Controle Inteligente de Veículos Autônomos: Automatização do Processo de Estacionamento de Carros

Fernando Osório, Farlei Heinen e Luciane Fortes

UNISINOS – Universidade do Vale do Rio dos Sinos
Centro de Ciências Exatas e Tecnológicas – C6
Mestrado em Computação Aplicada – PIPCA
Av. Unisinos, 950 – São Leopoldo, RS
E-mail: osorio@exatas.unisinos.br
farlei@indus.unisinos.br
luciane.fortes@gm.com

RESUMO:

Este trabalho tem por objetivo apresentar um sistema de controle inteligente de veículos autônomos. O sistema que estamos desenvolvendo é responsável pela automatização da tarefa de condução de um veículo, onde buscamos obter um sistema de controle robusto capaz de estacionar um carro em uma vaga paralela. O sistema SEVA (Simulador de Estacionamento de Veículos Autônomos) permite controlar o carro através da leitura de um conjunto de sensores, gerando os comandos de aceleração e de giro de direção, de modo a localizar e estacionar o carro em uma vaga. Atualmente o sistema conta com um controlador baseado em um sistema especialista (conjunto de regras heurísticas) e com um controlador gerenciado por uma Rede Neural Artificial com aprendizado supervisionado. Os resultados obtidos até o presente demonstram que ambos os controladores são capazes de estacionar corretamente um carro, baseados apenas nas informações provenientes de seus sensores externos.

PALAVRAS- CHAVE: robótica autônoma, controle sensorial-motor inteligente, condução autônoma de veículos, aprendizado neural, sistema especialista.

1. Introdução

Os veículos autônomos (RMA – Robôs Móveis Autônomos) tem atraído a atenção de um grande número de pesquisadores da área de Inteligência Artificial, devido ao desafio que este

novo domínio de pesquisas nos propõe: dotar sistemas de uma capacidade de raciocínio inteligente e de interação com o meio em que estão inseridos. Os RMAs podem "sentir" o ambiente em que estão inseridos através da leitura de seus sensores (e.g. sensores infra-vermelho, lasers, *bumpers*, câmeras de vídeo, etc), e através desta percepção sensorial eles podem planejar melhor as suas ações [Medeiros 98, Heinen 99].

Atualmente temos robôs móveis atuando em diferentes áreas, como por exemplo: robôs desarmadores de bombas, robôs usados para a exploração de ambientes hostis, e a condução de veículos (carros) robotizados. Alguns dos exemplos mais famosos de RMAs são: o sistema desenvolvido pelo NavLab da CMU [Pomerleau 90, Batavia 96] que é capaz de conduzir uma caminhonete pelas estradas americanas; o robô do tipo *rover* enviado para Marte pela NASA [Stone 96]; o robô Dante que explora vulcões [Lemonick 94]; o sistema de controle de um veículo *Ligier* elétrico desenvolvido pelos pesquisadores do INRIA na França [Paromtchik 96, Scheuer 98]. Todos estes sistemas possuem em comum a capacidade de receber leituras de sensores que lhes dão informações sobre o ambiente em que estão inseridos e de modo semi ou completamente autônomo, geram os comandos que fazem com que eles se desloquem em um ambiente de modo seguro: sem se chocar contra obstáculos ou colocar em risco sua integridade ou a dos diferentes elemento presentes no ambiente.

A partir de estudos e trabalhos de pesquisa desenvolvidos pelo Grupo de Inteligência Artificial do PIPCA [WebGIA 01], foram criadas as bases para o desenvolvimento de aplicações na área de robótica autônoma móvel. Destaca-se particularmente o desenvolvimento dos projetos HMLT (Hybrid Machine Learning Tools) e COHBRA / HyCAR (Controle Híbrido Inteligente de Robôs Autônomos), que contam com a participação de alunos do Mestrado em Computação Aplicada da Unisinos.

Especificamente neste trabalho, optamos pela implementação de um sistema capaz de controlar um veículo autônomo similar a um carro (robô não-holonômico – forma retangular) equipado com um conjunto de seis sensores infra-vermelhos e/ou ultra-sônicos de baixo custo. Baseando-se nos modelos da cinemática de um veículo real [Garnier 97] e dos sensores [Michel 96] (modelo este usado em simuladores amplamente adotados pela comunidade científica internacional, como o Khepera-SIM), implementamos o nosso simulador. O simulador, denominado de SEVA – Simulador de Estacionamento de Veículos Autônomos, foi desenvolvido em Visual C++ de forma modular, onde podemos facilmente adaptar os modelos da cinemática do veículo, dos sensores e o sistema de controle dos atuadores

(aceleração e rotação da direção). Sendo assim, este simulador permite que sejam definidas diferentes configurações de carros, bem como de módulos de controle aplicados em diferentes tarefas. Neste artigo iremos tratar especificamente da simulação de um carro equipado com seis sensores infra-vermelhos (medidores de distância dos obstáculos), com um sistema de controle desenvolvido a fim de permitir que este carro localize uma vaga de estacionamento paralelo, realizando a seguir o estacionamento de modo automático (sem a intervenção de um condutor humano).

