os potenciais de repouso se situam entre $-55mV$ e $-100mV$. Nas fibras dos músculos lisos, os potenciais de repouso estão entre $-30\,mV$ e $-55\,mV$ (DURAN, 2003).

2.3 Origem do potencial de repouso

Tanto o interior da célula, como o meio extracelular estão preenchidos por uma solução salina. Em soluções salinas muito diluídas, a maior parte das moléculas se decompõe em íons, que movem-se livremente pela solução aquosa. Os fluidos dentro e fora da célula são sempre neutros, isto é, a concentração de ânions em qualquer local é sempre igual à de cátions, não podendo haver um acúmulo local de cargas elétricas nesses fluidos. Podemos então imaginar a membrana celular exercendo o papel de um capacitor, no qual as duas ficam separadas e isoladas pela *membrana* (OKUNO; CALDAS; CHOW, 1982). As cargas elétricas em

excesso, $+Q$ e $-Q$, que provocam a formação do potencial de repouso se localizam em torno da membrana celular. Esse potencial se origina também na membrana celular: a superfície interna da membrana é coberta pelo excesso de ânions $(-Q)$, enquanto que, na superfície externa, há o mesmo excesso de cátions $(+Q)$ (OKUNO; CALDAS; CHOW, 1982).

Biologicamente, as estruturas responsáveis pela característica isolante da membrana são as proteínas e os lipídios. O suporte principal da membrana é composto de duas camadas de moléculas de fosfolipídios, cujas "cabeças" polares encontram-se posicionadas de frente para o citoplasma intracelular e para o espaço extracelular, separando a solução condutora interna da externa (KOCH, 1999). O conjunto formado pela membrana e pelas cargas elétricas adjacentes irá exercer a função de um capacitor, cuja unidade que o caracteriza é a **capacitância**. A capacitância, denotada por C

$$C = \frac{Q}{V}. \tag{2.17}$$

é uma medida de quanta carga Q tem de ser distribuída através da membrana para que um certo potencial elétrico V_m aumente. A carga Q armazenada no capacitor é proporcional a diferença de potencial elétrico da membrana V_m:

$$Q = CV_m. \tag{2.18}$$

Aqui, a capacitância é comumente especificada em termos da *capacitância específica de membrana* C_m, em unidades de microfarads por centímetro quadrado da área da membrana $(\mu F/cm^2)$. O valor real de C pode ser obtido multiplicando-se C_m pela área total da membrana. Quando a tensão sobre a capacitância é alterada, uma corrente elétrica é gerada. Esta *corrente capacitiva*, é obtida derivando-se a equação 2.18 em relação ao tempo,

$$I_C = C\frac{dV_m\,(t)}{dt}. \tag{2.19}$$

Para um determinado valor fixo de corrente, a existência da capacitância da membrana impõe uma restrição de quão rapidamente V_m pode variar em resposta a esta corrente; quanto maior a capacitância, menor será a mudança da tensão resultante. Quando a tensão é alterada de acordo com o tempo, a carga elétrica também é alterada e uma corrente fluirá, como descrito pela equação 2.19, porém nunca diretamente através da capacitância. A carga simplesmente será distribuída entre os dois lados através do restante do circuito (KOCH,

1999).

A alta resistividade dos lipídios previne a passagem de qualquer quantidade significante de carga pela membrana. Na realidade, a resistividade específica da membrana é aproximadamente um bilhão de vezes maior que a resistividade do citoplasma intracelular. Deste modo, do ponto de vista elétrico, as propriedades da membrana podem ser satisfatoriamente descritas por um único elemento: a capacitância (KOCH, 1999).

2.4 Resistência da membrana

Além das estruturas já citadas, existem também diversas *proteínas* fixadas na membrana celular. Na verdade, elas frequentemente penetram a membrana, permitindo que íons passem de um lado para outro. Moléculas proteicas compõem de 20 a 80% da membrana, sendo úteis para uma variedade de funções celulares, incluindo a formação de canais iônicos, enzimas, bombas e receptores. Elas funcionam como passagens através da barreira lipídica onde informações e susbtâncias podem ser transferidas além da membrana celular (KOCH, 1999). Existem diversos íons em fluxo pela membrana celular, (Na^+, K^+, Cl^-) e um dos principais papéis exercidos por tais proteínas é a função de canal iônico. Estes canais permitem a passagem dos íons de um lado ao outro da membrama, mantendo inalterado o potencial de membrana da célula (DURAN, 2003).

As concentrações iônicas nos fluidos, dentro e fora das células, são diferentes. Na parte interna a concentração de íons K^+ é bem maior que na parte externa. O oposto ocorre com os íons Cl^- e Na^+. A maior parte dos ânions intracelulares não são íons de Cl^-, mas grandes ânions protéicos designados por A^-(OKUNO; CALDAS; CHOW, 1982).

Podemos descrever o fluxo de corrente sobre tais canais através de uma resistência linear R. Levando-se em conta a existência do potencial de repouso da célula, o modelo elétrico mais simples que pode descrever uma membrana possui apenas três elementos: (1) uma capacitância C, (2) uma resistência R e (3) uma diferença de potencial V_{rest}. Por razões óbvias, este modelo também pode ser conhecido como um *circuito Resistor-Capacitor*, ou apenas circuito RC (KOCH, 1999).

A resistência da membrana é comumente especificada como *resistência específica da membrana* R_m, cuja unidade estabelece-se em termos do valor da resistência por unidade de área ($\Omega \times cm^2$). R é obtida dividindo-se R_m pela área da membrana celular. O inverso de R_m é definido como *condutância específica de escoamento* $G_m = 1/R_m$, é medida em unidades de siemens por centímetro quadrado (S/cm^2)(KOCH, 1999).

De acordo com Kock(1999) podemos imaginar a membrana celular como uma membrana composta por diversos microcircuitos RC. Entretanto, como as dimensões da célula são pequenas, o potencial elétrico além da membrana é o mesmo em qualquer ponto, motivo ao qual fisiologistas descrevem a membrana sendo *isopotencial*. Isto implica que o comportamento elétrico de uma célula pode ser adequadamente descrito como um único circuito dada uma fonte de corrente (Fig.2.2). A resistência do modelo R é calculada pela divisão da resistência específica da membrana R_m pela área total da membrana πd^2, enquanto que a capacitância total C é dada por C_m vezes a área da membrana.

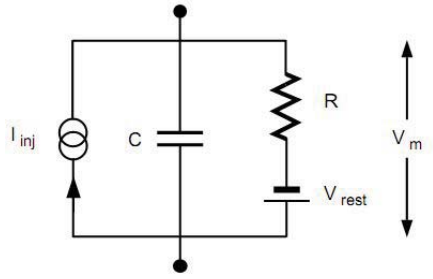

Figura 2.2: Circuito esquemático para membrana neuronal [Adaptado de (KOCH, 1999)].

Desta maneira, a dinâmica deste circuito pode ser descrita aplicando-se a *primeira lei de Kirchhoff* (IRWIN, 2000), que diz que a soma de todas as correntes fluindo para dentro ou para fora de qualquer nodo elétrico é nula. A corrente sobre a capacitância é dada pela equação 2.19. Já a corrente sobre a resistência é dada pela *Lei de Ohm*

$$I_R = \frac{V_m - V_{rest}}{R}. \tag{2.20}$$

Devido à conservação da corrente, as correntes capacitivas e resistivas devem ser iguais à corrente externa,

$$C\frac{dV_m(t)}{dt} + \frac{V_m(t) - V_{rest}}{R} = I_{inj}(t). \tag{2.21}$$

Seja $\tau = RC$, definido em unidades de $\Omega \times F = segundos$, podemos reescrever a equação acima como

$$\tau\frac{dV_m(t)}{dt} = -V_m(t) + V_{rest} + RI_{inj}(t). \tag{2.22}$$

Um pequeno e importante detalhe é o sinal da corrente externa. Por con-

venção, a corrente que flui de dentro do neurônio para o meio externo, é representada como sendo positiva (KOCH, 1999).

A equação 2.22 é conhecida como *Equação da Membrana*, e é uma equação diferencial ordinária de primeira ordem. Com as condições iniciais adequadas, a equação da membrana descreve uma trajetória única da tensão elétrica da célula, similar ao comportamento de carga e descarga de um capacitor (fig.2.3).

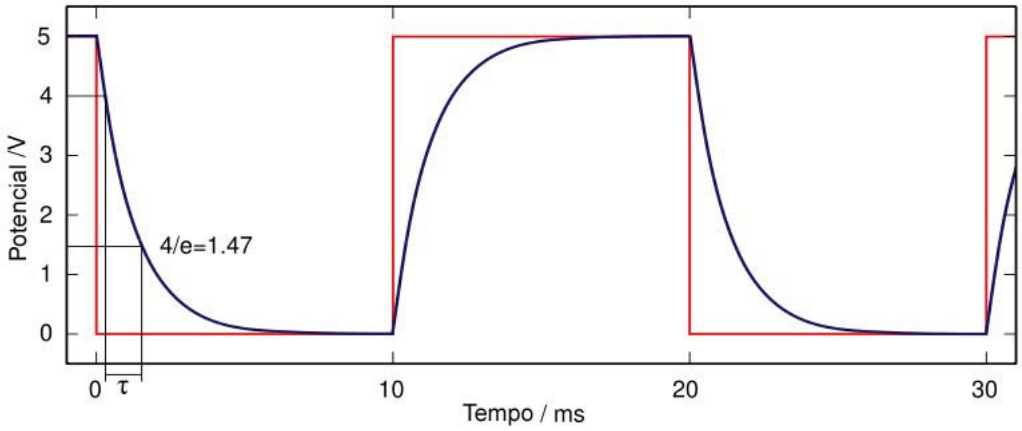

Figura 2.3: Comportamento de carga e descarga de um capacitor [Adaptado de (FOUNDATION, 2010)].

2.5 Equilíbrio de Donnan

O potencial de repouso é sempre observado quando há diferenças de concentrações iônicas dentro e fora da célula. Assim, essas diferenças de concentrações devem estar de alguma forma ligadas à existência desse potencial. Além disso, as diferentes concentrações iônicas logo se igualariam por difusão, se isso não fosse impedido pela membrana celular. Caso a membrana fosse completamente impermeável, as concentrações permaneceriam indefinidamente inalteradas em ambos os lados da membrana (OKUNO; CALDAS; CHOW, 1982).

A membrana celular, porém, não é completamente impermeável, podendo ocorrer a passagem de certos íons. De modo geral, ela é mais permeável para íons monovalentes inorgânicos e pequenos, bem menos para íons multivalentes e totalmente impermeável para íons orgânicos complexos (fosfatos orgânicos) e proteínas (DURAN, 2003). Sendo assim, a atividade elétrica no neurônio é mantida e propagada por correntes iônicas através da membrana neuronal. A maioria destas correntes transmembranais envolvem um destes quatro tipos de íons: sódio (Na^+), potássio (K^+), cálcio (Ca^{2+}) ou cloro (Cl^-) (IZHIKEVICH, 2007a).

As concentrações destes íons são diferentes dentro e fora da célula, criando gradientes eletroquímicos - as principais forças motrizes da atividade neural. O meio extracelular apresenta uma alta concentração de Na^+ e Cl^- e uma concentração relativamente alta de Ca^{2+}. Já o meio intracelular contém alta concentração de K^+ e de moléculas carregadas negativamente (denotadas por A^-), como pode ser observado na Fig. 2.4 (IZHIKEVICH, 2007a).

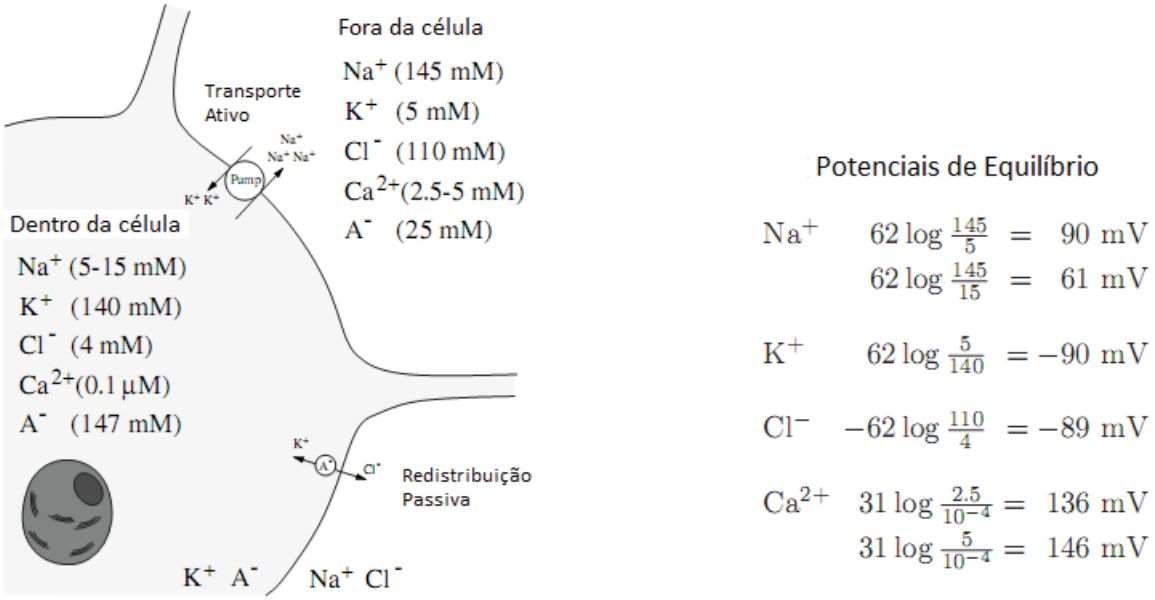

Figura 2.4: Concentrações iônicas e potencial de equilíbrio de Nernst em um típico neurônio de mamífero [Adaptado de (IZHIKEVICH, 2007a).]

Ao contrário da concentração de Cl^-, a concentração intracelular de K^+ não pode se modificar substancialmente. Os íons de potássio são necessários para manter a neutralidade elétrica no interior da célula, devido à presença de ânions intracelulares (A^-). Os ânions intracelulares são, principalmente, grandes moléculas de albumina, que não atravessam a membrana; sua concentração no interior da célula é, portanto, constante. A solução intracelular é eletricamente neutra, sendo o número de ânions igual ao de cátions. Como a concentração intracelular de Na^+ se mantém muito baixa, a neutralidade da solução deve ser garantida pelos íons K^+ (OKUNO; CALDAS; CHOW, 1982).

Por causa da presença de ânions impermeáveis A^- no interior da célula, a concentração dos cátions permeáveis deve ser maior que a de ânions permeáveis. Além disso, a concentração de íons K^+ é diferente dentro e fora da célula. Como a célula é permeável para íons de potássio, então é necessário que haja uma diferença de potencial elétrico através da membrana, para manter essa diferença de concentrações. Por ter uma concentração maior na parte interna da célula, os íons K^+ tendem a sair para o meio externo, atravessando a membrana. Contudo, devido à existência do potencial de repouso V_{rest}, uma força elétrica

dirigida para o interior da célula atua na membrana sobre cada um desses íons. Essa difusão produzirá na membrana um acúmulo de cargas elétricas positivas na superfície externa e negativas na superfície interna da célula. Assim, dois fenômenos físicos ocorrem em sentidos contrários, devendo haver um equilíbrio entre eles. Se uma dessas concentrações fosse alterada, o equilíbrio atingido seria diferente (fig.2.5). Raciocínio análogo pode ser desenvolvido para os outros íons permeáveis.

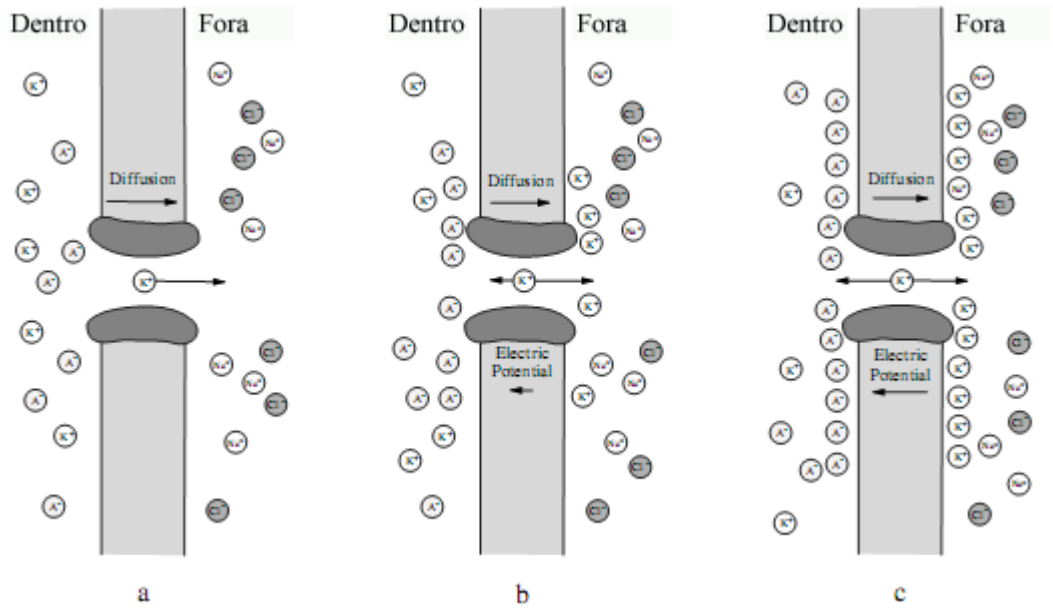

Figura 2.5: Difusão dos íons K^+ em direção ao gradiente de concentração através da membrana (a) cria-se uma força de potencial elétrico que aponta na direção oposta (b) até que as forças de difusão e elétrica se oponham uma a outra (c) [Adaptado de (IZHIKEVICH, 2007a)].

O *modelo de Donnan* considera a membrana uma barreira porosa, através da qual alguns íons monovalentes $(q = \pm e)$ podem se mover. O fluxo de cada tipo de íon permeável i corresponde à passagem de uma densidade de corrente elétrica j_i. Em equilíbrio, as concentrações iônicas interna e externa permanecem constantes,

$$j_i = 0 \tag{2.23}$$

e o campo elétrico através da membrana não se altera. Como foi descrito na seção anterior, a densidade de corrente elétrica (j), para cada tipo de íons que atravessa a membrana, é dada pela Equação de Nernst-Plank (2.14). O primeiro e segundo termos do lado direito da equação correspondem respectivamente às densidades de corrente elétrica, devidas ao gradiente de concentração iônica e ao campo elétrico através da membrana. A partir das equações 2.14 e 2.23 obtém-se a condição de equilíbrio

$$kT\frac{dC_i}{dx} + q_i C_i \frac{dV}{dx} = 0 \tag{2.24}$$

válida para cada tipo de íon permeável. Esta relação de equilíbrio entre as concentrações dos íons é conhecida como "equilíbrio de Donnan". Se a membrana for permeável a mais de duas espécies iônicas, o equilíbrio de Donnan exige que esses íons estejam distribuídos conforme o potencial de membrana. Assim, os íon se redistribuirão até alcançar esta situação de equilíbrio (DURAN, 2003).

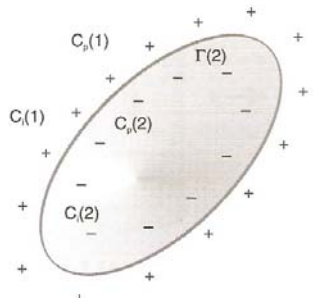

Figura 2.6: Membrana celular e distribuição de cargas elétricas [Adaptado de (DURAN, 2003)].

Na equação diferencial 2.24, C_i e V variam com a coordenada x no interior da membrana e são constantes no fluido extracelular (1) e no interior da célula (2) (Fig. 2.6). Resolvendo-se essa equação pode-se obter uma relação entre a diferença de potencial elétrico e as concentrações iônicas nos meios (1) e (2). De acordo com (OKUNO; CALDAS; CHOW, 1982), uma solução dessa equação é

$$V(2) - V(1) = -\frac{kT}{q_i} \ln \frac{C_i(2)}{C_i(1)}. \tag{2.25}$$

O lado esquerdo da equação 2.25 corresponde à diferença de potencial elétrico através da membrana e é o mesmo para qualquer tipo de íon. Essa diferença de potencial é denominada *potencial de Donnan* V^D

$$V^D = V(2) - V(1) \tag{2.26}$$

e deve ser igual ao potencial de repouso V_{rest}. O termo do lado direito da equação 2.26, dependente das concentrações iônicas, é conhecido como *potencial de Nernst*:

$$V_i^N = -\frac{kT}{q_i} \ln \frac{C_i(2)}{C_i(1)}. \tag{2.27}$$

Os potenciais V_i^N são calculados, para os íons i, a partir das medidas das concentrações iônicas $C_i(2)$ e $C_i(1)$. Através da equação 2.25 e das definições de V^D e V_i^N, observa-se que em equilíbrio,

$$V^D = V_i^N \tag{2.28}$$

para todos os íons permeáveis.

2.6 O fluxo de Na^+

Entretanto, experimentalmente, como relatado em (OKUNO; CALDAS; CHOW, 1982) pode-se verificar que apenas os potenciais de Nernst para os íons de potássio e cloro estão próximos ao valor observado do potencial de repouso (V_{rest}). Tal situação parece indicar que a membrana dessas células seria permeável apenas aos íons de potássio e cloro. Contudo, sabe-se que a membrana também é permeável aos íons Na^+, o que implica que apenas o equilíbrio de Donnan não é suficiente para explicar a baixa concentração de sódio no interior de células nervosas.

A concentração intracelular de K^+ é determinada pela concentração dos grandes ânions do interior da célula. A concentração intracelular do Cl^- ocorre de acordo com a membrana, sendo portanto uma consequência da distribuição dos íons K^+. Experimentos em laboratório demonstraram que apenas para altas concentrações extracelulares de potássio, os valores medidos concordam com a previsão teórica, enquanto que para baixas concentrações os valores medidos são maiores que os calculados.

Essa diferença entre os valores medidos é causada pela presença de íons de Na^+. Quando a solução contém Na^+, uma diferença entre V_{rest} e V_K^N é causada pelo afluxo de íons Na^+ através da membrana. Assim, esses íons, cuja concentração extracelular é bem maior que a interna, penetram na célula, anulando em parte o excesso de cargas elétricas negativas nas superfícies da membrana e tornando o seu potencial menos negativo.

O uso de isótopos radioativos permite verificar diretamente se um determinado tipo de íon i atravessa a membrana celular. Para isto são adicionados isótopos radioativos

desse íon ao meio externo ou interno. Medindo-se o número de isótopos radioativos que aparecem no outro meio é possível saber o fluxo desses íons através da membrana (OKUNO; CALDAS; CHOW, 1982).

Usando esta técnica, experiências com células nervosas e musculares mostraram que a membrana é permeável aos íons de sódio, embora menos do que aos íons de potássio ou cloro. Em (OKUNO; CALDAS; CHOW, 1982), os autores citam duas experiências utilizadas para compreender melhor o fluxo de sódio através da membrana. Nessas experiências, isótopos radioativos de $^{24}Na^{+}$ são adicionados ao interior da célula. O número desses íons que podem abandonar a célula devido à difusão deve ser insignificante devido a sua carga elétrica positiva e à menor concentração de sódio no interior da célula. Assim, os íons só podem ser levados para fora da célula com dispêndio de energia.

Na primeira experiência, inicialmente a densidade de corrente de $^{24}Na^{+}$ diminui lentamente a $18,3°C$, pois a própria corrente faz diminuir a concentração intracelular de $^{24}Na^{+}$. A seguir o neurônio é subitamente resfriado a $0,5°C$ o que provoca uma queda na densidade da corrente de escoamento dos isótopos $^{24}Na^{+}$. Com o reaquecimento, essa densidade retorna ao nível anterior. A grande variação de j^{*}_{Na} com a temperatura implica na existência de um processo químico *ativo* e não uma difusão *passiva*.

Na segunda experiência, inicialmente há um escoamento de íons $^{24}Na^{+}$, que foram adicionados ao meio intracelular. A seguir, foi adicionado dinitrofenol (DNP) à solução extracelular, o que provoca uma queda contínua e acentuada na densidade de corrente elétrica de escoamento até que ela, se torna praticamente nula. Após a dissolução do DNP, a densidade de corrente elétrica volta a se normalizar. O DNP é um veneno que penetra na célula e bloqueia os processos metabólicos do fornecimento de energia, sem entretanto afetar os processos de difusão através da membrana. A diminuição da densidade de corrente do $^{24}Na^{+}$, em presença do DNP, é portanto causada pela carência de energia metabólica (OKUNO; CALDAS; CHOW, 1982).

Os resultados dessas duas experiências mostram que o transporte de Na^{+} através da membrana não é apenas passivo, mas depende de um fornecimento de energia.

2.7 As Bombas de sódio

Uma célula no estado de equilíbrio, segundo o modelo de Donnan, apresenta um afluxo constante de íons de sódio para o interior da célula e um escoamento constante de íons de potássio para o fluído externo. Isso ocorre porque os potenciais de Nernst V^{N}_{Na} e V^{N}_{K}

diferem do potencial de repouso V_{rest}. Se, através da membrana, houvesse apenas o transporte passivo, a célula teria suas concentrações iônicas alteradas. Como essas concentrações são constantes, tal observação implica na existência de um outro tipo de transporte, denominado *ativo*, por ocorrer com dispêndio de energia (DURAN, 2003).

Sem o transporte ativo de íons através da membrana, haveria uma diminuição constante na concentração intracelular de K^+ e, consequentemente, um aumento do potencial de repouso V_{rest}. Com um potencial de repouso menos negativo, a concentração intracelular de Cl^- aumentaria. Assim, as concentrações intracelulares de Na^+, K^+ e Cl^- se aproximariam das extracelulares, deixando de existir o potencial de repouso V_{rest}. Isso causaria o desaparecimento da capacidade funcional das células, provocando nelas lesões irreversíveis. O abastecimento energético da célula, através de seu metabolismo, necessário para a manutenção do sistema de transporte ativo de Na^+ e K^+, pode se tornar insuficiente por uma carência de oxigênio ou ainda por um envenenamento (DURAN, 2003).

De acordo com (OKUNO; CALDAS; CHOW, 1982) existem dois tipos de transporte ativo, através das membranas celulares, presentes em quase todas as células. O primeiro é o transporte de nutrientes orgânicos essenciais para o interior da célula, como a glicose e os aminoácidos; o segundo é o transporte de íons para manter as concentrações intracelulares de Na^+ e K^+ necessárias à célula.

O transporte ativo de Na^+ e K^+ através da membrana celular é realizado por uma proteína complexa existente na membrana, denominada "*sódio-potássio-adenosina-trifosfatase*", ou simplesmente **bomba de sódio**. Quando a bicamada lipídica está isenta de proteínas, a membrana é altamente impermeável a íons, não importando quão pequenos eles sejam (DURAN, 2003).

Nos neurônios do cérebro humano, a energia armazenada em uma molécula de ATP (trifosfato de adenosina) liberada pela reação

$$ATP \rightarrow ADP + P + energia \tag{2.29}$$

é utilizada para retirar 3 íons de Na^+ da célula e levar 2 íons de K^+ para seu interior. Cada bomba de sódio desses neurônios pode transportar, por segundo até 200 Na^+ para fora e 130 K^+ para dentro da célula. Contudo, esse transporte é ajustado à necessidade da célula. Um pequeno neurônio possui cerca de um milhão de bombas de sódio, que podem transportar aproximadamente 200 milhões de Na^+ por segundo (OKUNO; CALDAS; CHOW, 1982).

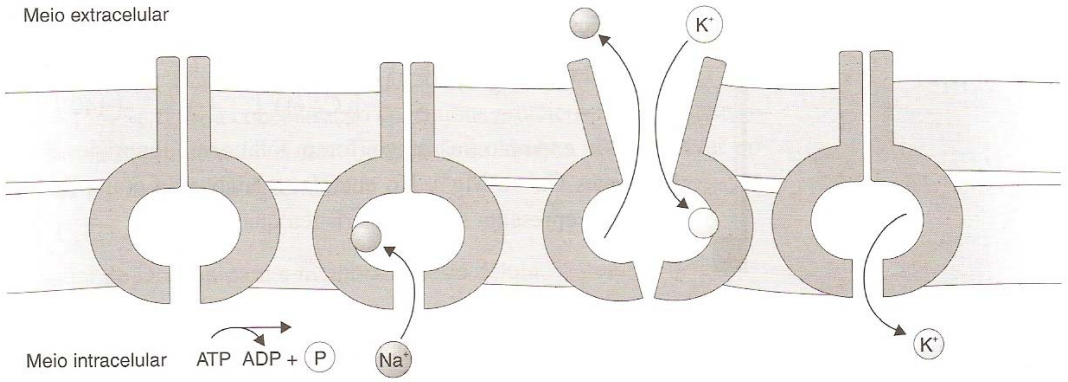

Figura 2.7: Esquema simplificado de funcionamento de uma bomba de sódio [Adaptado de (DURAN, 2003)].

O funcionamento de uma bomba acoplada de sódio e potássio pode ser visualizado na Fig. 2.7. Os íons de Na^+ na superfície interna da membrana se combinam com moléculas transportadoras denotadas por Y e o complexo resultante NaY pode atravessar a membrana. Ao chegar à superfície externa da membrana, esse complexo pode se desfazer, liberando o íon Na^+. Com a ajuda da molécula transportadora Y, os íons de Na^+ atravessam a membrana contra suas quedas de concentração e potencial.

Após se dissociar do Na^+, a molécula transportadora Y é transformada na molécula transportadora X. Essa reação é acelerada por uma enzima contida na membrana. A molécula X une-se a um íon extracelular K^+, formando KX, que atravessa a membrana para dentro da célula. Na superfície interna da membrana, KX se desfaz, tendo como resultado o transporte de K^+ para o interior da célula. O transporte de NaY para fora da célula e de KX para o seu interior ocorre por difusão, devido aos seus gradientes de concentração. Um novo ciclo de transporte é iniciado quando a molécula transportadora X, utilizando energia celular, se transforma na molécula transportadora Y.

O complexo NaY é eletricamente neutro. Durante o processo de transporte, nenhuma carga elétrica atravessa a membrana. Assim, o transporte acoplado de Na^+ e K^+ permite economizar energia. As células despendem até 20% de sua energia metabólica para manter o funcionamento das bombas de sódio. Esse dispêndio de energia seria ainda maior se não fosse o acoplamento discutido anteriormente (DURAN, 2003).

Todas estas características e propriedades dos neurônios, referentes a processos iônicos, caracterizam a emissão de potenciais de ativação. Tais potenciais são gerados próximos ou no corpo celular dos neurônios, sendo propagados através do axônio com velocidade e amplitude constantes. O axônio também é caracterizado por possuir uma alta resistência elétrica, além de uma capacitância muito grande, o que permite que tal estrutura

possa ser modelada como uma linha de transmissão Resistor-Capacitor (RC) (HAYKIN, 1999).

2.8 Spikes

O potencial de repouso é uma condição necessária para que células nervosas e musculares possam exercer suas funções específicas no organismo. Às células nervosas cabe a função de recolher informações, distribuí-las pelo corpo e coordená-las. As células musculares, comandadas pelas células nervosas, podem se contrair ou relaxar. Durante o desempenho dessas funções, surgem alterações breves e características no potencial de membrana dessas células, originando o fenômeno conhecido como *potencial de ativação* do neurônio (ou por *spike*) (OKUNO; CALDAS; CHOW, 1982).

2.8.1 O potencial de ativação

Do ponto de vista de sistemas dinâmicos, neurônios são excitáveis porque eles estão perto de uma transição, chamada *bifurcação*, entre o estado de repouso e o estado de atividade sustentada de *spikes*. Enquanto existe um grande número de possíveis mecanismos iônicos de excitabilidade e geração de *spikes*, existem apenas quatro mecanismos de bifurcação que resultam em tal transição. Considerando a configuração e forma de um espaço de fase de uma bifurcação é possível compreender algumas propriedades computacionais dos neurônios, como a natureza do valor de limiar, a coexistência de estados de repouso e *spikes*, etc. Além disso, podemos compreender como estas propriedades estão inter-relacionadas, porque algumas são equivalentes e outras são mutuamente exclusivas (IZHIKEVICH, 2007a). De maneira geral, podemos esquematizar o funcionamento de um neurônio de acordo com a fig.2.8:

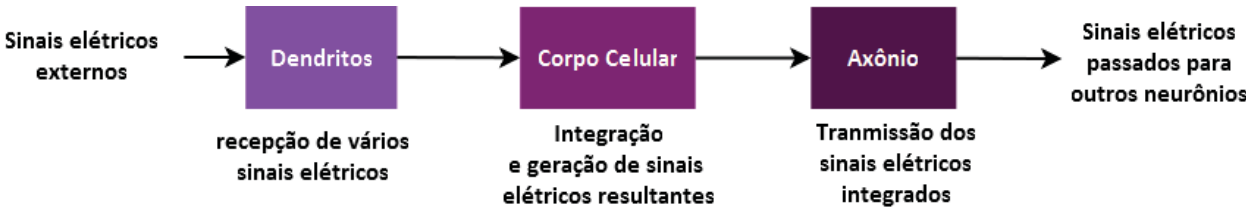

Figura 2.8: Funcionamento de um neurônio.

Quando um neurônio está em *repouso*, o potencial de membrana V_m permanece em um valor negativo e constante. Entretanto, um estímulo externo às células nervosas

e musculares produz uma variação v em seus potenciais de membrana

$$V_M = V_{rest} + v.$$

Essa variação rápida que se propaga ao longo de uma dessas células é denominada *potencial de ação* (DURAN, 2003). Um típico neurônio recebe conexões de mais de 10.000 outros neurônios através das sinapses. Estas "conexões " produzem correntes elétricas transmembranais que modificam o potencial de membrana do neurônio. Tais correntes sinápticas produzem alterações, os potenciais pós-sinápticos ($PSPs$). Pequenas correntes produzem pequenos $PSPs$; correntes maiores produzem significantes $PSPs$ que podem ser amplificados pelos canais sensíveis a voltagem, localizados na membrana neuronal, e desta maneira ocasionar a geração de um potencial de ação ou *spike* - uma mudança abrupta e transitória do potencial da membrana que se propaga aos outros neurônios por meio do seu axônio. Estes *spikes* compõem o principal meio de comunicação entre os neurônios. Em geral, neurônios não disparam por conta própria; eles disparam como consequência dos *spikes* recebidos dos outros neurônios (IZHIKEVICH, 2007a).

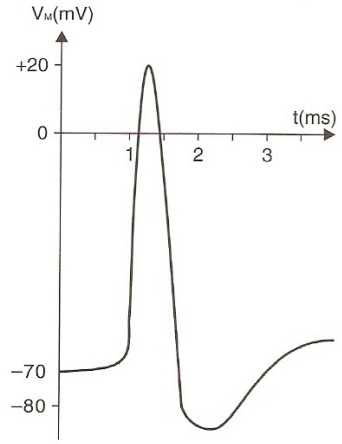

Figura 2.9: Forma típica do potencial de membrana de uma célula nervosa [Adaptado de (DURAN, 2003)].

Enquanto o potencial de repouso V_{rest}, medido em uma posição fixa, é constante, o potencial de membrana V_M, nessa mesma posição, durante a passagem de um potencial de ação, varia com o tempo. Em todos os potenciais de ação medidos, partindo do potencial de repouso V_{rest}, o potencial V_M se eleva rapidamente a um valor positivo e volta mais lentamente ao potencial de repouso, como pode ser observado na Fig. 2.9. Nas células excitáveis, se o potencial V_M em uma região da membrana celular ultrapassar em um instante t_0 um valor mínimo característico do tipo de célula, o *potencial limiar* V_L, ocorrerá a transmissão de um pulso de potencial elétrico através da membrana (KOCH, 1999).

Após a ocorrência de um potencial de ação, há uma fase de recuperação das propriedades químicas e físicas originais da membrana celular, na qual nenhum outro impulso nervoso pode ser deflagrado. Devido a este *período refratário*, ao ser o neurônio estimulado prolongadamente com um potencial despolarizante acima do limiar, gera-se em seu axônio uma série de potenciais de ação cuja frequência de ocorrência é determinada pela duração da fase refratária da célula. Dessa forma, a informação transmitida nas redes neuronais do sistema nervoso central é codificada na frequência do que pode ser chamado de *cadeia de impulsos* (CARVALHO, 1989).

A forma desse pulso e sua velocidade são características de cada tipo de célula e independem da amplitude da perturbação (desde que $V_M < V_L$). Esse pulso se propaga sem alterar sua forma através da célula com velocidade constante. Se a amplitude da perturbação inicial for menor que o valor limiar $V_M < V_L$, a perturbação é rapidamente atenuada. Portanto, existem apenas duas possíveis situações para uma perturbação localizada, conhecidas como **lei de excitação tudo-ou-nada**:

a. a propagação de um pulso de potencial elétrico característico ao longo de uma célula; ou

b. a rápida atenuação da perturbação, sem que ocorra a excitação da célula.

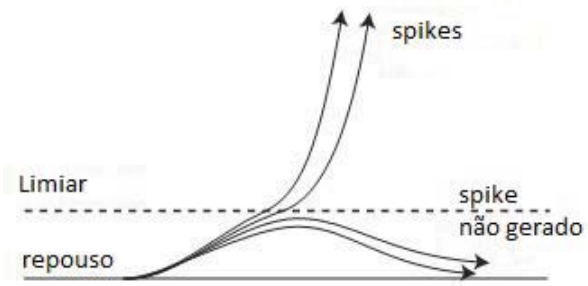

Figura 2.10: O conceito de limiar de disparo [Adaptado de (IZHIKEVICH, 2007a)].

A Fig.2.10 representa um *spike* - a curva que descreve a variação do potencial de membrana V_M com o tempo durante a propagação de um potencial de ativação. O limiar da figura representa o valor limiar V_L, quando o valor do potencial de membrana é menor que o limiar, o neurônio permanece em repouso; porém quando este valor é maior que o limiar, o neurônio dispara um *spike*, e retorna para seu estado de repouso (IZHIKEVICH, 2007a).

Durante a passagem do potencial de ativação, há uma mudança do sinal de V_M por certo intervalo de tempo, como indica a Fig.2.11. Isso significa que, durante este

intervalo de tempo, a superfície interna fica carregada positivamente e a superfície externa negativamente. No instante em que $V_M = 0$ a membrana fica momentaneamente descarregada (OKUNO; CALDAS; CHOW, 1982). Podemos descrever o potencial de ação como causador de um *desequilíbrio* de grandes proporções na carga da membrana e de duração próxima a um milissegundo, o que torna o potencial de membrana V_m temporariamente positivo (DURAN, 2003).

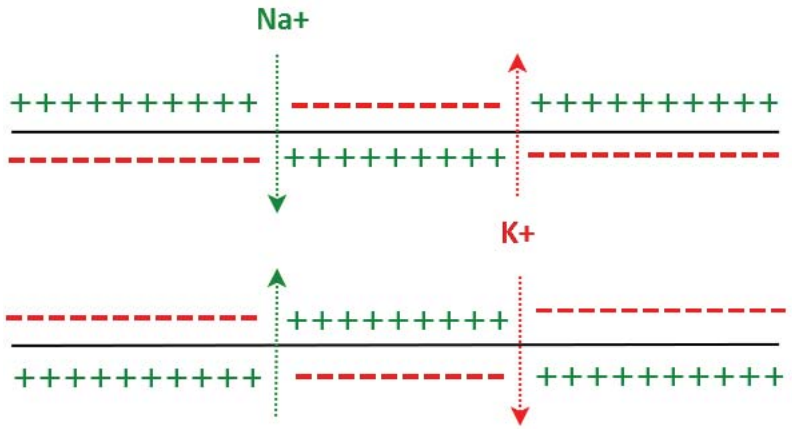

Figura 2.11: Variação da polarização da membrana.

Nos organismos dotados de sistema nervoso, o potencial de ação serve para comunicações de longa distância e são enviados de uma célula nervosa para outra (KOCH; SEGEV, 1989). Em geral, é necessária a contribuição de várias sinapses para que um potencial de ação seja gerado em determinado local. Embora haja similaridade em suas estruturas, duas células nervosas distintas podem se comportar de maneiras completamente diferentes quando alimentadas pelo mesmo conjunto de valores de entrada. Do ponto de vista biológico, uma possível explicação seria que a resposta do neurônio depende de muitos fatores, como a morfologia de sua árvore de dendritos, a localização das conexões de entrada, a estrutura dos canais iônicos expressos pelo neurônio, etc. Estes fatores determinam então as regras que regem a dinâmica do neurônio (IZHIKEVICH, 2007a).

2.8.2 Redes Neurais de Spikes (*Spiking neural networks*)

Qualquer coisa que interage com seu próprio ambiente necessita tomar atitudes apropriadas. Mimetizar mecanismos naturais simples em robôs é uma tarefa que tem se mostrado intensamente difícil. O que impulsiona novas tentativas é o fato dos animais desempenharem este comportamento com aparente facilidade. Para (VREEKEN, 2003) a razão para este comportamento incompreensível até então reside em uma estrutura natural destes seres vivos, o '*cérebro*'.

Milhões e milhões de neurônios estão interconectados cooperando mutuamente no processamento de sinais de entrada e agindo na decisão de ações. Em poucas palavras, pode-se dizer que o funcionamento do cérebro em si não é compreendido, já que não existe uma compreensão total do funcionamento de um único neurônio (VREEKEN, 2003). Os neurônios estão espalhados por diversas estruturas anatomicamente diferentes (cerebelo, córtex, etc.). Dentro de cada uma destas estruturas, diferentes tipos de neurônios existem, com diferentes padrões de conectividade, diferentes respostas características em relação à entradas e executam diferentes tarefas (BOHTE, 2004). Entretanto, o conceito básico de como um neurônio funciona é entendido: neurônios enviam breves pulsos de energia elétrica como sinais se tiverem recebido uma quantidade de sinais suficientes para tal ação. Este mecanismo tem sido moldado através de modelos matemáticos para uso do computador, tornando-os conhecidos como redes de neurônios artificiais ou redes neurais artificiais (HAYKIN, 1999).

As redes neurais artificiais já estão se tornando uma técnica antiga na ciência da computação. As primeiras idéias e modelos datam de mais de 60 anos atrás. A primeira geração de redes neurais artificiais surgiu do neurônio com função limiar de McCulloch-Pitts (MCCULOCH; PITTS, 1943). Em um segundo momento as funções degrau e limiar deixaram de ser utilizadas, sendo substituídas por funções de ativação contínuas, tornando os modelos adequados para entradas e saídas analógicas. Em uma etapa posterior, o realismo biológico das unidades das redes foi incrementado utilizando de *spikes* individuais, e caracterizando uma nova geração de redes neurais (MAASS; NATSHLAGER; MARKRAM, 2002). Esta modificação permite incorporar informações espaço-temporais na comunicação e computação, como neurônios reais fazem.

Se computadores comunicam-se através de *bits*, neurônios comunicam-se por *spikes*. Os *spikes* não podem simplesmente cruzar o espaço entre um neurônio e outro. Eles tem de ser "manuseados" pela sinapse. Inicialmente se pensava que a sinapse apenas transferia um sinal de um axônio para um dendrito, porém tem se provado ser um pré-processador de sinais bem complexo e de papel crucial no aprendizado e adaptação (ROWCLIFFE; FENG; BUXTON, 2006).

Com o aumento do foco sobre a neurociência computacional nos últimos anos, uma nova gama de modelos e algoritmos com plausibilidade biológica têm surgido na literatura específica (ROWCLIFFE; FENG, 2008). Muitas destas aplicações que utilizam de modelos biofísicos usam o modelo *intregrate-and-fire* para neurônios como unidade computacional, principalmente em tarefas destinadas à engenharia. Um trabalho que descreve este tipo de uso por meio de uma revisão bibliográfica é o trabalho de (FURBER; TEMPLE, 2006).

2.8.2.1 Codificação dos pulsos

As discussões sobre o código neural são vastas na literatura, e é provável que não exista consenso até que algumas perguntas (sobre as características dos componentes biológicos do neurônio) sejam respondidas (FURBER; TEMPLE, 2006). Os *spikes* gerados na saída dos neurônios podem representar uma informação em grande variedade de maneiras, cada uma das quais com suas especificidades e aplicabilidades (SIMOES, 2006). Um consenso sobre esta questão parece ainda estar distante.

Embora *spikes* apresentem diferentes amplitudes, durações ou formatos, eles tipicamente são tratados como eventos discretos. Isto significa que para caracterizar uma sequência de *spikes* a única informação necessária é a sucessão de tempos de emissão $t_i^{(f)}$ (ESCOBAR; AL, 2009). O índice inferior i identifica o neurônio, enquanto o índice superior f identifica o número do *spike*. Desta maneira, (DAYAN; ABBOTT, 2001) define uma sequência de *spikes* emitidos por um nerônio como

$$F_i = \left\{ t^{(1)}, \ldots, t^{(n)} \right\}. \tag{2.30}$$

De uma forma geral, a presença ou ausência de pulsos parece ter forte relevância para neurônios biológicos, enquanto sua forma e tamanho não. Com base nesse princípio, os projetistas de redes neurais de *spikes* têm feito uso de diferentes códigos para representar informações.