O controle do veículo autônomo pode ser realizado de duas formas distintas: baseado em um sistema especialista composto por um conjunto de regras heurísticas, ou, baseado em uma Rede Neural Artificial com aprendizado supervisionado. Ambos os sistemas de controle usam as informações provenientes dos sensores para definir o comportamento dos atuadores que irão controlar a direção do deslocamento, a velocidade do veículo e a rotação da direção.

O simulador permite a visualização, em uma janela gráfica, da trajetória e do comportamento do veículo durante o processo de estacionamento. Uma análise visual da evolução do comportamento do veículo controlado pelo sistema demonstra que o controlador realiza as tarefas de estacionamento com sucesso. Podemos também considerar nesta análise a taxa de respostas corretas no aprendizado neural, sendo da ordem de 98% em média, a qual demonstra que a rede é plenamente capaz de aprender a controlar corretamente o veículo. Estudos estão sendo realizados a fim de determinar de modo mais exato a precisão do sistema de controle, sua capacidade de generalização (Rede Neural), e a sua robustez, em relação a situações novas e a imprecisão dos seus sensores.

Nas seções seguintes vamos descrever o modelo de simulação adotado, e após descreveremos o controlador baseado em regras, seguido da descrição do funcionamento do controlador baseado na Rede Neural. Por fim, apresentaremos as perspectivas de melhorias que estão sendo estudadas a fim de serem implementadas no sistema.

2. Simulador SEVA

O simulador SEVA (Simulador de Estacionamento de Veículos Autônomos) possui os seguintes componentes principais:

- Modelo de simulação dos ***sensores***;
- Modelo de simulação da ***cinemática*** do veículo (deslocamento do carro);
- Comandos do atuador relacionado ao ***deslocamento*** (avançar / recuar e velocidade);
- Comandos do atuador relacionado ao ***giro*** do veículo (rotação da direção).

A figura 1 apresenta a interface gráfica do simulador SEVA, onde na parte superior encontramos uma zona de *status* que indica: o estados dos sensores, o estado da velocidade de deslocamento do veículo e o estado da rotação do veículo. Na janela principal também podemos visualizar uma representação simplificada do cenário hipotético no qual o veículo está inserido. Um desenho da trajetória do veículo foi sobreposto sobre a imagem da interface gráfica, permitindo visualizar o comportamento típico do veículo, obtido em modo de controle autônomo, sem interferência humana.