- **Taxa média de disparos**. Segundo (SIMOES, 2006) a taxa média de disparos não constitui propriamente uma representação temporal da informação, mas tem sido a grandeza mais utilizada para a codificação de informação transmitida por neurônios nos últimos anos. Ao menos três noções de taxas podem ser utilizadas: i) média sobre o tempo; ii) média sobre repetições; e iii) média sobre a população de neurônios. A taxa de disparos pode ser representada por uma variável de valor real, tendo todo o comportamento do neurônio biológico abstraído em seu funcionamento (FURBER; TEMPLE, 2006);

- **Latência**. A latência corresponde ao período de tempo entre o início da estimulação da célula até o início da emissão do potencial de ação na região sináptica do neurônio. Este código tem particular relevância quando os estímulos extremos são alterados abruptamente. O tempo de chegada do primeiro *spike* tem sido considerado de grande relevância neste contexto. Um modelo que utiliza de tal codificação pode ser encontrado no trabalho de (BOHTE, 2003).

- **Fase**. A idéia de codificação em relação ao tempo do primeiro *spike* pode ser utilizada para produzir um novo modelo de codificação se o sinal de referência for periódico ao invés de ser um evento isolado. De acordo com (SIMOES, 2006) existem evidências biológicas da utilização desse código no sistema olfativo de mamíferos, além de permitir tarefas de localização espacial;

- **Sincronia**. Nesta abordagem, os *spikes* provenientes de outros neurônios podem ser utilizados como sinal de referência para um código de *spikes*. O sincronismo, ou correlação, entre um par ou grupo de *spikes* pode conter informações que certamente não estão contidas na taxa média de disparos.

Nesta categoria de modelos de neurônios, a informação é transmitida exclusivamente nos tempos em que algum evento ocorre, seja na variação da ocorrência de *spikes* ou na relação entre sequência de *spikes* de dois neurônios distintos. Entretanto, em (FURBER; TEMPLE, 2006) os autores salientam que outros processos também podem ser importantes além da comunicação via *spikes*, como os processos químicos que acontecem na célula bem como a estrutura e organização das árvores dendríticas. Apesar disto, redes de neurônios de *spikes* são mais poderosas do que suas predecessoras pois podem codificar informação temporal em seus sinais, porém necessitam de diferentes regras de aprendizado com estruturas biologicamente plausíveis (ROWCLIFFE; FENG, 2008).

2.8.2.2 Plasticidade Sináptica

Um importante desafio para compreensão do comportamento é tentar entender como os sistemas nervosos permitem o aprendizado de sequências comportamentais que ocorrem em escalas arbitrárias de tempo, variando de segundos para milissegundos. Um exemplo dado por (MELAMED et al., 2004) é o de um pianista. Um pianista é capaz de tocar escalas ou repetições de um único tom rapidamente, a uma taxa de 12 ou mais notas por segundo, provavelmente utilizando de alguma sequência interna de comandos motores que é mais rápida do que isso. Entretanto, quando tocando melodias lentas, estes comandos são executados dez ou até vinte vezes mais devagar. Estas sequências temporais diferentes demonstram a existência de uma regra de aprendizado sináptico que opera em uma escala de tempo fixa de milissegundos.

Este é um exemplo de plasticidade sináptica, uma forma de mudança do pré-processamento que descreve mudanças da eficácia sináptica a curto ou longo prazo (VREEKEN, 2003). De acordo com (FURBER; TEMPLE, 2006) a sinapse apresenta uma natureza adaptativa

muito sutil. Sua habilidade de ajustar sua eficácia como uma conexão é visto como o principal mecanismo de memória de longo prazo presente no cérebro. Como e quando estas modificações ocorrem são perguntas que não foram totalmente respondidas até o momento, porém existem uma série de teorias e explicações elaboradas por pesquisadores.

A plasticidade Hebbiana (HEBB, 1949) é uma forma local de longa duração de potencialização (LTP) e depressão (LTD) das sinapses, baseada na correlação da atividade neural entre neurônios pré e pós-sinápticos. Segundo (ROWCLIFFE; FENG, 2008) este comportamento é comumente implementado utilizando da codificação por taxas; atividades neurais similares significam uma forte correlação.

Plasticidade sináptica dependente do tempo de *spike* (STDP) é uma forma de aprendizado Hebbiano competitivo que utiliza informações exatas do tempo de disparo de *spikes*. Experimentos neurocientíficos mostraram que o fortalecimento da sinapse ocorre se o potencial de ação pré-sináptico chega em um intervalo de tempo de 50 ms antes do *spike* pós-sináptico, caso contrário ocorre um enfraquecimento da conexão se ele chega atrasado. Devido a este mecanismo, STDP pode permitir distribuições estáveis de LTP e LDP, tornando os neurônios pos-sinápticos sensíveis ao tempo de disparo dos *spikes* de entrada. Esta sensibilidade permite a competição entre *spikes* e propagação de informação pela rede mais veloz (IZHIKEVICH, 2007b).

É interessante comentar que o aprendizado desenvolvido por Bohte (BOHTE; KOK; POUTRE, 2002) (PAUGAM-MOISY; BOHTE, 2009) é uma das poucas regras de aprendizado aplicadas em redes neurais de *spikes*, que não se baseia no aprendizado Hebbiano. De fato, muitas das regras de plasticidade sináptica desenvolvidas para modelos de redes com *spikes*, dependem da correlação Hebbiana como principal meio de modificação dos pesos sinápticos. Entretanto, (BOHTE; KOK; POUTRE, 2002) mostrou, com sua regra de aprendizado de *backpropagation*, que existem outras alternativas eficientes para treinar modelos de neurônio biologicamente inspirados (ROWCLIFFE; FENG, 2008).

3 MODELOS DE NEURÔNIOS BIOLÓGICOS

A grande maioria das células nervosas gera uma série de breves pulsos de tensão em resposta a fortes entradas sinápticas. Estes pulsos, potenciais de ação ou *spikes*, originam-se no corpo celular ou próximo a ele, são propagados através do axônio com velocidade e amplitude constantes. Potenciais de ação ocorrem em uma variedade de formas; sendo a mais comum a despolarização "tudo-ou-nada" da membrana: se o potencial da membrana falhar em exceder um valor particular limitante, nenhum spike será gerado e o potencial retornará ao nível de repouso. Se o potencial exceder o valor limiar, o potencial da membrana ao longo do tempo executa uma trajetória estereotipada que reflete as propriedades da membrana neural. Como apresentado na fig. 3.1, o formato do potencial de ação pode variar bruscamente de um tipo para outro de célula. Dependendo do tipo de célula neural e do sistema nervoso do animal ao qual tal célula pertence, o potencial de ativação da célula apresenta uma curva característica, diferente das demais células nervosas.

Apenas uma pequena fração de todos os neurônios não são capazes - em condições fisiológicas - de gerar potenciais de ação, fazendo uso exclusivo dos sinais de classificação. Exemplos de tais células podem ser encontradas na retina distal e em neurônios do circuito sensitivo-motor de invertebrados. Estas células parecem estar ausentes do córtex, cerebelo e estruturas associadas, embora seja difícil excluir totalmente a sua existência. Reeve e Hallam (REEVE; HALLAM, 2005) realizaram um trabalho abrangendo a análise de modelos de redes neurais artificiais adequadas para a coordenação motora de robôs. Eles concluíram que o realismo biológico é desejável porém em alguns casos conflita com o custo computacional. Além disso, modelos de neurônios com dinâmica interna possibilitam um repertório amplo com um menor número de neurônios em relação às outras implementações que utilizam de modelos mais tradicionais, como o modelo de McCulloch-Pitts (MCCULOCH; PITTS, 1943) e os modelos de neurônios sigmoidais (CHAVES, 2007).

Dentre os modelos de unidades computacionais de redes neurais apresentados na literatura específica, uma classe que vem tendo certo destaque da comunidade científica é

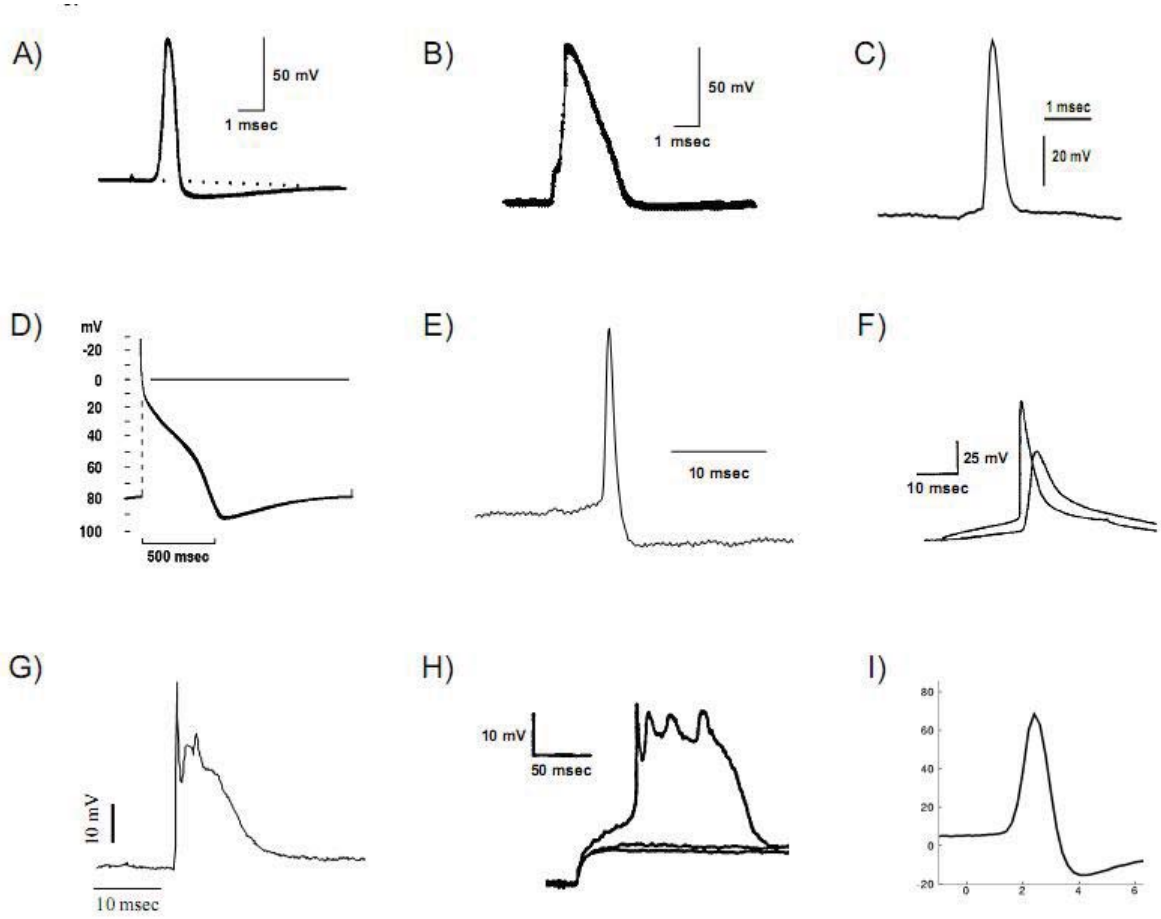

Figura 3.1: Potencial de ação em diferentes células de vertebrados e invertebrados [Adaptado de (KOCH, 1999)].

a dos neurônios de *spikes* (*spiking neurons*). Estes modelos matemáticos apresentam uma boa concordância com as medições de neurônios *in vitro* e *in vivo*, além de descreverem fielmente, e de certa maneira reproduzirem, atividades ligadas ao cérebro (GERSTNER; KISTLER, 2002).

Existe uma grande variedade de modelos pertencentes a esta classe, desde os mais detalhados (tratando de cada estrutura existente no neurônio) até aqueles que tratam o neurônio como uma estrutura homogênea. Para uma revisão completa dos modelos existentes na literatura, pode-se consultar os trabalhos de Gerstner e Kistler (GERSTNER; KISTLER, 2002), de Bohte (BOHTE, 2003; PAUGAM-MOISY; BOHTE, 2009) e Izhikevich (IZHIKEVICH, 2004; IZHIKEVICH, 2007a).

3.1 O modelo de Hodgkin-Huxley

Um dos principais modelos da Neurociência Computacional é o modelo de Hodgkin e Huxley (HODGKIN; HUXLEY, 1952) sobre a iniciação e propagação do potencial de

ativação no axônio de lula gigante. Usando técnicas experimentais pioneiras naquela época, eles conseguiram determinar que o axônio da lula possuía três correntes principais: uma corrente persistente de potássio ($K^{+"}$); uma corrente transiente de sódio (Na^{+}); e uma corrente residual, composta em sua maior parte por íons de cloro (Cl^{-}) (IZHIKEVICH, 2007a). O axônio da lula gigante, com diâmetro de aproximadamente meio milímetro, caracterizava um verdadeiro "Leviatã" entre os axônios, já que um típico axônio do córtex tem um diâmetro mil vezes inferior (KOCH, 1999).

Em geral, os cientistas se referem a todos os modelos baseados em condutância como sendo do "tipo Hodgkin-Huxley". Estes modelos não são apenas importantes devido ao significado biofísico de seus parâmetros, mas também porque permitem a investigação de questões relacionadas à integração sináptica, efeitos da morfologia dos dendritos, interação entre correntes iônicas e outras questões relacionadas à dinâmica de uma única célula (IZHIKEVICH, 2004).

Foi por meio de experimentações que Hodgkin e Huxley conseguiram desmembrar os componentes da corrente da membrana e determinar que a corrente total que passa por uma membrana é a soma das correntes iônicas com a corrente capacitiva (TUCKWELL, 1988a):

$$I_m(t) = I_{ionic}(t) + C_m \frac{dV(t)}{dt}. \tag{3.1}$$

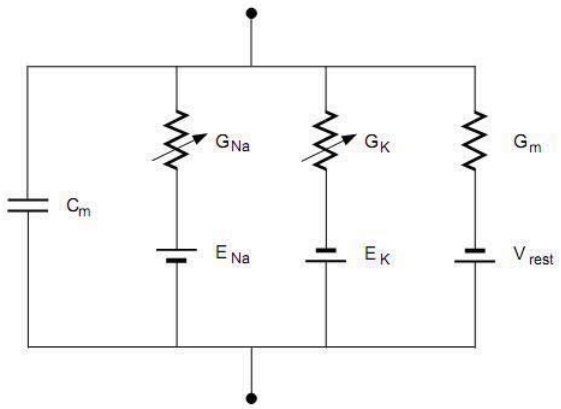

Figura 3.2: Circuito elétrico para um fragmento do axônio da lula [Adaptado de (KOCH, 1999)].

O potencial de ação envolve duas condutâncias iônicas principais, dependentes do potencial da membrana: uma condutância de sódio g_{Na}, e uma condutância de potássio g_K. Elas são independentes uma da outra. Existe também uma terceira condutância, menor, chamada de condutância "residual" (g_m) que independe do potencial da membrana. A

corrente iônica total que flui pela membrana pode ser expressa por

$$I_{ionic} = I_{Na} + I_K + I_{leak}. \tag{3.2}$$

onde I_{Na} e I_K são as correntes do sódio e do potássio, e I_{leak} é uma corrente residual composta por outros íons, em sua maioria íons de Cl^- (TUCKWELL, 1988a). Além disso, uma corrente iônica $I_i(t)$ é linearmente relacionada ao potencial de condução, via lei de Ohm:

$$I_i(t) = g_i(V(t), t)\ (V(t) - E_i), \tag{3.3}$$

onde o potencial de reversão iônico E_i é dado pela equação de Nernst para o tipo apropriado de íon, e $g_i(V(t), t)$ é a condutância associada ao canal iônico. Em geral, este valor de condutância é dependente do tempo e do potencial de membrana, porém pode também depender de vários mediadores químicos como cálcio intracelular (KOCH; SEGEV, 1989). Conceitualmente, o circuito da Fig.3.2 pode ser utilizado para representar a membrana axonal.

Cada uma das duas condutâncias iônicas é expressada como a máxima condutância, $\bar{g}_{Na}$ e $\bar{g}_K$, multiplicado por um coeficiente numérico representando a fração da condutância máxima atualmente aberta. Estes números são funções de uma ou mais partículas que Hodgkin e Huxley introduziram para descrever a dinâmica das condutâncias. No modelo original estas partículas foram denotadas por *partículas chaveadoras*(*gating particles*), podendo estar em um de dois possíveis estados, aberto ou fechado, dependendo do tempo e do potencial de membrana. Para que a condutância se "abra", todas essas partículas devem ser abertas simultaneamente. Toda a cinética do modelo está contida no funcionamento destas partículas (KOCH, 1999).

Quando as partículas transportadoras são sensíveis ao potencial de membrana, estes canais são ditos serem chaveados por voltagem (IZHIKEVICH, 2007a). Tais partículas são divididas em dois tipos: aquelas que ativam ou abrem um canal, e aquelas que inativam ou fecham um canal (Fig.3.3). De acordo com Hodgkin-Huxley, a probabilidade de uma partícula chaveadora estar em seu estado aberto é denotada pela variável m (algumas vezes a variável n é usada para os canais de K^+ e Cl^-). Já a probabilidade de uma partícula inibidora estar em seu estado aberto é denotada pela variável h. A proporção de canais abertos em uma população é dada por

$$p = m^a h^b \tag{3.4}$$

onde a é o número de portões ativadores e b o número de portões inativadores por canal. Dependendo dos valores de m e h pode-se gerar correntes transitórias (podem ser inativadas ou não) ou persistentes (não podem ser inativadas) (IZHIKEVICH, 2007a).

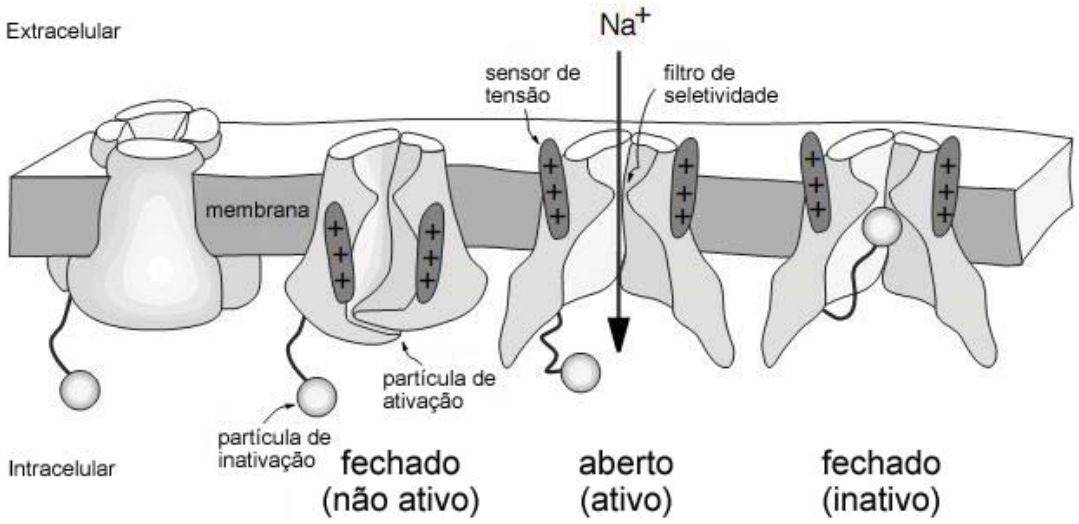

Figura 3.3: Estrutura de um canal iônico chaveado por voltagem. Sensores de voltagem abrem portões ativadores e permitem que determinados íons fluam pelo canal de acordo com seus gradientes eletroquímicos [Adaptado de (IZHIKEVICH, 2007a)].

3.1.1 A corrente do potássio

Hodgkin e Huxley modelaram a corrente do potássio como sendo

$$I_K = \bar{g}_K n^4 (V - E_K), \tag{3.5}$$

onde a condutância máxima é $\bar{g}_K = 36\, mS/cm^2$, e o potencial do potássio é $E_K = -12\, mV$, valores relativos ao potencial de repouso do axônio. A variável n descreve o estado da partícula de ativação, e sendo um número adimensional entre 0 e 1 (IZHIKEVICH, 2007a).

A probabilidade de encontrar uma partícula ativadora em seu estado permissivo ou estado aberto é n (e será $1 - n$ em seu estado não-permissivo ou fechado, quando nenhuma corrente flui através da condutância). A equação 3.5 indica que, para o canal ser aberto, quatro partículas ativadoras devem estar simultaneamente em seus estados abertos, resultando em uma corrente persistente de K^+. Também podemos pensar em n como uma proporção das partículas em seu estado permissivo. Assumindo que existe apenas dois estados para uma única partícula, e que a transição de um para o outro é dirigido por um mecanismo de primeira ordem, tal relação pode ser esquematizada como:

$$n \underset{\alpha_n}{\overset{\beta_n}{\rightleftharpoons}} 1 - n. \tag{3.6}$$

tal que α_n (e resp. β_n) é uma taxa constante dependente da voltagem (em unidades de $1/\text{seg}$),

especificando quantas transições ocorrem entre os estado aberto e fechado da partícula (e vice-versa) (KOCH; SEGEV, 1989).

Matematicamente, a dinâmica que descreve o funcionamento desta partícula é dado por uma equação diferencial de primeira ordem:

$$\frac{dn}{dt} = \alpha_n(V)(1-n) - \beta n(V)n. \tag{3.7}$$

Segundo Koch(1999), a chave do modelo de Hodgkin e Huxley foi a descrição quantitativa das taxas constantes dependentes do potencial da célula. Ao invés de utilizar as variações constantes α_n e β_n, pode-se escrever a equação 3.7 em termos de um tempo constante dependente do potencial $\tau_n(V)$, e um valor de estado estacionário $n_\infty(V)$:

$$\frac{dn}{dt} = \frac{n_\infty - n}{\tau_n}, \tag{3.8}$$

com

$$\tau_n = \frac{1}{\alpha_n + \beta_n}, \tag{3.9}$$

$$n_\infty = \frac{\alpha_n}{\alpha_n + \beta_n}. \tag{3.10}$$

Uma das propriedades mais marcantes da membrana da lula é declividade da relação entre a condutância e o potencial de membrana. Abaixo de aproximadamente $20\,mV$, a condutância de potássio no estado estacionário g_K aumenta e dobra V, variando $4.8\,mV$, enquanto a sensibilidade de potencial da condutância de sódio é ainda maior. Para altos níveis de despolarização, a saturação na condutância da membrana é iniciada. Esta forte relação é refletida na dependência de potencial das taxas constantes. Hodgkin e Huxley aproximaram as taxas constantes dependentes do potencial como

$$\alpha_n(V) = \frac{10 - V}{100(\exp^{(10-V)/10} - 1)}, \tag{3.11}$$

$$\beta_n(V) = 0.125 \exp^{-V/80}, \tag{3.12}$$

onde V é o potencial de membrana relativo ao potencial de repouso do axônio em unidades de milivolts.

A fração da condutância de potássio aberta para um particular valor de voltagem $\bar{V}$, isto é, para $t \rightarrow \infty$, é equivalente a $n_\infty(\bar{V})^4$. Na situação de repouso (V_{rest}) este valor é muito pequeno, $n_\infty(0)^4 = 0.01$, correspondente a apenas 1% da condutância total de potássio ativada. O crescimento da condutância de potássio é descrito pela equação

diferencial:

$$n(t)^4 = (n_\infty - (n_\infty - n_0)\exp^{-t/\tau_n(\bar{V})})^4, \tag{3.13}$$

onde n_0 é o valor inicial da ativação de potássio, $n_0 = n_\infty(0) = 0.32$, e n_∞ é o valor final, $n_\infty = n_\infty(\bar{V})$. O comportamento ao longo do tempo de qualquer variável de ativação descreve um comportamento exponencial, um reflexo do pressuposto subjacente de um modelo cinético de primeira ordem.

3.1.2 A corrente do sódio

A fim de ajustar o comportamento cinético da corrente de sódio, Hodgkin e Huxley postularam a existência de uma *partícula ativadora* do sódio m, bem como uma *partícula inibidora* h

$$I_{Na} = \bar{g}_{Na} m^3 h (V - E_{Na}), \tag{3.14}$$

onde a condutância máxima do sódio é $\bar{g}_{Na} = 120\, mS/cm^2$ e potencial de reversão do sódio é $E_{na} = 115\, mV$, valores relativos ao potencial de repouso do axônio. As variáveis m e h são valores adimensionais, com $0 \leq m, h \leq 1$ (IZHIKEVICH, 2007a).

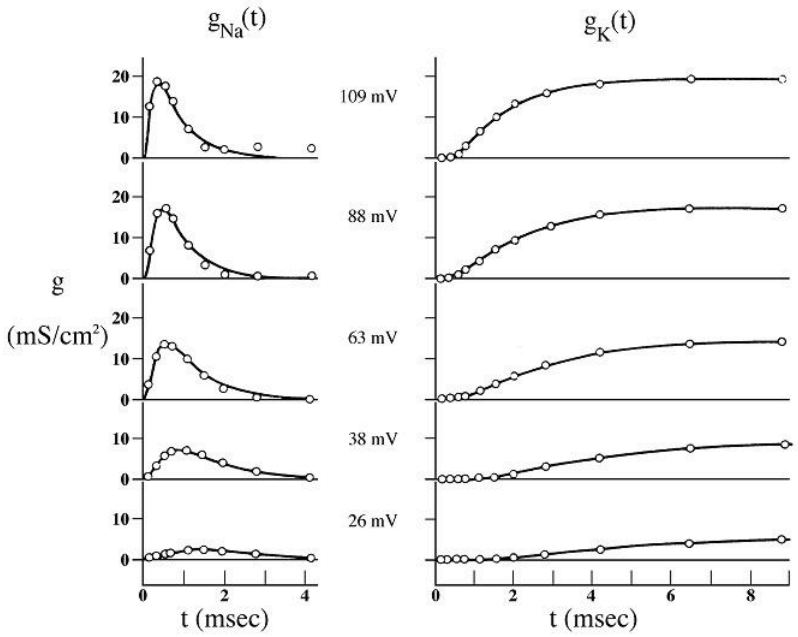

Figura 3.4: Condutâncias de K^+ e Na^+ durante uma fase de tensão [Adaptado de (KOCH, 1999)].

A amplitude da corrente de sódio depende de quatro partículas tornando-se independentes, através de transições de primeira-ordem entre um estado aberto e um estado fechado. Uma vez que estas partículas são independentes, a probabilidade de que três partícu-

las m e uma partícula h existam neste estado é m^3h. É interessante destacar que h é a probabilidade de que a partícula inibidora *não esteja* sem seu estado de inativação. Formalmente, a mudança desas partículas em relação ao tempo pode ser descrita por duas equações diferenciais de primeira ordem

$$\frac{dm}{dt} = \alpha_m(V)(1-m) - \beta_m(V)m \tag{3.15}$$

$$\frac{dh}{dt} = \alpha_h(V)(1-h) - \beta_h(V)h. \tag{3.16}$$

onde as taxas constantes para m e h são

$$\alpha_m(V) - \frac{25-V}{10(\exp^{(25-V/10)}-1} \tag{3.17}$$

$$\beta_m(V) = 4\exp^{-V/18} \tag{3.18}$$

$$\alpha_h(V) = 0.07\exp^{-V/20} \tag{3.19}$$

$$\beta_h(V) = \frac{1}{\exp^{(30-V)/10}+1}. \tag{3.20}$$

A fração de condutância de sódio aberta no estado estacionário é menor que 1% da condutância de sódio máxima. Acontece que para voltagens baixas ou próximas do potencial de repouso do axônio, a variável de ativação é próxima de zero enquanto que para potenciais positivos o valor da variável de inativação h é quase nulo. Então, a corrente de sódio no estado estacionário ($\bar{g}_{Na}m^3h(V-E_{Na})$), é sempre muito pequena. O segredo para obter uma alta corrente de sódio que despolarize a membrana reside na dinâmica temporal das variáveis m e h. Para valores baixos de potencial de membrana ou próximos do potencial de repouso, assume-se que h possui um valor próximo de um. Quando uma fase súbita de tensão despolarizante é imposta sobre a membrana, como na Fig 3.4, m modifica-se em uma fração de milissegundos para seu novo valor próximo de um, enquanto h necessita de cinco ou mais milissegundos para atenuar seu alto valor inicial para um valor pequeno. Em ouras palavras, dois processos controlam a condutância de sódio: um processo de ativação que rapidamente incrementa g_{na} em consequência da despolarização ultrapassando a inativação, um processo mais lento que reduz g_{na} em consequência da despolarização (KOCH, 1999).

3.1.3 O modelo completo

Similar à maioria das membranas biológicas, a membrana axonal contém uma condutância "*residual*", g_n, que não depende da tensão aplicada à membrana e permanece constante ao longo do tempo. O valor medido por Hodgkin e Huxley, $g_n = 0.3\ mS/cm^2$, corresponde a uma resistividade passiva de membrana com valor de $R_m = 3333\ \Omega cm^2$. Este componente passivo também tem um potencial de reversão a ele associado. Hodgkin e Huxley não mediram explicitamente o valor de V_{rest}, mas o ajustaram de maneira que a corrente total da membrana em um potencial de repouso $V = 0$ é nula. Em outras palavras, V_{rest} é definido através da equação

$$g_{Na}(0)E_{Na} + g_K(0)E_K + g_m V_{rest} = 0, \tag{3.21}$$

e possui o valor de $+10.613mV$. A capacitância da membrana é $C_m = 1\mu F cm^2$. No potencial de repouso, a resistência efetiva da membrana devido a presença da soma das condutâncias: residuais, do íon de potássio, e do íon de sódio; correspondem a $857\Omega cm^2$, equivalente a um tempo efetivo "passivo" de membrana constante de cerca de $0.85mseg$.

Podemos então escrever em uma única equação o fluxo de todas as correntes que fluem através da membrana neuronal

$$C_m \frac{dV}{dt} = \bar{g}_{Na} m^3 h(E_{Na} - V) + \bar{g}_K n^4 (E_K - V) + G_m(V_{rest} - V) + I_{inj}(t), \tag{3.22}$$

onde I_{inj} é a corrente que é injetada através de um eletrodo intracelular. Esta equação diferencial não-linear, em conjunto com as três equações, ordinárias, lineares, diferenciais de primeira ordem (3.8, 3.15 e 3.16), especificam a evolução das taxas constantes, constituindo o modelo de Hodgkin-Huxley para representação de um neurônio, ou pequeno pedaço da membrana neuronal (IZHIKEVICH, 2007a).

Em termos computacionais, o modelo de Hodgkin-Huxley é extremamente custoso de se implementar. Ele gasta 120 operações de ponto flutuante para avaliar $0.1\,ms$ de tempo do modelo, consequentemente, 1200 operações$/1\ ms$. Deste modo, costuma-se usar o formalismo de Hodgkin-Huxley somente para simulações envolvendo um pequeno número de neurônios, ou quando o tempo de simulação não é uma questão importante (IZHIKEVICH, 2004).

3.2 Modelo integrador (Integrate-and-fire)

A vantagem de utilizar modelos baseados no funcionamento das condutâncias iônicas é que cada variável e parâmetro tem um significado biofísico bem definido. Ás vezes não é necessário ou não se tem recursos para usar um modelo biofisicamente detalhado. Como alternativa, existem simples modelos que reproduzem fielmente as características neurocomputacionais do neurônio. Estes modelos ressaltam tais comportamentos do ponto de vista de sistemas dinâmicos (IZHIKEVICH, 2007a). Os modelos integradores (*integrate-and-fire models*) fazem essa redução de complexidade estipulando que um potencial de ação ocorre sempre que o potencial de membrana do modelo de neurônio atinge um valor limitante v_{thresh} (fig.3.5). Depois do potencial de ação, o potencial da membrana é reiniciado para um valor v_{reset} menor do que o limitante de disparo ($v_{reset} < v_{thresh}$) (DAYAN; ABBOTT, 2001).

O modelo integrador (LAPICQUE, 1907; STEIN, 1967; TUCKWELL, 1988a) é uma idealização de um neurônio que possui uma corrente resistiva 'residual' e um número de correntes sensíveis a voltagem que são completamente desativadas na condição de repouso da célula. O comportamento deste modelo de neurônio pode ser descrito pela equação linear

$$C\dot{V} = I - \overbrace{g_{leak}(V - E_{leak})}^{resistnciadefuga}, \tag{3.23}$$

onde os parâmetros possuem o mesmo significado biofísico já descritos nos capítulos anteriores. Quando o potencial de membrana V atinge o valor limitante E_{leak}, as correntes sensíveis à voltagem são instantaneamente ativadas, é dito que o neurônio dispara um potencial de ativação, e V é reiniciada para o valor E_K. Depois de algumas reduções apropriadas, o modelo integrador pode ser escrito na forma

$$\dot{v} = b - v,\ if\ v = 1,\ then\ v \leftarrow 0, \tag{3.24}$$

onde o estado de repouso é $v = b$, o valor de limiar é $v = 1$, e o valor de reinício (*reset*) é $v = 0$ (fig.3.6). Aparentemente o neurônio é excitável quando $b < 1$ e dispara uma cadeia de *spikes* quando $b > 1$ com um período $T = -\ln(1 - 1/b)$ (IZHIKEVICH, 2007a).

Algumas das características computacionais ilustradas pelo neurônio integrador são (KOCH, 1999; IZHIKEVICH, 2007a)

- Comportamento tudo-ou-nada (*all-or-none spikes*). Uma vez que a forma do *spike* não é simulada, implicitamente assume-se que todos os *spikes* são idênticos em tamanho e duração;

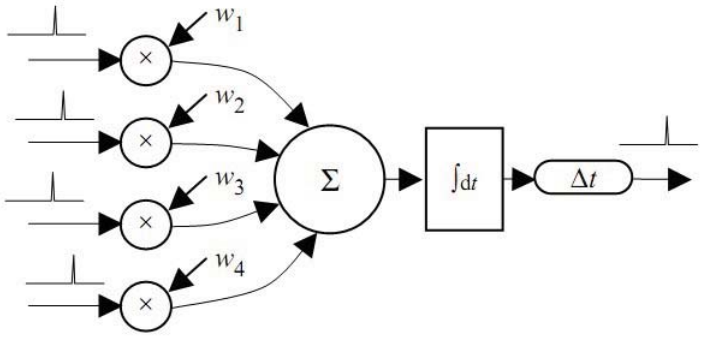

Figura 3.5: Representação do neurônio como uma unidade computacional. No modelo *integrate-and-fire* os *spikes* de entrada são multiplicado por seus respectivos pesos sinápticos, somados e integrados em relação ao tempo. Se a integral exceder o limiar, o neurônio dispara um *spike* e o processo reinicia [Adaptado de (FURBER; TEMPLE, 2006)].

- Valor limiar bem definido (*Well-defined threshold*). Um *spike* característico é disparado logo que $v = v_{thresh}$, não deixando espaço para nenhuma ambiguidade;
- Período refratário relativo. Quando $E_K < E_{leak}$, o neurônio é menos excitável imediatamente após o *spike*;
- Distinção entre inibição e excitação. Entradas excitatórias $(I > 0)$ tornam o potencial de membrana mais próximo do valor de limiar, e assim facilitando o disparo de um spike, enquanto entradas inibitórias $(I < 0)$ fazem exatamente o oposto;
- Excitabilidade Classe 1. O neurônio pode continuamente codificar a força de uma entrada em uma frequência de *spikes*.

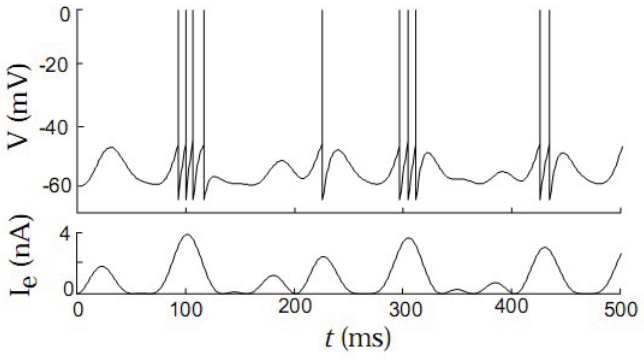

Figura 3.6: Um modelo *integrate-and-fire* dirigido por um corrente de eletrodo variando com o tempo. A curva superior é o potencial de membrana, enquanto o traço inferior é a corrente ao longo do tempo [Adaptado de (DAYAN; ABBOTT, 2001)].

Como o modelo integrador possui apenas uma variável, ele não pode reproduzir *spikes* fásicos (*phasic spiking*), explosões (*burstings*), e outras características neurobiológicas. Devido ao valor de limiar do disparo ser fixo, os *spikes* não possuem latências

(IZHIKEVICH, 2004). Em poucas palavras, o modelo parece ser um bom modelo para um integrador, e sendo aceitável para matemáticos que desejam provar teoremas e derivar expressões analíticas. Entretanto, usar este modelo pode ser considerado uma perda de tempo para neurocientistas que desejam simular grandes redes de neurônios (IZHIKEVICH, 2007a).

3.3 Modelo ressonador (Resonate-and-fire)

Um outro modelo para simulações de neurônio de *spike* é o neurônio ressonador (*resonate-and-fire*), desenvolvido por (IZHIKEVICH, 2001). Este modelo é similar ao modelo de neurônio integrador, exceto pelo fato da variável de estado ser complexa. Além disso, o modelo ressonador é também computacionalmente eficiente para simulações de grande redes de neurônios de *spikes*. Este modelo proporciona ilustrações geométricas de vários fenômenos que ocorrem em neurônios biológicos que apresentam oscilações amortecidas do potencial de membrana. Esse tipo de comportamento pôde ser observado em muitos neurônios biológicos e em quase todos os modelos biofísicos do tipo Hodgkin-Huxley (HODGKIN; HUXLEY, 1952; FITZHUGH, 1969; IZHIKEVICH, 2003).

Esta propriedade dinâmica torna os neurônios suscetíveis ao tempo do estímulo, pois a distância entre o potencial de membrana e o limiar mudam de acordo com a oscilação. Um estímulo forte pode provocar um *spike* ou não, de acordo com seu tempo relativo à fase de oscilação, como mostrado nas figuras 3.7 e 3.8. Tal mecanismo permite diversos fenômenos não-lineares, como oscilação da probabilidade de disparos (*spikes*), explosões de *spikes* (*bursting*), a comunicação seletiva, ressonância, etc (IZHIKEVICH, 2001).

O modelo ressonador proposto por (IZHIKEVICH, 2001) é simples, porém biologicamente inspirado. Além disso, ele ilustra como oscilação abaixo do potencial limiar pode afetar a dinâmica de *spikes* do neurônio. É também, de acordo com o autor, o modelo mais simples possível que apresenta oscilações atenuadas do potencial de membrana. Sua eficiência computacional é similar ao modelo de neurônio integrador (*integrate-and-fire*), o que o torna conveniente para simulações com redes com grandes quantidades de neurônios de *spikes*.

Neurônios geram potenciais de ativação porque suas membranas possuem canais iônicos dependentes de voltagem. Uma vez que há muitos tipos destes canais, podem existir milhares de mecanismos diferentes para excitabilidade e atividade de *spikes* baseados em condutâncias, e por consequência, milhares de modelos biofisicamente precisos de neurônios de *spikes*. A maioria destes modelos podem ser estudados utilizando-se teorias de sistemas

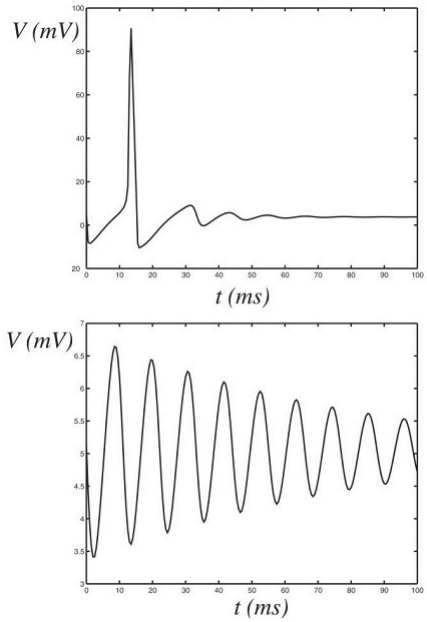

Figura 3.7: Exemplos de oscilações atenuadas do potencial de membrana no modelo de Hodgkin-Huxley (detalhe para a escala das voltagens) [Adaptado de (IZHIKEVICH, 2001)].

dinâmicos (IZHIKEVICH, 2007a). Uma analogia feita é que o estado de repouso da célula corresponde a um atrator de equilíbrio, e *spikes* repetitivos correspondem a um atrator de ciclo-limite.

Em contraste, neurônios situados próximos de uma bifurcação do tipo Andronov-Hopf exibem oscilações suavizadas do potencial de membrana, que modificam a distância em relação ao limiar de disparo (de um *spike*). O efeito de um segundo pulso depende de seu tempo relativo ao período de oscilação. Se o tempo entre dois pulsos é próximo da metade do período, os pulsos efetivamente cancelam um ao outro (fig 3.8). Se o tempo é próximo de um período, os pulsos se adicionam. Estes neurônios preferem entradas tendo uma certa frequência de ressonância que é igual à frequência de oscilação. Tais neurônios são conhecidos por serem *ressonadores* (LLINAS, 1988). Ao contrário dos integradores, aumentar a frequência da estimulação pode retardar ou até mesmo encerrar um disparo de um neurônio ressonador.

Os modelos H-H (HODGKIN; HUXLEY, 1952) e de FitzHugh (FITZHUGH, 1961) apresentam bifurcações do tipo Andronov-Hopf, sendo portanto, ressonadores. Por outro lado, o modelo de Connor (CONNOR; WALTER; MCKOWN, 1977) apresenta bifurcação do tipo *saddle-node*, portanto, é um integrador. Deste modo, neurônios de *spikes* que possuem mecanismos iônicos similares podem apresentar propriedades neurocomputacionais completamente diferentes. Uma particularidade presente nos modelos ressonadores é a existência de uma oscilação atenuada do potencial da membrana com um valor de sub-limiar, o que garante uma propriedade neurocomputacional única que não está presente no modelo integrador de neurônio.

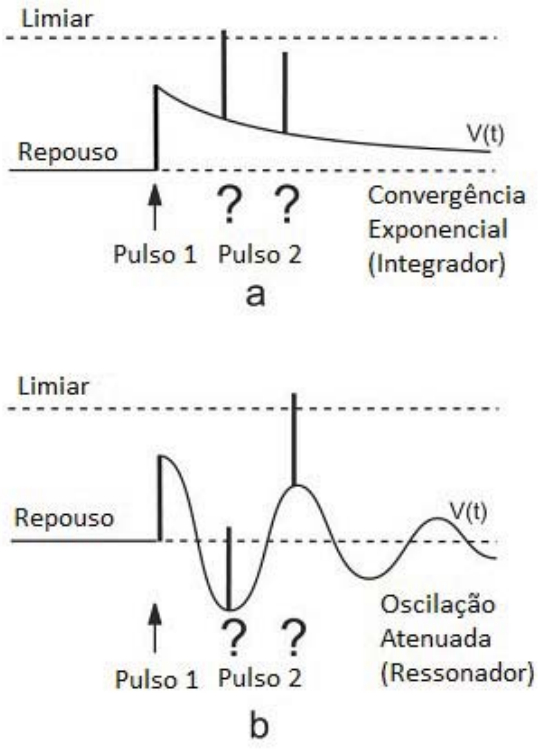

Figura 3.8: O potencial de membrana do neurônio, $V(t)$ é perturbado por um pulso de corrente (*pulso 1*). a) Sistemas com convergência exponencial para o potencial de repouso: o segundo pulso (*pulso 2*) pode fazer o neurônio emitir um *spike* se ele o estímulo chegar logo após o pulso 1. b) Sistemas com convergência oscilatória atenuada para o potencial de repouso: para disparar um *spike*, a distância entre os pulsos 1 e 2 deve ser próxima de um período de oscilação [Adaptado de (IZHIKEVICH, 2001)].

3.3.1 Linearização

De acordo com (IZHIKEVICH, 2001), qualquer modelo não-linear de neurônio que apresenta oscilação atenuada pode ser convertido em um modelo linear através de uma mudança de variáveis locais. O sistema linear resultante

$$\begin{aligned} \dot{x} &= bx - wy \\ \dot{y} &= wx + by \end{aligned} \tag{3.25}$$

pode ser escrito na equivalente forma complexa

$$\dot{z} = (b + iw)z. \tag{3.26}$$

Aqui $z = x + iy \in \mathbb{C}$ é uma variável complexa que descreve a atividade oscilatória do neurônio. A parte real x, é a variável análoga à corrente. Ela descreve a dinâmica dos canais sensíveis à voltagem, e das correntes sinápticas. A parte imaginária y é a variável análoga à voltagem. Já a expressão $b + iw \in \mathbb{C}$, é um parâmetro onde $b > 0$ é a taxa

de atração ao repouso e $w > 0$ é a frequência das oscilações. Quando for $w = 0$, o neurônio funciona como um neurônio integrador.