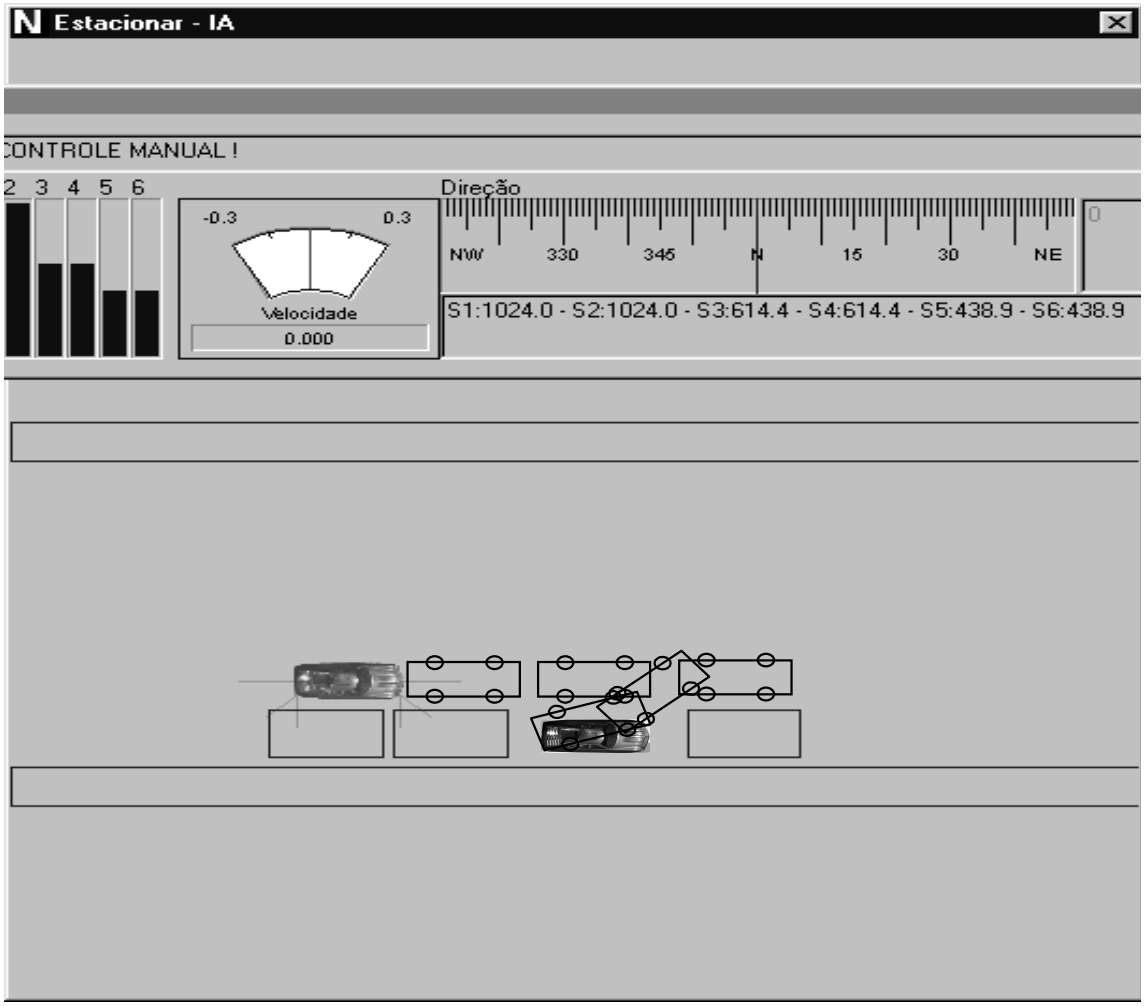

Figura 1 – Simulador SEVA

2.1. Modelo dos Sensores

Os sensores de distância simulam sensores infra-vermelhos, sendo capazes de determinar a distância entre o carro e os obstáculos presentes no ambiente: outro carros e a calçada. Os seis sensores utilizados estão distribuídos em pontos estratégicos do carro (vide figura 2), estando localizados na frente, na traseira e na lateral direita do carro. Foram implementados apenas os sensores da lateral direita do veículo, pois nossos estudos atuais se restringiram ao estacionamento em vagas paralelas localizadas no lado direito do carro, caso típico em pistas de duas vias.

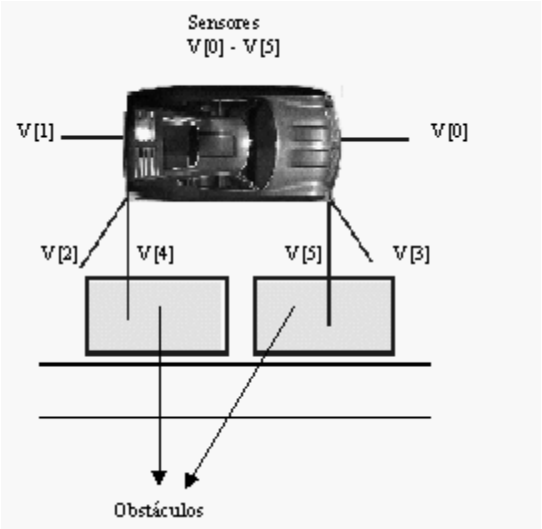

Figura 2 – Sensores do Veículo

Os sensores retornam um valor numérico entre 0 e 1024, onde o valor 0 indica que houve um contato com o objeto (colisão) e 1024 indica que não há obstáculos "visíveis" no campo de percepção do sensor (caminho livre). O simulador apresenta este campo de percepção representado graficamente por segmentos de reta, onde quando este segmento intercepta um objeto isto é convertido para um valor entre 0 e 1024 representando a distância relativa entre o sensor e o objeto interceptado. Os valores numéricos dos sensores também são apresentados na janela gráfica a fim de permitir um melhor acompanhamento das informações referentes a simulação do estacionamento.

2.2. Modelo da Cinemática do Veículo

A movimentação do veículo respeita o modelo da cinemática de um carro, conforme modelo também adotado em [Garnier 97]. Neste modelo é simulado um veículo representado

por um volume retangular suportado por quatro rodas, onde as rodas traseiras possuem um eixo fixo e as rodas dianteiras podem ser direcionadas, através do giro da barra da direção. As coordenadas do veículo são definidas por Vp = (X, Y, θ), onde ‘X’ e ‘Y’ definem o ponto médio do eixo traseiro do veículo e ‘θ’ indica a sua orientação (ângulo em relação a direção de referência). A velocidade do veículo é representada por ‘V’ e o ângulo de giro (rotação) da direção é denotado por ‘Φ’. O valor de ‘L’ indica o tamanho do eixo das rodas. O deslocamento do veículo é descrito pelas seguintes equações:

$$X = V * \cos(\Phi) * \cos(\theta)$$
$$Y = V * \cos(\Phi) * \cos(\theta) \qquad \text{Eq. 1}$$
$$\theta = V / L * \sin(\Phi)$$

A figura 3 apresenta um esquema do modelo da cinemática do veículo que foi adotado em nossas simulações.

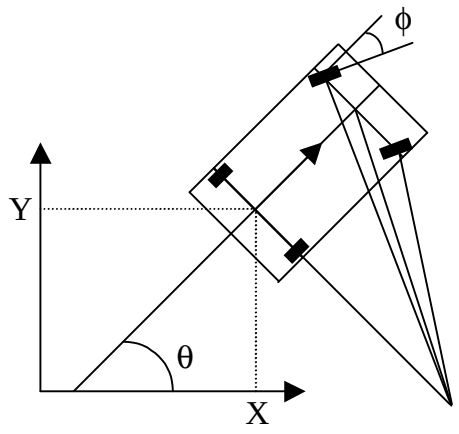

Figura 3 – Cinemática do veículo com eixo de rotação dianteiro

2.3. Atuador de Deslocamento

O deslocamento do veículo é obtido na simulação através do controle de sua velocidade ‘V’. Para que fosse possível controlar o veículo na tarefa de estacionamento, definiu-se cinco estados referentes a sua velocidade de deslocamento: Avanço_Rápido (AR), Avanço_Lento (AL), Parado (P), Ré_Lenta (RL), Ré_Rápida (RR). Cada um destes estados é associado a um valor numérico correspondente a velocidade real adotada pelo veículo.

2.4. Atuador de Rotação

O giro da direção, assim como a velocidade do veículo, é definido através de um conjunto de quatro valores adotados como ângulos de rotação: Giro_Esquerda_Max (GPM), Giro_ Esquerda_Pequeno (GEP) , Direção_Reta (DR) , Giro_Direita_Pequeno (GDP), Giro_ Direita_Max (GDM). Cada um destes estados é associado a um valor numérico que corresponde ao giro da direção adotado pelo veículo.

3. Controle baseado em um Sistema Especialista

O controle do estacionamento do carro foi implementado através da criação de um conjunto de regras que descrevem o comportamento de um motorista (especialista) ao realizar esta tarefa. O responsável pela criação das regras atuou como analista de conhecimentos, codificando a base de regras, e também como especialista do problema, visto que este era um condutor experiente, o que lhe possibilitou explicitar seus próprios conhecimentos a respeito do problema.

As regras explicitadas identificaram a necessidade de se definir um conjunto de estados pelos quais passa o condutor durante as diferentes etapas do processo de estacionamento. As etapas identificadas permitiram que fosse definido um autômato de estados finitos (FSA), composto por 7 estados, representados na figura 4: procurando_vaga, posicionando, entrando_vaga, posicionando_ vaga, otimizando_vaga, alinhando e parado.