3.3.2 O Modelo

De acordo com (IZHIKEVICH, 2004) o neurônio ressonador é um modelo bidimensional (2D) análogo ao modelo integrador, e pode ser escrito na forma 3.26, com um *reset* pós-pulso (*after-spike reset*) dado por

$$if\ Im\, z = A_{threshold}\ then\, z \rightarrow z_0(z) \tag{3.27}$$

onde $A_{threshold}$ é um parâmetro limitante, e $z_0(z)$ é uma função arbitrária que descreve a atividade do *reset* pós-*spike*. Indo um pouco além, uma rede de neurônios ressonadores pode ser descrita por

$$\dot{z_i} = (b_i + +iw_i)z_i + \sum_{j=1}^{n} c_{ij}\delta(t - t_j^*), \tag{3.28}$$

onde $z_i \in \mathbb{C}$ descreve o estado do i-ésimo neurônio, $b_i + iw_i \in \mathbb{C}$ é um parâmetro interno do neurônio, cada $c_{ij} \in \mathbb{C}$ é um coeficiente sináptico, δ é a função delta de Dirac, e t_j^* é o instante de tempo mais recente no qual o j-ésimo neurônio emitiu um *spike*. Cada disparo produz um pulso que substitui as atividades dos outros neurônios por uma constante complexa c_{ij}. Depois que z_i dispara um potencial de ação, o neurônio é reiniciado com um novo valor ($i \in \mathbb{C}$) (IZHIKEVICH, 2001). Mesmo que todas as equações envolvidas em 3.28 sejam lineares, o comportamento da rede não o é, devido às conexões entre os neurônios e o mecanismo de *reset*.

Além disso, outra característica neurocomputacional que diferencia integradores de ressonadores é que os neurônios ressonadores podem facilmente disparar um *spike* em resposta a um estímulo inibitório. Característica não presente nos neurônios integradores. Como já discutido, existe uma similaridade entre o modelo integrado e o modelo ressonador. Ambos podem ser usados em simulações com grandes redes de neurônios de *spikes*. Entretanto, os modelos não se repetem, mas complementam um ao outro (IZHIKEVICH, 2001).

3.4 Modelo integrador quadrático (Quadratic Integrate-and-fire)

Os modelos *integrate-and-fire* e *resonate-and-fire* são similares em alguns aspectos: ambos são descritos por equações diferenciais lineares, ambos possuem um limiar

fixo para disparo de *spikes*, e ambos apresentam um equilíbrio estável no estado de repouso. A única diferença é que este equilíbrio é um nodo no modelo integrador, mas um ponto de convergência (*focus*) no modelo ressonador. Estes dois modelos se complementam, e são competentes para provar teoremas e derivar expressões analíticas (IZHIKEVICH, 2004).

Muitos cientistas se referem as estes modelos neurais como 'modelos de *spikes*', porém faltam-lhes alguns mecanismos para geração dos *spikes*. Desta maneira, o modelo de *spike* mais simples que existe é o modelo integrador quadrático (*quadratic integrate-and-fire* (IZHIKEVICH, 2007a). Substituindo $-v$ por v^2 na equação 3.24 resulta-se o modelo de integrador quadrático.

$$\dot{v} = b + v^2,\ if\ v = v_{peak},\ then\ v \leftarrow v_{reset}, \tag{3.29}$$

Aqui, v_{peak} não é um limiar, mas sim o pico de um *spike*. Para estudos analíticos é interessante utilizar $v_{peak} = +\infty$. Já para simulações, o valor de pico é considerado elevado, porém finito, sendo assim pode-se normalizar $v_{peak} = 1$. Um ponto destacado por (IZHIKEVICH, 2007a) é que a equação $\dot{v} = b + v^2$ é uma forma topológica da bifurcação do tipo *saddle-node*, de maneira que ela descreve mecanismos de qualquer modelo similar ao modelo de Hodgkin-Huxley.

Ao contrário dos modelos lineares predecessores, o modelo integrador quadrático é um genuíno integrador. Ele apresenta uma bifurcação do tipo *saddle-node*; possui um limiar dinâmico; e gera *spikes* com latências, da mesma maneira que muitas células de mamíferos fazem. Além disso, o modelo quadrático é recomendado para simulações como redes de neurônios integradores.

3.5 Modelo de Izhikevich

Dentre os vários modelos existentes, o modelo que reproduz mais detalhadamente as características de um neurônio real é o modelo de Hodgkin-Huxley. Porém, ele possui um alto custo computacional, já que é descrito através de quatro equações diferenciais não-lineares acopladas, o que inviabiliza o seu uso em simulações em uma rede com um grande número de neurônios (OLIVEIRA, 2007). Por outro lado, usar um modelo do tipo *integrate-and-fire* é computacionalmente efetivo, porém o modelo é demasiadamente simples e incapaz de reproduzir determinados tipos de dinamismos exibidos por neurônios corticais.

Contextualizado com tal realidade, (IZHIKEVICH, 2003) desenvolveu um modelo de neurônio de *spikes* que tem como mérito apresentar plausibilidade biológica como no modelo de Hodgkin-Huxley, além de ser computacionalmente eficiente como os modelos do

tipo *integrate-and-fire*. Através de parâmetros, o modelo consegue reproduzir comportamentos de pulsos e de *bursting* de tipos conhecidos de neurônios corticais (como ilustrado na fig. 3.9).

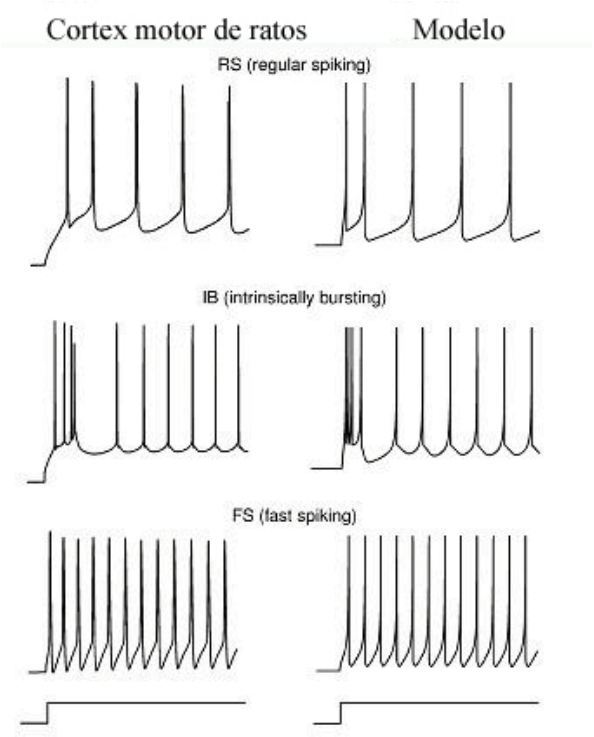

Figura 3.9: Reprodução de padrões de ativação de neurônios oriundos do córtex motor de ratos através do modelo de Izhikevich [Adaptado de (IZHIKEVICH, 2003)].

O modelo de Izhikevich é composto de um sistema bidimensional de equações diferenciais ordinárias

$$\dot{v} = 0.04v^2 + 5v + 140 - u + I \tag{3.30}$$

$$\dot{u} = a(b\,v - u) \tag{3.31}$$

com um *reset* pós-pulso (*after-spike reset*) dado por

$$if\ v \geq 30\,mV,\ else \begin{cases} v \leftarrow c \\ u \leftarrow u + d. \end{cases} \tag{3.32}$$

onde as variáveis v e u, bem como os parâmetros a, b, c e d são adimensionais. As variáveis presentes no modelo são:

- v representa o potencial membranal do neurônio, medida em mV;
- u contabiliza a ativação da corrente iônica de potássio, K^+, e a inativação da corrente de sódio, Na^+, além de fornecer uma realimentação negativa à v.

- I é uma variável que representa uma corrente originada de uma sinapse ou uma perturbação externa recebida pela rede.

Já o conjunto de parâmetros de ajuste é composto por:

- o parâmetro a determina a escala de tempo da variável u. Um valor típico para a é $a = 0.02$;
- o parâmetro b descreve a sensibilidade da variável u, além de influenciar na bifurcação juntamente com o parâmetro I. Um valor típico para b é $b = 0.2$;
- os parâmetros c e d correspondem, respectivamente, a valores impostos a v e adicionados à u após um *spike* (CHAVES, 2007). Valores típicos para c e d são $c = -65\,mV$ e $d = 2$.

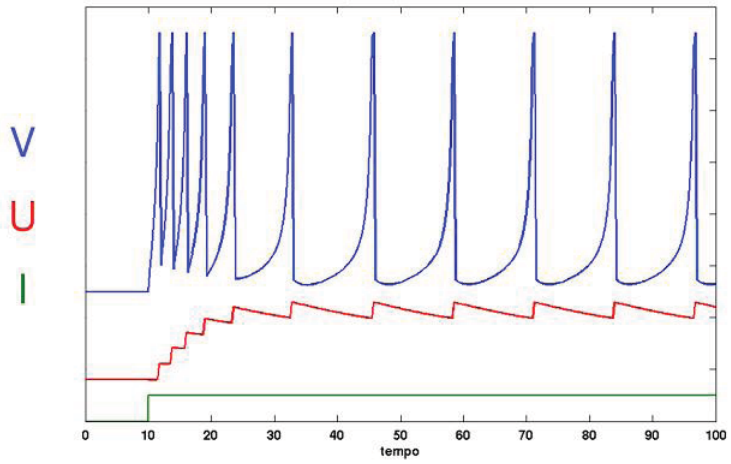

Figura 3.10: Evolução temporal do modelo de Izhikevich [Adaptado de (CHAVES, 2007)].

Os valores do conjunto de parâmetros não mudam durante o tempo de execução. Tais valores estabelecem qual tipo de dinamismo de neurônio deseja-se simular, sendo possível reproduzir mais de 20 tipos de neurônios reais (IZHIKEVICH, 2003). Em linhas gerais, u representa uma variável de recuperação da membrana (CHAVES, 2007). Quando o potencial de membrana atinge seu ápice $(30\ mV)$, o potencial de membrana (v) e a variável u são reiniciadas de acordo com a equação 3.32. O potencial de repouso do modelo situa-se entre -70 e $-50\ mV$ dependendo do valor de b. Como a maioria dos neurônios reais, o modelo não possui um limiar fixo; dependendo do histórico do potencial de membrana antecedente ao *spike*, o valor de potencial limiar pode ser baixo como $-55\ mV$ ou alto como $-40\ mV$ (OLIVEIRA, 2007).

A expressão $0.04v^2 + 5v + 140 - u + I$ foi obtida por meio de ajuste de parâmetros de tal modo que o potencial de membrana v é representado em unidades de mV

e o tempo t tenha escala de ms. Esta expressão é também a melhor que se ajusta para a simulação de redes de neurônios pulsantes de larga escala. Porém, caso seja necessário apenas a simulação de um simples neurônio a expressão $0.04v^2+4.1v+108-u+I$ se apresenta como uma opção melhor (IZHIKEVICH, 2003). O custo computacional dispendido para simulação de $1\ ms$ do modelo é de 13 operações de ponto flutuante, por isso é bastante eficiente em simulações de grande escala de redes corticais (IZHIKEVICH, 2004).

3.6 Modelo probabilístico

Ao longo dos anos muitos modelos de neurônios foram propostos na literatura. Entretanto, tais modelos são determinísticos e quando usados para modelar neurônios biológicos, eles apresentam limitações para resolver problemas complexos de engenharia. Sendo assim, novos modelos de redes neurais de *spikes* são necessários para substituir os já existentes para modelagem em larga escala (IZHIKEVICH; ELDELMAN, 2008).

Recentemente tem-se descoberto novas informações sobre os níveis neuronais, genéticos e níveis quânticos de processamento de informação em redes neurais biológicas. Exemplo disso pode ser encontrado em (KASABOV, 2010), onde o autor afirma que se um neurônio emite ou não um *spike* em um dado momento de tempo depende não apenas dos sinais de entrada, mas também das expressões de genes e proteínas, das propriedades físicas das conexões, da probabilidade de *spikes* serem recebidos nas sinapses, dos neurotransmissores sendo emitidos, dos canais iônicos estarem abertos, entre outros fatores.

Como processos envolvendo *spikes* nos neurônios biológicos são estocásticos por natureza, seria apropriado procurar novas inspirações para melhorar os modelos atuais de neurônios com parâmetros probabilísticos e obter modelos de neurônios de *spike*(pSNM) que amplie sua aplicabilidade (KASABOV, 2009). O modelo proposto por (KASABOV, 2010) armazena a informação nos pesos sinápticos e nos parâmetros probabilísticos relacionados com a ocorrência e propagação de *spikes*.

Um pSNM é esquematicamente mostrado na fig.3.11, onde um neurônio n_i recebe impulsos (*spikes*) de entrada dos seus neurônios pré-sinápticos n_j $(j = 1, 2, \ldots, m)$. O estado do neurônio n_i é descrito pela soma das entradas recebidas por todas as m sinápses - o potencial pós-sináptico $PSP_i(t)$. Quando o PSP_i atinge um determinado limiar de disparo $\upsilon_i(t)$, o neurônio n_i emite um *spike*. Os pesos das conexões $(w_{j,i}, j = 1, 2, \ldots, m)$, associado à sinapse, são estabilizados durante o aprendizado utilizando a regra de Thorpe (THORPE;

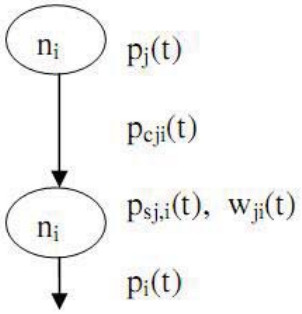

Figura 3.11: Representação de um pSNM mostrando uma única conexão sináptica [Adaptado de (KASABOV, 2010)].

DELORME; RULLE, 2001):

$$\Delta w_{i,j} = mod^{order(j)} \tag{3.33}$$

onde "mod" é um fator de modulação (um parâmetro entre 0 e 1) e $order^j$ é a ordem em que o *spike* emitido pelo neurônio n_j chega à sinapse $s_{j,i}$ em relação ao tempo de chegada dos *spikes* dos outros neurônios, depois que o neurônio n_i emitiu um *spike*. Esta é uma regra de aprendizado veloz que requer que os dados sejam propagados apenas uma vez (KASABOV, 2009).

Além dos pesos de sinapses $w_{j,i}(t)$, o pSNM possui três parâmetros probabilísticos:

1. A probabilidade $p_{cj,i}$ que um *spike* emitido pelo neurônio n_j irá alcançar o neurônio n_i no momento t através de uma conexão entre n_i e n_j. Este parâmetro é atribuído a cada conexão para representar suas incertezas estruturais e funcionais;

2. A probabilidade $p_{sj,i}$ da sinapse $s_{j,i}$ contribuir para o $PSP_i(t)$ depois que ele tenha recebido um *spike* do neurônio n_j. Este parâmetro pode mudar durante o aprendizado da rede, na maioria dos casos seu valor é $p_{sj,i} = 1$, a menos que existam razões para incluir este parâmetro no modelo de neurônio;

3. O parâmetro probabilístico $p_i(t)$ de um neurônio n_i emitir um *spike* de saída em um determinado tempo t, uma vez que o $PSP_i(t)$ total tenha alcançado um valor em torno do limiar do potencial pós-sináptico. Este conceito é conhecido como densidade de probabilidade de disparo de um *spike* em um neurônio biológico com ruído. O valor do $PSP_i(t)$ é calculado usando-se a seguinte fórmula

$$PSP_i(t) = \sum_{p=t_0,\ldots,t} \sum_{j=1,\ldots,m} e_j g(p_{cj,i}(t-p)) f(p_{sj,i}(t-p)) w_{j,i}(t) + \eta(t+t_0) \tag{3.34}$$

onde e_j é 1 se um *spike* foi emitido, caso contrário será 0; $g(p_{cj,i}(t))$ é 1 com uma probabilidade $p_{cj,i}(t)$, e 0 em caso contrário; $f(p_{sj,i}(t))$ é 1 com uma probabilidade $p_{sj,i}(t)$, e 0 em caso contrário; t_0 é o instante de tempo do último *spike* emitido pelo neurônio n_i; $\eta(t - t_0)$ é um termo adicional representando enfraquecimento do PSP.

Como um caso especial, quando todos ou alguns dos parâmetros probabilísticos são fixados em "1", o pSNM será simplificado e assemelhar-se-á a algum outro modelo de neurônio já conhecido, como o modelo *integrate-and-fire* (KASABOV, 2010).

Apesar de ser um modelo mais recente, ele vem sendo aplicado em vários tipos de aplicações, incluindo tarefas de classificação, como reconhecimento de face; autenticação pessoal baseada em informação audiovisual, etc (KASABOV, 2007a; KASABOV, 2007b).

4 NAVEGAÇÃO AUTÔNOMA E MODELOS BIO-INSPIRADOS

Na última década, tem ocorrido um aumento no número de robôs construídos que implementam comportamentos biológicos de navegação. Sistemas bioinspirados trazem contribuições significantes para dois campos de pesquisa: primeiramente eles fornecem testes reais para os modelos de navegação biológicos; em segundo lugar eles criam novos mecanismos para navegação disponíveis para aplicações técnicas.

Em seu sentido original, o termo navegação descreve o processo de direcionar um navio ao seu destino. Este processo consiste basicamente da repetição de três etapas: (a) o navegador determina a posição do navio em um mapa com a maior precisão possível; (b) no mapa, o navegador relaciona a posição do navio com o destino, relacionando também as posições circundantes e possíveis riscos; (c) baseado nesta informação, ele define o novo rumo do navio (FRANZ; MALLOT, 2000).

Levitt e Lawton (LEVITT; LAWTON, 1990) definiram a navegação como sendo um processo que responde a três perguntas: (a) "Onde eu estou?"; (b) "Onde estão as demais posições em relação à minha localização?"; e (c) "Como eu posso ir aos outros lugares a partir da posição em que me encontro?". Aplicado na área de navegação de robôs, isto significa que as entradas sensoriais do robô são usadas para atualizar uma representação única e global do ambiente, a partir da qual as ações motoras são determinadas pelo procedimento de inferência elaborado. Esta visão de navegação tem sido adotada na robótica tradicional e compõe a base de muitos sistemas de navegação de robôs.

Entretanto nenhum destes sistemas tradicionais alcançou ainda a flexibilidade e o desempenho da navegação de abelhas ou formigas, ou pássaros e peixes migratórios, por exemplo. Tal cenário tem motivado pesquisadores de robótica a procurar por mecanismos de navegação biológica que possam ser implementados em robôs móveis autônomos. Além disso, algumas pesquisas etológicas (disciplina que estuda o comportamento animal) mostraram que muitos animais (incluindo os seres humanos) são capazes de navegar sem responder todas

as três questões de Levitt e Lawton, ou até mesmo nenhuma delas. "Onde eu estou?" não é necessariamente a pergunta mais importante a se responder. Para animais 'navegantes' a principal questão existente é "Como eu posso alcançar o meu objetivo?".

A modelagem do comportamento adaptativo através do desenvolvimento de agentes autônomos é uma abordagem amplamente investigada nas áreas de inteligência artificial e controle autônomo, e um modelo particular de comportamento é representado por um agente que está motivado para tentar sobreviver em um ambiente definido, sem qualquer ajuda externa. O agente pode gerar suas ações exclusivamente das informações sensoriais disponíveis, ou pode usar algum tipo de experiência prévia (NOVELLINO et al., 2007).

4.1 Conceitos básicos

Robôs inteligentes podem ser utilizados para diversas aplicações. Mas de acordo com Murphy (MURPHY, 2000) eles são empregados quando: (1) existe um risco significativo à segurança humana (nuclear, espacial, militar); (2) a economia ou a natureza do serviço resulta no uso ineficiente de mão-de-obra humana (serviços industriais, agricultura); e (3) para ações humanitárias de grande risco (retirada de minas terrestres, serviços de busca e resgate).

De acordo com Gibilisco (GIBILISCO, 2003), um agente que demonstra comportamento autônomo é auto-suficiente, recebe seu próprio controlador e não depende de nenhum computador central para estabelecimento de comandos. Ele realiza a navegação em seu ambiente de trabalho através de seus próprios meios, que normalmente são rodas ou uma unidade de pista. Uma vez que a noção clássica de navegação captura somente uma parte dos fenômenos biológicos observados, Franz e Mallot (FRANZ; MALLOT, 2000) adotam a seguinte definição para navegação:

> *"Navigation is the process of determining and maintaining a course or trajectory to a goal location".* [1]

As capacidades mínimas para navegação são, portanto, mover-se no espaço e determinar se o objetivo foi ou não encontrado. Para este fim, as características sensoriais que determinam o objetivo devem ser armazenadas em alguma forma de memória de longa duração. Esta noção de navegação não implica que a posição corrente do agente deve ser

[1] Navegação é o processo de determinar e manter um curso ou trajetória até uma posição desejada (objetivo).

reconhecida, nem que uma representação da forma de um mapa deva ser usada para encontrar o objetivo.

A referência a uma posição objetivo distingue a navegação das outras formas de comportamento espacial, como a *exploração, prevenção de colisões, estabilização de curso*, entre outras. As formas de comportamento espacial capturadas pela definição prévia de navegação podem ser agrupadas em duas categorias: esquemas de navegação local (*local navigation*) e esquemas que encontram uma rota (*way-finding*). Modelos de navegação local são processos de movimentação do agente em seu ambiente imediato, isto é, na porção do ambiente onde os objetos são perceptíveis ao agente. Além disso, necessitam do reconhecimento de uma única posição, a meta (ou objetivo). O agente escolhe suas ações com base nas informações sensoriais correntes, sem auxílio de nenhum tipo de representação de objetos ou lugares situados fora da sua capacidade de percepção (TRULLIER et al., 1997). Já encontrar um caminho envolve o reconhecimento de vários lugares, e a representação das relações entre lugares que podem estar fora da área de sensoriamento corrente.

4.2 A hierarquia da navegação

De acordo com Trullier et al. (TRULLIER et al., 1997) os esquemas de navegação podem ser classificados de acordo com a complexidade das tarefas que se deseja realizar. Problemas de navegação local podem ser divididos em quatro níveis: (i) navegação por busca (*searching*); (ii) navegação seguindo uma direção (*direction-following*); (iii) navegação por apontamento (*aiming*); e (iv) navegação com orientação (*guidance*). Já as técnicas que visam encontrar um caminho ou rota (*way-finding*) podem ser divididas em três níveis: (i) navegação por respostas de reconhecimento desencadeado (*recognition-triggered responses*); (ii) navegação topológica (*topological navigation*); e (iii) navegação com pequisa (*survey navigation*) (figuras 4.1 e 4.2).

- *Navegação por busca (search).* Um agente navegando através de busca não recebe nenhum tipo de orientação em direção ao objetivo. A posição final é encontrada ao acaso, apenas se o agente depara-se com ela enquanto move-se pelo ambiente. É o esquema mais simples de navegação e requer apenas as competências básicas para locomoção e detecção do objetivo, sem necessitar de nenhum tipo de representação espacial. A busca exige uma grande quantidade de tempo comparada com os outros métodos de navegação. Uma vez que a direção do objetivo não precisa ser conhecida, este modelo pode ser usado como uma estratégia de *backup* para quando o agente não

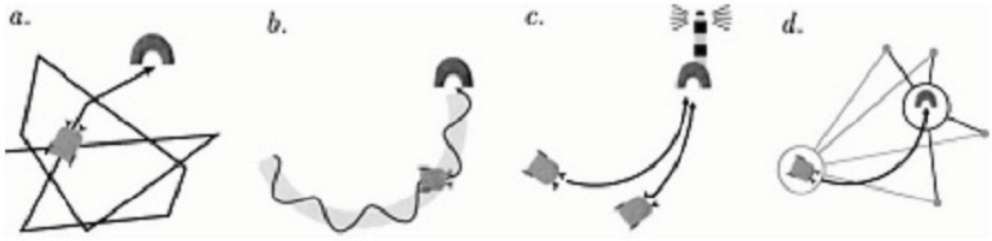

Figura 4.1: Esquemas de navegação local: (a) um agente utilizando um esquema de busca sem nenhum tipo de orientação até o objetivo; (b) esquema *direction-following* que permite que um agente encontre o objetivo por meio de uma trilha; (c) o esquema de 'apontamento' (*aiming*) necessita de uma dica sensorial saliente indicando a direção do objetivo; (d) já um agente utilizando esquema com orientação consegue encontrar seu objetivo através das relações espaciais dos objetos que o circulam [Adaptado de (FRANZ; MALLOT, 2000)].

consegue encontrar o objetivo.

- *Navegação seguindo uma direção (direction-following).* Neste tipo de navegação o agente deve ser capaz de direcionar seu percurso com uma direção local disponível (uma trilha) a fim de localizar o objetivo. O objetivo não precisa necessariamente ser perceptível ao longo da movimentação. Uma vez que este modelo é mais eficiente do que o esquema por busca, ele permite que o agente localize o objetivo somente quando o robô movimenta-se na trilha definida pela direção. Se o agente é deslocado da trilha, ele não encontrará o objetivo. Se além disso, a distância até o objetivo é conhecida, o esquema torna-se mais eficiente, uma vez que o agente pode alternar para outra estratégia depois que o objetivo não tenha sido detectado. Uma estratégia para adquirir a direção e distância da posição objetivo é chamada de integração de caminho na biologia, ou de *odometria* na robótica. Um agente utilizando de odometria é capaz de retornar à posição inicial a partir de qualquer ponto de trajetória, enquanto o processo de integração não seja interrompido.

- *Navegação por apontamento (aiming).* Um agente que encontra-se no objetivo tem que orientar o eixo de seu corpo de acordo com a posição espacial do objetivo. O objetivo deve estar associado à alguma dica relevante, seja ela olfativa, auditiva, visual, ou de outra maneira que seja sempre perceptível durante a movimentação. Diferentemente do modelo anterior (orientado por uma tilha), o objetivo pode ser alcançado a partir de várias posições diferentes sem o perigo de acumular o erro.

- *Navegação com orientação (guidance).* Quando o objetivo não é identificado por uma dica visível, o agente pode ser guiado pela configuração espacial dos objetos circundantes. A orientação é um processo por meio do qual uma certa relação egocêntrica, no

que diz respeito a um objeto ou paisagem particular, é mantida. Assim, a informação espacial não é somente uma única direção ou localização, mas uma relação espacial entre a posição corrente, o objetivo e o ambiente perceptível no momento. Esse tipo de navegação é um esquema de navegação local já que necessita apenas do processamento das informações atuais (sensoriais ou internas). Um exemplo de navegação com orientação poderia ser um navio que tenta alcançar uma posição fixa entre algumas ilhas.

- *Navegação por respostas de reconhecimento desencadeado (recognition-triggered responses).* Este tipo de navegação conecta duas localidades por meio de um método de navegação local. Um modelo desta natureza pode ser denotado formalmente por um par ordenado (posição inicial, método de navegação local), envolvendo não só o reconhecimento do objetivo, como também da posição inicial. Não existe um planejamento da sequência de movimentos, somente da seção da ação mais próxima. Navegação por reconhecimento desencadeado pode ser usada como um processo de navegação elementar para a criação de *rotas*. Rotas são sequências de respostas de reconhecimento desencadeado, em que a realização do objetivo de uma etapa desencadeia o início da próxima etapa. Se um segmento da rota é bloqueado por um obstáculo, o agente tem de recorrer a uma estratégia de busca até que ele reconheça uma posição novamente.

- *Navegação topológica (topological navigation).* Um agente utilizando de reconhecimento desencadeado é limitado a usar sempre a mesma sequência de localizações (mesma rota). As rotas são geradas independentemente uma da outra e cada objetivo necessita de sua própria rota. Para isso, o agente precisa ter a competência de detectar onde duas rotas passam no mesmo local. Duas possibilidades diferentes de configurações sensoriais associadas com rotas distintas que passam pelo mesmo lugar devem ser mescladas via uma 'integração de rotas'. Uma coleção de trajetórias integradas torna-se então uma representação topológica do ambiente. Esta representação pode ser expressada matematicamente através de um grafo, onde os vértices representam posições e as arestas representam um método de navegação local conectando dois vértices. O fato de que rotas alternativas convergem para um mesmo objetivo exige o uso de técnicas de planejamento que geram rotas a partir do grafo. Como consequência, um agente dependente de um esquema de navegação topológica não gera novas trajetórias ao longo de terrenos não visitados.

- *Navegação por pesquisa (survey navigation).* Navegação por pesquisa requer a incorporação de todos os lugares conhecidos e de suas relações espaciais em um quadro de referência comum. A representação espacial deve ser manipulada e acessível como um

todo, assim, as relações espaciais entre quaisquer duas localizações representadas podem ser inferidas. Exemplos deste tipo de navegação incluem verificação de atalhos ou desvio de obstáculos em terrenos desconhecidos.

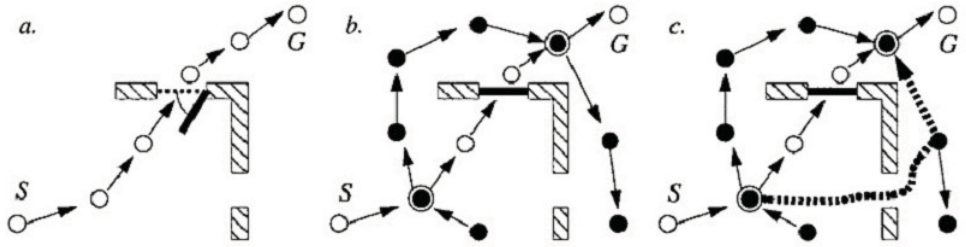

Figura 4.2: Esquemas para localizar rotas: (a) navegação por respostas de reconhecimento desencadeado não pode adaptar-se a novos obstáculos, como por exemplo, a existência de um porta fechada (S - posição inicial, G - posição objetivo); (b) navegação topológica permite uma concatenação flexível por rotas; (c) navegação por pesquisa permite que o agente encontre atalhos em um ambiente desconhecido [Adaptado de (FRANZ; MALLOT, 2000)].

Exemplos de mecanismo de navegação local, por respostas de reconhecimento desencadeado e navegação topológica podem ser encontrados no mundo animal, ao passo que navegação por pesquisa é limitada aos vertebrados. A abordagem clássica da robótica para a navegação é rememorativo à navegação por pesquisa, uma vez que o conhecimento espacial é representado em um mapa global comum.

Técnicas bioinspiradas podem ser encontradas em todos os níveis da hierarquia de navegação, com exceção da navegação por busca e pesquisa. Talvez devido a sua baixa eficiência, o modelo por busca não tem recebido muita atenção dos pesquisadores. Por outro lado, a busca por pesquisa exige a presença de todas as habilidades dos níveis inferiores da hierarquia, muitos deles continuam pouco compreendidos.

4.3 Prevenção de colisões (*Obstacle avoidance*)

Técnicas para planejamento de movimento computam trajetórias livre de colisões de acordo com as restrições do veículo, geralmente assumindo um modelo perfeito do ambiente e do robô. Entretanto, quando o ambiente circundante é desconhecido e imprevisível tais técnicas apresentam falhas. Uma alternativa de complementação do problema de navegação é a prevenção de colisões com obstáculos (*obstacle avoidance*). O objetivo é mover o veículo até o objetivo sem nenhuma colisão, tendo possíveis obstáculos detectados pelos sensores durante a execução. A vantagem de se utilizar um modelo de prevenção de colisões reativo é que o movimento passa a ser computado com o auxílio das informações obtidas

pelos sensores, adaptando o movimento a qualquer situação imprevisível e incompatível com os planos iniciais da navegação.

4.3.1 Definição

Seja A o robô (um objeto rígido) movendo-se no espaço de trabalho W, cuja configuração do espaço é CS. Sendo q uma configuração, em um instante de tempo t é dada por q_t e $A(q_t) \in W$ é o espaço ocupado pelo robô nesta configuração. No robô existem também sensores, que no instante q_t medem uma porção do espaço $S(q_t) \subset W$, identificando um conjunto de obstáculos $O(q_t) \subset W$ (SICILIANO; KHATIB, 2008)..

Seja b um vetor de controle constante e que $u(q_t)$ seja esse vetor de controle aplicado em q_t durante um intervalo de tempo δt. Dado $u(q_t)$, o veículo descreve uma trajetória $q_{t+1} = f(u, q_t, \delta t)$, onde $\delta t \geq 0$. Sendo $Q_{t,T}$ o conjunto de configurações da trajetória partindo de q_t com $\delta t \in [0, T]$, um dado intervalo de tempo. $T > 0$ é chamado de *período de amostragem (sampling period)*.

Seja $F : CS \times CS \rightarrow \Re^+$ uma função que avalia o progresso de uma configuração para outra.

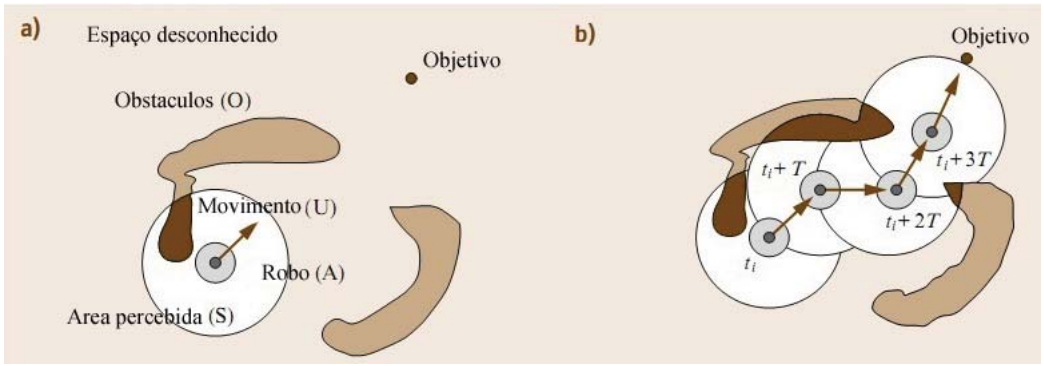

Figura 4.3: Prevenção de colisões (a) definição do problema, (b) o resultado da técnica aplicada a cada passo de tempo gerando uma trajetória livre de colisões (SICILIANO; KHATIB, 2008).

Seja q_{target} uma configuração objetivo. Então, de acordo com as definições prévias, no tempo t_i o robô A está em q_{t_i}, onde os valores de medida dos sensores $S(q_{t_i})$ são obtidos, e consequentemente uma descrição dos obstáculos $O(q_{t_i})$. O objetivo é comparar um controle de movimento u_i tal que: (i) a trajetória gerada seja livre de colisões $A(Q_{t_i}, T) \cap O(qt_i) = \emptyset$; e (ii) que ele faça o robô se aproximar de sua posição objetivo $F(q_{t_i}, q_{target}) < F(q_{t_i+T}, q_{target})$.

O resultado de resolver este problema a cada intervalo de amostragem é uma sequência de controles de movimento $u_1 \ldots u_n$, computados em tempo de execução que evitam

os obstáculos através dos valores medidos pelos sensores, enquanto fazem o robô aproximar de sua posição final (objetivo) $q_{t_i} \ldots q_{target}$ (fig.4.3). Em (SICILIANO; KHATIB, 2008) os autores afirmam que existe pelo menos três aspectos que afetam o desenvolvimento de um método de prevenção de colisões: a técnica aplicada, o tipo de sensoriamento do robô, e o tipo de cenário explorado.

4.3.2 Técnicas para prevenção de colisão com obstáculo

Existem dois grupos de técnicas para prevenção de colisões: métodos que computam o movimento em uma etapa, e métodos que computam em mais de uma. Métodos de uma etapa transformam diretamente a informação dos sensores em instruções de controle. Subdividem-se em dois tipos:

- Métodos heurísticos: foram as primeiras técnicas utilizadas para gerar movimentação baseada nos sensores.
- Métodos de analogias físicas: assimilam o comportamento de prevenção de colisões a um problema físico conhecido.

Os métodos de mais de uma etapa calculam informações intermediárias, que são processadas para a obtenção da ação de controle do movimento. Tais métodos podem ser:

- Métodos de *subconjuntos de controle*, que calculam um conjunto intermediário de ações de movimentação, e depois escolhem uma como solução. Estes métodos podem computar um subconjunto de direções de movimento (método do histograma de campo vetorial, método de restrição de obstáculo); ou computar um subconjunto de velocidades de controle (abordagem de janela dinâmica).
- Outros métodos calculam algumas *informações de alto nível* como informações intermediárias, que são traduzidas como o próximo comando de movimento (diagrama de navegação de proximidade (SICILIANO; KHATIB, 2008)).

Todos estes métodos descritos possuem vantagens e desvantagens dependendo do contexto da navegação, como movimentação em alta velocidade, mundos incertos, movimentação em espaços restritos, etc.

4.3.2.1 Método do campo potencial (*Potential Field Method - PFM*)

Um dos métodos mais populares é o método do campo potencial (PFM), que utiliza uma analogia em que o robô é uma partícula que se move no espaço de configuração sob influência de um campo de força (HOLLAND, 2004). Enquanto a posição objetivo exerce uma força atrativa à partícula $(Fatt)$, os obstáculos exercem uma força repulsiva (F_{rep}). Em cada instante de tempo t_i, o movimento é calculado para seguir a direção da força artificial resultante inferida pela soma de ambos os potenciais (atrativo e repulsivo) $F_{tot}(q_{t_i}) = F_{att}(q_{t_i}) + F_{rep}(q_{t_i})$ (fig.4.4):

$$F_{att}(q_{t_i}) = K_{att} n_{q_{target}} \tag{4.1}$$

$$F_{rep}(q_{t_i}) = \begin{cases} K_{rep} \sum_j \left(\frac{1}{d(q_{t_i}, p_j)} - \frac{1}{d_0} \right) n_{p_j}, \ if \ d(q_{t_i}, p_j) < d_0 \\ 0, \ otherwise. \end{cases} \tag{4.2}$$

K_{att} e K_{rep} são as constantes das forças, d_0 é a distância da influência dos obstáculos p_j, q_{t_i} é a configuração atual do veículo, e n_{target} e n_{p_j} são os vetores unitários que apontam de q_{t_i} para a posição objetivo e cada obstáculo p_j, respectivamente. A partir de $F_{tot}(q_{t_i})$, a ação de controle u_i pode ser obtida com uma posição ou força de controle. Este método é largamente utilizado devido ao seu fácil entendimento e por ter um formalismo matemático bem claro (SICILIANO; KHATIB, 2008).

4.3.2.2 Método do histograma de campo vetorial (*Vector Field Histogram - VFH*)

O método VHF resolve o problema de colisões em dois passos: cálculo de um conjunto de direções candidatas, e seleção de uma destas direções de movimento. Primeiramente o espaço é dividido em setores a partir da localização do robô. O método utiliza um histograma polar H construído em torno do robô, onde cada componente representa uma densidade polar de obstáculo no setor correspondente. A função que mapeia a distribuição de obstáculos no setor k em um componente correspondente do histograma $h^k(q_{t_i})$ é:

$$h^k(q_{t_i}) = \int_{\Omega_k} P(p)^n \left(1 - \frac{d(q_{t_i}, p)}{d_m ax} \right)^r dp \tag{4.3}$$

O domínio da integral é $\Omega_k = \{p \in W | p \in k \wedge d(q_{t_i}, p) < d_0\}$. A densidade

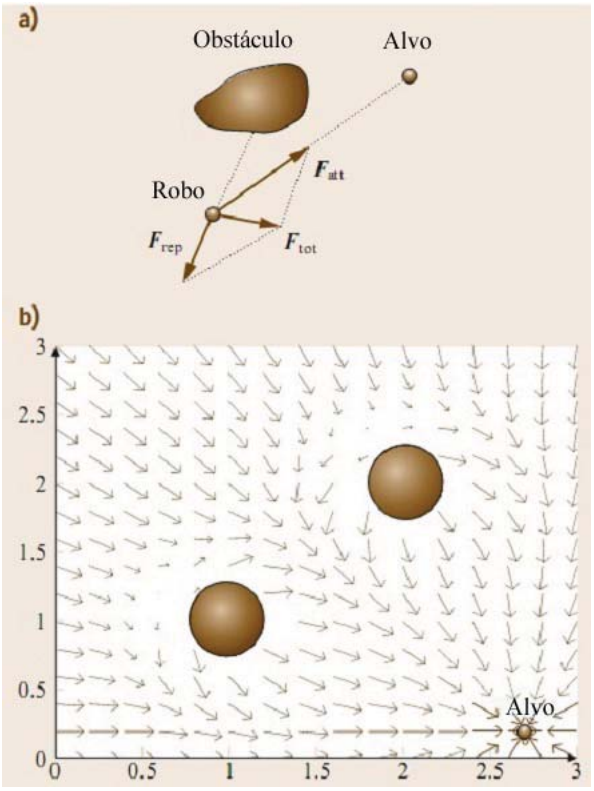

Figura 4.4: (a) Cálculo da direção de movimento através do método do campo potencial. (b) Cálculo das direções calculadas em cada ponto do espaço com o método [Adaptado de (SICILIANO; KHATIB, 2008)].

$h^k(q_{t_i})$ é proporcional à probabilidade $P(r)$ de que um ponto estivesse sendo ocupado por um obstáculo, e proporcional também ao fator que aumenta à medida que a distância até o ponto diminui.

Tipicamente, o histograma resultante apresenta picos (direções com uma alta densidade de obstáculos) e vales (direções com baixa densidade). O conjunto de direções candidatas é o conjunto de componentes adjacentes com menor densidade que dado um valor limiar, e mais perto do componente que contém a direção do objetivo. Este conjunto de componentes (setores) é identificado como vale selecionado, e representa um conjunto de direções candidatas (fig.4.5).

O objetivo da próxima etapa é selecionar uma direção deste conjunto de componentes elaborado. A estratégia é aplicar três heurísticas que dependem do componente que contém o alvo ou o tamanho do vale selecionado. O casos são verificados em sequência:

- Caso 1: setor objetivo está no vale selecionado. Solução: $k_{sol} = k_{target}$, onde k_{target} é o setor que contém a posição objetivo.
- Caso 2: setor objetivo não está no vale selecionado e o número de setores no vale é

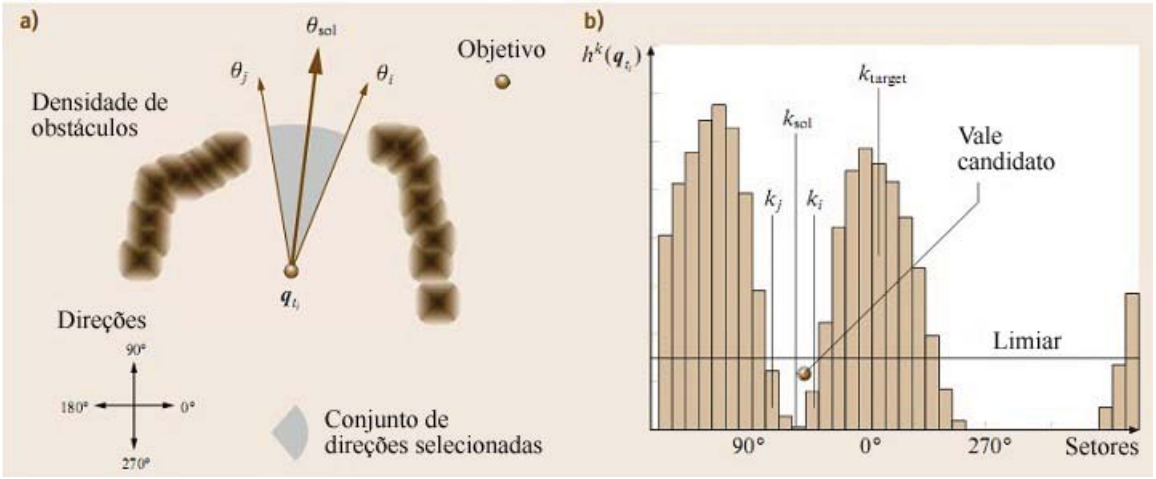

Figura 4.5: (a) Distribuição do robô e obstáculos no espaço. (b) Escolha do vale candidato e composição da ação de movimentação [Adaptado de (SICILIANO; KHATIB, 2008)].

maior que m. Solução: $k_{sol} = k_i \pm \frac{m}{2}$, onde m é um número fixo de setores e k_i é o setor do vale mais próximo de k_{target}.

- Caso 3: setor objetivo não está no vale selecionado e o número de setores no vale é menor ou igual a m. Solução: $k_{sol} = \frac{k_i + k_j}{2}$, onde k_i e k_j são os setores extremos do vale.

O resultado é o setor k_{sol} cuja bissetriz é a direção solução θ_{sol}. A velocidade v_{sol} é inversamente proporcional à distância ao objeto mais próximo. A ação de controle é $u_i = (v_{sol}, \theta_{sol})$. O método VFH é um método formulado para trabalhar com a probabilidade de distribuição dos obstáculos, sendo bem adaptado para trabalhar com sensores incertos como sonares ultra-sônicos (SICILIANO; KHATIB, 2008).

4.3.2.3 Método de restrição ao obstáculo (*The obstacle restriction method - ORM*)

O método resolve o problema em três etapas, onde o resultado das duas primeiras etapas é um conjunto de direções candidatas ao movimento. O primeiro passo é calcular um sub-objetivo instantâneo, se necessário. O próximo passo então associa uma restrição de movimento para cada obstáculo, juntando-as em seguida para calcular o conjunto de direções desejáveis. O último passo é uma estratégia para calcular o movimento, dado este conjunto elaborado.

A primeira parte do algoritmo é denominada de *seleção de alvos instantâneos*. Este passo calcula um sub-objetivo para situações onde é melhor mover em direção a uma determinada região do espaço, ao invés de mover-se diretamente ao objetivo. Estes sub-

objetivos são localizados entre obstáculos ou na borda de um obstáculo (fig.4.6a). Em seguida o processo verifica (através de um algoritmo local) se a meta pode ser alcançada a partir da localização do robô. se isto não for possível, então o sub-objetivo mais próximo é selecionado.