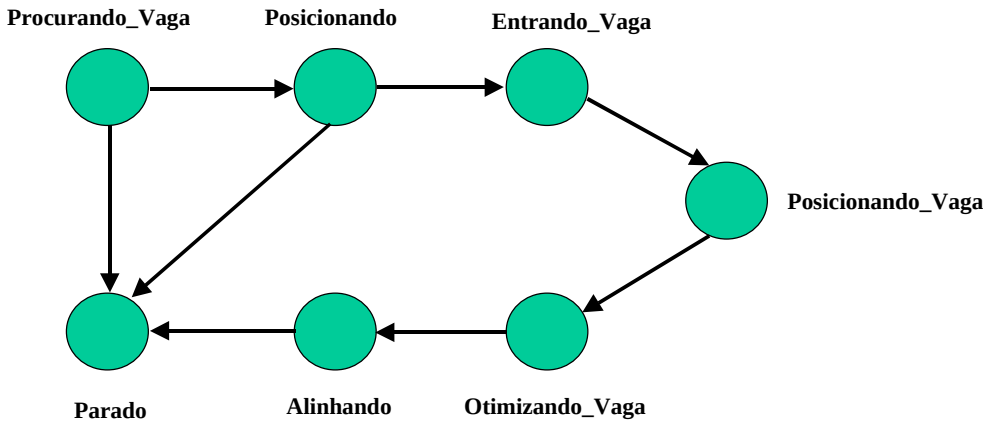

Figura 4 – Autômato com os estados usados no processo de estacionamento do carro

O uso de um diagrama de estados que definem o autômato permitiu que fosse simplificado o tratamento do problema, visto que o especialista reage de modo diferente aos estímulos sensoriais do ambiente, segundo o estado em que se encontra. Por exemplo, ao procurar uma vaga paralela estamos usualmente andando em linha reta e buscando localizar uma "abertura" entre dois carros, e quando vamos colocar o carro na vaga estamos usualmente dando uma ré, controlando os veículos ao nosso lado de modo a girar corretamente a direção e entrar na vaga sem bater neles.

Para que fosse possível realizar a tarefa de estacionamento, também foi adicionado um odômetro, que nos permite controlar o quanto o carro se deslocou. O odômetro é usado em casos onde, após acharmos uma vaga, é necessário primeiro avançar uma certa distância antes de entrar de ré na vaga. O comportamento final do veículo controlado pelo sistema especialista é aquele que foi apresentado na figura 1.

As regras usadas consultam o estado atual do autômato, dos sensores e do odômetro, e geram como efeito o controle dos atuadores de deslocamento (velocidade – controlada pela variável *Speed*) e de rotação (giro da direção – controlado pela variável *RotVel*). O sistema completo possui um total de 30 regras, onde listamos a seguir alguns exemplos destas regras empregadas no simulador:

```
Se Estado_Atual(Procurando_Vaga) e Próximo_ao_Obstáculo(V[4]) e
   Próximo_ao_Obstáculo(V[5])
Então Speed = Avanço_Rápido e RotVel =  Direção_Reta;
Se Estado_Atual(Procurando_Vaga) e Longe__do_Obstáculo(V[2]) e
   Longe__do_Obstáculo(V[3]) e Longe__do_Obstáculo(V[4]) e
   Longe__do_Obstáculo(V[5])
Então Troca_Estado(Posicionando) e Inicializa(Odômetro);
Se Estado_Atual(Posicionando)
Então Speed = Avanço_Rápido e Rotvel = Direção_Reta;
Se Estado_Atual(Posicionando) e Longe_do_Obstáculo(V[4]) e
   Deslocamento_Suficiente(Odômetro)
Então Estado_atual(Entrando_Vaga) e Inicializa(Odômetro);
Se Estado_Atual(Entrando_Vaga)
Então Speed = Ré_Rápida e RotVel = Giro_Esquerda_Max;
```

Quadro 1 – Exemplo das regras usadas no sistema de controle do veículo

4. Controle baseado em uma Rede Neural Artificial

Os resultados obtidos com o controle baseado em um sistema especialista nos levaram ao estudo de uma alternativa para a aquisição de conhecimentos, visto que o processo de codificação de regras é muito trabalhoso, além de não garantir uma grande robustez do sistema na presença de ruídos (e.g. valores inexatos dos sensores e situações não previstas). Buscou-se então a implementação de uma solução baseada no "aprendizado prático" a partir de uma base de exemplos de condução do veículo autônomo. A solução adotada foi a utilização de Redes Neurais Artificiais (RNA) [Braga 00, Osorio 99], com aprendizado supervisionado, capazes de aprender bases de exemplos que demonstram como se deve estacionar o carro.

O primeiro passo foi a construção das bases de exemplos empregadas no aprendizado da Rede Neural. Adaptou-se o simulador de modo a gerar um "arquivo de log", contendo o registro do estado dos sensores, estado do autômato e comandos enviados aos atuadores (velocidade e rotação). Este arquivo permite que sejam gravados todos os dados referente a execução de uma tarefa de estacionamento, seja esta tarefa controlada pelo sistema especialista, ou então, por um usuário que pode controlar o veículo através do teclado.

Em nossos experimentos iniciais com a RNA, geramos um arquivo de log que registrou o comportamento de controle do veículo, baseando-se no controle gerado pelo sistema especialista. Nosso objetivo foi o de realizar o aprendizado da rede neural de modo que ela pudesse substituir o controlador baseado em regras. Em uma etapa posterior pretendemos testar o aprendizado da rede usando um controle supervisionado baseado no comportamento de uma pessoa controlando o estacionamento do veículo.