Para cada obstáculo i é calculado um conjunto das direções não desejáveis para movimentação S^i_{nD}. Este conjunto é a união de dois subconjuntos: S^i_1 e S^i_2. O conjunto S^i_1 representa o lado do obstáculo inadequado para evitar a colisão, e S^i_2 é uma área de exclusão em volta do obstáculo (fig.4.6b). A restrição do movimento para o obstáculo é a união destes dois conjuntos $S^i_{nD} = S^i_1 \cup S^i_2$. O conjunto de direções desejáveis para movimentação é o conjunto complementar $S_D = \{[-\phi, \phi] \mid dS_{nD}\}$, onde $S_{nD} = \bigcup_i S^i_{nD}$.

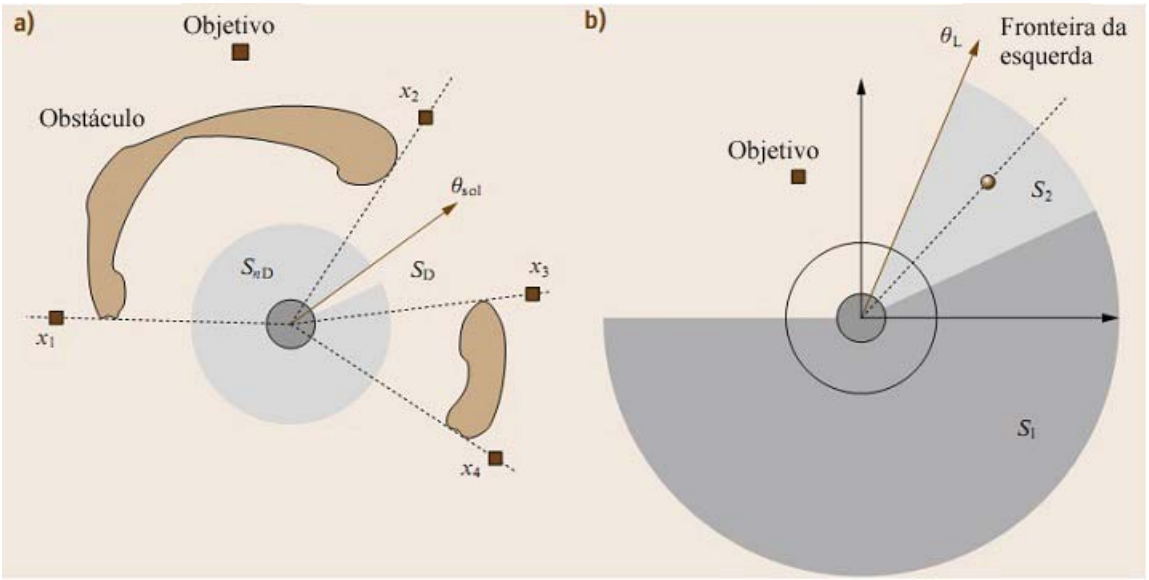

Figura 4.6: (a) Distribuição de sub-objetivos x_i. (b) Os dois conjuntos de direções indesejadas S_1 e S_2 para um dado obstáculo [Adaptado de (SICILIANO; KHATIB, 2008)].

A etapa final é selecionar a direção do movimento. Existem três casos, dependendo do conjunto de direções desejáveis S_D e da direção do objetivo θ_{target}:

- Caso 1: $S_D \neq \emptyset$ e $\theta_{target} \in S_D$. Solução: $\theta_{sol} = \theta_{target}$.
- Caso 2: $S_D \neq \emptyset$ e $\theta_{target} \notin S_D$. Solução: $\theta_{sol} = \theta_{lim}$, onde θ_{lim} é a direção de S_D mais próxima de θ_{target}.
- Caso 3: $S_D = \emptyset$. Solução: $\theta_{sol} = \frac{\phi^l_{lim} + \phi^r_{lim}}{2}$, onde ϕ^l_{lim} e ϕ^r_{lim} são as direções de S_{Dr} e S_{Dl} mais próximas a θ_{target} (S_{Dl} e S_{Dr} são os conjuntos de direções desejáveis dos obstáculos à esquerda e à direita do objetivo, respectivamente).

O resultado é uma solução θ_{sol} para a direção do movimento. A velocidade v_{sol} é inversamente proporcional à distândia ao obstáculo mais próximo. A ação de controle

é $u_i = (v_{sol}, \theta_{sol})$. Este é um método baseado geometricamente em estudo de casos, e apresenta vantagens para execução de movimentos efetivos em espaços restritos (SICILIANO; KHATIB, 2008).

4.3.2.4 Aproximação por janela dinâmica (*Dynamic window approach - DWA*)

O método DWA é um método que resolve o problema de navegação livre de colisões em duas etapas, utilizando um subconjunto do espaço de controle U como informação intermediária. Considerando uma ação de controle de movimento como um par ordenado composto por velocidades de translação e rotação (v, w). O conjunto U é definido por:

$$U = (v, w) \in \Re^2 \mid v \in [-v_{max}, v_{max}] \wedge w \in [-w_{max}, w_{max}]. \tag{4.4}$$

O conjunto candidato de ações de controle U_R contém ações que: (i) estão dentro das velocidades máximas do veículo U; (ii) geram trajetórias seguras U_A; e (iii) podem ser alcançadas dentro de um curto período de tempo dadas as acelerações do veículo U_D. O conjunto U_A contém as ações admissíveis. Estas ações de controle podem ser canceladas antes de uma colisão aplicando-se a desaceleração máxima (a_v, a_w):

$$U_A = \left\{(v, w) \in U \mid v \leq \sqrt{2d_{obs}a_v} \wedge w \leq \sqrt{2\theta_{obs}a_w}\right\} \tag{4.5}$$

onde d_{obs} e θ_{obs} são a distância ao obstáculo e a orientação do plano tangente à trajetória ao longo do obstáculo, respectivamente. O conjunto U_D contém as ações de controle alcançáveis em um curto período:

$$U_D = \{(v, w) \in U \mid v \in [v_0 - a_v T, v_0 + a_v T] \wedge w \in [w_0 - a_w T, w_0 + a_w T]\} \tag{4.6}$$

onde $\dot{q_{t_i}} = (v_0, w_0)$ é a velocidade corrente.

O subconjunto resultante de ações de controle é (fig.4.7)

$$U_R = U \cap U_A \cap U_D. \tag{4.7}$$

O próximo passo é selecionar uma ação de controle $u_i \in U_R$. O problema é

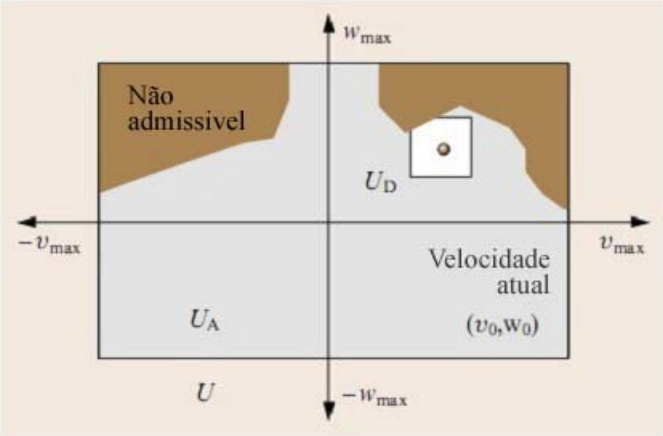

Figura 4.7: Subconjuntos de ações de controle $U_R = U \cap U_A \cap U_D$, onde U contém as ações com as velocidades máximas, U_A os controles admissíveis, e U_D os controles alcançáveis a partir de um curto período de tempo [Adaptado de (SICILIANO; KHATIB, 2008)].

definido como a maximização de uma função objetivo:

$$G(u) = \alpha_1 \cdot Goal(u) + \alpha_2 \cdot Clearance(u) + \alpha_3 \cdot Velocity(u). \tag{4.8}$$

Esta função é um "compromisso" entre Goal(u), que favorece as velocidades que oferecem progresso ao objetivo; Clearance(u), que favorece as velocidades longe dos obstáculos, e Velocity(u) que favorece às altas velocidades. A solução é uma ação de controle u_i que maximiza essa função. O método DWA resolve o problema no espaço de controle utilizando de informações de dinamismo do veículo, assim sendo, o método é melhor adaptado para trabalhar com veículos de baixa capacidade dinâmica ou que funcionem à altas velocidades (SICILIANO; KHATIB, 2008).

Resumindo, o problema de prevenção de colisões pode ser formalizado e tratado sob várias técnicas, sendo cada uma delas detentora de particularidades na maneira que abstraem e representam o problema. Constrastando com as técnicas aqui descritas, o próximo capítulo traz a modelagem de um modelo conexionista bioinspirado que será aplicado na mesma problemática de prevenção de colisões.

5 METODOLOGIA

Neste capítulo serão descritas as atividades e procedimentos realizados no processo de modelagem do problema explorado pela dissertação. O foco é a utilização de um modelo de redes neurais de *spikes* aplicado em uma tarefa de navegação autônoma de robôs, gerando movimentações livre de colisões. Desta maneira, serão descritos os processos de modelagem conceitual da rede, destacando o algoritmo de aprendizado STDP - que codifica um fluxo de informação utilizando os tempos de emissão de *spikes* dos neurônios da rede; a etapa de simulação numérica - para validação do modelo e ajuste de parâmetros; o processo de prototipação explorando um ambiente virtual - MRDS; e o processo empregado para realização das experimentações físicas.

5.1 Modelagem Conceitual da Rede

Por modelagem conceitual da rede designa-se o processo de especificação dos principais elementos de uma SNN:

- a unidade computacional - modelo artificial de neurônio que reproduza *spikes*;
- a topologia da rede - estrutura e organização dos neurônios e suas interações (sinapses);
- o algoritmo de aprendizado - regra que reproduza a plasticidade sináptica das células nervosas, capacidade que o sistema possui de adaptar-se a um conjunto de estímulos do ambiente, representando de maneira adequada as regularidades dos mesmos.

5.1.1 Unidades de processamento

O primeiro passo é encontrar um modelo de neurônio artificial que reproduza o comportamento de um neurônio biológico, principalmente à emissão de *spikes*. No capítulo **??** (*Modelos de Neurônios Biológicos*) foi realizada uma pesquisa bibliográfica sobre

alguns modelos existentes na literatura, partindo de modelos clássicos até modelos mais atuais. Com base em tal pesquisa efetuada, e em trabalhos correlacionados na literatura ((ARENA et al., 2009), (ESCOBAR; AL, 2009), entre outros) optou-se pelo uso do modelo de Izhikevich (IZHIKEVICH, 2003)(equações 3.30, 3.31, 3.32).

O formalismo do modelo de Izhikevich é retratado de maneira simples, e consegue reproduzir vários padrões neurais de potenciais de ativação modificando-se apenas os parâmetros do modelo, o que facilita até mesmo sua implementação. Consequentemente, optou-se por este modelo para representar o funcionamento de cada neurônio da SNN.

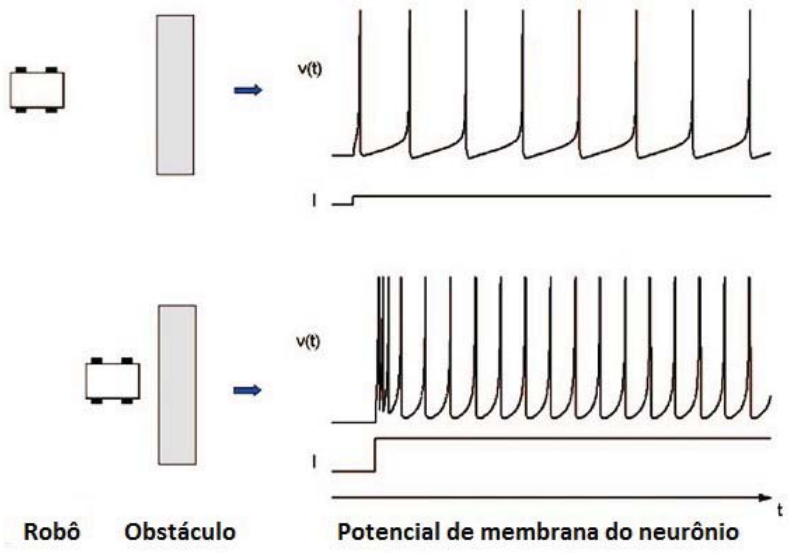

Figura 5.1: Neurônios excitáveis do tipo *Class I* codificam a distância do robô a um obstáculo por meio de uma sequência de *spikes* [Adaptado de (ARENA et al., 2009)].

Além disso, tais neurônios serão modelados como sendo neurônios do tipo *Class I*, presentes na região do hipocampo e responsáveis por tarefas de locomoção e localização espacial (DAYAN; ABBOTT, 2001). A característica principal destes neurônios é o fato de apresentarem uma resposta característica por meio de *spikes* que é proporcional à amplitude do sinal de entrada (valor da corrente elétrica recebida). Esta propriedade é importante pois funciona como um mecanismo de codificação, codificando qualquer medida em uma frequência característica de *spikes* emitidos (fig.5.1). Os parâmetros do modelo de Izhikevich para representação do comportamento de um neurônio do tipo *Class I* são: $a = 0.02$, $b = -0.1$, $c = -55$ e $d = 6$ (IZHIKEVICH, 2004), enquanto a variável de entrada *I* recebe tanto os valores dos estímulos externos (estímulos sensoriais) como os valores das entradas sinápticas.

Com respeito à entrada sináptica, consideremos um neurônio j que recebe conexões sinápticas de n neurônios, e que t_i indique o instante que neurônio genérico i, conectado ao neurônio j, emite um *spike* (fig.5.2). A entrada sináptica do neurônio j é dada

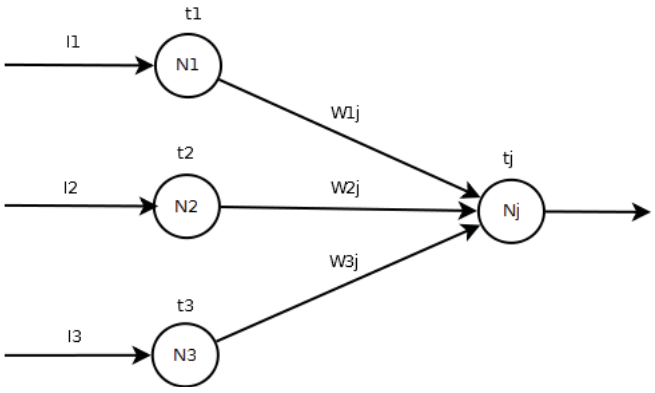

Figura 5.2: Entrada sináptica de um neurônio N_j.

então pela equação:

$$I_j(t) = \sum_{i=1}^{n} w_{ij}\ \epsilon(t - t_i) \tag{5.1}$$

onde t corresponde ao tempo atual, w_{ij} representa o peso da sinapse que conecta o neurônio i ao neurônio j, e a função $\epsilon(t)$ é definida por

$$\epsilon(t) = \begin{cases} \frac{t}{\tau}\exp(1 - \frac{t}{\tau}) & if\ t \geq 0 \\ 0, & if\ t < 0. \end{cases} \tag{5.2}$$

A equação 5.2 descreve a contribuição de um *spike*, emitido de um neurônio pré-sináptico no instante $t = 0$. Os parâmetros utilizados durante as simulações e experimentações podem ser encontrados em (ARENA et al., 2009).

5.1.2 Topologia

Na SNN empregada no problema, existem duas categorias de neurônios: neurônios sensitivos (*sensory neurons*), e neurônios motores (*motor neurons*). Os neurônios sensitivos são os neurônios conectados aos sensores do robô e responsáveis por computar os estímulos percebidos pelo robô. Os valores lidos pelos sensores são injetados diretamente na variável de corrente elétrica do neurônio (variável *I* da equação 3.30). Por outro lado, os neurônios motores são os neurônios responsáveis por controlar os motores do agente. Com base nos dinamismos e nos estímulos de saída dos neurônios motores (cadeias de *spikes* emitidos), são geradas as ações de controle que movimentam o agente.

O modelo de rede adotado contém duas camadas de neurônios de *spikes*:

- Camada sensitiva: composta por neurônios sensitivos, computa os estímulos percebidos pelos sensores do agente e propaga a informação para a camada seguinte através de emissões sucessivas de *spikes*. Esta camada é composta por 6 neurônios sensitivos: 4 deles recebem como estímulo os valores de leitura de dois sensores de toque, posicionados à direta e à esquerda do robô. Os outros 2 neurônios recebem como estímulo os valores de leitura de dois sensores ultra-sônicos, espaçados de maneira similar a dos sensores de contato;

- Camada motora: composta por neurônios motores, recebem os *spikes* da camada sensitiva e os transformam em uma ação motora, que controla diretamente a movimentação do agente. Esta camada é composta por 4 neurônios. Estes neurônios geram cadeias de *spikes* que serão interpretadas como uma ação de movimento (movimento frontal, ou mudança de direção).

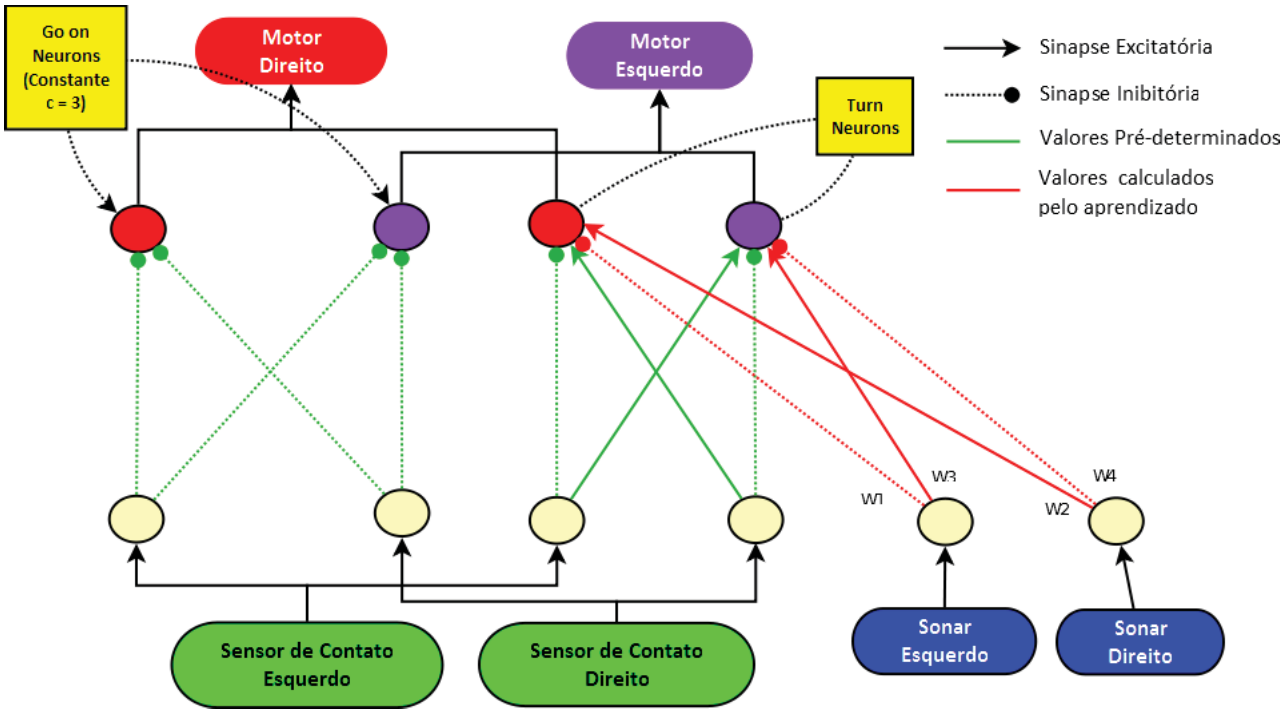

Figura 5.3: Diagrama da SNN.

A estrutura das conexões entre neurônios é inspirada no trabalho clássico sobre veículos de Braitenberg (BRAITENBERG, 1984) e no trabalho de Arena *et al.* (ARENA et al., 2009). Tais modelos são simples veículos reativos dotados com um mecanismo de controle bioinspirado baseado no acoplamento direto entre sensores e motores. O modelo utilizado e retratado na fig.5.3 possui sinapses inibitórias diretas e sinapses excitatórias cruzadas. Este tipo de modelo é biologicamente relevante pois tem sido utilizado para modelar o sistema

nervoso de grilos na tarefa de reconhecimento de fontes de som (*cricket phonotaxis*) (WEBB; SCUTT, 2000).

No modelo abordado existem sinapses excitatórias e sinapses inibitórias. O efeito de uma sinapse excitatória é despolarizar o potencial pós-sináptico. Já o efeito de uma sinapse inibitória é o inverso, ou seja, o de hiperpolarizar este potencial. De acordo com (CARVALHO, 1989) as sinapses excitatórias fornecem um sentido de cooperação entre os neurônios aferentes (sensitivos) e eferentes (motores), enquanto as sinapse inibitórias sugerem uma idéia de competição entre as mesmas células Além disso, tais sinapses podem ter pesos: pré-determinados (identificados pela cor verde na fig.5.3) e àquelas atualizadas pelo algoritmo de aprendizado STDP (identificadas pela cor vermelha).

Estes pesos pré-determinados não são alterados pelo algoritmo de aprendizado, e correspondem ao comportamento básico do robô para o tratamento de uma colisão (ARENA et al., 2009). Sendo assim, com tal estrutura definida, se o sensor de contato direito é acionado, o robô é capaz de efetuar um giro para à esquerda (uma ação de *turn* para a esquerda), e vice-versa. Quando não há estímulo externo por parte dos sensores de contato, existe uma contante (como pode ser visto na fig.5.3) que é inserida na variável de corrente dos neurônios motores e que garante a atividade neural responsável pela movimentação do robô.

5.1.3 Aprendizado da Rede - STDP

O postulado de Hebb(1949) diz que uma sinapse é fortalecida caso uma célula pré-sináptica seja ativada e se o potencial pós-sináptico excitatório resultante (EPSP) contribua para a emissão de um potencial de ativação de uma célula pós-sináptica. Esta idéia de plasticidade sináptica foi reforçada por estudos que constataram que muitas sinapses sofrem potencializações de longa duração (*long-term potentiation - LTP*), e que este processo requer tanto atividades pré-sinápticas como a ocorrência de fortes despolarizações pós-sinápticas (LISMAN; SPRUSTON, 2005).

Recentes estudos experimentais e teóricos indicam a existência de um mecanismo de processamento de informação baseado no tempo preciso da emissão de *spikes* pelos neurônios do cérebro. A modificação persistente da eficácia sináptica em função dos tempos relativos de *spikes* pré e pós-sinápticos é um fenômeno conhecido como 'Plasticidade Sináptica Dependente do Tempo de *Spike*' (*Spike timing-dependent plasticity - STDP*) (FLORIAN, 2007). Por meio deste mecanismo estímulos externos podem ser codificados através do tempo relativo entre *spikes* de neurônios pré e pós-sinápticos (HOSAKA; ARAKI; IKEGUCHI, 2008). O uso da regra de aprendizado STDP abre novos caminhos para a compreensão da codificação da

informação no cérebro. Este mecanismo de aprendizado sináptico é biologicamente plausível, e tem sido observado em sistemas neurais biológicos, principalmente em neurônios piramidais da região CA1 do hipocampo (DAN; POO, 2004).

Como já descrito, a regra STDP depende estritamente deste tempo relativo entre os potenciais de ação pré e pós-sinápticos. Esta forma de modificação sináptica pode balancear automaticamente as conexões sinápticas para fazer com que neurônios pós-sinápticos disparem de maneira irregular, porém mais sensíveis ao tempo dos *spikes* pré-sinápticos (SONG; MILLER; ABBOTT, 2000).

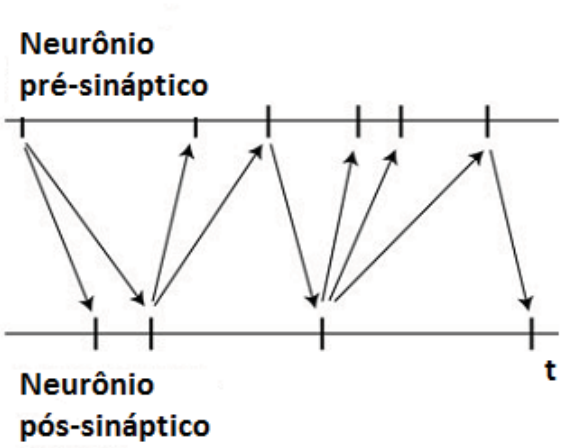

Figura 5.4: Apenas *spikes* conectados por setas contribuem para a plasticidade sináptica [Adaptado de (IZHIKEVICH; GALLY; EDELMAN, 2004)].

A rede adotada para controle do agente autônomo utilizará modificações em sua estrutura sináptica de acordo com a regra sináptica STDP. Seja w uma variável que representa um peso sináptico. *Spikes* pré e pós-sinápticos modificam o peso w ($w \rightarrow w+\Delta w$), por meio de uma função $F(\Delta t)$. Δt é a diferença entre os tempos dos *spikes* pré e pós-sinápticos:

$$\Delta t = t_{pre} - t_{post} \tag{5.3}$$

Além disso, (HOSAKA; ARAKI; IKEGUCHI, 2008) nos diz que o cálculo de Δt, utilizado pela regra STDP, pode ser feito de duas maneiras:

- a primeira delas é considerar todos os pares possíveis de *spikes* pré e pós-sinápticos (SONG; MILLER; ABBOTT, 2000);
- ou então considerar apenas os pares de *spikes* mais próximos uns dos outros. Estratégia adotada neste trabalho, e que pode ser observada na fig.5.4 (IZHIKEVICH; GALLY; EDELMAN, 2004).

Com base em tais valores de Δt, podemos calcular o ajuste sináptico através da regra STDP:

$$\Delta W = \begin{cases} A_+ \exp^{\Delta t/\tau_+} & if\ \Delta t < 0 \\ A_- \exp^{-\Delta t/\tau_-} & if\ \Delta t \geq 0 \end{cases} \tag{5.4}$$

Os parâmetros τ_+ e τ_- determinam a variação do intervalo entre *spikes* no qual as modificações sinápticas de fortalecimento e enfraquecimento ocorrem. Já os parâmetros A_+ e A_-, determinam a quantidade máxima da modificação sináptica (Δw) que ocorrerá quando Δt for próximo de zero (BENUSKOVA; ABRAHAM, 2007). Os valores de parâmetros utilizados durante as simulações podem ser encontrados em (ARENA et al., 2009).

Se o neurônio pós-sináptico emitir um *spike* depois que o neurônio pré-sináptico ($\Delta t < 0$) o peso sináptico será fortalecido. Se a ordem reversa de disparos ocorrer ($\Delta t > 0$), o peso sináptico será decrementado (fig.5.5). As sinapses modificadas pelo STDP competem pelo controle do tempo de emissão dos potenciais pós-sinápticos. Entradas que levam neurônios pós-sinápticos a emitirem *spikes* com uma curta latência, ou que atuam em grupos correlacionados entre si, são capazes de competir com mais sucesso e desenvolver sinapses fortes, enquanto sinapses com longas latências ou entradas menos efetivas são enfraquecidas durante este processo (SONG; MILLER; ABBOTT, 2000).

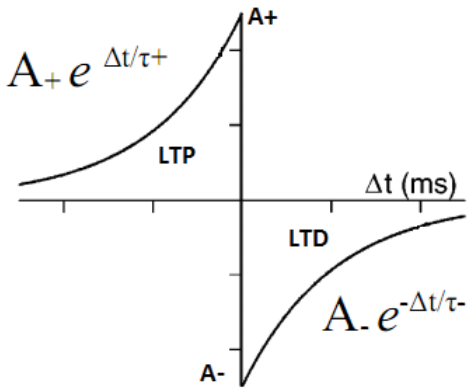

Figura 5.5: A função de modificação sináptica do STDP [Adaptado de (SONG; MILLER; ABBOTT, 2000)].

Utilizar o STDP pode ocasionar um crescimento demasiado dos pesos sinápticos, já que por ser uma técnica de aprendizado não-supervisionado o processo atualiza os valores sinápticos a cada instante de tempo. Uma estratégia utilizada para contornar esse problema é adotar valores limitantes para os pesos sinápticos. Além disso, alguns autores utilizam uma 'taxa de decaimento' na regra de aprendizado. Com base nos trabalhos de (ARENA et al., 2009), utilizou-se uma taxa de decaimento de 5% do valor do peso sináptico que é execu-

tada repetidamente a cada 3000 milissegundos durante a simulação. A regra que descreve a atualização sináptica com a taxa de decaimento pode ser expressa por:

$$w(t+1) = \begin{cases} 0.95\ w(t) + \Delta w, & if\ t\ mod\ 3000 = 0 \\ w(t) + \Delta t & caso\ contrario \end{cases} \tag{5.5}$$

O STDP Hebbiano possui uma sensibilidade particular para casualidade: se um neurônio pré-sináptico contribuir com o disparo de um neurônio pós-sináptico, o mecanismo de plasticidade irá fortalecer a sinapse, então o neurônio pré-sináptico irá tornar-se mais efetivo em causar disparos no neurônio pós-sináptico. Este mecanismo pode ajudar a rede a associar saídas estáveis a uma entrada em particular (FLORIAN, 2007). De acordo com (SONG; MILLER; ABBOTT, 2000) a casualidade é o elemento chave do STDP.

Vários estudos teóricos têm explorado as implicações funcionais do STDP. Esta regra de aprendizado sináptico possui várias características distintas: (a) a bidirecionalidade da modificação sináptica através de LTDs e LTPs balanceadas permite que o circuito neural mantenha a excitação sináptica da rede em um nível estável, evitando excitações demasiadas. A bidirecionalidade do STDP também fornece um mecanismo de competição entre as entradas convergentes da rede; (b) a dependência da sequência dos *spikes* permite ao circuito aprender determinados padrões de sequências e a codificar a casualidade dos eventos externos, torna-se capaz de prever eventos futuros com base nos estímulos adquiridos previamente; (c) a restrita janela temporal permite que o sistema selecione entradas baseado nas suas latências de resposta com precisão de milissegundos, modelando assim o dinamismo temporal do circuito (DAN; POO, 2004).

Embora exerça uma grande contribuição para a competição, em (SONG; MILLER; ABBOTT, 2000) os autores sugerem que o STPD talvez não seja a única fonte de plasticidade em situações onde o aprendizado Hebbiano acontece. Como qualquer outra regra de aprendizado Hebbiano o STDP não pode fortalecer uma sinapse se não houver um disparo em um neurônio pós-sináptico. Se, por alguma razão, as sinapses excitatórias são tão fracas que não levam o neurônio a disparar, o STDP não consegue "resgatar" tais atividades neurais. Uma sugestão dos autores para contornar tais situações seria o uso de um dimensionamento sináptico.

O tamanho da janela temporal onde cada sinapse é fortalecida ou enfraquecida é crítico para determinar os efeitos do STDP. Este tempo pode afetar diretamente as propriedades das respostas temporais de um circuito neural. A plasticidade destes circuitos é essencial para o desenvolvimento e para as funções integrativas do sistema nervoso. Incorpo-

rar estas modificações celulares em modelos neurais realistas será um passo importante para a compreensão da base neural das funções cognitivas de alto nível, tais como a aprendizagem e a memória. (DAN; POO, 2004)

Nas simulações e experimentações das próximas seções o peso sináptico das sinapses ligadas aos estímulos não condicionados (sensores de toque) foram fixadas em -8 (sinapses inibitórias) e $+8$ (sinapses excitatórias). Os valores iniciais das sinapses dos estímulos condicionados (sensores ultra-sônicos) que sofrem influência do STDP são iniciados em 0.05 (sinapses excitatórias) e -0.05 (sinapses inibitórias). Além disso, todos os pesos são limitados dentro do intervalo $[-8, 8]$ (ARENA et al., 2009).

5.2 Estrutura de Controle

O robô utilizado nas experimentações foi modelado por meio do *kit* de robótica *LEGO Mindstorms NXT*. Ao longo das experimentações, utilizou-se a dimensão de comprimento do robô como uma unidade de medida, referida por *unidade de robô (robot unit - r.u.)*. O robô é um veículo com três rodas (*tribot*), sendo duas delas dirigidas pelos dois motores - identificados como motores direito e esquerdo. A atividade motora das rodas é controlada diretamente pela saída da rede neural de *spikes*, apresentada nas seções anteriores. Além disso, o robô possui dois sensores de colisão (toque) e dois sensores ultra-sônicos, localizados na região frontal do agente, possuindo uma faixa de percepção de [0,+45°] (para os sensores do lado direito) ou de [-45°,0] (para o sensores do lado esquerdo) (fig.5.6).

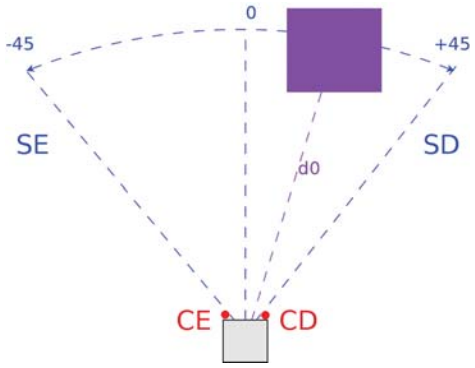

Figura 5.6: Posicionamento dos sensores do agente. Os sonares são representados por quadrados azuis, indicador por SD e SE (sonar direito, sonar esquerdo). Já os sensores de contato estão representados por pequenos círculos vermelhos indicados por CD e CE (contato direito, contato esquerdo). A distância entre o robô e o objeto mais próximo é identificada por d_0 (na cor roxa).

Para especificação das medidas do labirinto, faixas de percepção dos sensores e distâncias de deslocamento do agente serão utilizadas as dimensões do robô como uma

unidade de medida, caracterizada por r.u. (*robot units* - unidade de robô) (ARENA et al., 2009). Seja d_0 a distância entre o robô e o objeto mais próximo, calculada em r.u. Os sensores de colisão serão ativados quando a distância d_0 for igual a $d_0 = 0.6\ r.u.$, nesse caso a entrada associada ao estímulo é definida como um valor constante de $I = 9$. Este valor provoca geração de *spikes* regulares, fazendo com que o robô evite um obstáculo mudando seu movimento para uma direção.

Para os sensores ultra-sônicos utilizados para detecção e prevenção de obstáculos, o estímulo associado à entrada dos neurônios sensitivos é descrito por meio de uma função exponencial:

$$I = 9 \cdot \exp^{-0.6\ d_0} + 2.2. \tag{5.6}$$

Utilizando esta função, os neurônios começam a disparar quando a distância entre o robô e objeto mais próximo é menor que $11\ r.u.$ (ARENA et al., 2009).

A movimentação do robô é feita em um ambiente preenchido por obstáculos espaçados randomicamente. Cada passo da execução da rede simula um tempo de janela de $300\ ms$. O movimento do robô é gerado de acordo com o número de *spikes* gerados pelos neurônios motores durante este intervalo de tempo. O número de *spikes* emitidos pelos dois sensores motores esquerdos (direitos) são somados para gerar a ação de controle do robô. O sinal que controla cada motor depende do número de *spikes* emitidos pelos neurônios motores a eles associados. O algoritmo 5.1 mostra passo-a-passo como é feito o controle de navegação do robô autônomo.

Uma particularidade que pode ser observada na fig.5.3 é que os neurônios motores podem ser categorizados como sendo *go-on neurons* ou *turn neurons*. Os neurônios identificados por *go-on* geram o comportamento necessário para permitir que o robô avançe frontalmente, mesmo com a ausência de estímulos externos. Os *spikes* dos demais neurônios motores (*turn*) são somados aos neurônios *go-on*, de maneira que o sinal de controle dos motores do agente seja composto pela soma destas duas cadeias de *spikes*. Na ocorrência de uma colisão, a estrutura da rede inibe a atividade dos neurônios *go-on*, suprimindo movimentos frontais. Quando ambos os motores emitirem o mesmo número de *spikes*, o movimento do agente será frontal. Na ausência de estímulos condicionados (rotações) a amplitude do movimento é de aproximadamente $0.3\ r.u.$ para cada passo de simulação.

Quando a informação coletada pelos sensores produz um número de *spikes* diferente nos motores direito e esquerdo, o robô sofre uma rotação. Seja então ΔN_R e ΔN_L o número de *spikes* presentes no sinal de controle do motor direito e esquerdo, respectivamente,

Algoritmo 5.1 Navegação de robô autônomo por meio de SNN

Inicializa-se os parâmetros da SNN
for $step = 1$ até 25.000 **do**
  for $timeWindow = 1$ até 300 **do**
    Recebe os estímulos dos sensores;
    Atribui os valores lidos à entrada sináptica I_{syn} dos neurônios da camada 01;
    Calcula-se V_m dos neurônios da camada 01;
    Calcula-se a entrada sináptica I_{syn} dos neurônios da camada 02;
    Calcula-se V_m dos neurônios da camada 02;
    if Ocorreu um spike na camada 01 ou na camada 02 **then**
      Calcula o valor de Δt para cada sinapse;
      Calcula-se o Δw correspondente a cada Δt;
      $w(t+1) \leftarrow \Delta W + w(t)$; {*Atualiza os pesos sinápticos*}
    end if
  end for
  Calcula-se ΔN_r; {*número de spikes dos neurônios motores direito*}
  Calcula-se ΔN_l; {*número de spikes dos neurônios motores esquerdo*}
  Calcula-se $\Delta N_s \leftarrow \Delta N_r - \Delta N_l$;
  Gera a ação de controle;
  if $\Delta N_s == 0$ **then**
    Ação $\leftarrow$ MoveForward; {*movimento frontal*}
  else
    $\Theta \leftarrow 0.14\ \Delta N_s$;
    Ação $\leftarrow$ Turn(Θ); {*mudança de direção*}
  end if
  Enviar ação para robô executá-la
end for

e seja

$$\Delta N_s = \Delta N_R - \Delta N_L \tag{5.7}$$

a diferença entre estes valores, então o ângulo de rotação (no sentido anti-horário) é dado por

$$\theta = 0.14\ \Delta N_s rad. \tag{5.8}$$

Isto significa, por exemplo, que se $\Delta N_R = 3$ e $\Delta N_L = 2$, a diferença ΔN_s será $\Delta N_2 = 1$ e o robô irá rotacionar com um ângulo de $\theta = 0.14$ rad (aproximadamente 8°). Após a rotação o robô realiza um movimento frontal. Sem ter executado a etapa de aprendizado da rede, o robô é conduzido pelo ambiente apenas com seu comportamento não-condicionado. Neste caso, o robô é controlado apenas pelas informações coletadas pelos sensores de contato sendo capaz de evitar um obstáculo após colidir com o mesmo. No caso de obstáculos frontais, e direção de rotação é randômica.

5.3 Simulação numérica

Entre as etapas de modelagem da rede e prototipação da solução houve uma etapa de simulação numérica. O objetivo desta etapa foi simular todos os algoritmos envolvidos no processo de elaboração da SNN: modelo de neurônio de Izhikevich, regra STDP, taxa de decaimento, etc.; a fim de se compreender o funcionamento do modelo da rede como um todo.

O programa utilizado foi a ferramenta computacional Matlab[1] na versão $R2008b$. Por se tratar de uma simulação numérica, além dos algoritmos envolvidos no processamento da SNN foram criadas funções para:

- simular o deslocamento do agente e a aproximação a possíveis obstáculos;
- simular existência de colisões;
- simular os estímulos externos captados pelos sensores ultra-sônicos.
- plotagem e análise de gráficos com os resultados das simulações.

Durante a execução do programa procurou-se observar a saída de cada neurônio, procurando identificar um comportamento similar ao descrito pela teoria. A figura

[1] `http://www.mathworks.com/`

5.7 mostra o primeiro intervalo de janela da execução da simulação. Na situação inicial todos os neurônios emitem um *spike* inicial sendo a única atividade neural gerada neste intervalo, uma vez que não existem estímulos externos percebidos pelos sensores de toque nem pelos sonares. Consequentemente, a atividade neural dos neurônios motores é gerada apenas pela constante de movimento (já discutidas nas seções anteriores) inserida nos neurônios 1 e 2 da camada motora que garante a emissão de um número fixo de *spikes* para realização de um movimento frontal.

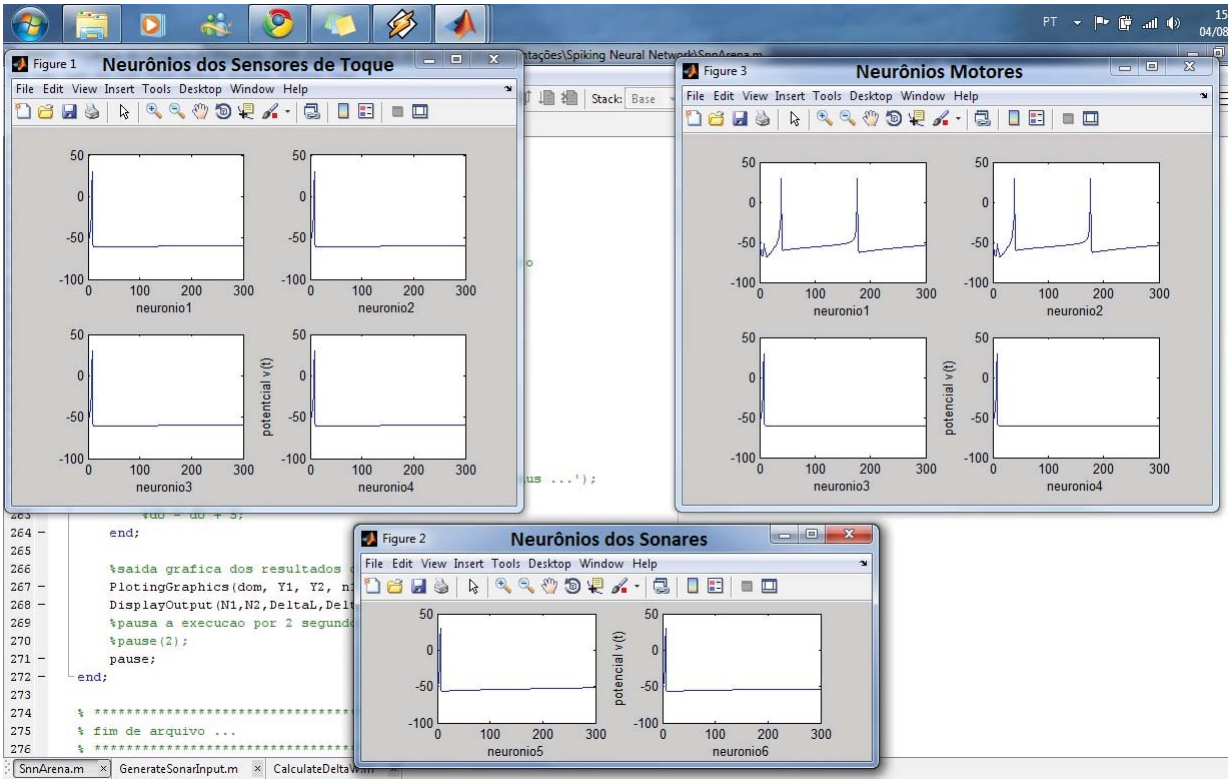

Figura 5.7: Atividade neural dos neurônios da rede em uma etapa inicial da simulação. (Esquerda) Saída dos neurônios sensitivos ligados aos sensores de toque. (Abaixo) Saída dos neurônios sensitivos ligados aos sonares. (Direita) Saída dos neurônios motores.

Uma segunda situação que é interessante ser apresentada é a "eminência de uma colisão", cuja atividade neural pode ser vista na figura 5.8. Na figura, podemos ver que os neurônios ligados aos sensores de toque não realizam nenhum tipo de contato com o objeto resultando na ausência de *spikes* nos neuronios de 1 a 4 da camada sensitiva. Já o neurônio 5, que recebe estímulos externos do sonar esquerdo, emite uma cadeia de *spikes* sinalizando a aproximação de um obstáculo em sua faixa de percepção. Como o neurônio 6, que recebe estímulos do sonar direito, não possui atividade neural, isso sinaliza a eminência de uma colisão a um objeto localizado à esquerda do robô.