O modelo de RNA adotado foi o MLP (*Multi-Layer Perceptron*) [Braga 00, Osorio 99], com aprendizado supervisionado do tipo *Cascade-Correlation* [Fahlman 90]. O algoritmo *Cascade-Correlation* foi empregado no lugar do tradicional algoritmo de *Back-Propagation* [Rumelhart 86] devido ao fato deste outro algoritmo simplificar a definição da rede (não precisamos indicar o número de neurônios da camada oculta) e por ser um algoritmo comprovadamente mais eficiente que o Back-Propagation [Joost 93, Fahlman 90].

As variáveis (atributos) de entrada da rede que empregamos foram: o estado dos seis sensores e uma indicação do estado atual do processo de estacionamento (um dos sete estados possíveis: procurando_vaga, posicionando, etc). Na saída da rede iremos obter o estado dos atuadores (velocidade e rotação), assim como uma indicação do próximo estado do processo

de estacionamento. Deste modo, a rede irá controlar os atuadores, mas também irá decidir quando devemos passar de um estado a outro no autômato de controle. A figura 5 apresenta o esquema das entradas e saídas da RNA adotada. Note que o estado previsto pela rede pode ser re-inserido nela como sendo o estado atual, sendo então usado no instante de tempo seguinte.

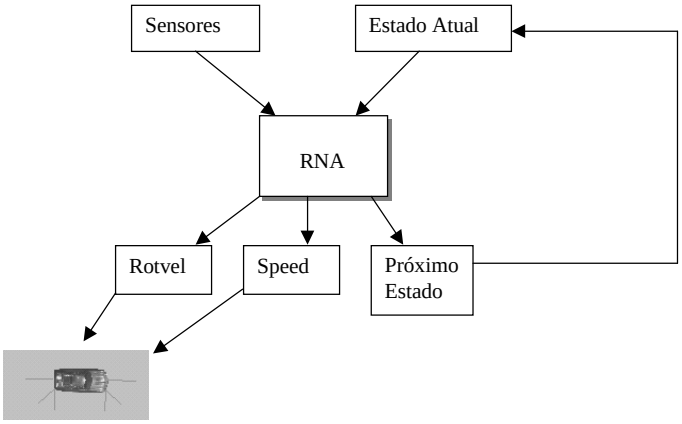

Figura 5 – Descrição da Rede Neural Artificial utilizada para controlar o veículo

Salientamos a importância de se conhecer o estado atual, para podermos interpretar corretamente os sensores, por exemplo:

- Quando estamos procurando uma vaga, temos os sensores laterais indicando a proximidade de um carro, onde isto indica que devemos continuar a avançar até encontrar uma vaga, quando então os sensores laterais irão indicar que o obstáculo não está mais tão próximo;
- Quando estamos entrando na vaga, temos os sensores laterais indicando a proximidade de um carro (como na situação anterior), mas isto indica que devemos engatar uma ré para colocar o carro na vaga previamente localizada.

Apesar de precisarmos de uma indicação do estado do processo de estacionamento em que nos encontramos, a rede neural é capaz de aprender a detectar a passagem de um estado a outro, e uma vez terminada a fase de aprendizado, ela será capaz de realizar o estacionamento de forma autônoma sem a intervenção humana.

A figura 6 apresenta uma amostra da base de aprendizado utilizada para treinar a rede neural, onde foi usada uma codificação binária do tipo “1 entre N” para representar: o estado atual (7 estados possíveis), a velocidade (5 velocidades possíveis), o giro da direção (onde usamos apenas 4 rotações possíveis) e o próximo estado (que também codifica 7 estados possíveis).