A situação seguinte à "eminência de uma colisão" é o "tratamento da colisão"

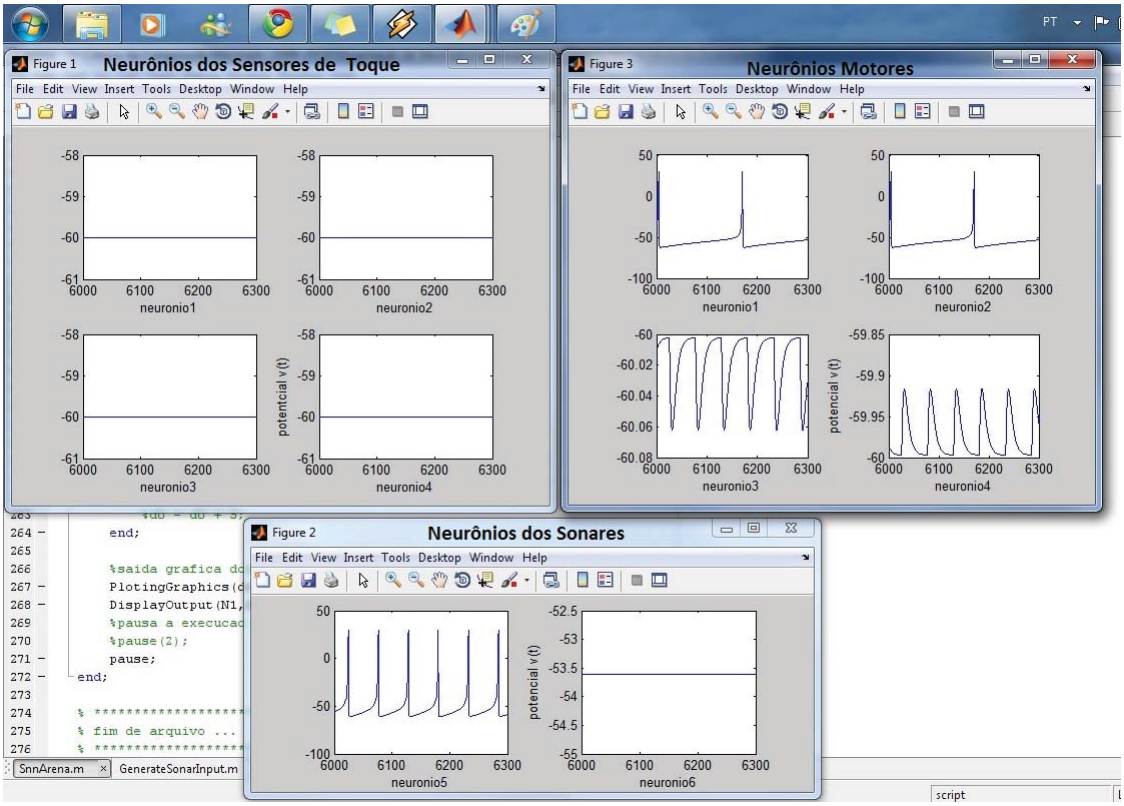

Figura 5.8: Atividade neural dos neurônios da rede em uma situação de colisão eminente. (Esquerda) Saída dos neurônios sensitivos ligados aos sensores de toque. (Abaixo) Saída dos neurônios sensitivos ligados aos sonares. (Direita) Saída dos neurônios motores.

em si. Na figura 5.9 é possível ver a atividade neural quando o robô colide com um obstáculo. Nesta situação o sensor de contato esquerdo é pressionado, fazendo com que os neurônios 1 e 3 da camada sensitiva emitam uma cadeia de *spikes* sinalizando a colisão. Esta atividade na camada sensitiva gera atividade neural na camada motora: os neurônios motores responsáveis pelo motor esquerdo (neurônios motores 2 e 4) emitem um número maior de *spikes* do que os neurônios motores que controlam o motor direito. Como consequência $\Delta S > 0$, e o robô realizará uma rotação de $\theta°$no sentido horário (*colisão na esquerda* $\rightarrow$ *rotação para direita*).

Caso a colisão fosse detectada pelo sensor de toque da direita o comportamento oposto poderia ser verificado (*colisão na direita* $\rightarrow$ *rotação para esquerda*). Se ambos sensores indicassem a existência de um objeto frontal, o robô escolheria uma direção com base nos valores dos pesos sinápticos alterados pelo STDP. Com base em tais análises podemos dizer que uma prevenção de colisão é caracterizada pela emissão de *spikes* dos neurônios motores em consequência de estímulos captados pelos neurônios ligados aos sonares, de maneira que o robô realize uma mudança de direção antes que as situações de "eminência de colisão" ou "tratamento de colisão" aconteçam.

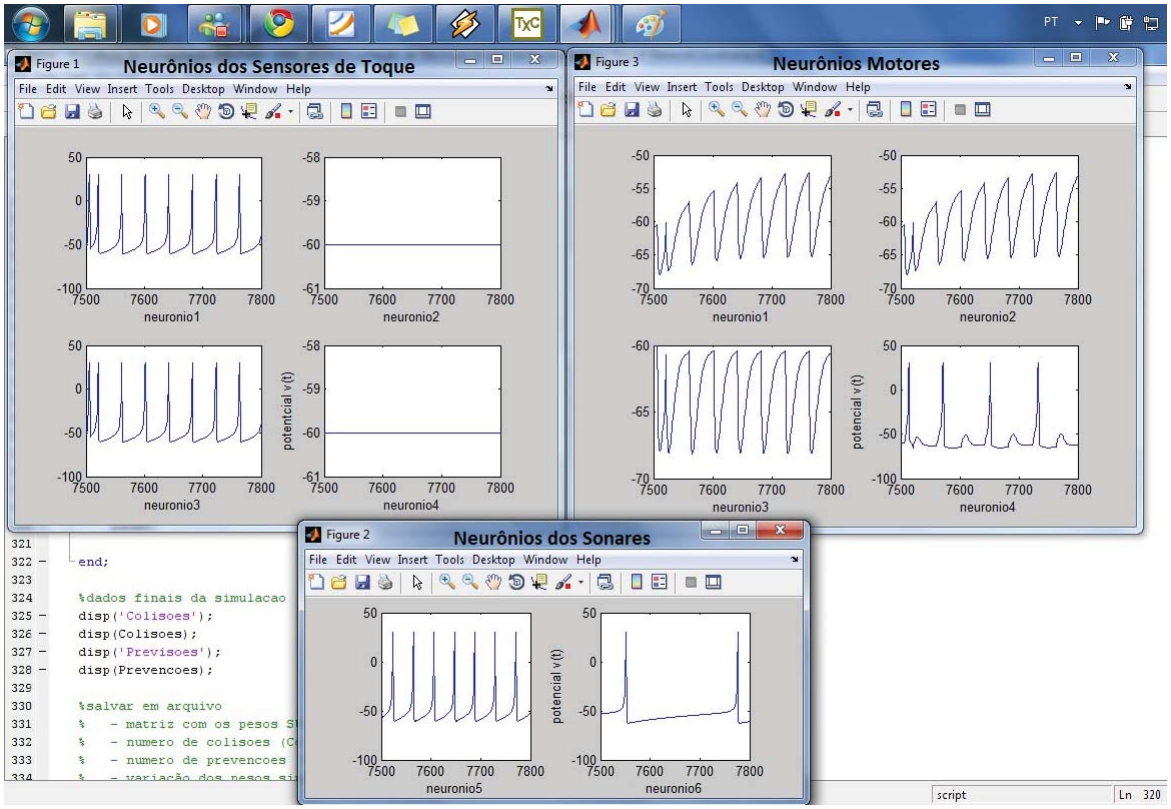

Figura 5.9: Atividade neural dos neurônios da rede durante o tratamento de uma colisão. (Esquerda) Saída dos neurônios sensitivos ligados aos sensores de toque. (Abaixo) Saída dos neurônios sensitivos ligados aos sonares. (Direita) Saída dos neurônios motores.

5.4 Modelagem da Aplicação

Uma etapa importante no desenvolvimento do trabalho é a etapa de prototipação. Nesta etapa o modelo será colocado em execução, parâmetros testados, comportamentos observados para otimizar o desenvolvimento do modelo e corrigir prováveis empecilhos. De acordo com (JOHNS; TAYLOR, 2007) o uso de um simulador durante a etapa de prototipação oferece muitas vantagens, pois:

- prototipar novos robôs no mundo real é uma tarefa que envolve a manipulação de muitas peças (fios, ferro, etc). Modificar ou incrementar a estrutura do robô durante etapas de desenvolvimento pode levar semanas ou meses. Já no simulador é possível realizar tais modificações e aperfeiçoamentos na estrutura do robô em alguns dias;

- além disso o simulador também facilita o processo de desenvolvimento e correção de um *sofwtare* de controle. Quando o robô está executando no ambiente simulado é fácil inserir *breakpoints* nos serviços de controle e encontrar problemas no código-fonte;

- outro ponto positivo é que o simulador também pode ser utilizado em salas de aula,

servindo de mecanismo de ensino para alunos que necessitam utilizar um robô e são limitados pela disponibilidade do *hardware*. Os estudantes podem então utilizar o tempo de trabalho desenvolvendo programas de controle, revisando comportamentos de um robô simulado antes de aplicar a versão final no *hardware* do robô;

- outra vantagem é a segurança física, pois o roboticista não corre nenhum risco ao realizar experimentações no ambiente de simulação tridimensional;

5.4.1 Microsoft Robotics Developer Studio - MRDS

Durante a realização de trabalhos prévios, muitas dificuldades foram encontradas principalmente nas etapas de experimentações. Realizar modificações e analisar mudanças no comportamento do robô consumiam muito tempo, um tempo que poderia ser otimizado utilizando-se um ambiente de simulação. Uma ferramenta que disponibiliza recursos para o desenvolvimento de aplicações bem como uma *engine* para simulações tridimensionais é o *MRDS - Microsoft Robotics Developer Studio*[2], ferramenta utilizada no desenvolvimento do presente trabalho. O MRDS é um *framework* para desenvolvimento de aplicações robóticas. O *kit* de desenvolvimento de *software* (SDK) do MRDS possui vários componentes, sendo os principais: o CCR (*Concurrency and Coordination Runtime*) e o DSS (*Decentralized Software Services*).

O CCR é um modelo de programação para manipulação de concorrência (*multi-threading*), sincronização de tarefas e manipulação de erros, enquanto o DSS é usado para construir aplicações baseadas em modelos de serviços acoplados. Quando se trabalha com robótica muitos eventos ocorrem simultaneamente. Desta forma, qualquer aplicação feita no MRDS irá conter um ou mais serviços, onde cada serviço será responsável por controlar determinado processo, como receber dados de um sensor de toque, controlar um motor, etc. Combinar estes serviços realizando troca de mensagens entre eles é uma das tarefas do DSS. O CCR irá tratar da comunicação assíncrona entre os serviços que estarão executando concorrentemente.

A unidade básica no MRDS é o "serviço", terminologia utilizada para designar um programa. Quem inicializa e finaliza estes serviços é o DSS, que também gerencia o fluxo de mensagens entre tais serviços. Serviços podem ser combinados para criar aplicações utilizando de relações de "parceria". Este processo todo é denominado de "orquestração" (JOHNS; TAYLOR, 2007). Os serviços que compõem uma aplicação podem ser criados e destruídos

[2] http://www.microsoft.com/robotics/

dinamicamente. Entretanto, na maioria dos casos eles são especificados no início da aplicação através de um 'manifesto'. O manifesto é um arquivo XML que lista todos os serviços necessários e suas relações de parceria. Um serviço pode ter muitas relações de parceira conforme necessário.

Um serviço simulado é um serviço que cria, manipula e lê dados de entidades (objetos) existentes no ambiente de simulação. Uma aplicação que utiliza do ambiente de simulação possui uma parceria com o serviço 'SimulationEngine' que fornece os recursos necessários para a criação de um ambiente virtual de experimentação. Para informações detalhadas sobre o MRDS consulte (JOHNS; TAYLOR, 2007) e (MORGAN, 2008).

5.4.2 Estrutura do protótipo

Seguindo as diretrizes do trabalho de (ARENA et al., 2009) criou-se uma arena de simulação para testes e prototipação. O ambiente simulado é uma arena totalmente cercada com obstáculos dispostos randomicamente. As dimensões do labirinto são de 75 x 75 $r.u.$ (unidades de robô), enquanto cada objeto possui dimensão de 10 x 10 $r.u.$ (fig.5.10). A posição inicial do robô neste espaço também é aleatória (fig.5.11). Assim, o objetivo é projetar um robô que navegue dentro deste espaço evitando possíveis colisões com os obstáculos e paredes. O desenvolvimento do protótipo aconteceu em duas etapas: 1) na primeira parte foi elaborado um protótipo simulado apenas com os recursos do MRDS, um robô autônomo reativo que não utilizava de nenhuma estrutura neural; 2) em um segundo momento a estrutura de controle foi alterada inserindo-se a SNN, passando a controlar os movimentos do robô de acordo com as atividades neurais (*spikes*) resultantes dos dados percebidos pelos sensores do robô.

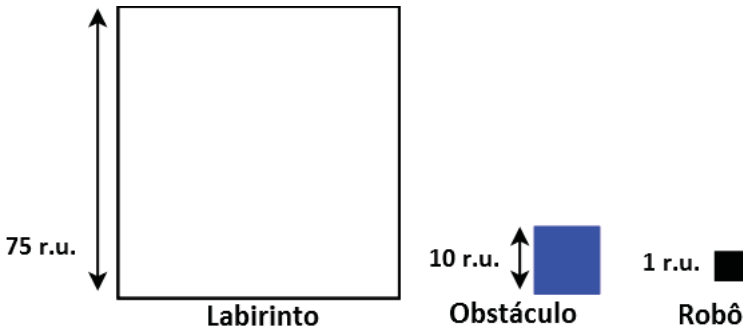

Figura 5.10: Dimensões dos elementos constituintes do ambiente simulado.

Uma aplicação simulada é composta por diversos serviços, alguns fornecidos e implementados pelo SDK do MRDS, e outros criados e especificados pelos usuários. A estrutura da primeira fase de prototipação é mostrada na fig.5.12, os serviços são diferenciados

de acordo com a origem dos mesmos. Serviços fornecidos pelo SDK são identificados pela cor amarela, enquanto serviços criados e definidos são identificados pela cor preta.

Os serviços do SKD utilizados na prototipação foram:

- **SimulationEngine**: serviço com os recursos da *engine* gráfica;
- **SimulationDifferentialDrive**: serviço para controle dos motores do robô, através do envio de ações de movimentação frontal, recuo, rotação, etc;
- **SimulatedSonar**: serviço que realiza o sensoriamento do sonar, retornando a menor distância entre o robô e um obstáculo do ambiente;
- **SimulatedWebCam**: serviço para controle da câmera simulada com a vista frontal do robô, atualiza os quadros de imagem de acordo com a movimentação do robô dentro do ambiente;
- **SimulatedBumperArray**: serviço que realiza o sensoriamento dos sensores de toque, indica a existência de colisões do robô com algum obstáculo do ambiente.

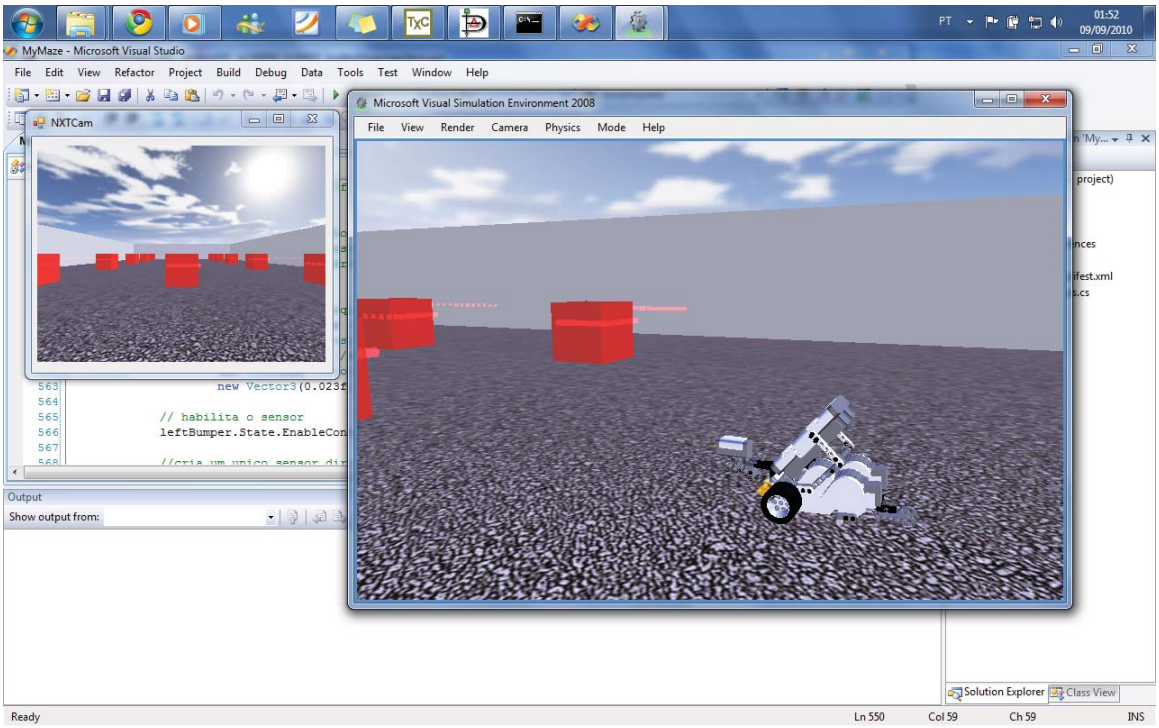

Figura 5.11: Ambiente de simulação virtual. (Esquerda) Visão frontal do robô. (Direita) Agente simulado posicionado no ambiente próximo a obstáculos (cubos vermelhos).

Além destes serviços, houve a necessidade de se criar outros serviços para a elaboração da aplicação:

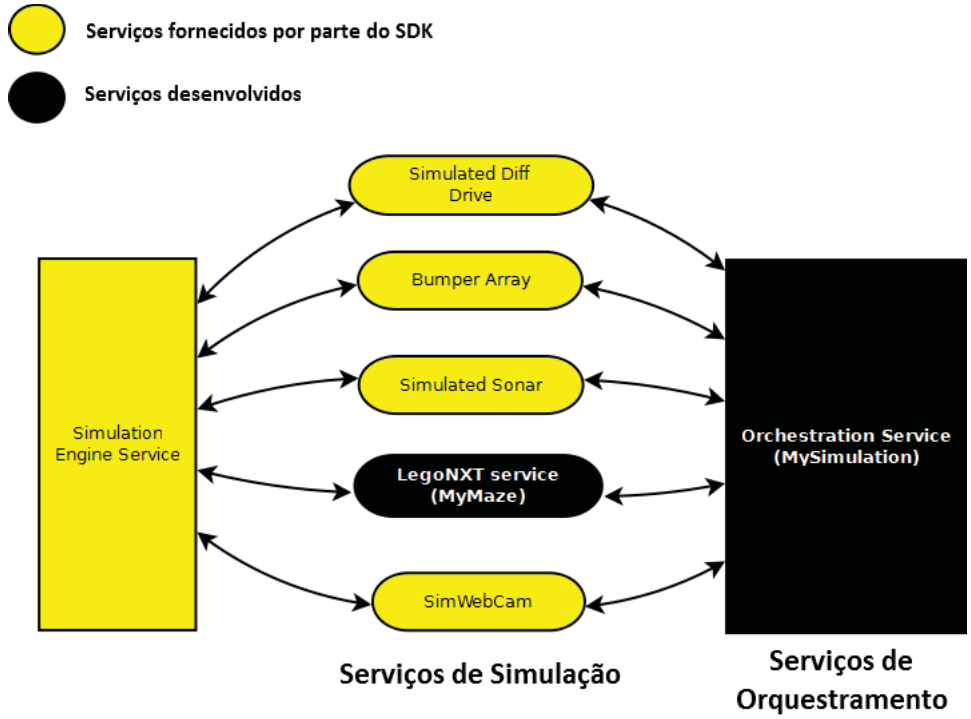

Figura 5.12: Diagrama de serviços envolvidos em protótipo simulado.

- **MyMaze**: é um programa (serviço) que cria um ambiente virtual de simulação estabelecendo uma parceria com a engine 3D do MRDS, através do serviço "SimulationEngine". É no serviço MyMaze que acontece a inserção de todas as entidades que compõem a cena simulada (céu, solo, fontes luminosas, labirinto, obstáculos, robôs, sensores). Além disso, o MyMaze é responsável por iniciar os serviços controladores dos sensores, que irão coletar os estímulos percebidos pelo robô. Este serviço é então responsável por povoar o ambiente, já que o controle das entidades é feito por um outro serviço, descrito a seguir.

- **MySimulation**: é o programa responsável pelo controle da simulação em si. É também um serviço de "controle", pois é em seu código que as parcerias com os demais serviços são estabelecidas. O serviço "MySimulation" recebe as informações decorrentes dos estímulos externos captados pelos sensores e gera uma ação de controle para o robô. Na primeira versão do protótipo a navegação e a velocidade do agente durante a execução são controladas pela menor distância entre o robô e um objeto: quanto maior a distância, maior a velocidade. Em caso de colisões, uma rotação com ângulo randômico é realizada (*randon turn*) e o movimento é reiniciado.

Na segunda parte do processo de prototipação foi integrada a SNN à primeira versão do protótipo (fig.5.13). A SNN é inserida dentro do serviço de orquestração, participando diretamente do controle do robô. Os valores lidos pelos sensores, e passados para

"MySimulation" através de seus respectivos serviços alimentam as entradas da rede. Dentro de "MySimulation" existe um método de controle que a cada 300 ms (tempo de janela) calcula a atividade neural da rede e gera uma ação de controle. Uma vez definida a ação, o serviço de controle envia uma mensagem para o serviço responsável por controlar os motores do robô, fazendo com que o agente se movimente. Este processo é repetido n vezes, onde n é definido antes da execução.

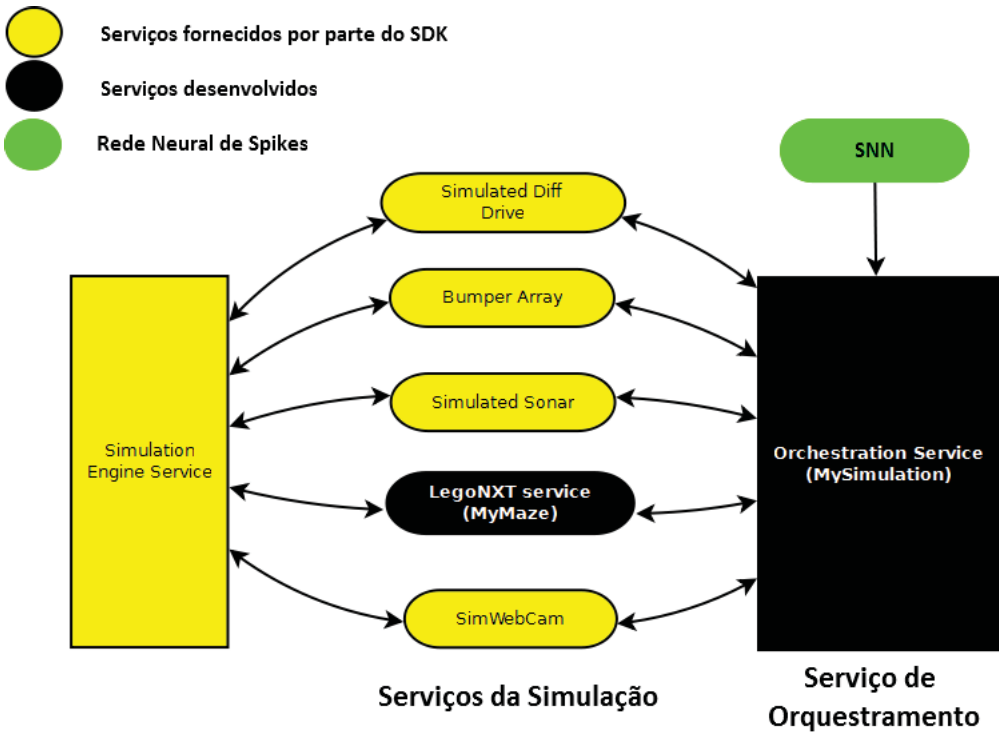

Figura 5.13: Diagrama de serviços envolvidos em protótipo simulado (com SNN integrada).

Embora utilizar um simulador apresente suas vantagens, tal procedimento também possui suas limitações: a) o mundo real é um ambiente complexo e nem sempre se consegue reproduzir situações fisicamente reais no simulador com a mesma escala de tempo com que acontecem naturalmente; b) o mundo real também é um lugar repleto de ruídos, sensores reais frequentemente capturam dados ruidosos ao invés de dados "limpos"; c) existem também uma diferença no poder computacional entre o ambiente de simulação e o robô real (*hardware*). O simulador pode executar em um computador com alta capacidade de processamento enquanto o robô real terá menor capacidade computacional. São por tais motivos que experimentações físicas ainda são necessárias, e esta é a próxima etapa e assunto abordados na próxima seção do trabalho.

5.4.3 Estrutura da experimentação física

Após as simulações com o MRDS, a estrutura do protótipo foi refeita para controle da plataforma de *hardware* empregada na modelagem do robô. Neste trabalho foi utilizado o *kit* de robótica LEGO Mindstorms NXT. Uma vantagem do uso do MRDS é que ele já contém serviços implementados para controle do NXT. As chamadas de métodos de controle do NXT são parecidas às dos métodos necessários para controle de um agente simulado, restando portanto apenas a adaptação do código com as chamadas de funções/métodos necessárias. A estrutura do programa de controle de navegação das experimentações físicas é descrita na fig.5.14.

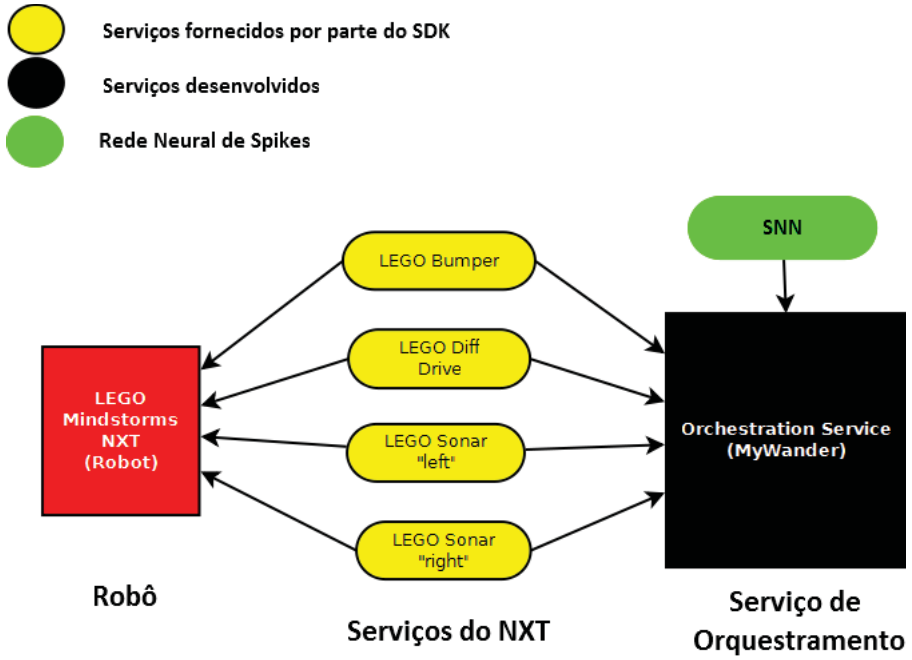

Figura 5.14: Diagrama de serviços envolvidos no programa de navegação da experimentação física (com SNN integrada).

Os serviços fornecidos pelo SDK são identificados pela cor amarela, enquanto serviços criados e definidos são identificados pela cor preta. Os serviços do SKD utilizados durante a experimentação física são:

- **LEGODifferentialDrive**: serviço para controle dos motores do LEGO NXT, através do envio de ações de movimentação frontal, recuo, rotação, etc;
- **LEGOSonar**: serviço que realiza o sensoriamento do sonar, retornando a menor distância entre o NXT e um obstáculo do ambiente. Como foram utilizados dois sonares, existem dois serviços, cada um controlando um dos sonares: sonar direito e sonar esquerdo;

- **LEGOBumperArray**: nas experimentações os sensores de toque são emulados pelos sonares, mas caso fosse necessário a inserção destes sensores, o serviço responsável pelo controle seria o "LEGOBumperArray";

Além destes serviços, há o serviço de controle da aplicação:

- **MyWander**: é o serviço de "controle", pois é em seu código que as parcerias com os demais serviços são estabelecidas. O serviço "MyWander" recebe as informações decorrentes dos estímulos externos captados pelos sensores e gera uma ação de controle para o robô (NXT);

A diferença entre o protótipo de simulação e do programa de controle da experimentação física consiste na modificação das chamadas de métodos e da substituição da *engine* de simulação pelo *hardware* do robô (o NXT), com quem o serviço de controle realiza troca de informações: recebendo a percepção dos sensores e retornando uma ação de controle.

6 RESULTADOS E DISCUSSÕES

Neste capítulo são discutidos alguns dos aspectos referentes às experimentações realizada. Será feita inicialmente uma análise dos dados das experimentações simuladas, seguida da análise dos dados das experimentações físicas.

6.1 Experimentações Simuladas

Para validação do modelo foram executados 15 cenários de navegação distintos, tendo 5 posições iniciais diferentes, verificadas para 3 cenários diferentes. Em tais cenários a posição inicial do agente é definida aleatoriamente. Além disso o número de obstáculos e a posição que ocupam no ambiente também são randômicas. O número de obstáculos varia entre $4 \leq obs \leq 7$. O comprimento de percepção do sonar é de $10\ r.u.$, em unidades de robô. Cada cenário foi simulado por um período de tempo de $T = 25.000$ passos de simulação (fig.6.1) (ARENA et al., 2009). As dimensões utilizadas nos experimentos para criação dos obstáculos, do labirinto, comprimento do movimento do robô, etc, estão descritas na tabela 1.

Tabela 1: Comparativo entre dimensões das simulações

Dimensões	**Arena et al.(2009)**	**Mantovani(2011)**	**Unidade**
Unidades de robô (r.u.)	0,35	0,25	centímetros
Obstáculos	3,5	2,5	metros
Labirinto	26,25	18,75	metros
Movimento Frontal	0,11	0,08	centímetros
Alcance do Sonar	3,5	2,0	metros
Distância de Colisão	0,21	0,15	centímetros
Taxa de decaimento	3000	3000	passos de simulação

A máquina usada para simulação dos cenários foi um *notebook* Dell Vostro 1520 com: processador *Intel Core 2 Duo* de 2.20 GHz, memória RAM de 4 GB, placa de

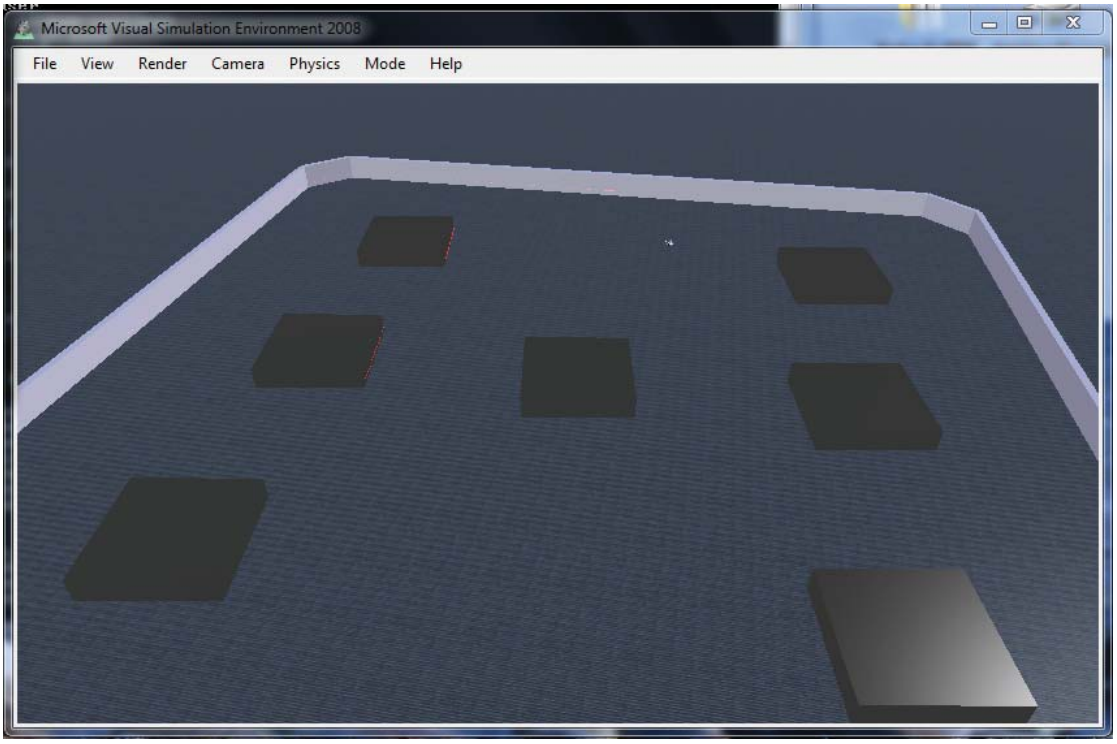

Figura 6.1: Tomada aérea do ambiente de simulação.

vídeo *NVidia GeForce 9300M GS*; rodando um sistema *Windows* 7 de 64 bits. As versões de *software* utilizadas foram: *Microsoft Robotics Developer Studio R2* e *Microsoft Visual Studio 2008*. O protótipo foi escrito em linguagem C#.

Para ilustrar o desempenho da navegação do robô na tarefa de prevenção de colisões foram utilizadas as medidas de desempenho descritas em Arena et al.(2009). Um movimento de mudança de direção (*turn*) é a resposta do robô para um estímulo externo desencadeado por um estímulo condicionado (sonares) ou por um estímulo incondicionado (sensores de contato). Para testar o comportamento do modelo neural analisaremos então a quantidade de *turns* gerados ao longo da simulação. Uma mudança de direção resultante do processamento das informações dos sonares pela SNN caracterizará uma "previsão", enquanto um *turn* resultante das informações dos sensores de toque caracterizará uma "colisão". Se o modelo neural tiver um bom desempenho ao longo da simulação o número de prevenções irá crescer, enquanto o número de colisões diminuirá.

A figura 6.2 mostra o comportamento descrito pelo robô em um dos cenários de simulação observados (Simulação 06 - S6). O diagrama do lado esquerdo mostra a trajetória do robô durante a fase de aprendizagem, fase em que o robô evita obstáculos apenas por meio de colisões. Por outro lado, o diagrama do lado direito mostra que após esta etapa de adaptação dos pesos sinápticos o robô consegue evitar a presença de obstáculos utilizando a

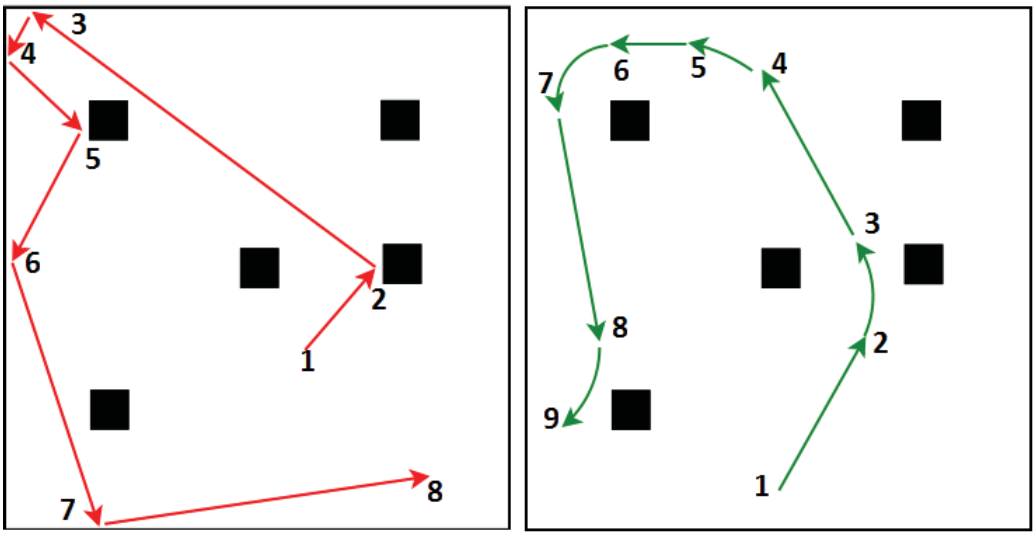

Figura 6.2: Simulação 06.(Esquerda) Antes do aprendizado. (Direita) Depois do aprendizado.

informação dos sensores ultra-sônicos. Os trechos [2,3] e [8,9] da figura à direita mostram o comportamento de prevenção do robô, que ao se aproximar de um obstáculo utiliza dos estímulos captados pelos sonares para identificar a localização do objeto e realizar o movimento de prevenção. Já o trecho entre os pontos [4,7] mostra como esta 'solução' que o robô encontrou também evita colisões com as extremidades do labirinto.

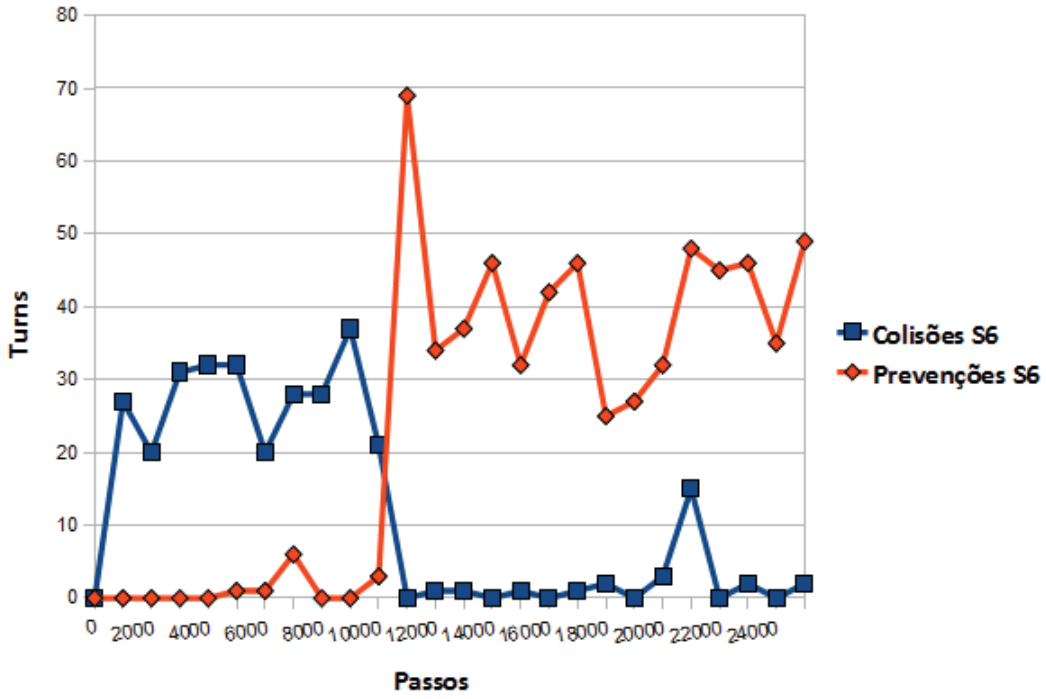

Figura 6.3: Comparação entre colisões e prevenções obtidas na simulação 06 (S6).

O gráfico com as medidas de desempenho do cenário 01 (S6) pode ser visualizado na figura 6.3. Na imagem são comparados o número de prevenções e colisões

executadas pelo agente no cenário simulado. Estes números são calculados a cada 1000 passos de simulação durante a navegação do robô. Podemos notar que no início o número de prevenções é nulo, enquanto o número de colisões existentes é alto. Com o passar da simulação, o número de colisões diminui ao mesmo tempo que o número de prevenções cresce, indicando que o robô aprendeu a tarefa de prevenção utilizando os sonares. Outro ponto a ser destacado no gráfico apresentado é o pico obtido no tempo de simulação $T = 11.000$. Entre os passos [10.000, 11.000] da simulação a SNN começa a utilizar as informações processadas pelos neurônios ligados aos sonares, percebendo a aproximação de obstáculos ao longo do ambiente, resultando em um considerável número de ações de $turn$ com angulações pequenas ($\pm 8°$ou $\pm 16°$). Conforme o aprendizado continua, a saída da rede consegue convergir para outros valores de ângulos ($\pm 32°$, $\pm 40°$, etc.), reduzindo o número de movimentos de mudança de direção dos períodos subsequentes da simulação.

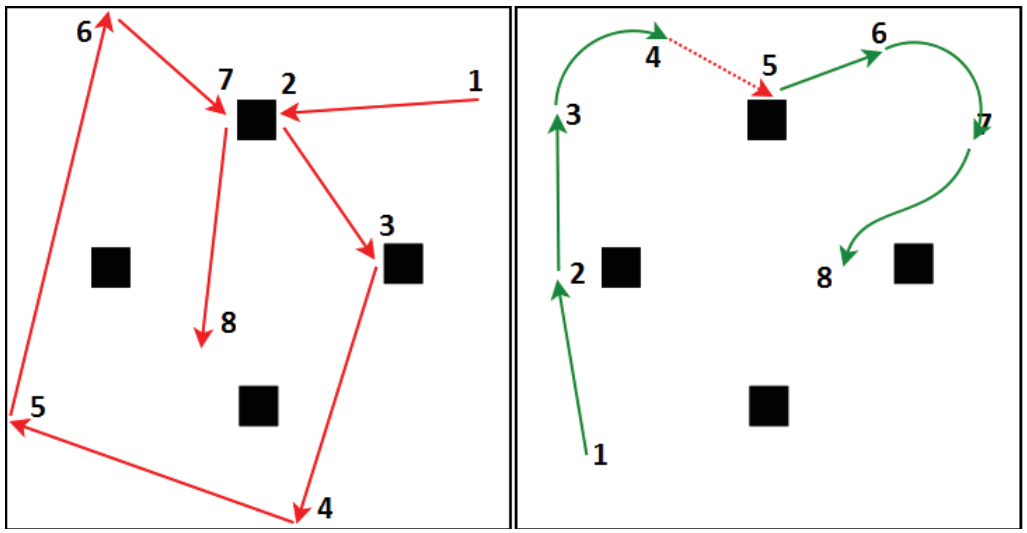

Figura 6.4: Simulação 07.(Esquerda) Antes do aprendizado. (Direita) Depois do aprendizado.

Um segundo cenário observado ao longo das experimentações e que deve ser destacado está retratado na figura 6.4 (Simulação 07 - S7). O que podemos observar é que durante o aprendizado (imagem à esquerda) ocorrem colisões com maior frequência do lado esquerdo do robô do que pelo lado direito, fazendo com que os pesos sinápticos responsáveis pelas prevenções daquele lado convirjam mais rápido que os demais. Do lado direito da fig.6.4 é possível observar que o robô aprende a prevenir possíveis obstáculos posicionados à sua esquerda (ponto 2 e trecho [7,8]), embora alguns obstáculos posicionados à direita, e/ou perceptíveis a ambos os sonares, possam gerar colisões (trecho [4,5]). Estas eventuais colisões ocorridas após o período de convergência dos pesos sinápticos podem ser visualizadas no gráfico da fig.6.5 entre os passos de simulação [18.000 - 23.000].

Uma terceira situação observada foi quando a SNN conseguiu aproximar os

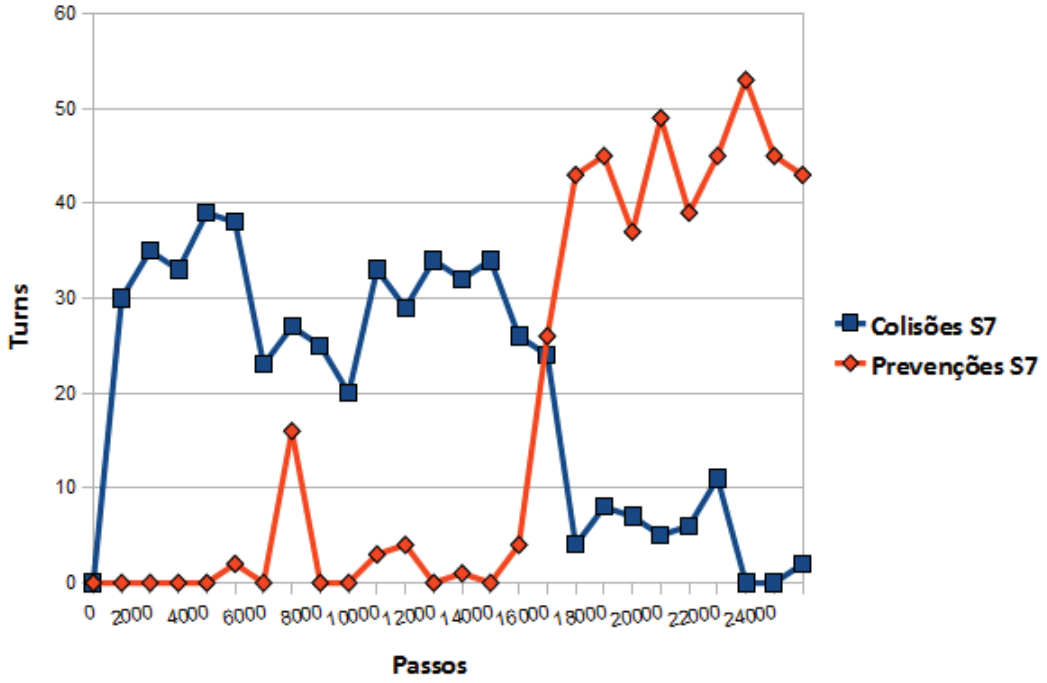

Figura 6.5: Comparação entre colisões e prevenções obtidas na simulação 07 (S7).

pesos sinápticos evitando colisões de ambos os lados do robô. Tal cenário simulado é retratado na fig.6.6 (Simulação 08 - S8). Depois do período de aprendizado o robô consegue evitar obstáculos posicionados em ambos os lados, como retratado na fig.6.6, lado direito. Para reproduzir este comportamento de trajetória o número de *turns* realizados é relativamente grande comparado aos demais cenários simulados, considerando também que este foi um dos poucos cenários gerados com 7 obstáculos, o que implica em um maior período de percepção de eventuais obstáculos. O gráfico "Prevenções x Colisões" da S8 (fig.6.7) retrata bem este número superior de movimentos evasivos realizados pelo robô. A trajetória fica caracterizada por um percurso de ziguezague, porém totalmente livre de colisões, como representado entre os passos de simulação [18.000 - 25.000].

Caso não existam obstáculos, dependendo do lado que sofra um maior número de colisões, a trajetória do robô descreverá percursos circulares pelo ambiente, podendo ser em sentido horário ou anti-horário. A figura 6.8 descreve a trajetória do robô caso ele tenha encontrado um número maior de colisões pela esquerda em sua fase de aprendizagem. Caso o número de colisões pela direita fosse superior, a trajetória teria comportamento similar porém com sentido anto-horário.

Embora a estrutura da rede seja simétrica fica perceptível que os pesos sinápticos não o são. Tal particularidade fornece ao robô estratégias de decisão caso um obstáculo frontal seja percebido. Neste sentido, o comportamento da rede depende da configuração dos

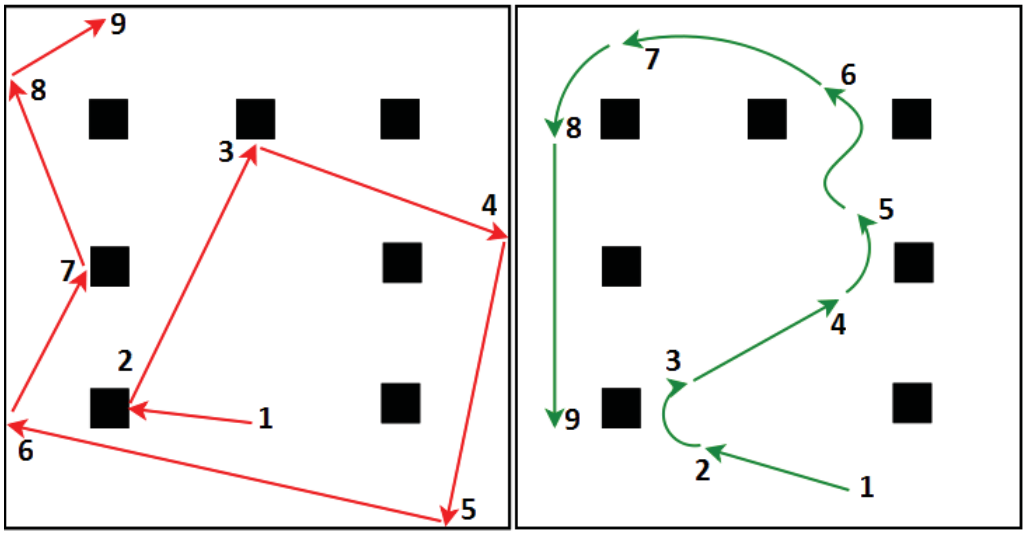

Figura 6.6: Simulação 08.(Esquerda) Antes do aprendizado. (Direita) Depois do aprendizado.

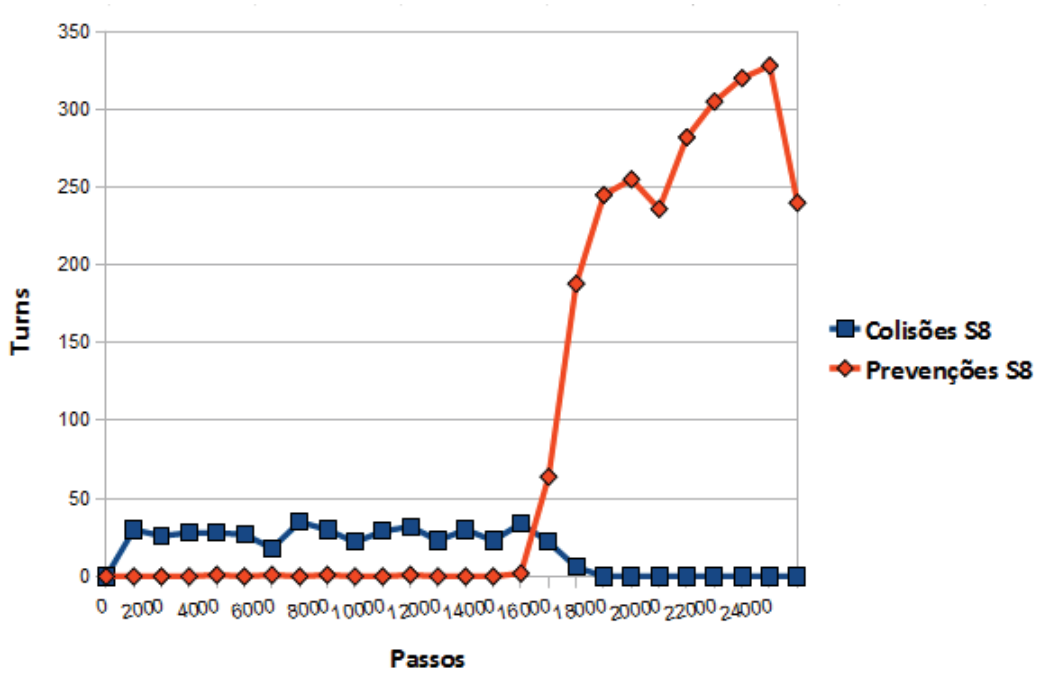

Figura 6.7: Comparação entre colisões e prevenções obtidas na simulação 08 (S8).

obstáculos encontrados usados para treinar o robô, e depende também diretamente do número de colisões à direita ou à esquerda encontrados durante a fase de aprendizado sináptico. Além disso, com base na médias dos valores de todas as simulações verificamos que a SNN começa a aprender a prevenir os obstáculos por volta do tempo de simulação $T = 15.000$ (fig.6.9).

Uma segunda medida de desempenho aplicada nas experimentações é a "distância média evasiva", que é a distância média em que um movimento preventivo é realizado pelo robô. Toda vez que um movimento de rotação for detectado, armazena-se a distância

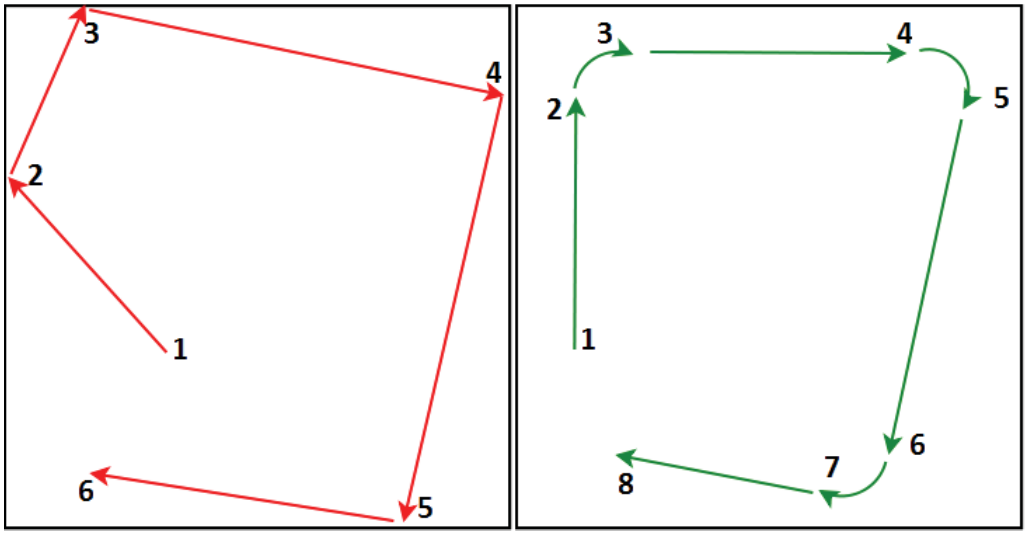

Figura 6.8: Simulação padrão sem obstáculos.(Esquerda) Antes do aprendizado. (Direita) Depois do aprendizado.

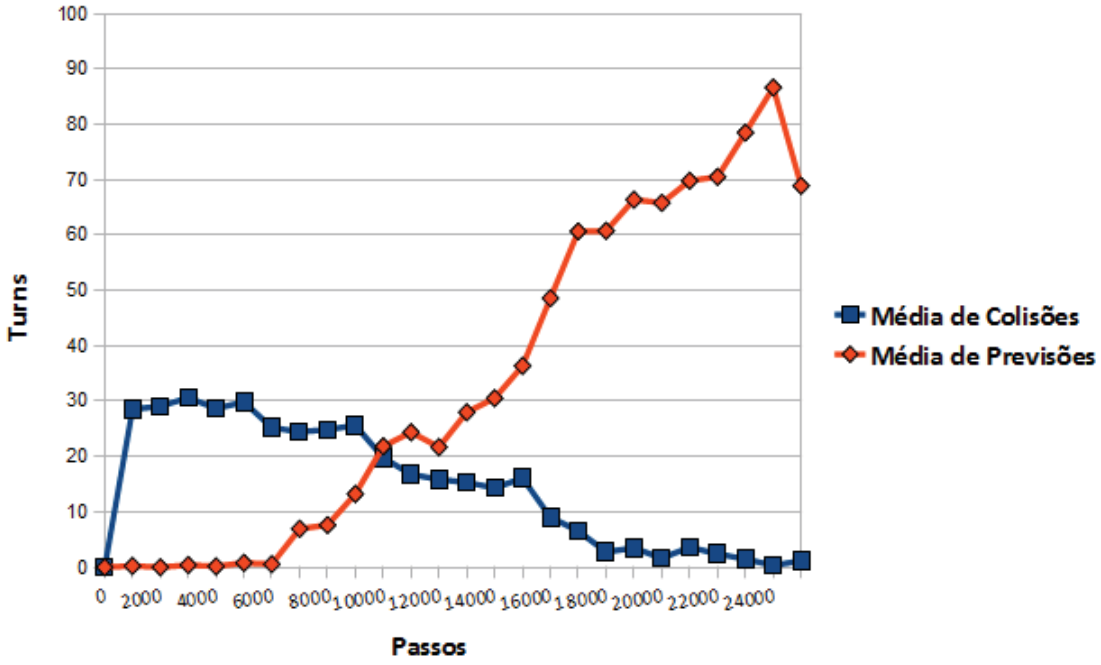

Figura 6.9: Comparação entre colisões e prevenções (Média das simulações).

do robô ao objeto mais próximo, e estima-se o valor médio a cada 1000 passos de simulação. Espera-se que a longo da navegação do robô a distância média cresça e tenha uma convergência próxima ao alcance máximo do sonar.

A fig.6.10 mostra o comportamento observado através da "distância média evasiva" calculada sobre todos os cenários simulados. Pode-se notar que ao iniciar o movimento, por existir um maior número de colisões, a distância dos movimentos evasivos é relativamente pequena, com valores situados entre 0 e $0.4\ m$. A partir do momento que o

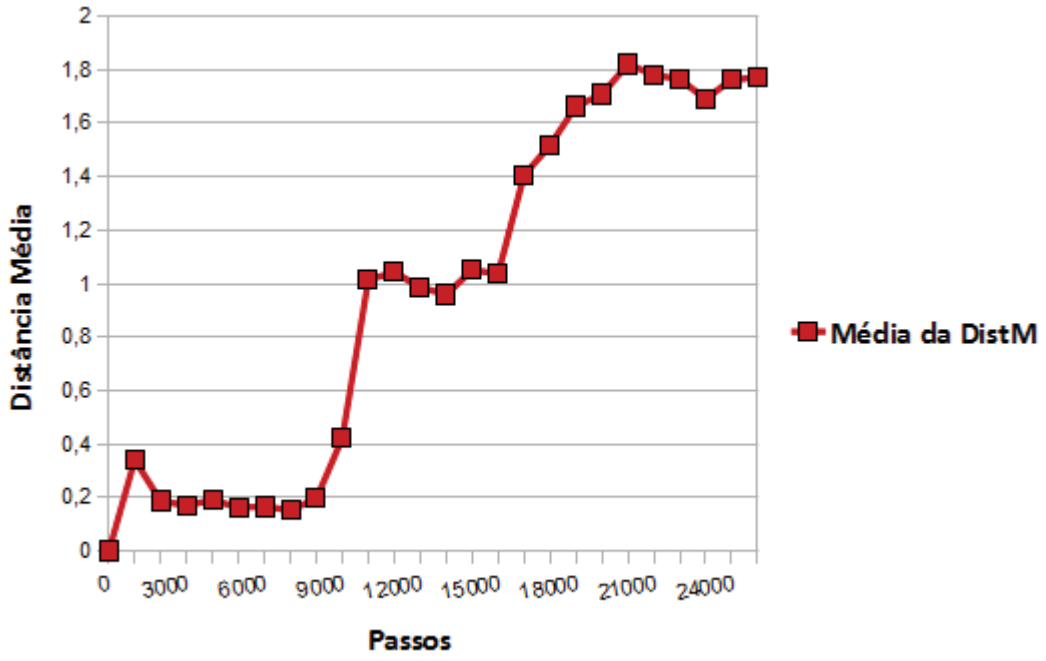

Figura 6.10: Comparação entre colisões e prevenções (Média das simulações).

aprendizado começa a realizar suas primeiras prevenções, o valor de distância evasiva observado cresce para valores entre 0.8 e $1.2\ m$. Por fim, após estabilização dos pesos sinápticos, este valor de distância cresce e se aproxima do comprimento de percepção do sonar ($2.0\ m$), apresentando valores entre 1.6 e $1.9\ m$ e caracterizando assim, uma navegação segura.

6.2 Experimentações Físicas

Subsequentemente ao processo de experimentação simulada procedeu-se a experimentação física do modelo neural. O controle de navegação do agente através de SNN foi realizado em 5 cenários distintos, onde o número de obstáculos varia entre 2 e 4, devido às limitações das dimensões da arena elaborada. Cada experimentação contém um total de 5000 passos simulados, onde cada passo equivale à uma ação tomada pelo robô. O *hardware* utilizado para modelagem do agente foi o modelo NXT do *kit* de robótica *Lego Mindstorms*. O robô é provido de dois sensores ultra-sônicos para percepção de obstáculos, sensores posicionados à direita e à esquerda do mesmo, garantindo uma faixa de percepção de aproximadamente 45°em ambos os lados (como pode ser observado na fig.6.11).

A arena elaborada possui dimensões de $2,0 \times 1,5\ m$ e é circundada por placas de isopor. As dimensões envolvidas na experimentação são descritas na tabela2. O agente inserido em seu ambiente de experimentação pode ser visto na fig.6.12. Ao movimentar-se

Tabela 2: Comparativo entre dimensões das experimentações

Dimensões	**Arena et al.(2009)**	**Mantovani(2011)**	**Unidade**
Unidades de robô (r.u.)	0,35	0,25	centímetros
Obstáculos	0,5	0,5	metros
Labirinto	2,5 x 2,5	2,0 x 1,5	metros
Movimento Frontal	0,11	0,08	centímetros
Alcance do Sonar	0,8	0,7	metros
Distância de Colisão	0,23	0,23	centímetros

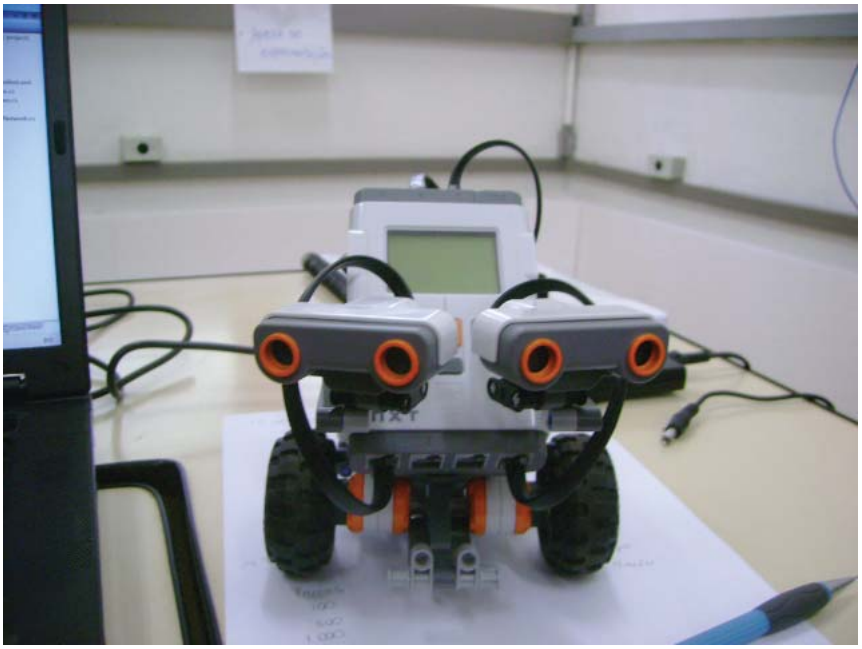

Figura 6.11: Robô utilizado durante as experimentações físicas.

em direção à caixa branca posicionada a sua frente, o robô a identifica como um obstáculo e dispara o tratamento necessário para evitar uma possível colisão.

A implementação do programa foi feita utilizando-se o MRDS e *Microsoft Visual Studio*. O uso MRDS permite reaproveitamento do código já desenvolvido no simulador, necessitando apenas algumas mudanças nas chamadas dos métodos-base do *framework*. Além disso, o programa é executado em uma máquina *host*, que faz requisições e envia as ações para o robô, que as recebe, processa e as executa, retornando sucesso ou falha em sua execução. Esta é uma característica interessante, pois como o programa não roda diretamente no robô, seu desempenho não é afetado por questões referentes ao manuseio de memória interna do *kit* de robótica.

A figura fig.6.13 mostra o comportamento do agente durante sua navegação antes, e depois do aprendizado sináptico. O tempo aproximado de convergência do algoritmo

Figura 6.12: Robô realizando percepção durante experimentação física.

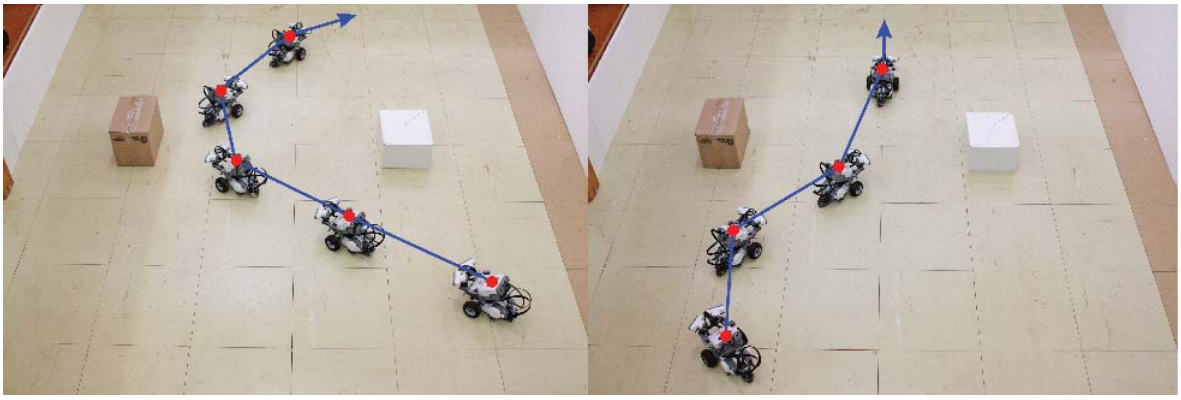

Figura 6.13: Trajetórias descritas pelo robô antes (a esquerda) e depois (a direita) do aprendizado sináptico.

foi de 40 minutos. Na figura fica caracterizada a mudança de trajetória realizada pelo robô como consequência das modificações realizadas nos pesos sinápticos da rede ao longo da movimentação do robô. Um movimento inicialmente caracterizado por mudanças de direção muito próximas aos obstáculos (fig.6.13 - (esquerda)) torna-se um movimento dirigido pelas informações obtidas pelo par de sonares (fig.6.13 (direita)).

O gráfico com o desempenho da navegação do robô ao longo dos 5 cenários experimentados pode ser observado na fig.6.14. Entre os passos [0 - 2000] nota-se um elevado número de colisões e a inexistência de movimentos preventivos. Aproximadamente no intervalo de passos [2500 - 2600] os pesos começam a se ajustar incrementando o número de movimentos evasivos, e reduzindo sutilmente o número de colisões. Já no intervalo [4000 - 5000] pode-se

distinguir uma queda acentuada no número de colisões, em comparação com os intervalos anteriores. Ao mesmo tempo, a distância em que movimentos evasivos ocorrem aumenta, caracterizando movimentos "mais seguros". Conforme o número de prevenções cresce, esta distância é incrementada aproximando-se do valor máximo de alcance do sonar (como mostrado na fig.6.15).

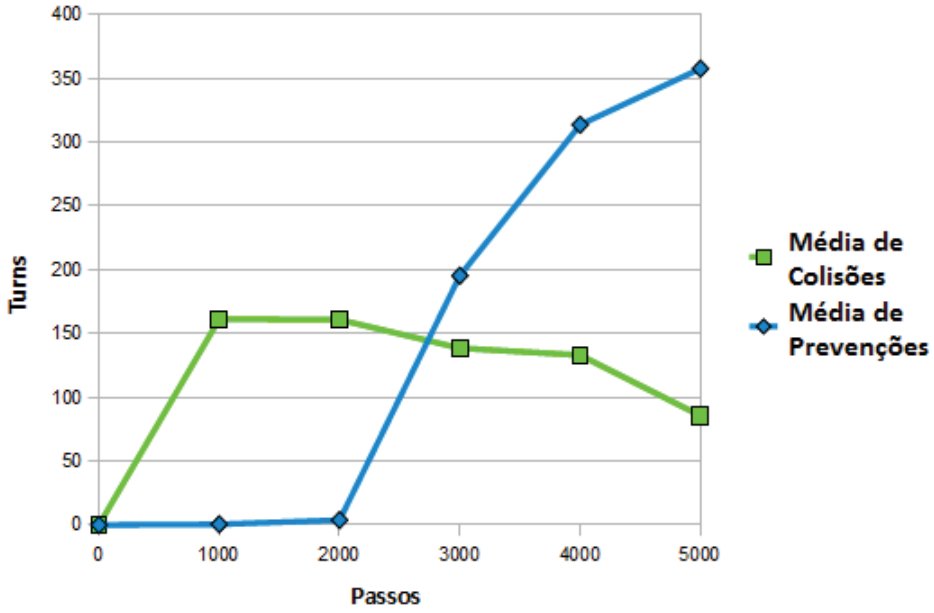

Figura 6.14: Média de Colisões e Prevenções das experimentações realizadas.

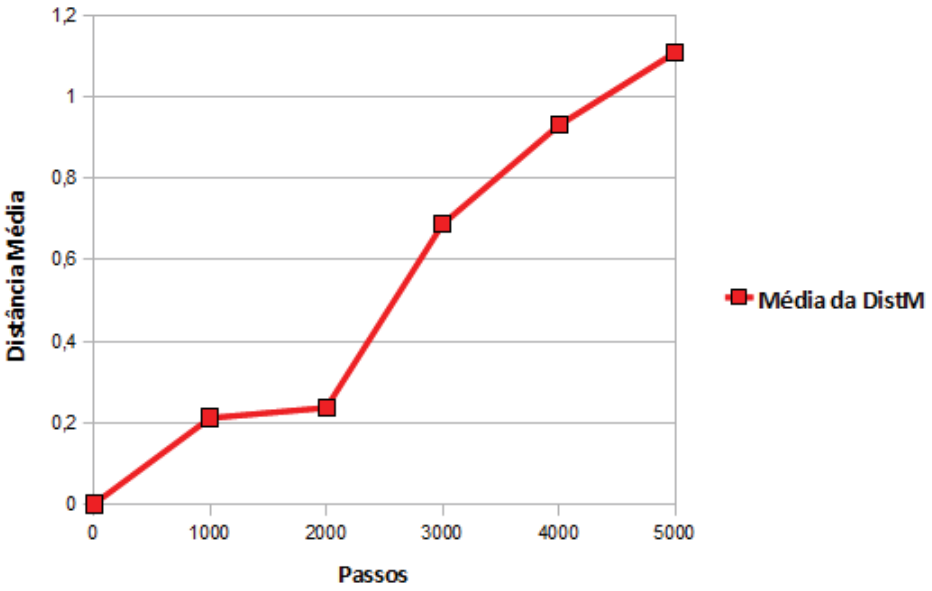

Figura 6.15: Distância média em que os movimentos evasivos são realizados.

Apesar do comportamento observado durante as experimentações físicas ser dentro do esperado, foi possível notar uma diferença no número de colisões verificadas ao final

das experimentações em relação aos dados obtidos ao final das simulações (gráficos 6.9 e 6.14). Nos experimentos simulados o número de colisões decresceu acentuadamente e permaneceu em um valor muito baixo, caracterizando uma situação praticamente "ideal". Por outro lado, nas experimentações, embora diminua, o número de colisões permanece consideravelmente alto, o que sugere a interferência de algum fator (ou fatores) ligado diretamente à experimentação, como:

- questões referentes a bateria do robô, que ao desgastar-se pode interferir no desempenho dos sensores, atuadores, e da comunicação com a máquina *host*;
- problemas de calibração dos sensores;
- atrasos de comunicação via *bluetooth*, que é o mecanismo utilizado pelo MRDS para comunicação com o NXT;
- tempo de varredura dos sensores (sonares), que ocorre em uma determinada janela de tempo (em milissegundos) e de acordo com a ocorrência de um evento. Dependendo da distância entre robô e máquina *host*, e do tempo de janela de leitura pode haver um atraso na execução de uma ação e no retorno de valores pelos sensores, caracterizando ações erradas e comportamentos inesperados;
- interferências do ambiente utilizado para experimentação física, que foi longe do ideal devido a problemas de infra-estrutura no laboratório de robótica;
- material utilizado para elaboração do labirinto, etc.

Embora os dados obtidos pelas experimentações físicas não sejam "ideais", o comportamento se aproxima ao verificado nas simulações, e ambos cenários de testes conseguiram diminuir o número de colisões e aumentar o número de prevenções.

7 CONCLUSÕES

No corrente trabalho foi discutido o uso de Redes Neurais de *Spikes* (SNN) para controle de um agente robótico em uma tarefa de navegação autônoma. Inicialmente discutimos todo um contexto biofísico e bioquímico para a apresentação do formalismo da equação da membrana neural. Junto de tal equação, apresentamos o conceito de potencial de ativação (*spike*), principal meio de comunicação entre neurônios biológicos e mecanismo utilizado para representação de informação a nível neural.

Para modelar este fenômeno existem muitos modelos neurocomputacionais disponíveis na literatura específica, variando quanto à complexidade e custo computacional para implementação. Um dos poucos modelos que consegue balancear ambas características é o modelo proposto por Izhikevich, formalizado por duas equações diferenciais e um *reset* pós-*spike*. A vantagem de tal modelo é que além de possuir uma implementação não-custosa, consegue reproduzir uma vasta gama de comportamentos de células nervosas utilizando apenas um pequeno conjunto de parâmetros.

Subsequentemente descrevemos alguns dos conceitos de navegação e da problemática de prevenção de colisões, seguida de algumas das técnicas tradicionais empregadas para a realização desta tarefa. Com isso, conseguimos estabelecer todo um conhecimento necessário para modelagem de uma SNN empregada em uma tarefa de navegação livre de colisões.

O uso do modelo de Izhikevich (IZHIKEVICH, 2003), a topologia da rede adotada e inspirada nos trabalhos de Braitenberg (BRAITENBERG, 1984) e Arena et al.(ARENA et al., 2009), junto do algoritmo de aprendizado sináptico adotado (STDP) incorporam plausibilidade biológica ao modelo neural desenvolvido ao longo da dissertação, tanto para representação como para manipulação da informação na rede via *spikes*. O uso do *timing* dos *spikes* dos neurônios para codificação da informação pode ser observado em células do hipocampo, responsáveis pelas tarefas de representação espacial e memória de longa duração, como constatado por trabalhos neurocientíficos.

Foi realizada uma etapa de simulação numérica do modelo, objetivando o aprendizado e compreensão dos processos que compõem o modelo como um todo. Por meio desta etapa foi possível ajustar parâmetros do modelo de neurônio, ajustar o funcionamento do algoritmo de aprendizado, além de compreender como informações externas captadas (sentidas) pelos sensores são capazes de modificar a saída da rede, gerando ou não prevenções. Posteriormente, a estrutura elaborada durante a simulação numérica foi transcrita para um *framework* de programação provido de simulações tridimensionais, o MRDS. O uso de um simulador ajuda a corrigir situações não previstas ao longo da modelagem, corrigindo uma falha em um tempo menor do que o necessário para corrigir na aplicação física final. Além disso, programar no MRDS para o ambiente simulado ou para a plataforma de hardware do robô dispende pouco tempo, já que as modificações necessárias no código-fonte são poucas mantendo-se uma base de raciocínio intacta. Outro ponto positivo do MRDS é que a aplicação roda em uma máquina *host* e comunica-se com o robô através de dispositivo *bluetooth*, não carregando assim a memória interna do robô e não limitando a implementação do programa.

Para validar o modelo foram criados quinze cenários de simulação com robô, e quinze cenários de experimentações físicas onde os obstáculos foram posicionados randomicamente dentro de um labirinto completamente cercado. Se a SNN convergisse seus pesos sinápticos, o robô seria capaz de evitar os obstáculos utilizando as informações dos sonares ao invés das informações fornecidas pelos sensores de colisão. E o que pudemos observar ao longo das experimentações foi exatamente tal comportamento. Após um período de aprendizado, o número de colisões detectadas através dos sensores de toque decaiu, enquanto o número de movimentos evasivos gerados pelas informações dos sonares cresceu, caracterizando prevenções de colisões. A distância relativa entre robô e obstáculo ao realizar uma mudança de direção também cresceu. Inicialmente tal valor era relativamente baixo, visto que colisões eram detectadas. Conforme os pesos sinápticos eram ajustados pelo algoritmo STDP, esta distância aumentou consideravelmente a ponto de evitar um obstáculo nas simulações e experimentações, aproximando-se do alcanço máximo dos sonares.

Nas experimentações físicas, embora os dados obtidos não sejam próximos do cenário ideal obtido pelas simulações, os resultados mostraram comportamentos similares e dentro do esperado, apesar dos "ruídos. Tais fatos comprovam que além de aprender a utilizar as informações dos sonares, a SNN consegue utilizá-las com uma precisão satisfatória. Os conjunto de dados obtidos por meio de simulações e de experimentações físicas demonstram que a navegação realizada pelo agente autônomo torna-se "mais segura" ao longo do tempo, já que o número de possíveis colisões decai ao mesmo passo que a SNN ajusta seus pesos sinápticos.

7.1 Trabalhos futuros

Como contribuições futuras, seguem algumas sugestões de modificações e melhorias do trabalho:

- Explorar outros modelos matemáticos de neurônios. Além dos formalismos de Izhikevich (IZHIKEVICH, 2003), alguns modelos clássicos poderiam ser testados, como também o modelo probabilístico de Kasabov (KASABOV, 2010);
- Investigar o uso de outros mecanismos de aprendizado sináptico;
- Inserção de *delay* na emissão dos *spikes* nas sinapses da rede, uma vez que para o modelo adotado a transmissão de um *spike* pré-sináptico até dois neurônios pós-sinápticos distintos leva a mesma quantidade de tempo. Um aspecto a ser explorado e analisado no algoritmo de aprendizado seria a inserção do *delay* de transmissão;
- Durante a atualização dos pesos sinápticos do algoritmo STDP apenas o último *spike* emitido contribui para a eficiência sináptica. Uma possível modificação na implementação deste mecanismo seria considerar um número fixo de *spikes* e observar o comportamento gerado pelo aprendizado da rede, bem como sua convergência;
- Outro aspecto seria o incremento dos estímulos visuais da rede (câmera com visão do agente, dicas visuais) incrementando também a tarefa de navegação para a qual o modelo é designado;
- Reajustar o intervalo de leitura dos sonares, na tentativa de contornar a execução de ações "ruidosas";
- Modificar os parâmetros do aprendizado sináptico para tentar acelerar o processo de ajuste dos pesos sinápticos;
- Remodelar o ambiente substituindo os materiais utilizados na construção, visando facilitar a reflexão das ondas sonoras e melhorar a percepção através dos sonares;
- Otimizar o processo de experimentação física, para aproximá-lo da situação ideal observada nas simulações;
- Por fim seria interessante construir através do MRDS uma aplicação virtual para simulação de diversos tipos de problemas de navegação, criando um *framework* de experimentação 3D. O MRDS permite execução paralela de programas distribuídos dentro de

uma mesma rede, e esta característica de paralelismo poderia ser explorada no intuito de comparar diversos modelos (robôs, arquiteturas) em diversos *hardwares*.

APÊNDICE A – Diagrama do problema

Diagrama com as teorias e ferramentas envolvidas na modelagem e resolução do problema.

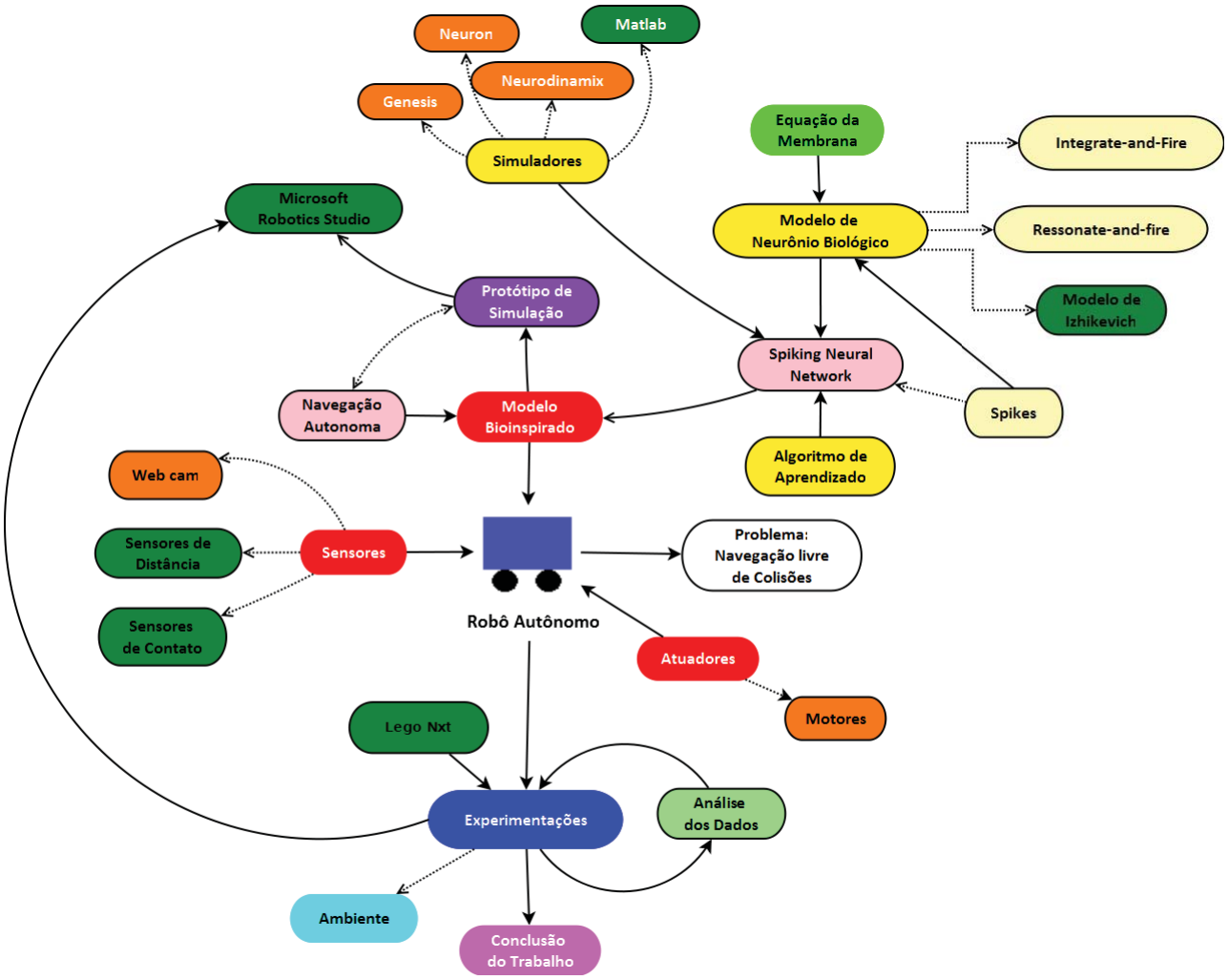

Figura A.1: Diagrama do problema

APÊNDICE B – Artigo desenvolvido em projetos relacionados à dissertação

Segue artigo desenvolvido em projetos paralelos e relacionados à dissertação que foi aceito e publicado no evento *INTERTECH 2010 - XI International Conference on Engineering and Technology Education*, que ocorreu na cidade de Ilhéus - BA, Brasil, de 07/03 a 10/03/2010.

UTILIZANDO A ROBÓTICA COMO RECURSO DIDÁTICO NO ENSINO DE ENGENHARIA E COMPUTAÇÃO

Rafael Gomes Mantovani[1], Valter Takahiro Kamiji[2], Pedro Paulo da Silva Ayrosa[3]

Abstract *— The task of mobile robot navigation can be understood as a problem of determining the trajectories with the aim of providing conditions that allow the execution of tasks. One type of problem that has attracted attention from scientific community is the autonomous navigation: the problems where the agent is who determines its trajectory in an unknown environment. The interdisciplinary, from areas such as computer graphics, artificial intelligence and related ones, assists in learning and development of knowledge in technology. In this paper we describe an experiment involving autonomous navigation of robots, using the model NXT robotics kit from Lego Mindstorms, widely used in educational robotics. Through mechanisms of computer vision, environmental characteristics are transmitted to the robot so that decision can be taken. This paper explores the possibility of using teaching kits for the treatment of problems of engineering and computing that students may encounter in their future professional life.*

Index Terms — Autonomous navigation, Lego Mindstorms, Teaching Engineering and computation.

INTRODUÇÃO

O campo científico de robótica móvel é relativamente jovem. Suas raízes incluem uma variedade de disciplinas, compartilhando conceitos de mecânica, engenharia elétrica e eletrônica, computação, ciências cognitivas e até mesmo sociais. A aplicação de tais modelos é extremamente vasta. Agentes móveis podem ser designados para exploração de ambientes inóspitos, perigosos e de difícil acesso aos seres humanos, como também podem dividir espaço com os seres humanos em ambientes tradicionais ([1]).

O projeto de desenvolvimento de um robô móvel envolve a integração de muitos campos do conhecimento. Para resolver problemas de locomoção, o projetista precisa conhecer conceitos de cinemática, dinâmica e teoria de controle. Para criar sistemas de percepção robustos, ele precisa demonstrar conhecimento de análise de sinais e campos específicos como visão computacional para empregar corretamente uma vastidão de sensores tecnológicos. Já a localização e navegação demandam compreensão de conhecimento de algoritmos computacionais, teoria da informação, inteligência artificial e teoria de probabilidade ([1]).

Um tipo de problema que tem atraído atenção da comunidade cientifica é a navegação autônoma: navegação onde o próprio agente, sem ajuda externa, determina a sua trajetória em um ambiente desconhecido ([2]). Um agente ser autônomo significa que, além de agir por conta própria, ele tem a capacidade de se auto-regular gerando as próprias regras que regem sua atuação. Pode-se dizer também, que um agente autônomo é automático, pois ele tem a capacidade de operar em um ambiente, percebê-lo e impactá-lo visando o cumprimento de tarefas definidas ([3]). Além disso, ele deve se autodirigir, com base em sua capacidade de aprender e adaptar seus comportamentos. Os processos de aprendizagem e adaptação devem ocorrer quando o agente está operando no ambiente.

Agentes autônomos artificiais são ideais para se estudar os princípios de inteligência. Segundo [4], uma das motivações para o uso destes agentes envolve a idéia de emergência. Agentes autônomos apresentam comportamentos emergentes, que surgem pela interação do agente com o ambiente sem que tenham sido programados a *priori* pelo projetista. A pesquisa de sistemas inteligentes autônomos é classificada como uma metodologia sintética, cuja idéia se resume em "construir para entender" ([3]).

O objetivo deste trabalho é realizar um experimento de navegação autônoma de robôs, baseado no trabalho descrito em [5]. O experimento consiste do desenvolvimento de um agente de navegação autônoma utilizando o *kit* de robótica Lego Mindstorms. Através de mecanismos de visão computacional, informações do ambiente são transmitidas ao robô para que o mesmo possa tomar decisões. As informações passadas ao robô correspondem à sua localização, localização do objetivo final e de possíveis obstáculos dispostos aleatoriamente no ambiente. Usando tais informações, o trajeto é calculado por um algoritmo de busca que, utilizando de funções heurísticas, tenta escolher o melhor percurso dentro do ambiente previamente mapeado.

1 Rafael Gomes Mantovani, LABRE - Laboratório de Robótica Educacional, Departamento de Computação - Universidade Estadual de Londrina, Caixa Postal 6041 – CEP 86057-980, Londrina, PR, Brasil, rgmantovani@gmail.com
2 Valter Takahiro Kamiji, LABRE - Laboratório de Robótica Educacional, Departamento de Computação - Universidade Estadual de Londrina, Caixa Postal 6041 – CEP 86057-980, Londrina, PR, Brasil, vtkamiji@gmail.com
3 Pedro Paulo da Silva Ayrosa, LABRE - Laboratório de Robótica Educacional, Departamento de Computação - Universidade Estadual de Londrina, Caixa Postal 6041 – CEP 86057-980, Londrina, PR, Brasil, ayrosa@uel.br

METODOLOGIA

Para execução do experimento desenvolvido, baseado nos trabalhos de [5], dividimos a aplicação em dois módulos principais: o Módulo de Navegação Autônoma (MNA) e o Módulo de Processamento Visual (MPV).

O MPV é responsável por elaborar uma representação do ambiente, através de sensoriamento prévio. Uma câmera localizada acima do ambiente disponibiliza uma visão panorâmica do ambiente por completo. A partir de imagens capturadas pela câmera, o MPV "mapeia" o ambiente no qual o agente desenvolvido terá de se locomover. Este mapeamento fornece a posição do agente, a posição do objetivo final, e os eventuais obstáculos que possam atrapalhar o movimento do agente dentro do ambiente. As posições do ambiente são tratadas como coordenadas do plano cartesiano. Desta forma, o mapa do ambiente é descrito através de uma matriz *M* onde as dimensões de cada célula *M(i,j)* são baseadas nas dimensões físicas do agente ([5]). Após todo esse processo, o fluxo da aplicação passa a ser trabalhado pelo MNA.

O MNA é responsável pelo cálculo da trajetória tomada pelo agente, e consequentemente pela tomada de decisão que ocorre a cada iteração do algoritmo de busca. Este módulo também gerencia todas as funções e estruturas responsáveis pela movimentação do robô dentro do ambiente. O percurso é obtido através de um algoritmo de busca guloso, utilizando como auxílio três diferentes funções heurísticas ([5]-[7]). Ao final da execução, uma seqüência de instruções é elaborada e transmitida para o robô realizar a movimentação.

A implementação dos algoritmos foi feita através da linguagem *Python* ([8] e [9]). Foi utilizada a versão 2.5.4 do *Python*. Nas etapas que envolveram processamento visual utilizou-se o módulo *PIL* para manipulação de imagens ([10]), a versão utilizada é compatível com Python 2.5.4. Já o programa que executa as instruções do robô é descrito em *NXT Python* ([11]). As técnicas e processos utilizados ao longo do trabalho serão descritos a seguir.

Robô Utilizado

Para confecção do agente robótico utilizou-se o *kit* Lego Mindstorms. Inicialmente a idéia era utilizar o modelo RCX, mas após alguns problemas, tanto de sensores como de atuadores, optou-se pela versão mais nova do *kit*: o modelo NXT, lançado pela Lego no ano de 2006. Características adicionais de como manipular estes *kits* podem ser encontrados em [12] e [13]. A figura 1 nos identifica o agente utilizado.

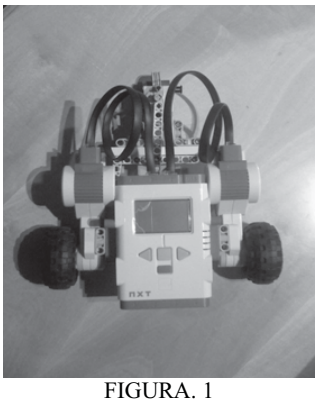

FIGURA. 1
ROBÔ UTILIZADO NO EXPERIMENTO.

Processamento Visual

Na primeira etapa do algoritmo de processamento visual, captura-se uma imagem panorâmica do cenário, constituído por um robô (de cor cinza), o alvo (na cor azul), e alguns obstáculos na cor preta dispostos em um piso de cor laranja. Para a captura da imagem seguiu-se às instruções descritas em [5].