13 16 392 0

Valsens[0]	Valsens[1]	Valsens[2]	Valsens[3]	Valsens[4]	Valsens[5]	Estado Inicial	Speed	Rotvel	Prox. Estado
1024.000000	1024.000000	614.399536	614.399536	438.857056	438.857056	0 0 0 0 1 0 0	0 0 0 1 0	0 1 0 0	0 0 0 0 0 0 1
1024.000000	1024.000000	614.399536	614.399536	438.857056	438.857056	0 0 0 0 0 0 1	0 1 0 0 0	0 0 0 1	0 0 0 0 0 0 1
1024.000000	1024.000000	614.399536	614.399597	438.857056	438.857056	0 0 0 0 0 0 1	0 1 0 0 0	0 0 0 1	0 0 0 0 0 0 1
1024.000000	1024.000000	614.399536	614.399597	1024.000000	438.857056	0 0 0 0 0 0 1	0 1 0 0 0	0 0 0 1	0 0 0 0 0 0 1
1024.000000	1024.000000	614.399536	1024.000000	1024.000000	438.857056	0 0 0 0 0 0 1	0 1 0 0 0	0 0 0 1	0 0 0 0 0 0 1
1024.000000	1024.000000	614.399536	1024.000000	438.857056	438.857056	0 0 0 0 0 0 1	0 1 0 0 0	0 0 0 1	0 0 0 0 0 0 1
1024.000000	1024.000000	716.796997	1024.000000	438.857056	1024.000000	0 0 0 0 0 0 1	0 1 0 0 0	0 0 0 1	0 0 0 0 0 0 1
1024.000000	1024.000000	853.330139	1024.000000	438.857056	1024.000000	0 0 0 0 0 0 1	0 1 0 0 0	0 0 0 1	0 0 0 0 0 0 1
1024.000000	1024.000000	614.399658	1024.000000	438.857056	1024.000000	0 0 0 0 0 0 1	0 1 0 0 0	0 0 0 1	0 0 0 0 0 0 1
1024.000000	1024.000000	614.399658	1024.000000	438.857056	1024.000000	0 0 0 0 0 0 1	0 1 0 0 0	0 0 0 1	0 0 0 0 0 0 1
1024.000000	1024.000000	614.399536	1024.000000	438.857056	1024.000000	0 0 0 0 0 0 1	0 1 0 0 0	0 0 0 1	0 0 0 0 0 0 1
1024.000000	1024.000000	614.399658	1024.000000	438.857056	1024.000000	0 0 0 0 0 0 1	0 1 0 0 0	0 0 0 1	0 0 0 0 0 0 1
1024.000000	1024.000000	614.399536	1024.000000	438.857056	1024.000000	0 0 0 0 0 0 1	0 1 0 0 0	0 0 0 1	0 0 0 0 0 0 1
1024.000000	1024.000000	614.399536	1024.000000	438.857056	1024.000000	0 0 0 0 0 0 1	0 1 0 0 0	0 0 0 1	0 0 0 0 0 0 1
1024.000000	1024.000000	614.399658	1024.000000	438.857056	1024.000000	0 0 0 0 0 0 1	0 1 0 0 0	0 0 0 1	0 0 0 0 0 0 1
1024.000000	1024.000000	614.399536	1024.000000	438.857056	1024.000000	0 0 0 0 0 0 1	0 1 0 0 0	0 0 0 1	0 0 0 0 0 0 1
1024.000000	1024.000000	614.399597	1024.000000	438.857056	1024.000000	0 0 0 0 0 0 1	0 1 0 0 0	0 0 0 1	0 0 0 0 0 0 1

Figura 6 – Arquivo usado para o aprendizado supervisionado da Rede Neural

Os resultados das simulações do aprendizado da RNA, registrados em um conjunto de 10 simulações que realizamos, foram os seguintes: obtivemos uma taxa de aprendizado média com um acerto de 98.649% nas respostas da rede; foram usadas (inseridas) apenas 2 unidades na camada intermediária (adicionadas pelo algoritmo Cascade-Correlation) e o número médio de épocas de aprendizado foi de 572 épocas. Como foi usada uma base de 392 exemplos, notamos que apenas cerca de 6 exemplos não foram corretamente aprendidos pela rede.

5. Conclusão e Perspectivas

Este trabalho teve como objetivo o desenvolvimento de um simulador para o controle de veículos autônomos em uma tarefa de estacionamento em vagas paralelas. Os resultados obtidos nas simulações realizadas com a ferramenta que desenvolvemos, o SEVA, demonstraram que o sistema de controle, seja ele implementado através de um sistema especialista ou através de uma Rede Neural Artificial, possui a capacidade de controlar corretamente o veículo, cumprindo seu objetivo principal: estacionar o veículo na vaga, sem bater nos demais obstáculos que estão ao seu redor.

Atualmente estamos estudando a possibilidade de integrar os conhecimentos representados sob a forma de regras e os conhecimentos representados nas bases de exemplos, de modo a permitir um melhor aproveitamento de toda a “experiência” disponível em relação ao problema. No futuro pretendemos aplicar técnicas de aprendizado híbrido e sistemas híbridos [Osorio 98 e 99b] como forma de tornar o controle do veículo ainda mais robusto.

Agradecimentos: Agradecemos a Fapergs, ao CNPq e a Unisinos pelo financiamento de nossas pesquisas e dos bolsistas envolvidos no desenvolvimento destes trabalhos.