A partir das imagens capturadas, o MPV deve identificar, usando coordenadas cartesianas, a posição do robô e o local do destino. No início do processamento no MPV uma imagem colorida é armazenada, com dimensões de 320x240 ([5]). A partir desta imagem o algoritmo funciona em duas etapas:

- **Localização do Robô e Posição Objetivo:** é realizada uma varredura pela imagem tentando identificar pixels que compõem a região do agente e do objetivo. Estas coordenadas são identificadas e armazenadas para cálculo do centro de massa da região. Com as coordenadas cartesianas identificadas na imagem (através do centro de massa), o procedimento as converte para uma coordenada matricial com o respectivo valor de dígito (discutido nas seções a seguir);
- **Localização de Obstáculos:** a imagem do ambiente é segmentada, aplicando-se processos para a conversão em níveis de cinza, realiza-se uma filtragem gaussiana da imagem seguida de uma etapa de limiarização. O objetivo aqui é identificar as áreas que compõem obstáculos na imagem. Estes processos (filtragem, limiarização, etc.) auxiliam na redução de ruídos existentes na cena capturada ([5]). Abordagens convencionais para segmentação de imagens são normalmente baseadas nas propriedades básicas dos níveis de cinza da imagem, buscando detectar *descontinuidades* ou *similaridades* na imagem ([14]). Já a existência de ruídos pode levar os métodos a distorcer as formas dos objetos, comprometendo seu reconhecimento ([14]). Conseqüentemente, obtém-se

uma imagem auxiliar contendo os obstáculos segmentados. Uma vez identificado se um pixel é pertencente a um obstáculo, ele é expandido recursivamente através da vizinhança-de-4, marcando cada pixel já percorrido com uma determinada tonalidade de cinza, de maneira que o algoritmo não visite dentro das recursões um pixel já percorrido. Além disso, é armazenado o número de pixels componentes de uma região segmentada. Regiões com uma determinada quantidades de pixels são identificadas como obstáculos, enquanto as demais são consideradas como ruído. Ao final da recursão, as dimensões máximas do obstáculo são identificadas e convertidas para a representação matricial (discutida a seguir).

Após esse processamento, uma representação matricial do ambiente é construída.

Ambiente

Após a etapa de processamento visual pelo MPV, o ambiente é convertido em uma representação em malha, com seis linhas e sete colunas (*matriz de ordem 6x7*). Estático, completamente observável e de um único agente, o ambiente é de fácil manipulação e tratamento de eventos ([7]). Dentro do ambiente podem existir obstáculos, representando possíveis "interferências naturais" ao deslocamento do agente. Sendo assim, a configuração inicial do problema é composta por:

- **Estado Inicial:** posição inicial do agente dentro do ambiente;
- **Estado Final:** posição final (ou objetivo) do agente;
- **Obstáculos:** que representam as dificuldades naturais ao movimento do agente.

A figura 2 apresenta um exemplo de configuração de ambiente capturada pelo MPV e sua respectiva representação matricial.

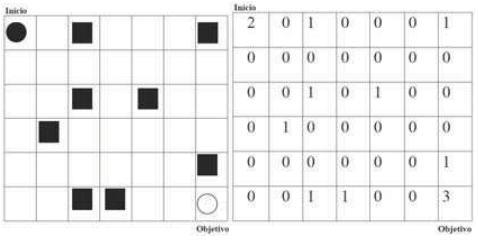

FIGURA. 2
AMBIENTE COM OBSTÁCULOS DISPOSTOS E MATRIZ DIGITALIZADA CORRESPONDENTE.

Cada dígito existente na matriz digitalizada da figura 2 apresenta um significado próprio:

- **Dígito 0:** representa coordenada livre para o ação do agente;
- **Dígito 1:** representa coordenada onde há um obstáculo situado;
- **Dígito 2:** representa a coordenada onde o agente se encontra inicialmente;
- **Dígito 3:** representa a coordenada do objetivo final do agente.

Com base em tais critérios de representação do ambiente, podemos então calcular a trajetória.

Cálculo da Trajetória

Para o cálculo da trajetória do agente foi utilizado um algoritmo de busca baseado em informação. A abordagem utilizada é o que se chama de **busca pela melhor escolha,** que consiste do processo onde um nó é selecionado para expansão com base em uma **função de avaliação** *f(n)*. Desta maneira o nó com a avaliação *mais baixa* é selecionado para expansão, porque a avaliação mede a distância até o objetivo ([7]). A busca gulosa tenta expandir o nó mais próximo à meta, na suposição de que isso provavelmente levará a uma solução rápida. O algoritmo guloso usado, embora não seja ótimo, é adequado para o problema visto que combina vantagens das buscas em amplitude e profundidade e que o custo de cada passo do robô é unitário ([7]).

A movimentação do robô pode ser feita nas direções 0º, 90º, 180º e 270º como pode ser observado na figura 3. Diante da facilidade de representação do ambiente foram utilizadas para *f(n)*, três diferentes medidas de distância conhecidas como: *City-Block* (CB), *Chessboard* (CS) e *Distância Euclidiana* (DE), representadas pelas seguintes equações:

$$\begin{aligned} CB(p,q) &= |x-u| + |y-v| \\ CS(p,q) &= \max(|x-u|,|y-v|) \\ DE(p,q) &= \sqrt{(x-u)^2 + (y-v)^2} \end{aligned} \quad (1)$$

onde *p(x,y)* e *q(u,v)* denotam coordenadas do plano cartesiano, estados que compõem o ambiente ([5]).

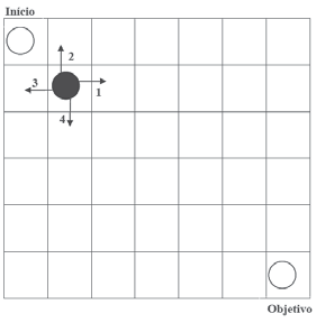

FIGURA. 3
ESQUEMA DE MOVIMENTAÇÃO DO AGENTE

Acionamento do Robô

Após etapa de processamento visual da imagem, e cálculo da rota de acordo com as heurísticas propostas, as coordenadas deste roteamento são transformadas em pseudocódigo e enviadas ao robô por intermédio de um transmissor sem fio. Tais dados são obtidos pelo agente robótico e então o mesmo executa a seqüência de instruções prevista na solução encontrada.

RESULTADOS E DISCUSSÕES

Durante as experimentações efetuamos análises de duas naturezas: 1 – ambientes com soluções, onde existiam rotas até a posição objetivo; e 2 – ambientes sem solução, ou seja, quando qualquer rota até a posição final é obstruída pela existência de obstáculos. Em um primeiro momento fixamos as posições de origem e destino, para analisar as trajetórias obtidas com as diferentes funções já citadas acima. Uma amostra deste comportamento observado está representada graficamente nas figuras 4 e 5.

A figura 4 mostra um ambiente *default*, ou seja, um ambiente que não contém obstáculos, explorado aqui no intuito de captar e perceber o comportamento natural das funções conforme a aproximação ao estado objetivo.

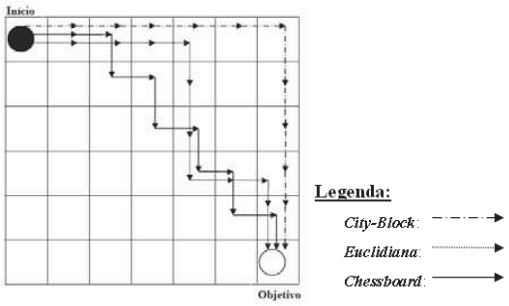

FIGURA. 4
COMPORTAMENTO DA TRAJETÓRIA DO AGENTE EM UM AMBIENTE SEM OBSTÁCULOS

Já na figura 5, podemos visualizar que as funções descritas adotam escolhas distintas de caminho conforme a própria análise do problema local, isto a cada iteração do algoritmo de busca. Às vezes, tais escolhas podem coincidir para certos tipos de funções, em outros casos, não.

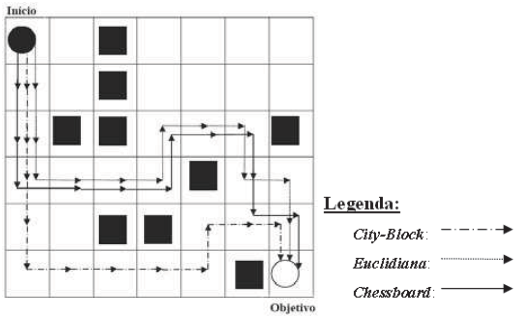

FIGURA. 5
COMPORTAMENTO DA TRAJETÓRIA DO AGENTE EM UM AMBIENTE COM OBSTÁCULOS DISPERSOS

Em um segundo momento, os testes passaram a adotar posições inicial e final variáveis dentro do ambiente, e não mais fixa e pré-determinada como nos exemplos anteriores. Uma análise de tal natureza pode ser vista na figura 6.

Além da comparação visual, podemos explicitar os resultados obtidos do desempenho de cada função por meio de tabelas, denotando o desempenho obtido para um número fixo de testes. Cada linha da tabela I contém o número do teste realizado (um identificador), a quantidade de obstáculos inseridos no ambiente, e o número de posições percorridas para cada função heurística aplicada: *City-Block, Chessboard e Distância Euclidiana*, respectivamente.

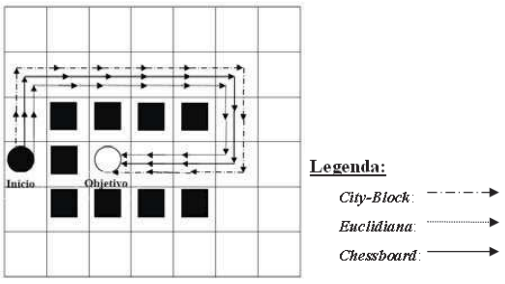

FIGURA. 6
COMPORTAMENTO DA TRAJETÓRIA DO AGENTE EM UM AMBIENTE COM OBSTÁCULOS DISPERSOS E POSIÇÕES DE ORIGEM E DESTINO NÃO-FIXAS

De acordo com tais valores podemos observar que a distância *Chessboard* perde em desempenho quando comparada com as demais, já que percorre um número maior de posições no percurso entre o estado inicial e o estado final. Já as funções *City-Block* e *Distância Euclidiana* apresentaram resultados semelhantes.

TABELA I
COMPARATIVOS ENTRE AS SOLUÇÕES ENCONTRADAS USANDO AS 3 FUNÇÕES DE HEURÍSTICA.

Nº Teste	Nº. Obstáculos	CB	CS	DE
1	0	11	11	11
2	5	11	11	11
3	6	11	11	11
4[1]	3	0	0	0
5	6	11	11	11
6	11	14	11	11
7[1]	6	0	0	0
8[1]	7	0	0	0
9	19	19	19	19
10	9	16	15	15
	Custo Médio	9.3	8.9	8.9

[1] – são ambientes onde o objetivo encontra-se cercado por obstáculos, ou seja, não há uma trajetória qualquer que chegue à posição final desejada.

Outro fator a ser destacado é que embora os custos obtidos sejam quase os mesmos, as trajetórias obtidas, em sua maioria, divergem muito quanto à semelhança de trajeto, diferença explicada pela natureza de cada função. Além disso, durante as experimentações constatou-se que o tempo de execução de uma instrução de mudança de direção para direita/esquerda (*turn left*/*right*) é maior do que o tempo de execução de uma instrução para avançar à frente (*forward*), quando utilizados os *kits* de robótica Lego Mindstorms. Sendo assim, embora os resultados mostrem que a quantidade de posições percorridas para cada função é próxima, o desempenho real é influenciado diretamente pelo tipo de trajetória obtida. Desta premissa, e avaliando o desempenho em relação ao tempo real de execução observamos uma ligeira vantagem para a função de Distância Euclidiana, já que ela proporcionou trajetórias mais "uniformes", sem muitas instruções de mudança de direção, resultando em rotas mais "eficientes".

CONCLUSÃO

O presente trabalho, além de ferramenta para o aprendizado e desenvolvimento de atividades ligadas à tecnologia, serviu como trabalho introdutório a diversas disciplinas da grade curricular do curso de Ciências da Computação. Explorando processamento gráfico, técnicas de inteligência artificial e afins, o experimento foi um fator agregador multidisciplinar. Além disso, foi desenvolvida uma aplicação de navegação autônoma utilizando os *kits* de robótica Lego Mindstorms. O algoritmo de roteamento quando utilizando da função heurística da *Distância Euclidiana* (DE) convergiu para soluções mais satisfatórias do que quando usadas as funções de distância *Chessboard* (CS) e *City-Block* (CB). Tal fato se deve principalmente à natureza do trajeto, que contém comportamentos mais "regulares", evitando mudanças de orientação desnecessárias, e reduzindo eventuais atrasos de execução das instruções. Possíveis trabalhos futuros poderiam ser elaborados visando explorar melhor as características de um ambiente mais complexo. Outro fator a ser pensado é melhorar a parte de segmentação e representação do ambiente, além de otimizar o algoritmo de elaborado, avaliando outras possíveis maneiras de roteamento, visando diminuir problemas em relação às condições do ambiente.

REFERÊNCIAS

[1] Siegwart, R.; Nourbakhsh, I. R. "Introduction to Autonomous Mobile Robots", *MIT Press*, 2004.

[2] Araújo, S. A.; Librantz, A. F. H., "Navegação Autônoma de Robôs", *Exacta,* São Paulo, v. 4, No especial, 2006, PP 81-83.

[3] Cazangi, R. R., "Uma proposta evolutiva para Controle Inteligente em Navegação Autônoma de Robôs", *Dissertação Mestrado*, Universidade Estadual de Campinas – Faculdade de Engenharia Elétrica e Computação, 2004.

[4] Pfeifer, R; Sheier, C., "Understanding Intelligence", *MIT Press*, 1999.

[5] Araújo, S. A; Librantz, A. F. H.; Flório Filho, O., "Navegação Autônoma de Robôs: uma implementação utilizando o kit Lego Mindstorms", In: *Congresso Sul Catarinense de Computação*, Criciúma: SulComp, 2006.

[6] Librantz, A. F. H., Araújo, S. A., Koyama, C., "Sistema de Navegação Autônoma de Robôs: uma proposta de uso como instrumento pedagógico multidisciplinar no curso de Ciência da Computação", *Exacta*, São Paulo, v.4, No especial, PP. 123-125, 2006.

[7] Russel, S; Norvig, P., "Inteligência Artificial", *Campus*, Rio de Janeiro, 2004.

[8] Python. "Python Programming Language: Official Website". Disponível em: http://www.python.org/. Acesso em 25 de março de 2009.

[9] Lutz M., "Pyhton: Guia de Bolso", *Atla Books Ltda*, Rio de Janeiro, 2006.

[10] NXTPython."NXT Python". Disponível em: http://home.comcast.net/~dplau/nxt_python/. Acesso em 04 de abril de 2009.

[11] PIL. "Python Imaging Library". Disponível em: http://www.pythonware.com/products/pil/. Acesso em 25 de março de 2009.

[12] Sartori, B., "Estudo da Linguagem Legolog", *Trabalho de Conclusão de Curso*, Univesidade Estadual de Londrina – Departamento de Computação, 2005.

[13] LEGO. "Lego.com: Mindstorms NXT Home". Disponível em http://mindstorms.lego.com/. Acesso em 04 de abril de 2009.

[14] Pedrini, H.; Schwartz, W. R. "Análise de Imagens Digitais: Princípios e Aplicações", *Thomson Learning,* São Paulo, 2008.

APÊNDICE C – Artigo desenvolvido em projetos relacionados à dissertação

Segue artigo desenvolvido em projetos paralelos e relacionados à dissertação que foi aceito no evento *ticEDUCA 2010 - I Encontro Internacional TIC e Educação*, que ocorreu na cidade de Lisboa, Portugal, de 19/11 a 20/11/2010.

I Encontro Internacional TIC e Educação

Utilização da Robótica Educativa como fator integrador no Curso de Graduação em Ciência da Computação.

PEDRO PAULO DA SILVA AYROSA, JACQUES DUÍLO BRANCHER, RAFAEL ROBSON NEGRÃO, RAFAEL GOMES MANTOVANI, LAIS CONTRIN

Universidade Estadual de Londrina, Brasil

ayrosa@uel.br, jacques@uel.br, rafael@uel.br, rgmantovani@gmail.com, lalacontrin@gmail.com

RESUMO: As duas últimas décadas foram extremamente ricas, quando se fala à respeito dos avanços das Tecnologias de Informação e Comunicação. Na área da educação, estes avanços puderam ser sentidos a partir da utilização de computadores e internet nas salas de aula. Nesta linha, o desenvolvimento de hardwares especializados para a educação, em especial, a robótica educacional estão tornando o aprendizado um processo ainda mais efetivo e por que não dizer prazeroso. Tendo em vista isto, o objetivo do presente trabalho é a apresentação dos resultados parciais da utilização da robótica educacional na grade curricular de um curso de graduação ciência da computação. Os primeiros resultados obtidos, nos permitem afirmar com certeza de que o caminho a ser trilhado é este.

Palavras-chave: Ensino Integrado, Integração Curricular, Robótica Educativa,

ABSTRACT: The last two decades have been extremely rich when talking about the advances of Information Technology and Communication. In education, these advances could be felt from the use of computers and the Internet in classrooms. Along these lines, the development of specialized hardware for education, in particular the educational robotics are making the learning process more effective and why not pleasant. In view of this, the aim of this work is the presentation of partial results from the use of robotics education in the curriculum of an undergraduate computer science. The first results allow us to say with certainty that the way to go is this.

INTRODUÇÃO

Em geral os estudantes encontram dificuldades de adaptação quando saem do mundo acadêmico e entram no mundo profissional. Tal dificuldade é proveniente do fato de que os estudantes não estão habituados a trabalhar em equipe, não estão familiarizados com os problemas encontrados no mundo real e raramente têm a oportunidade de ter um pensamento crítico. Com base nisto, o estudo da robótica pode vir a ser um meio de suprimir as deficiências dos alunos de Ciência da Computação na medida em que ela vai lhes proporcionar a visão de como trabalhar em equipe e também os conhecimentos multidisciplinares (eletrônica, mecânica, etc) que envolvem o projeto, construção e programação de um robô, além de lhes proporcionar uma clara visão dos reais problemas que ocorrem em um projeto fora do mundo acadêmico (ATTROT & AYROSA, 2002). Com base do acima exposto o objetivo do presente trabalho é apresentar o relato de um projeto em andamento que visa usar a robótica como ferramental de motivação e integração às práticas atuais do ensino.

ROBÓTICA EDUCATIVA

A robótica educativa pode ser definida como a montagem de modelos e sistemas robóticos tendo como finalidade o aprendizado de conceitos científicos (física, matemática, eletrônica, mecânica, etc) entre outros por parte de quem realiza a montagem de tais sistemas (KAY, 2003). Os sistemas robóticos são sistemas mecânicos e eletrônicos que realizam alguma espécie de atividade física, tal como se locomover, movimentar um braço ou levantar um objeto. O objetivo da montagem de tais sistemas é a visualização de conceitos físicos e matemáticos de maneira concreta (ANDERSON & KLASSNER, 2001). Os sistemas robóticos são em geral montados com motores, sensores e diversas outras peças. São

controlados por computador ou podem ser programados para que apresentem um determinado comportamento. Existem empresas que fabricam e comercializam *kits* para a montagem de robôs, os quais podem ser utilizados tendo-se em vista propósitos educacionais. A possibilidade de se utilizar a robótica como meio de reforço ao ensino tradicional (no qual os alunos ficam sentados em carteiras prestando atenção ao que o professor escreve na lousa) é estudada desde os anos 50, pois além de ser uma forma mais estimulante de aprendizado, ela faz com que o aluno entre em contato com tecnologias novas e também com novos conhecimentos (MASSONI, 2007).

Segundo Beer, Chiel e Drushel (1999), a transição de um estudante para a vida profissional é em geral difícil. Ainda segundo estes autores, tal dificuldade se deve ao fato de que da grande parte do trabalho de programação que muitos estudantes de ciência e de engenharia da computação realizam no setor industrial, após estarem formados, o software, é apenas um componente de um sistema muito maior. Este software deve ser projetado tendo-se os outros componentes em mente e também através da colaboração e da interação com as outras pessoas responsáveis pelas outras partes do sistema. Geralmente essa dificuldade na transição da condição de estudante para a condição de profissional se deve por quatro razões principais:

- Os estudantes não são treinados para lidar com problemas que necessitem de uma aproximação integrada;
- Eles raramente são expostos a problemas do mundo real;
- Eles raramente são encorajados a resolver problemas em um grupo de trabalho, ou trazer informações de outras áreas de conhecimento;
- Eles raramente têm a oportunidade de realizar um pensamento crítico.

Aproximações integradas são essenciais para a solução de problemas em ciência e engenharia. Uma idéia que muitos estudantes têm é que cada peça de um projeto complexo funciona como um sistema isolado, e o sistema completo irá funcionar como um todo unificado. Na prática isto nem sempre é verdade. Problemas inesperados surgem caso não seja feito um exame das propriedades especiais de cada peça que integra o sistema, e de suas interações. Fora da sala de aula, os problemas da ciência e da engenharia compartilham certas características do mundo real. Ao contrário dos problemas presentes nos livros texto, pode não haver uma única resposta correta. Além disso, ao invés de um professor, é o mundo quem decide se um projeto de engenharia ou hipótese científica está correta ou não. Não se podem ignorar recursos limitados de tempo, dinheiro e materiais. Finalmente, os dispositivos do mundo real nunca seguem os modelos ideais. Um exemplo é um programa que assume que um sensor sempre irá retornar valores exatos e válidos e que motores irão funcionar na velocidade ordenada e que o torque não será afetado por ruídos, entradas espúrias, saídas não confiáveis ou quebras. Os estudantes não têm experiência para estimar a importância destas questões. O progresso nos problemas da ciência e da engenharia é feito através da colaboração de grupos interdisciplinares e não por uma pessoa trabalhando de forma isolada. As práticas educacionais correntes enfatizam a solução individual do problema com base em disciplinas bem definidas. Como conseqüência, os estudantes não têm as características interpessoais necessárias para participarem de um grupo de trabalho, desta forma eles não terão a capacidade de transpor a barreira lingüística que torna a comunicação interdisciplinar tão difícil. Os estudantes frequentemente sentem que a educação é algo que é feito para eles, ao invés de algo que eles fazem por si próprios porque eles não são encorajados a pensar criticamente. Isto não limita somente a maneira como eles aprendem, mas geralmente os afasta de uma preparação para o aprendizado contínuo ao longo da vida, de que eles irão necessitar enquanto profissionais. A Robótica Educativa pode contribuir na discussão e na proposta de uma educação inovadora que faça uso da tecnologia baseada nas quatro metas educacionais: integração, problemas do mundo real, interdisciplinaridade nas equipes, e pensamento crítico. Construir um robô requer que os estudantes integrem controle (software), eletrônica e mecânica em um dispositivo funcional, desta forma, eles irão se confrontar com as interações entre os diferentes subsistemas e terão a oportunidade de explorar as diferenças existentes entre os mesmos

durante a construção de seu robô. Construir um autêntico robô, ao invés de programar uma simulação, imediatamente confronta os estudantes com as imperfeições dos dispositivos do mundo real, bem como proporciona o retorno imediato do sucesso ou falha de suas idéias. Se os estudantes trabalharem em equipe, eles serão encorajados a reunir suas habilidades individuais, permitindo-lhes se especializarem em tarefas específicas, dando-lhes experiência em desenvolver habilidades interpessoais para articular e defender seus pontos de vista, mas principalmente de alcançar um consenso do que é melhor para o grupo. Pelo fato da robótica envolver uma grande e variada gama de áreas de conhecimento, os estudantes aprendem que muitas perspectivas podem ser úteis para solucionar um problema difícil. Eles são motivados a aprender a "língua" de cada uma dessas áreas de conhecimento para quebrar as barreiras da interdisciplinaridade, desta forma os estudantes estão adquirindo conhecimentos de diversas áreas e começarão a ver um sistema como um todo e não mais como subsistemas isolados e reconhecerão que não há uma única resposta correta para tudo, o que encoraja e motiva a sua criatividade.

A PROPOSTA EM ANDAMENTO

O trabalho iniciado em abril de 2009, com duração prevista de 36 meses, está dividido em 5 fases gerais que estão sendo desenvolvidas de forma paralela e integrada com constantes realimentações:

- Fundamentação pedagógica;
- Diagnóstico das práticas atuais no ensino em Ciência da Computação;
- Elaboração de proposta de integração da robótica (através da adequação ou atualização de algumas disciplinas selecionadas;
- Execução de projeto piloto;
- Crítica e disseminação dos resultados.

RESULTADOS PARCIAIS

Como resultado do processo de realimentação e integração de abordagens inovadoras foi introduzida a utilização das ferramentas sociais de aprendizagem (aplicações computacionais que objetivam comunicação, compartilhamento e construção de conhecimento entre professores e alunos através de comunidades de interesse temático). A comunidade Robótica Educativa foi criada no *Learning Space*, espaço destinado a criação de comunidades de aprendizes subordinado ao Projeto *OpenLearning*. Cabe ressaltar que a escolha do *OpenLearning* está justificada pela grande quantidade de pesquisadores de língua portuguesa que desenvolvem pesquisa neste ambiente, além do caráter livre/aberto da plataforma. No momento estão sendo finalizados os tutoriais para o uso das ferramentas como suporte ao desenvolvimento do projeto.

Um experimento piloto de integração da Robótica Educativa com a disciplina Inteligência Artificial foi realizado e está descrito em MANTOVANI, KAMIJI e AYROSA (2010).

Todos os experimentos que estão sendo realizados estão utilizando *kits* NXT da Lego, contudo os problemas expostos em (MASSONI, 2007) e (CHUNG & ANNBERG, 2003) devem ser observados, como:

- Nem sempre a escolha do kit ideal para determinada estratégia pedagógica é uma tarefa fácil de ser feita, dependendo muitas vezes que o professor teste os diversos modelos existentes no mercado, tarefa essa dificultada pelo fato de não serem produtos fáceis de encontrar para realizar apenas testes;
- A necessidade que o professor pode vir a ter de equipamentos adicionais que não acompanharam o kit adquirido quando a compra foi feita pode não ser atendida ou então ser feita com tempo maior do que pode ser aguardado;
- Não se pode deixar de destacar também os preços, em geral muito elevados, que os kits e seus componentes adicionais têm no mercado;
- A arquitetura das salas de aula convencionais não é apropriada para se trabalhar com os kits de maneira confortável, necessitando de espaço e mesas de trabalho maiores para manipulação de peças e melhor visualização das montagens.

CONCLUSÃO

Quando o aluno trava contato com uma metodologia que explora a utilização da robótica no contexto educacional, ele é estimulado a refletir, questionar e arquitetar soluções ou procedimentos em que o conjunto de conhecimento e competências está mobilizado no processo de aprendizado efetivo. Assim, alunos que possam ter acesso a esta oportunidade tecnológica, têm a chance de poder desenvolver modelos que simulem as atividades que por ventura estejam desenvolvendo apenas no campo teórico. Com isso, a Robótica Educativa proporciona uma vivência intuitiva de conceitos de matemática, física e das diversas técnicas computacionais, no caso específico desta proposta, além de permitir que conceitos abstratos da computação sejam exemplificados de forma concreta na montagem e programação de robôs didáticos.

REFERÊNCIAS BIBLIOGRÁFICAS

ANDERSON, S.D.; KLASSNER, F. (2001) ,*Lego mindstorms: not just for k-12 anymore.* ACM Computing Curriculum.

ATTROT, W. ; AYROSA, P. P. S. (2002) *Aplicações da Robótica no Ensino de Ciência da Computação*. In: WEI 2002 - SBC2002, Florianópolis.

BEER, CHIEL, DRUSHEL (1999, jun), *Using Automo-mous Robotics to Teach Science and Enginnering.* Communications of the ACM.

CHUNG, Chan J.; ANNEBERG, Lisa (2003). *Robotics Contests And Computer Science And Engineering Education*.

KAY, J.S.; O´HARA .J. (2003), *Investigating open source software and educational robotics. Tech Report H6*, Computer Science Department, Rowan University.

MANTOVANI, R., KAMIJI, V., AYROSA, P.P.S, (2010) *Utilizando a Robótica com Recurso Didático no Ensino de Engenharia e Computação. XI International Conference on Engineering and Technology Education - INTERTECH'2010*. pág. 234-239.

MASSONI, C. (2007). *Utilização da robótica como ferramenta de motivação e ensino de programação para alunos de curso de graduação ligados a informática*. Monografia (Graduação em Ciência da Computação) – Universidade Estadual de Londrina.

APÊNDICE D – Artigo contendo os resultados da dissertação

Segue artigo gerado como consequência da dissertação de mestrado, e em processo de submissão.

Use of spiking neural networks for navigation tasks in autonomous robotic agents

Rafael G. Mantovani; Pedro Paulo da S. Ayrosa; Ernesto Y. Saito; Estefânia M. Fuzyi
Department of Computing - DC
Educational Robotics Laboratory - LABRE
State University of Londrina - UEL
Londrina-PR, Brazil
{*rgmantovani, eysaito88, faniaem*}*@gmail.com,ayrosa@uel.br*

***Abstract*—Detect and avoid possible collisions is one of the most important aspects in mobile robotics. Although this task seems easy when performed by animals (and humans), presents difficulty when modeled and executed by autonomous robotic agents. With increasing focus on computational neuroscience in recent years, a new range of bioinspired models and algorithms have emerged in the specific literature, such as the Spiking Neural Networks (SNN). These models enhanced the biological realism of their computational units using 'individual spikes', allowing the incorporation of spatial and temporal information in the processes of communication and computing, like real neurons do. The aim of the current work is to use a SNN to prevent collisions in obstacle avoidance task. During the experiments it was used the robotics kit Lego Mindstorms NXT. The prototyping and implementation of the model was done using the robotics framework Microsoft Robotics Developer Studio (MRDS), which besides providing a virtual environment for simulation, provides resources for the programming of variety of robotics hardware, included the NXT.**

***Keywords*-Mobile Robotics, Bioinspired Models, Spiking Neural Networks, Obstacle Avoidance, Lego Mindstorms NXT, Microsoft Robotics Developer Studio.**

I. Introduction

Detect and prevent collisions is an important aspect in mobile robotics. For an autonomous robot that performs a navigation task, detect and avoid obstacles is a key issue for his own security [1]. Although the prediction of collisions is a relatively simple task when performed by living beings, it is little difficult for robots, being a well researched topic in robotics. The question of how biological systems become all sensitive information in motion is a constant search for many neuroscientists and experts in robotics [2]. These experts are always found in nature an inexhaustible source of inspiration.

Izhikevich [3], [4] describes in his works the existence of a growing interest in neural models bioinspired, mainly in relation to Spiking Neural Networks - SNN. These neural models focus on the mathematical formalization of the computational properties of biological neurons. These models capture the nature of neuron spikes and capture the essence of the behavior being modeled artificial [3]. An interesting feature is that analog variables can be encoded by the "time difference" between the spikes emitted by one or more neurons.

An intelligent agent is able to learn and adapt to new environmental conditions. This characteristic is the result of continuous and intense interaction of the brain with the external world, mediated by the body [5]. Bioinspired different mechanisms are currently used in robotics, taking advantage of the characteristics of neural cells such as cells found in the hippocampus and parahippocampal region of the brain to implement location and navigation capabilities in biological agents [6]. Neuroscientific research found that the hippocampus plays a key role in the control of navigation in animals, encoding spatial positions and acting as a mechanism of associative memory [7], [8] and [9]. What the researchers could observe was that communication between neurons could be done using the timing of the spikes emitted by neurons of a synapse, process subsequently formalized and named "Spike-timing-dependent plasticity" or STDP.

Spurred by advances in technology and neuroscience, many works that employ SNN can be found in the literature. In addition to reviews, it has been found many successful applications covering areas such as computer vision [10] [11]; image segmentation [12] [13]; forecasting time series [14]; speech recognition [15] [16]; control robotic arm [17] [18]; identification and lateralization of sound sources [19] to navigation autonomous robots [20], [2] and [6].

The aim of this work is to use a spiking neural network for control and navigation of an autonomous robotic agent in the treatment and prevention of collisions with obstacles in an unknown environment. By using the concepts of spikes in neural structure of the model there is the incorporation of biological plausibility, both for representation and manipulation of information to the network. The environment where the agent is inserted is populated by randomly spaced obstacles, where the presence of the same is done by the robot during the execution of movement. To facilitate the implementation and handling of errors during navigation of the agent was using the environment Microsoft Robotics Developer Studio (MRDS) [21]. Apart from allowing programming and control over the robotics kit used

(Lego Mindstorms NXT), this Microsoft's framework has an engine with physical environment for three-dimensional simulations. Such tools are essential for the realization of the prototype model because it reduces the time required for testing and adjusting the data by simulating various scenarios that would take considerable time to be physically validated.

II. Spiking Neural Model

Among the various neural neural models existing the model that more accurately reproduces the features of a real neuron is the model of Hodgkin-Huxley [22]. However this model has a high computational cost, since it is described by four nonlinear differential equations coupled, which prevents its use in simulations on a network with a large number of neurons [23]. On the other hand, using models of type integrate-and-fire [24] proves computationally effective, but such models are too simple and unable to play certain types of dynamics exhibited by cortical neurons. Contextualized with such reality, Izhikevich [25] developed a spiking neuron model that have merit as present as the biological plausibility of Hodgkin-Huxley model, besides being computationally efficient as integrators models. Through a simple set of parameters the model can reproduce the potential activation of known types of cortical neurons.

The Izhikevich's model consists of a two-dimensional system of ordinary differential equations

$$\dot{v} = 0.04v^2 + 5v + 140 - u + I \tag{1}$$

$$\dot{u} = a(b\,v - u) \tag{2}$$

with an after-reset-spike given by

$$if\ v \geq 30\,mV,\ else \begin{cases} v \leftarrow c \\ u \leftarrow u + d. \end{cases} \tag{3}$$

where the variables v and u, and the parameters a, b, c e d are dimensionless.

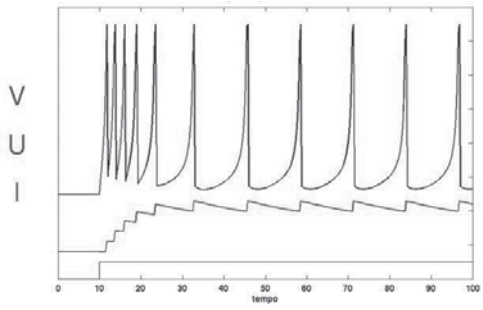

Figure 1. Temporal evolution of the model of Izhikevich [Adapted from Chaves [26]].

The values of the parameter set (a , b, c and d) do not change during runtime. These values establish which type of neuron dynamics is desired to simulate, this model is able to play more than 20 types of real neurons [25]. In general, u represents a membrane recovery variable [26]. When the membrane potential reaches its peak ($30\ mV$), the membrane potential (v) and the variable u are reset according to equation 3. The resting potential of the model is between -70 and $-50\ mV$ depending on the value of b. Like most real neurons, the model does not have a fixed threshold, depending on the history of membrane potential preceding the spike, the potential value threshold may be as low as $-55\ mV$ or as high as $-40\ mV$ [23].

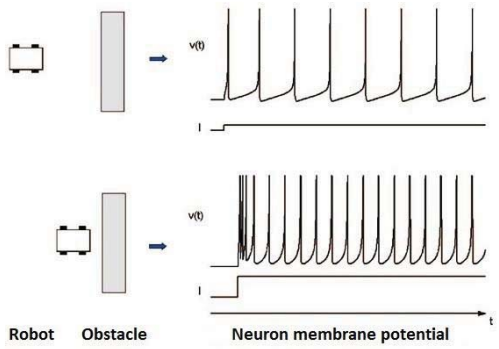

Figure 2. Excitable Class I neurons encode the distance between the robot and an obstacle through a sequence of spikes [Adapted from Arena et al.[6]].

A neural dynamics that catches attention is the dynamics found in the neurons of type Class I, located in the region of the hippocampus and responsible for tasks of movement and spatial localization [24]. The main feature of these neurons is the fact that they have a characteristic response through spikes which is proportional to the amplitude of the input signal (electric current value received). This is important because it acts as a mechanism for coding, encoding any measure as a characteristic frequency of spikes issued. The Izhikevich model parameters used to represent the behavior of a Class I neuron are: $a = 0.02$, $b = -0.1$, $c = -55$ and $d = 6$ [3], while the input variable *I* takes the values of both external stimuli (sensory stimuli) as the values of synaptic inputs. These types of neurons will be exploited for modeling the spatial navigation task of autonomous agent.

The synaptic input of a neuron is given by equation 4. Given a neuron j that receives synaptic connections from other neurons n, and knowing that t_i is the moment when the generic neuron i (connected to neuron j) sends a spike, the synaptic input from neuron j can be calculated by the following equation [6]:

$$I_j(t) = \sum_{i=1}^{n} w_{ij}\ \epsilon(t - t_i) \tag{4}$$

where the variable t corresponds to the current time, w_{ij} represents the weight of synapse connecting neuron i to neuron j and function $\epsilon(t)$ is defined by

$$\epsilon(t) = \begin{cases} \frac{t}{\tau}\exp(1-\frac{t}{\tau}) & if\ t \geq 0 \\ 0, & if\ t < 0. \end{cases} \quad (5)$$

and describes the contribution of a spike emitted by a presynaptic neuron at time $t = 0$ [6].

Recent experimental and theoretical studies indicate the existence of a mechanism of information processing based on the precise timing of the spike issue from neurons of the brain. A persistent change in synaptic efficacy depending on the relative timing of spikes pre and postsynaptic is phenomenon known as spike-timing-dependent plasticity or simply STDP [27]. Through this mechanism external stimuli can be encoded using the relative time between spikes of pre and postsynaptic neurons [28]. The use of the STDP learning rule opens new avenues for understanding the encoding of information in the biological brain. This mechanism of synaptic learning is biologically plausible, and has been observed in biological neural systems, especially in CA1 pyramidal neurons of the hippocampus [29]. This form of synaptic modification can automatically balance the synaptic connections to make postsynaptic neurons to fire irregularly, but more sensitive to the time of presynaptic spikes [30]. This synaptic learning algorithm was adopted to make changes in SNN synapses.

Formally, let w a variable that represents a synaptic weight. Pre- and postsynaptic action potentials change the weight w ($w \rightarrow w + \Delta w$) by means of a function $F(\Delta t)$. Δt represents the difference between the times of pre- and postsynaptic spikes:

$$\Delta t = t_{pre} - t_{post} \quad (6)$$

Moreover, according to Hosaka [28], the calculation of Δt used in the STDP rule can be done in two ways:

- the first is to consider all possible pairs of pre and postsynaptic spikes [30];
- or, consider only the pair of spikes closer to each other, that was the strategy adopted in this study (fig.3) [31].

Once defined the values of Δt, we can calculate the synaptic adjustment by the STDP rule itself [30] [6]:

$$\Delta W = \begin{cases} A_+ \exp^{\Delta t/\tau_+} & if\ \Delta t < 0 \\ A_- \exp^{-\Delta t/\tau_-} & if\ \Delta t \geq 0 \end{cases} \quad (7)$$

The parameters τ_+ and τ_- determine the variation of the interval between spikes in which synaptic changes of strengthening and weakening occur. The other parameters, A_+ and A_-, determine the maximum amount of synaptic modification (Δw) that occur when Δt is close to zero [32]. The parameter values used during the simulations can be found in [6].

Using STDP can lead to a overgrowth of synaptic weights, since being a technique of unsupervised learning the process updates the synaptic values at each instant of time. The strategy used to solve this problem is to take limited values for the synaptic weights [33] [34]. Moreover, some authors use a 'decay rate' rule in learning to control the values of the synaptic efficiencies [35] [36] [6].

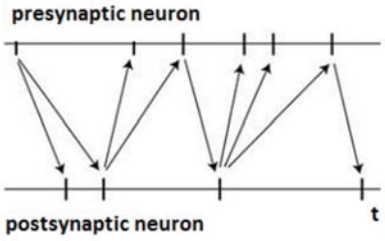

Figure 3. Only connected spikes by arrows contribute to synaptic plasticity [Adapted from Izhikevich [31]].

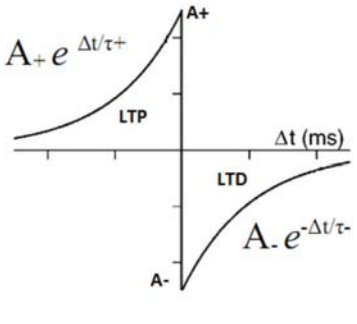

Figure 4. The STDP function of synaptic modification [Adapted from Song et al.[30]].

III. Structure Control and Simulation

The robot used in the trials was modeled using the robotics kit LEGO Mindstorms NXT. The robot is a vehicle with three wheels (tribot), two of them driven by two motors - identified as right and left motor. The motor activity of the wheels is directly controlled by the output of the spiking neural network. Furthermore, the robot has two touch sensors and two ultrasonic sensors located on the front of the agent, taking a range of perceptions of [0, +45 grad] (for the sensors on the right side) or [-45 grad, 0] (for the sensors on the left side).

Let d_0 the distance between the robot and the nearest object, measured in "robot units (ru)" [6]. The collision sensors are activated when the distance d_0 is equal to $d_0 = 0.6\ ru$, in this case the input associated with the stimulus is defined as a constant value of $I = 9$. This value causes the generation of regular spikes, allowing the robot to avoid an obstacle by changing its movement in one direction.

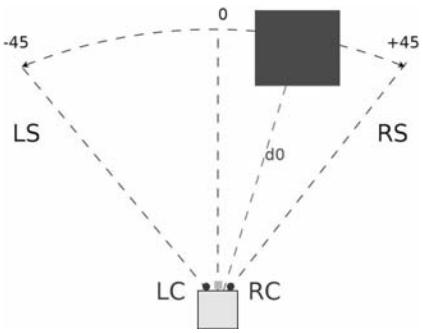

Figure 5. Positioning of the sensors of the agent. The sonars are represented by squares indicated by RS and LS (right sonar, left sonar). The contact sensors are represented by small circles indicated by RC and LC (right contact, left contact). The distance between the robot and the nearest object is identified by d_0.

For the ultrasonic sensors used for detecting and preventing barriers, the stimulus associated with the input of sensory neurons is described by an exponential function:

$$I = 9 \cdot \exp^{-0.6\ d_0} + 2.2. \tag{8}$$

Using this function the neurons start firing when the distance between the robot and the closest object is less than $11\ r.u.$ [6].

The robot's moving is done in an environment filled with randomly spaced obstacles. Each step of the implementation of the network simulates a time window of $300\,ms$. The movement of the robot is generated according to the number of spikes generated by motor neurons during this time interval. The number of spikes emitted by the two left (right) motor neurons (right) are summed to generate the control action of the robot. The signal that controls every motor depends on this number of spikes emitted by motor neurons associated with them.

A characteristic that can be observed in fig.6 is that motor neurons can be categorized as "go-on neurons" or "turn neurons" [6]. The go-on neurons generate the required behavior to allow the robot advance forward, even in the absence of external stimuli. The spikes of the other motor neurons (turn neurons) are added to the go-on ones, so that the motor control signal of the agent is composed of the sum of these two chains of spikes. In the event of a collision, the network structure inhibits the activity of go-on neurons, suppressing frontal movements. When both motors emit the same number of spikes, the agent will move forward. In the absence of conditioned stimuli (rotations) the range of motion is approximately $0.3\ ru$ for each simulation step.

When the information collected by the sensors produces a number of spikes different in left and right motors, the robot rotates. Let then ΔN_R and ΔN_L the number of spikes present in the right and left motor control signal, respectively, and that

$$\Delta N_s = \Delta N_R - \Delta N_L \tag{9}$$

is the difference between these values, then the angle of rotation (counterclockwise way) is given by:

$$\theta = 0.14\ \Delta N_s rad. \tag{10}$$

This means, for example, that if $\Delta N_R = 4$ and $\Delta N_L = 2$, the difference ΔN_s is $DeltaN_2 = 2$ and the robot will rotate with an angle of about $16\ ^\circ$. Without having run the network learning stage, the robot is driven by the environment only with its behavior not conditioning. In this case, the robot is controlled solely by the information collected by the sensors of contact and being able to avoid an obstacle after colliding with it. In case of frontal obstacles the direction of rotation is random.

During the simulations and experiments the synaptic weights of synapses not conditioned (touch sensors) were fixed at -8 (inhibitory synapses) and $+8$ (excitatory synapses). The initial values of the synapses of the conditioned stimuli (ultrasonic sensors) that are influenced by the STDP are initialized at 0.05 (excitatory synapses) and -0.05 (inhibitory synapses). Other values for the algorithms can be obtained in [30] and [6].

IV. Modeling of the Problem

A. The Network

In the SNN used in the problem of obstacle avoidance there are two types of neurons: sensory and motor neurons. The sensory neurons are the neurons connected to the robot sensors and they are responsible for computing the stimuli perceived by the robot. The values read by the sensors are set directly into the electric current variable of the neuron (variable *I* of equation 1). On the other hand, motor neurons are the neurons responsible for controlling the motors of the agent. Based on the dynamics and stimuli output of motor neurons (chains of spikes), control actions that move the agent are generated.

The network model adopted contains two layers of spiking neurons:

- sensitive layer: composed of sensory neurons it computes the stimuli perceived by the agent's sensors and propagates the information to the next layer through successive issues of spikes. This layer is composed of six sensory neurons: four of them receive as input the reading values of two touch sensors, positioned to the right and left of the robot. The other two neurons receive as input the reading values of two ultrasonic sensors, spaced similarly to the contact sensors;
- motor layer: composed of motor neurons, receive the spikes of the sensitive layer and turn it into a motor action, which directly controls the movement of the agent.

This layer is composed of four neurons. These neurons generate chains of spikes that will be interpreted as an action of motion (frontal, or change direction).

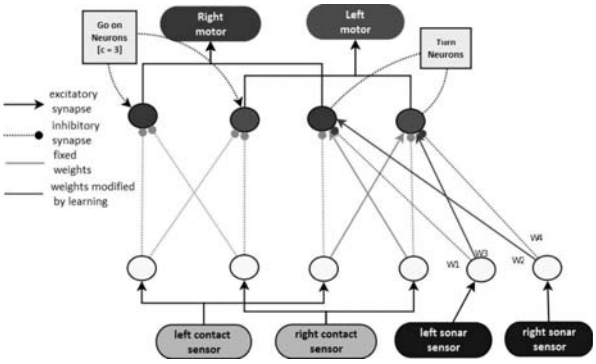

Figure 6. Diagram of SNN

The network structure and connections between neurons is inspired by Braitenberg vehicles [37] and by the work of Arena et al [6]. Such models are simple reactive vehicles equipped with a bioinspired control mechanism based on direct coupling between sensors and motors. The network model used is depicted in fig.6 and has direct inhibitory synapses and excitatory crossed synapses. This type of model is biologically relevant as it has been used to model the nervous system of crickets in the task of recognition of sound sources (cricket phonotaxis) [38].

The effect of excitatory synapses is depolarize the post-synaptic potential, while the inhibitory synapses one is the reverse, ie hyperpolarize this potential. According to Carvalho [39] excitatory synapses provide a sense of cooperation between the afferent neurons (sensory) and efferent (motor) neurons, while inhibitory synapses suggest an idea of competition between the same cells. Moreover, these synaptic weights may be predetermined or upgraded by the learning algorithm STDP. These predetermined weights are not changed by the learning algorithm and correspond to the basic behavior of the robot to handle a collision [6]. So with such structure defined, if the right contact sensor is triggered, the robot is able to make a turn to the left, and vice versa. When there is no external stimulus sensed by contact sensors, there is a constant (as can be seen in fig.6) that is inserted in the electric current variable of motor neurons and ensures that the neural activity responsible for the robot's movement.

B. Microsoft Robotics Developer Studio - MRDS

The implementations of simulated prototype and physical robot were performed using the Microsoft Robotics Developer Studio (MRDS)[1], a framework for application development in robotics. In addition to providing an environment for virtual simulation, the MRDS provides a set of tools for programming various robotics kits, included the LEGO Mindstorms NXT, which was used in this work. The software development kit (SDK) of the MRDS has several components, which mainly include: the Concurrency and Coordination Runtime (CCR) and the Decentralized Software Services (DSS).

The CCR is a programming model for handling multi-threading, task synchronization and error handling, while the DSS is used to build applications based on service models coupled. When working with robotics is known that many events can occur simultaneously. Thus, any application made in MRDS will contain one or more services, where each service is responsible for controlling certain process, such as receiving data from a touch sensor, a control motor, etc. Combine these services by performing message exchanges between them is one of the tasks of DSS. The CCR will deal with asynchronous communication between services that are running concurrently.

The basic unit in the MRDS is the "service", the terminology used to describe a program. Who stops and starts these services is the DSS, which also manages the flow of messages between such services. Services can be combined to create applications using the relations of "partnership". This whole process is called "orchestration" [40]. The services that compose an application can be created and destroyed dynamically. However, in most cases it is specified in the application startup via a 'manifest'. The manifest is an XML file that lists all the services needed and their relationships. A service can have many partner relationships as required. Additional information about the MRDS can be obtained in [40] and [41].

V. Experiments and Results

In this section are discussed various aspects related to the simulations and physical experiments carried out as well as on the results. Following the guidelines of the work of Arena et al. [6] was created a simulation arena for testing and prototyping. The simulated environment is a fully enclosed arena with obstacles placed randomly. The initial position of the robot in this space is also random (as obstacles) (fig.7). Thus, the aim is to design a robot that navigates in this space avoiding obstacles and walls.

A. Simulated Trials

Model validation carried out 10 different scenarios of navigation along the simulations. In such scenarios the initial position of the agent is $4 \leq obs \leq 7$. The length of the sonar perception is $10\ r.u.$. Each scenario was simulated for a period of time $T = 25.000$ simulation steps.

The machine used for simulation of the scenarios was a notebook Dell Vostro 1520 with: Intel Core 2 Duo from 2.20 GHz, RAM 4 GB, video card NVIDIA GeForce 9300M GS;

[1] http://www.microsoft.com/robotics/

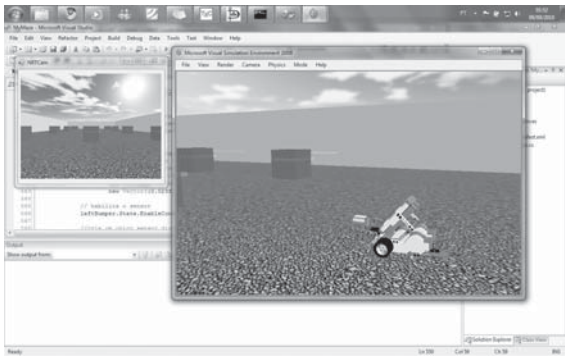

Figure 7. Virtual simulation environment. (Left) Frontal view of the robot. (Right) Simulated agent positioned close to obstacles.

running a Windows 7 - 64 bits. The versions of software used were Microsoft Robotics Developer Studio R2 and Microsoft Visual Studio 2008. The sources were written in C#.

To illustrate the performance of the robot in the navigation task of obstacle avoidance we use the performance measures described in Arena et al.[6]. A motion turn is the robot's response to an external stimulus triggered by a conditioned stimulus (sonar) or by an unconditioned stimulus (contact sensor). To test the behavior of the neural model then we analyze the amount of turns generated during the simulation. A change of direction by the information of sonar feature a"prediction" as a turn result of information from touch sensors feature a "collision". If the neural model has performed well throughout the simulation the number of preventions will grow, while the number of collisions decreases.

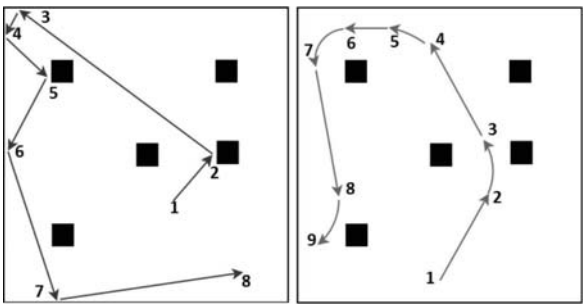

Figure 8. Simulated scenario. (Left) Before learning. (Right) After learning.

Figure 8 shows the behavior described by the robot in a simulation scenario observed. The diagram on the left shows the trajectory of the robot during the learning phase in which the robot avoids obstacles only through collisions. The diagram on the right shows that after this stage adaptation of the synaptic weights of the robot it can avoid the presence of obstacles using the information from ultrasonic sensors. The paths [2-3] and [8-9] in the right figure shows the preventive behavior of the robot, that when approaching an obstacle it uses the stimuli received by sonar to identify the location of the object and make the movement prevention. But the stretch between the points [4-7] shows how this 'solution' that the robot found also avoids collisions with the ends of the maze.

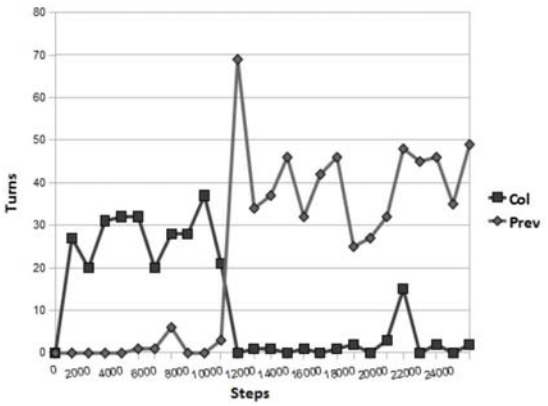

Figure 9. Comparison between collision and prevention obtained in the simulated scenario.

The chart with the performance measures of this scenario can be seen in fig.9. In the figure are compared the number of collisions and preventions executed by the agent in the simulated scenario. These numbers are calculated every 1000 simulation steps during robot navigation. We note that early prevention is null, while the number of collisions is high. Over the simulation, the number of collisions decreases while the number of preventions grows, indicating that the robot learned the task of prevention using sonar. Another point to be highlighted in the graphic is the peak reached in simulation time $T = 11{,}000$. Among the steps [10.000 - 11.000] of the simulation the SNN starts to use the information processed by the neurons linked to sonar, sensing the approach of foreign objects, resulting in a considerable number of turn actions with small angles ($\pm 8°$or $\pm 16°$). As the learning continues, the output of the network converges to other values of angles ($\pm 32°$, $\pm 40°$, etc.), reducing the number of moves to change direction in the subsequent periods of the simulation .

Although the network structure is symmetrical is noticeable that the synaptic weights are not. This particularity gives the robot decision strategies if an front obstacle is perceived. In this sense, the behavior of the network depends on the configuration of obstacles used to train the robot, and also directly dependent on the number of collisions on the right or the left found during the synaptic learning. Furthermore, based on average values of all the simulations we found that the SNN starts to learn to avoid obstacles around the

simulation time $T = 15{,}000$ (fig.10).

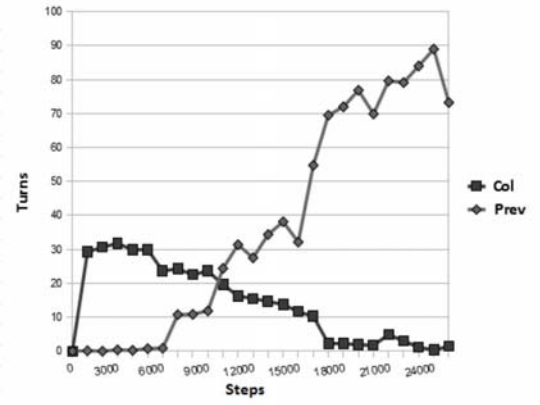

Figure 10. Comparison between collision and prevention (Average of all simulations).

A second performance measure used in trials is the ”average evasive distance” which is the average distance that a preemptive move is performed by the robot. Every time a rotation is detected, it stores the distance of the robot to the nearest object, and average value is estimated every 1000 simulation steps. It is expected that throughout the robot navigation the average distance grows and converge to a value near the maximum range of sonar.

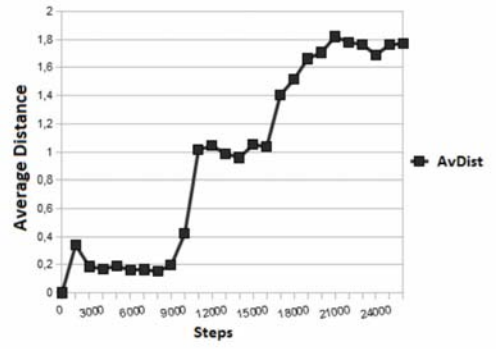

Figure 11. Mean distance from where the evasive movements are performed (Average of all simulations).

Figure 11 shows the behavior observed by the evasive average distance calculated over all simulated scenarios. It may be noted that when the robot starts moving,due to a larger number of collisions, the distance of the maneuvering is relatively small, with values between 0 e $0.4\ m$. From the moment that learning begins to hold its first preventions, the evasive distance value observed grows to between 0.8 e $1.2\ m$. Finally, after stabilization of the synaptic weights, the distance value increases and approaches the length perception of sonar ($2.0\ m$), with values between 1.6 e $1.9\ m$ and thus characterizing a safe navigation.

B. Physical experimentations

Subsequently the simulations was realized physical experimentations of the neural model. The navigation control agent through SNN was conducted in five different scenarios, where the number of obstacles ranging from 2 e 4, due to limitations of the dimensions of the arena prepared. Each experiment contains a total of 5000 simulated steps, where each step is equivalent to an action taken by the robot. The hardware used for agent modeling was the NXT model of robotics kit LEGO Mindstorms. The robot is equipped with two ultrasonic sensors for the perception of obstacles, sensors placed on the right and left of it by ensuring a range of perception of about 45°on both sides (as shown in fig.12).

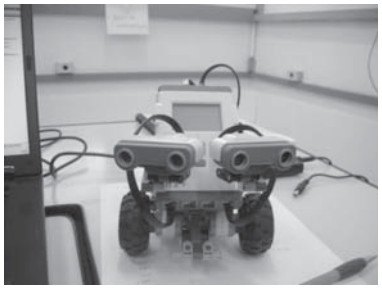

Figure 12. Robot used during the physical experiments.

The arena has dimensions of $2,0 \times 1,5\ m$ and is surrounded by Styrofoam plates to facilitate the reflection of sound waves emitted by sonar. The agent embedded in its environment of experimentation can be seen in fig.13. By moving toward the white box positioned in its front the robot identifies it as an obstacle and fires the treatment needed to avoid a possible collision.

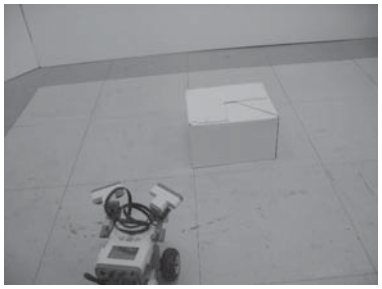

Figure 13. Robot perceiving the environment during physical experimentation.

The implementation of the program was made using the MRDS and Microsoft Visual Studio. Using MRDS allows reuse of code already developed in the simulator,

requiring only minor changes in some method-base calls of the framework. In addition, the program runs on a host machine, which makes requests and sends the actions to the robot, which receives, processes and executes them, returning success or failure in its execution. This is an interesting feature, because as the program runs directly on the robot, its performance is not affected by issues relating to the handling of internal memory of the robotics kit.

Figure 14. Trajectories described by the robot before (right) and after (left) synaptic learning.

The figure fig.14 shows the behavior of the agent during your navigation before and after synaptic learning. The approximate time of convergence of the algorithm was 40 minutes. In the figure is characterized the change in trajectory performed by the robot as a result of modifications of the synaptic weights of the network over the robot's movement. A move initially characterized by changes of direction that are very close to obstacles (fig.14 - (right)) becomes a movement driven by information obtained by the pair of sonars (fig.**??** (left)).

The chart with the performance of robot navigation over the five scenarios tested is shown in fig.15. Among the steps [0000-2000] can be noted a high number of collisions and the lack of preventive movements. Approximately in the range of steps [2500 - 2600] the weights begin to adjust by increasing the number of evasive movements, and reducing slightly the number of collisions. In the interval [4000 - 5000] can be distinguished a sharp drop in the number of collisions, compared to previous times. At the same time, the distance at which avoidance occur increases, characterizing 'safer' movements. As the number of preventions grows, this distance is increased approaching the maximum range of the sonar (as shown in Fig. ref fig: Distancia.Media.Fisico).

Although the behavior observed during physical experiments to be within expectations, it was possible to notice a difference in the number of collisions recorded at the end of the trials in relation to data obtained at the end of simulations (figures 10 and 15). In simulated experiments the number of collisions decreases sharply and remained at a very low value, "featuring an almost ideal situation". Moreover, in trials, although it has decreased the number of collisions remains considerably high, suggesting the interference of some factor (or factors) linked directly to trial, such as:

- questions regarding the battery of the robot, which can wear out and interfere with the performance of sensors, actuators, and communication with the host machine;
- problems of calibration of sensors;
- communication delays via bluetooth, which is the mechanism used by MRDS for communication with the NXT;
- scan time of the sensors (sonar), which occurs in a certain window of time (in milliseconds) and in accordance with the occurrence of an event. Depending on the distance between robot and host machine, and the time of the reading window may occur a delay in the execution of an action and return values by the sensors, featuring wrong and unexpected behaviors;
- interference from the environment used for physical experimentation, which was far from ideal due to infrastructure problems in the robotics lab;
- materials used for preparation of the maze, etc..

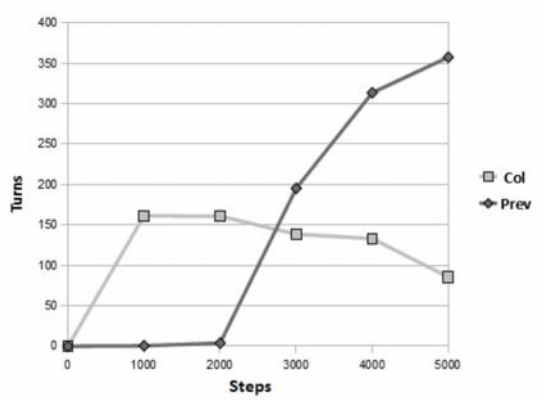

Figure 15. Comparison between collision and prevention (Average of all experimentations).

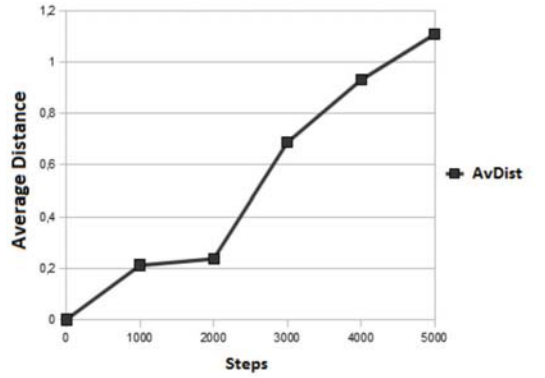

Figure 16. Mean distance from where the evasive movements are performed (Average of all experimentations).

Despite the data obtained by physical experiments are not "ideal", the behavior approaches that was seen in the

simulations, and both sets of tests were able to decrease the number of collisions and increase the number of preventions.

VI. Conslusions

In the current paper is discussed the use of Spiking Neural Networks (SNN) to control a robotic agent in a task of autonomous navigation (obstacle avoidance). To model this phenomenon there are many neurocomputational models available in specific literature, varying in complexity and computational cost for implementation. One of the few models that can balance both of these characteristics is the model proposed by Izhikevich, formalized by two differential equations and a post-reset after spike. The advantage of this model is that besides having a non-expensive implementation, it can play a wide range of behavior of nerve cells using only a small set of parameters.

The use of the Izhikevich's model, the network topology adopted and based on the works of Braitenberg (1984) and Arena et al. (2009), with the synaptic learning algorithm used (STDP), incorporate biological plausibility of the neural model developed over the paper, both for representation and manipulation of information to the network via spikes. The use of the spike timing of neurons to encode the information could be observed in cells of the hippocampus, responsible for the tasks of spatial representation and long-term memory, as evidenced by neuroscience studies.

With respect to MRDS, the use of a simulator helps to correct situations not covered during the modeling, correcting a flaw in a shorter time than necessary to correct the final physical implementation. Moreover, programming in the MRDS facing simulated environment or the hardware of the robot spends little time since the necessary chances in the source code are few, keeping a base reasoning intact. Another plus point of MRDS is that the application runs on one machine host and communicates with the robot device through bluetooth, thus not loading the internal memory of the robot and not limiting the implementation of the program.

In order to validate the model were created simulation and physical scenarios with robot and obstacles placed randomly within a maze completely surrounded. If the SNN could play his role well, the robot could avoid obstacles using the information of sonar instead of the information provided by the sensors of a collision. And what we have seen over the trials was exactly such behavior. After a learning period, which varied depending on the configuration of the scenario, the number of collisions detected by using touch sensors decreased, while the number of evasive movements generated by the sonar information grew, featuring obstacle avoidance. The relative distance between the robot and obstacles while turning in a direction has also increased. Initially this value was relatively low, since collisions were detected. While the synaptic weights were adjusted by the STDP algorithm this distance increased significantly as to avoid an obstacle. This proves that the SNN learns well and use the informations of sonar with a satisfactory accuracy.

A. Future works

As future contributions, here are some suggestions for changes and improvements at work:

- Explore other mathematical models of neurons. Besides the Izhikevich's formalisms [25], some classical models could be tested, as well as the Kasabov probabilistic model [42];
- Investigate the use of other mechanisms of synaptic learning;
- Insertion of a delay in emission of spikes in the synapses of the network, since for the model adopted the transmission of a presynaptic spike by two distinct postsynaptic neurons takes the same amount time. One aspect to be explored and analyzed in the learning algorithm would be the insertion of delay transmission;
- During the update of synaptic weights of the STDP algorithm only the latter spike issued contributes to synaptic efficiency. A possible modification in the implementation of this mechanism would be to consider a fixed number of spikes and observe the behavior generated by the network learning, as well as their convergence;
- Another aspect is the increase of visual stimuli of the network (camera with agent's view, visual cues) also pushing the navigation task for which the model is designated;
- Reset the reading range of sonar in an attempt to circumvent the implementation of 'noisy' actions;
- Modify the parameters of synaptic learning to try to accelerate the adjustment of synaptic weights;
- Optimize the process of physical experimentation, to bring it to the ideal situation observed in the simulations
- Finally it would be interesting to build an application using the MRDS virtual simulation of various types of navigation problems, creating a framework for 3D experiments. The MRDS allows parallel execution of programs distributed within a local network, and this feature of parallelism could be exploited in order to compare different models (robots, architecture) in several hardware.

References

[1] S. Soumare, A. Ohya, and S. Yuta, "Real-time obstacle avoidance by an autonomous mobile robot using an active vision sensor and a vertically emitted laser slit," 2002, pp. 3068–3073.

[2] T. K. Horiuchi, "A spike-latency model for sonar-based navigation in obstacle fields," *Trans. Cir. sys. Part I*, vol. 56, no. 11, pp. 2393–2401, 2009.

[3] E. M. Izhikevich, "Which model to use for cortical spiking neurons?" *IEEE Trans. Neural Netw.*, vol. 15, no. 5, pp. 1063–1070, 2004.

[4] ——, *Dynamics systems in neuroscience: The Geometry of excitability and bursting*. The MIT Press, 2007.

[5] A. Novellino, P. D'angelo, L. Cozzi, M. Chiappalone, V. Sanguineti, and S. Martinoia, "Connecting neurons to a mobile robot: an in vitro bidirectional neural interface," *Intell. Neuroscience*, vol. 2007, pp. 2–17, 2007.

[6] P. Arena, L. Fortuna, M. Frasca, and L. Patane, "Learning anticipation via spiking networks: application to navigation control," *Trans. Neur. Netw.*, vol. 20, no. 2, pp. 202–216, 2009.

[7] N. Burgess and J. O'Keefe, "Neuronal computations underlying the firing of place cells and their rola in navigation," *Hippocampus*, no. 6, pp. 749–762, 1996.

[8] N. Burgess, S. Becker, J. A. King, and J. O'Keefe, "Memory for events and their spatial context: Models and experiments," *Philos. Trans. roy. Soc. London B*, vol. 356, 2001.

[9] O. Jensen and J. E. Lisman, "Hippocampal sequence-encoding driven by a cortical multi-item working memory buffer," *Trends in Neurosciences*, vol. 28, no. 2, pp. 67–72, 2005.

[10] C. Christodoulou, G. Bugmann, and T. G. Clarkson, "A spiking neuron model: Applications and learning," *Neural Networks*, vol. 15, no. 7, pp. 891–908, 2002.

[11] M. J. Escobar and E. AL, "Action recognition using a bio-inspired feedforward spiking network," *Int. Journal Comput. Vis.*, vol. 82, pp. 284–301, 2009.

[12] P. Rowcliffe, J. Feng, and H. Buxton, "Clustering within integrate-and-fire neurons for image segmentation." London, UK: Springer-Verlag, 2002, pp. 69 – 74.

[13] J. M. Buhmann, T. M. Lange, and U. M. Ramacher, "Image segmentation by networks of spiking neurons," *Neural Comput.*, vol. 17, no. 5, pp. 1010–1031, 2005.

[14] H. Burgsteiner, M. Kroll, A. Leopold, and G. Steinbauer, "Movement prediction from real=world images using a liquid state machine," *Applied Intelligence*, vol. 26, no. 2, pp. 99–109, 2007.

[15] W. Maass, T. Natshlager, and H. Markram, "Real-time computing without stable states: a framework for neural computation based on perturbation," *Neural Comput.*, vol. 17, no. 8, pp. 2531 – 2560, 2002.

[16] D. Verstraeten, B. Schrauwen, D. Strootbandt, and J. V. Campenhout, "Isolated word recognition with the liquid state machine: a case study," *Information Processing Letters*, vol. 95, no. 6, pp. 521–528, 2005.

[17] P. Joshi and W. Maass, "Movement generation with circuits of spiking neurons," *Neural Comput.*, vol. 17, no. 8, pp. 1715–1738, 2005.

[18] P. Rowcliffe and J. Feng, "Training spiking neuronal networks with applications in engineering tasks," *IEEE Transactions on Neural Networks*, vol. 19, no. 9, pp. 1626 – 1640, 2008.

[19] K. Voutsas and J. Adamy, "Biologically inspired spiking neural network for sound source lateralization," *IEEE Trans. on Nueral Netw.*, vol. 18, no. 6, pp. 1785–1799, 2007.

[20] X. Wang, Z. G. Hou, A. Zou, M. Tan, and L. Cheng, "A behavior controller based on spiking neural networks for mobile robots," *Neurocomput.*, vol. 71, no. 4–6, pp. 655–666, 2008.

[21] W.-H. Hung, P. Liu, and S.-C. Kang, "Service-based simulator for security robot," in *Advanced robotics and Its Social Impacts, 2008. ARSO 2008. IEEE Workshop on*, 2008, pp. 1–3.

[22] A. L. Hodgkin and A. F. Huxley, "A quantitative description of ion currents and its applications to conduction and excitation in nerve membranes," *J. Physiol. (London)*, vol. 117, pp. 500–544, 1952.

[23] G. M. C. Oliveira, "Emergência de sincronicidade em um sistema de neurônios pulsantes acoplados," Master's thesis, Centro Federal de Educação Tecnológica de Minas Gerais - CEFET-MG, Dezembro 2007.

[24] P. Dayan and L. F. Abbott, *Theoretical neuroscience: Computational and mathematical modeling of neural systems*, 1st ed. The MIT Press, 2001.

[25] E. M. Izhikevich, "Simple model of spiking neurons," *IEEE Trans. Neural Netw.*, vol. 14, no. 6, pp. 1569–1572, 2003.

[26] J. F. Chaves, "Síntese de estruturas bio-inspiradas baseada em redes de neurônios pulsantes," Master's thesis, Centro Federal de Educação Tecnológica de Minas Gerais - CEFET-MG, Dezembro 2007.

[27] R. Florian, "Reinforcement learning through modulation of spike-timing-dependent synaptic plasticity," *Neural Computation*, vol. 19, no. 6, pp. 1468–1502, 2007.

[28] R. Hosaka, O. Araki, and T. Ikeguchi, "Stdp provides the substrate for igniting synfire chains by spatio-temporal input patterns," *Neural Computational*, vol. 20, no. 2, pp. 415–435, 2008.

[29] Y. Dan and M. ming Poo, "Spike timing-dependent plasticity of neural circuits," *Neuron*, vol. 44, pp. 23–30, 2004.

[30] S. Song, K. D. Miller, and L. F. Abbott, "Competitive hebbian learning through spike-timing-dependent synaptic plasticity," *Nature Neuroscience*, vol. 3, no. 9, pp. 919–926, 2000.

[31] E. M. Izhikevich, J. A. Gally, and G. M. Edelman, "Spike-timing dynamics of neuronal groups," *Cerebral Cortex*, vol. 14, pp. 933–944, 2004.

[32] L. Benuskova and W. Abraham, "Stdp rule endowed with the bcm sliding threshold accounts for hippocampal heterosynaptic plasticity," *Journal of Computational Neuroscience*, vol. 22, pp. 129–133, 2007.

[33] S. Song and L. F. Abbott, “Cortical development and remapping through spike timing-dependent plasticity,” *Neuron*, vol. 32, pp. 339–350, 2001.

[34] E. M. Izhikevich, “Solving the distal reward problem through linkage of stdp and dopamine signaling,” *Cereb. Cortex*, vol. 17, no. 10, pp. 2443–2452, 2007.

[35] P. F. M. J. Verschure, B. J. A. Kröse, and R. Pfeifer, “Distributed adaptive control: The self-organization of structured behavior,” *Robot. Autonom. Syst.*, vol. 9, pp. 181–196, 1992.

[36] P. F. M. J. Verschure and R. Pfeifer, “Categorization, representations, and the dynamics of system-environment interaction: A case study in autonomous systems,” J. A. Meyer, H. Roitblat, and S. Wilson, Eds., 1992, pp. 210 – 217.

[37] V. Braitenberg, *Experiments in synthetic psychology*. The MIT Press, 1984.

[38] B. Webb and T. Scutt, “A simple latency dependent spiking neuron modelo of cricket phonotaxis,” *Biol. Cybern.*, vol. 82, no. 3, pp. 247–269, 2000.

[39] L. A. V. de Carvalho, “Síntese de redes neuronais com palicações à representação do conhecimento e à otimização,” Ph.D. dissertation, Universidade Federal do Rio de Janeiro - UFRJ, Rio de Janeiro, Brasil, Março 1989.

[40] K. Johns and T. Taylor, *Professional Microsoft Robotics Developer Studio*. Wiley Publishing, Inc., 2007.

[41] S. Morgan, *Programing Microsoft Robotics Studio*. Microsoft Press, 2008.

[42] N. Kasabov, “To spike or not to spike: A probabilistic spiking neuron model,” *Neural Netw.*, vol. 23, no. 1, pp. 16–19, 2010.

Referências

ARENA, P.; FORTUNA, L.; FRASCA, M.; PATANE, L. Learning anticipation via spiking networks: application to navigation control. *Trans. Neur. Netw.*, v. 20, n. 2, p. 202–216, 2009.

BARRERA, A.; WEITZENFELD, A. Biologically-inspired robot spatial cognition based on rat neurophysiological studies. *Auton. Robots*, v. 25, n. 1–2, p. 147–169, 2008.

BENUSKOVA, L.; ABRAHAM, W. Stdp rule endowed with the bcm sliding threshold accounts for hippocampal heterosynaptic plasticity. *Journal of Computational Neuroscience*, Springer Netherlands, v. 22, p. 129–133, 2007.

BOHTE, S. M. *Spiking Neural Networks*. 1-145 p. Tese (PhD in Computer Science, Netherlands) — Universitet Lieden, 2003.

BOHTE, S. M. The evidence for neural information processing with precise spike-times: A survey. *Natural Computing*, v. 3, n. 2, p. 195–206, 2004.

BOHTE, S. M.; KOK, J. N.; POUTRE, L. H. Error-backpropagation in temporally encoded networks of spiking neurons. *Neurocomputing*, v. 48, n. 1-4, p. 17 – 37, 2002.

BRAITENBERG, V. *Experiments in synthetic psychology*. [S.l.]: The MIT Press, 1984.

BUHMANN, J. M.; LANGE, T. M.; RAMACHER, U. M. Image segmentation by networks of spiking neurons. *Neural Comput.*, v. 17, n. 5, p. 1010–1031, 2005.

BURGESS, N.; BECKER, S.; KING, J. A.; O'KEEFE, J. Memory for events and their spatial context: Models and experiments. *Philos. Trans. roy. Soc. London B*, v. 356, 2001.

BURGESS, N.; O'KEEFE, J. Neuronal computations underlying the firing of place cells and their rola in navigation. *Hippocampus*, n. 6, p. 749–762, 1996.

BURGSTEINER, H.; KROLL, M.; LEOPOLD, A.; STEINBAUER, G. Movement prediction from real=world images using a liquid state machine. *Applied Intelligence*, v. 26, n. 2, p. 99–109, 2007.

CARVALHO, L. A. V. de. *Síntese de redes neuronais com palicações à representação do conhecimento e à otimização*. Tese (Doutorado) — Universidade Federal do Rio de Janeiro - UFRJ, Rio de Janeiro, Brasil, Março 1989.

CHAVES, J. F. *Síntese de estruturas bio-inspiradas baseada em redes de neurônios pulsantes*. Dissertação (Mestrado) — Centro Federal de Educação Tecnológica de Minas Gerais - CEFET-MG, Dezembro 2007.

CHRISTODOULOU, C.; BUGMANN, G.; CLARKSON, T. G. A spiking neuron model: Applications and learning. *Neural Networks*, v. 15, n. 7, p. 891–908, 2002.

CONNOR, J. A.; WALTER, D.; MCKOWN, R. Neural repetitive firing: Modifications of the hodgkin-huxley axon sugested by experimental results from crustacean axons. *Biophysical Journal*, v. 81-102, p. 18, 1977.

CURTIS, H. J.; COLE, K. S. Membrane action potentials from the squid giant axon. *J. Cell. & Comp. Physhiol.*, v. 15, p. 147–157, 1940.

CURTIS, H. J.; COLE, K. S. Membrane resting and action potentials from the squid giant axon. *J. Cell. & Comp. Physhiol.*, v. 19, p. 135–144, 1942.

DAN, Y.; POO, M. ming. Spike timing-dependent plasticity of neural circuits. *Neuron*, v. 44, p. 23–30, 2004.

DAYAN, P.; ABBOTT, L. F. *Theoretical neuroscience: Computational and mathematical modeling of neural systems*. 1st. ed. [S.l.]: The MIT Press, 2001.

DURAN, J. E. R. Biofísica: Fundamentos e aplicações. In: _____. [S.l.]: Pearson Prentice hall, 2003. cap. Membranas excitáveis, potenciais de ação, eletrorreceptores e peixes-elétricos, p. 163–186.

ESCOBAR, M. J.; AL, E. Action recognition using a bio-inspired feedforward spiking network. *Int. Journal Comput. Vis.*, v. 82, p. 284–301, 2009.

FITZHUGH, R. Impulses and physiological states in theoretical models of nerve membrane. *Biophysical Journal*, v. 1, n. 6, p. 445–466, 1961.

FITZHUGH, R. Biological engineering. In: _____. [S.l.]: McGraw-Hill, 1969. cap. Mathematical models of excitation and propagation in nerve, p. 1–85.

FLOREANO, D.; EPARS, Y.; ZUFFEREY, J.; MATTIUSSI, C. Evolution of spiking neural circuits in autonomous mobile robots. *Int. J. Intell. Syst.*, v. 21, n. 9, p. 1005–1024, 2006.

FLORIAN, R. Reinforcement learning through modulation of spike-timing-dependent synaptic plasticity. *Neural Computation*, v. 19, n. 6, p. 1468–1502, 2007.

FOUNDATION, W. *Wikimedia Commons*. april 2010. Disponível em: <http://commons-.wikimedia.org/>.

FRANZ, M.; MALLOT, A. H. Biomimetic robot navigation. *Robotics and Autonomous Systems*, v. 30, p. 133–153, 2000.

FURBER, S.; TEMPLE, S. Neural systems engineering. *J. R. Soc. Interface*, v. 4, n. 13, p. 193–206, 2006.

GERSTNER, W.; KISTLER, W. M. *Spiking neuron models: Single neurons, population, platicity*. [S.l.]: Cambridge University Press, 2002.

GIBILISCO, S. *Concise encyclopedia of robotics*. [S.l.]: McGraw-Hill, 2003.

HAYKIN, S. *Redes neurais: Princípios e prática*. 2. ed. [S.l.]: Bookman, 1999.

HEBB, D. *The organization of behavior: A neuropsychological theory*. [S.l.]: Wiley, 1949.

HODGKIN, A. L.; HUXLEY, A. F. A quantitative description of ion currents and its applications to conduction and excitation in nerve membranes. *J. Physiol. (London)*, v. 117, p. 500–544, 1952.

HOLLAND, J. *Designing autonomous mobile robots*. [S.l.]: Elsevier, 2004.

HOPPENSTEADT, F. C. *An introduction to the mathematics of neurons*. [S.l.]: Cambridge University Press, 1986.

HORIUCHI, T. K. A spike-latency model for sonar-based navigation in obstacle fields. *Trans. Cir. sys. Part I*, v. 56, n. 11, p. 2393–2401, 2009.

HOSAKA, R.; ARAKI, O.; IKEGUCHI, T. Stdp provides the substrate for igniting synfire chains by spatio-temporal input patterns. *Neural Computational*, v. 20, n. 2, p. 415–435, 2008.

HUNG, W.-H.; LIU, P.; KANG, S.-C. Service-based simulator for security robot. In: *Advanced robotics and Its Social Impacts, 2008. ARSO 2008. IEEE Workshop on*. [S.l.: s.n.], 2008. p. 1–3.

IRWIN, J. D. *Análise de circuitos em engenharia*. [S.l.]: Pearson Makron Books, 2000.

IZHIKEVICH, E. M. Resonate-and-fire neurons. *Neural Netw.*, v. 14, n. 6, p. 883–894, 2001.

IZHIKEVICH, E. M. Simple model of spiking neurons. *IEEE Trans. Neural Netw.*, v. 14, n. 6, p. 1569–1572, 2003.

IZHIKEVICH, E. M. Which model to use for cortical spiking neurons? *IEEE Trans. Neural Netw.*, v. 15, n. 5, p. 1063–1070, 2004.

IZHIKEVICH, E. M. *Dynamics systems in neuroscience: The Geometry of excitability and bursting*. [S.l.]: The MIT Press, 2007.

IZHIKEVICH, E. M. Solving the distal reward problem through linkage of stdp and dopamine signaling. *Cereb. Cortex*, v. 17, n. 10, p. 2443–2452, 2007.

IZHIKEVICH, E. M.; ELDELMAN, M. G. Large-scale model of mammalian thalamocortical systems. *Proceedings of the National Academy of Sciences of the USA*, v. 105, n. 9, p. 3593–3598, 2008.

IZHIKEVICH, E. M.; GALLY, J. A.; EDELMAN, G. M. Spike-timing dynamics of neuronal groups. *Cerebral Cortex*, v. 14, p. 933–944, 2004.

JENSEN, O.; LISMAN, J. E. Hippocampal sequence-encoding driven by a cortical multi-item working memory buffer. *Trends in Neurosciences*, v. 28, n. 2, p. 67–72, 2005.

JOHNS, K.; TAYLOR, T. *Professional Microsoft Robotics Developer Studio*. [S.l.]: Wiley Publishing, Inc., 2007.

JOSHI, P.; MAASS, W. Movement generation with circuits of spiking neurons. *Neural Comput.*, v. 17, n. 8, p. 1715–1738, 2005.

KASABOV, N. Challenges for computational intelligence. In: _____. [S.l.]: Pearson Prentice hall, 2007. v. 63, cap. Brain-, gene-, and quantum inspired computational intelligence: Challenges and opportunities, p. 193–219.

KASABOV, N. *Evolving connectionist systems: The knownledge engineering approach*. London: Springer, 2007.

KASABOV, N. Integrative connectionist learning systems inspired by nature: current models, future trends and challenges. *Natural Computing: an international journal*, Kluwer Academic Publishers, v. 8, n. 2, p. 199–218, 2009.

KASABOV, N. To spike or not to spike: A probabilistic spiking neuron model. *Neural Netw.*, v. 23, n. 1, p. 16–19, 2010.

KOCH, C. *Biophysics of computation: Information processing in single neurons*. [S.l.]: Oxford University Press, 1999.

KOCH, C.; SEGEV, I. *Methods in neuronal modeling: From synapses to networks*. [S.l.]: The MIT Press, 1989.

LAPICQUE, L. Recherches quantitatives sur l'excitation electrique des nerfs traitee comme une polatization. *J. Physiol. Pathol. Gen.*, v. 9, p. 620–635, 1907.

LEVITT, T. S.; LAWTON, D. T. Qualitative navigation for mobile robots. *Artif. Intell.*, v. 44, n. 3, p. 305–306, 1990.

LISMAN, J.; SPRUSTON, N. Postsynaptic depolarization requirements for ltp and ltd: a critique of spike timing-dependent plasticity. *Nature Neuroscience*, v. 8, n. 7, p. 839–841, 2005.

LIU, J.; PEREZ-GONZALEZ, D.; REES, A.; ERWIN, H.; WERMTER, S. A biomimetic spiking neural network of the auditory midbrain for mobile robot sound localization in reverberant enironments. In: . [S.l.]: IEEE Press, 2009. p. 648–655.

LLINAS, R. R. The intrinsic electrophysiological properties of mammalian neurons: Insights into central nervous system function. *Science*, v. 242, p. 1654–1664, 1988.

MAASS, W.; NATSHLAGER, T.; MARKRAM, H. Real-time computing without stable states: a framework for neural computation based on perturbation. *Neural Comput.*, v. 17, n. 8, p. 2531 – 2560, 2002.

MCCULOCH, W. S.; PITTS, W. A logical calculus of the ideas immanent in nervous activity. *Bulletin of Mathematical Biology*, Springer, v. 5, n. 4, p. 115–133, 1943.

MELAMED, O.; GERSTNER, W.; MAASS, W.; TSODYKS, M.; MARKRAM, H. Coding and learning of behavioral sequences. *Trends in Neurosciences*, v. 27, n. 1, p. 11–14, 2004.

MOIOLI, R. *Sobre cognição, adaptação e homeostase: uma análise e síntese de ferramentas computacionais bioinspiradas aplicadas à navegação autônoma de robôs*. Campinas, Brazil: [s.n.], 2008.

MORGAN, S. *Programing Microsoft Robotics Studio*. [S.l.]: Microsoft Press, 2008.

MURPHY, R. R. *Introduction to AI robotics*. [S.l.]: The MIT Press, 2000.

NOVELLINO, A.; D'ANGELO, P.; COZZI, L.; CHIAPPALONE, M.; SANGUINETI, V.; MARTINOIA, S. Connecting neurons to a mobile robot: an in vitro bidirectional neural interface. *Intell. Neuroscience*, v. 2007, p. 2–17, 2007.

OKUNO, E.; CALDAS, I. L.; CHOW, C. *Física para ciências biológicas e biomédicas*. [S.l.]: Harper e Row do Brasil, 1982.

OLIVEIRA, G. M. C. *Emergência de sincronicidade em um sistema de neurônios pulsantes acoplados*. Dissertação (Mestrado) — Centro Federal de Educação Tecnológica de Minas Gerais - CEFET-MG, Dezembro 2007.

PAUGAM-MOISY, H.; BOHTE, S. M. Computing with spiking neuron networks. In: _____. *Handbook of natural computing*. [S.l.]: Springer Verlag, 2009. p. 1–47.

REEVE, R.; HALLAM, J. An analysis of neural models for walking control. *Neural Netw.*, v. 16, n. 3, p. 733–742, 2005.

ROWCLIFFE, P.; FENG, J. Training spiking neuronal networks with applications in engineering tasks. *IEEE Transactions on Neural Networks*, v. 19, n. 9, p. 1626 – 1640, 2008.

ROWCLIFFE, P.; FENG, J.; BUXTON, H. Clustering within integrate-and-fire neurons for image segmentation. In: . London, UK: Springer-Verlag, 2002. p. 69 – 74.

ROWCLIFFE, P.; FENG, J.; BUXTON, H. Spiking perceptrons. *IEEE Transactions on Neural Networks*, v. 17, n. 3, p. 803 – 807, 2006.

SCHMICKL, T.; HAMANN, H.; WORN, H.; CRAILSHEIM, K. Two different approaches to a macroscopic model of a bio-inspired robotic swarm. *Robot. Auton. Syst.*, v. 57, n. 9, p. 913–921, 2009.

SICILIANO, B.; KHATIB, O. (Ed.). *Handbook of robotics*. [S.l.]: Springer, 2008.

SIMOES, A. da S. *Aprendizado não-supervisionado em redes neurais pulsadas de base radial*. Tese (Doutorado) — Tese (Doutorado) - Escola Politécnica da Universidade Estadual de São Paulo, São Paulo, Brazil, 2006.

SONG, S.; MILLER, K. D.; ABBOTT, F. Competitive hebbian learning through spike-timing-dependent synaptic plasticity. *Nature Neuroscience*, v. 3, n. 9, p. 919–926, 2000.

SOUMARE, S.; OHYA, A.; YUTA, S. Real-time obstacle avoidance by an autonomous mobile robot using an active vision sensor and a vertically emitted laser slit. In: . [S.l.: s.n.], 2002. p. 3068–3073.

STEIN, R. Some models of neuronal variability. *Biophysical Journal*, v. 7, n. 1, p. 37 – 68, 1967.

THORPE, S.; DELORME, A.; RULLE, R. V. Spike-based strategies for rapid processing. *Neural Netw.*, v. 14, n. 6-7, p. 715–725, 2001.

TRULLIER, O.; WIENER, S. I.; BERTHOZ, A.; MEYER, J. A. Biologically based aritificial navigation systems: reviews and prospects. *Progress in Neurobiology*, v. 51, p. 483–544, 1997.

TUCKWELL, H. *Introduction to theoretical neurobiology Vol. 2: Nonlinear and stochastic theories*. [S.l.]: Cambrigde Univesity Press, 1988.

TUCKWELL, H. *Introduction to theoretical neurobiology Vol.1: Linear cable theory and dendritic structure*. [S.l.]: Cambridge Univesity Press, 1988.

VERSTRAETEN, D.; SCHRAUWEN, B.; STROOTBANDT, D.; CANPENHOUT, J. V. Isolated word recognition with the liquid state machine: a case study. *Information Processing Letters*, v. 95, n. 6, p. 521–528, 2005.

VOUTSAS, K.; ADAMY, J. Biologically inspired spiking neural network for sound source lateralization. *IEEE Trans. on Nueral Netw.*, v. 18, n. 6, p. 1785–1799, 2007.

VREEKEN, J. *Spiking neural networks - An introduction*. 2003.

WANG, X.; HOU, Z. G.; ZOU, A.; TAN, M.; CHENG, L. A behavior controller based on spiking neural networks for mobile robots. *Neurocomput.*, v. 71, n. 4–6, p. 655–666, 2008.

WEBB, B.; SCUTT, T. A simple latency dependent spiking neuron modelo of cricket phonotaxis. *Biol. Cybern.*, v. 82, n. 3, p. 247–269, 2000.