Bibliografia

[Batavia 96] Batavia, Parag; Pomerleau, Dean & Thorpe, Charles. Applying Advanced Learning Algorithms to ALVINN. CMU Technical Report CMU-RI-TR-96-31. Carnegie Mellon University. Pittsburgh. 1996.

[Braga 00] Braga, Antônio de Pádua; Ludermir, Tereza Bernarda; Carvalho, André Carlos Ponce de Leon Ferreira. Redes Neurais Artificiais, Teoria e Aplicações; LTC Editora, Rio de Janeiro. 2000.

[Fahlman 90] Fahlman, Scott E. & Lebiere, Christian. The Cascade-Correlation Learning Architecture. Carnegie-Mellon University – CMU, Computer Science Techincal Report. CMU-CS-90-100. 1990.

[Garnier 97] Garnier, Philippe; Fraichard, Thierry; Laugier, Christian; Paromtchik, I. and Scheuer. Motion Autonomy Through Sensor-Guided Manoeuvres. IEEE-RSJ International Conference on Intelligent Robots and Systems, Procedings of the Intelligent Cars and Automated Highway Systems Workshop. Sept. 1999. Grenoble, France.

[Heinen 99] Heinen, Farlei José. Robótica Autônoma: Integraçào entre Planificação e Comportamento Reativo; UNISINOS Editora, São Leopoldo, Novembro, 1999.

[Joost 93] Joost M, W.Schiffmann, R. Werner. Comparison of Optimized Backpropagtion Algoritms. Presented at ESANN 93, Brüssel.

[Lemonick 94] Lemonick, Michel. Dante Tours the Inferno. Time Magazine – Time Domestic/Science. Vol. 144, No. 7. August 15, 1994.

[Medeiros 98] Medeiros, Adelardo. Introdução a Robótica. ENA-98 Encontro Nacional de Automática (50° Congresso da SBPC). Natal, RN. pp.56-65. 1998.

[Michel 96] Michel, Olivier. Khepera Simularor Version 2.0 – User Manual. Université de Nice – Sophia Antipolis. Laboratoire I3S - CNRS, France / EPFL – Lausanne, Swiss. March 1996.

[Osório 98] Osório, Fernando Santos. *INSS: Un Systemè Hybride Neuro-Symbolique pour l'Apprentissage Automatique Constructif.* Thèse de Doctorat (Ph.D) en Informatique. Laboratorie Leibniz - IMAG / INPG. Grenoble, France, 1998.

[Osório 99] Osório, Fernando S. Redes Neurais Artificiais: Do aprendizado Natural ao Aprendizado Artificial. Tutorial – I Fórum de Inteligência Artificial / Ulbra; Canoas, Agosto, 1999.

[Osório 99a] Osório, Fernando S. & Vieira, Renata. Sistemas Híbridos Inteligentes. Tutorial apresentado no ENIA'99 – Encontro Nacional de Inteligência Artificial, Congresso da SBC, Rio de Janeiro 1999.

[Osório 99b] Osório, Fernando S & Amy, Bernard. INSS: A hybrid system for constructive machine learning. Neurocomputing, Elsevier Press. Netherlands. 1999.

[Paromtchik 96] Paromtchik, I. E. & Laugier, C. Autonomous Parallel Parking of a Nonholonomic Vehicle. Procedings of the IEEE International Symposium on Intelligent Vehicles., pp. 13-18. September, 1996.

[Pomerleau 90] Pomerleau, D. Neural network based autonomous navigation. In: Thorpe, C.(Ed). Vision and Navigation: The CMU Navlab. Kluwer Academic Publihers, 1990.

[Rumelhart 86] Rumelhart, D.; Hinton, G; Williams, R. L. Learning Internal Representations by Error Propagation. In: Parallel Distributed Processing – Vol. 1. MIT Press, Cambridge. 1986.

[Scheuer 98] Sheuer, A. & Laugier, C. Planning Sub-Optimal and Continuous-Curvature Paths for Car-Like Robots. IEEE-RSJ International Conference on Intelligent Robots and Systems. Victoria, British-Columbia, Canada. Oct. 1998.

[Stone 96] Stone, H. W. Mars Pathfinder Microver: A low-cost, low-power Spacecraft. Proceeding of the 1996 AIAA. Forum on advanced developments in Space Robotics. Madison, WI. August 1996.

[WebGIA 01] Grupo de Inteligência Artificial da Unisinos. URL: http://inf.unisinos.br /~osorio/gia.html (Jul. 2001).