O Sistema Integrado do Time de Futebol de Robôs USPDroids

Marcelo O. Silva*, Diogo O. Melo†, Ben Hur Gonçalves‡, Ralph C. R. Silva§
Roseli A. F. Romero††
*mcl.os@grad.icmc.usp.br, †xyxper@grad.icmc.usp.br, ‡benhur@usp.br, §ralph.silva@usp.br
††rafrance@icmc.usp.br
LAR, ICMC, USP-São Carlos
Av. Trabalhador São-Carlense, 400
13566-590 - Caixa Postal 359
São Carlos - SP - Brasil

***Resumo*— Este trabalho descreve o sistema integrado de controle do time de futebol de robôs USPDroids. Este sistema integra um sistema de visão baseado em cores que utiliza a técnica de "Blob Coloring" juntamente com um controlador PID baseado na posição dos robôs obtido no sistema de visão. O artigo também apresenta um sistema de estratégia no qual foi implementado uma modificação da técnica de campo potencial baseado em Problemas de Valor de Contorno.**

I. INTRODUÇÃO

Disputas de futebol de robôs podem ser utilizadas como ambientes para testes de diversas técnicas de Inteligência Artificial, Visão Computacional, Controle e outras áreas. Por este motivo, desenvolveu-se um time de futebol de robôs no ICMC-USP para a criação de uma plataforma de aprendizado e teste de diversas técnicas em um ambiente real e dinâmico. O time recebeu o nome de *USPDroids* e, foi projetado para as competições da categoria *IEEE Very-Small Soccer*.

O time é composto por cinco módulos (Figura 1):

- **Visão** Módulo responsável pela localização dos robôs e classificação entre robôs *aliados* e robôs *inimigos*;
- **Estratégia** Módulo responsável pela Inteligência Artificial do time, que irá controlar autonomamente os robôs;
- **Controle** Módulo que irá interpretar os dados enviados pela estratégia e tratá-los de modo que os resultados sejam informações para a Eletrônica/Mecânica;
- **Eletrônica** Módulo que irá enviar dados do computador para o robô, recebê-los e controlar os motores;
- **Mecânica** Módulo que constitui a parte física do robô (carcaça, rodas, engrenagens e etc).

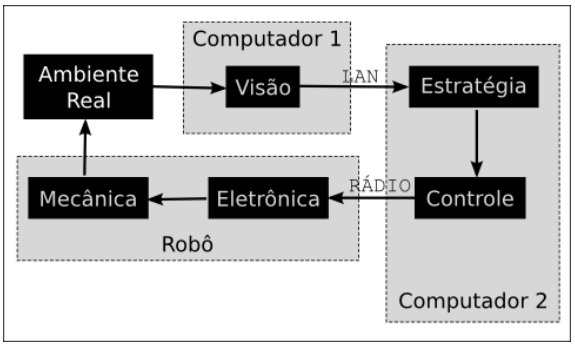

Fig. 1. Interligação entre os módulos

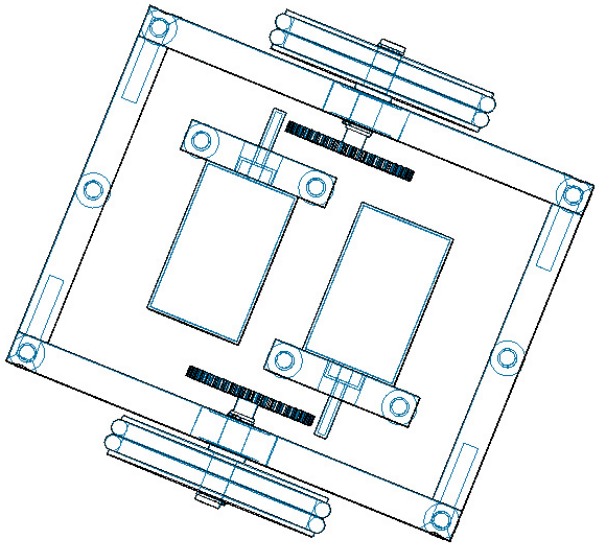

Fig. 2. Visão superior da mecânica do robô.

II. ROBÔS

Os robôs foram construídos, seguindo as normas da IEEE, em alumínio com espessura de 5mm. Tais robôs são construídos com a arquitetura de acionamento diferencial, na qual seus movimentos são definidos pela relação entre as velocidades de cada roda. Por exemplo, se a roda direita girar com uma velocidade superior a da esquerda (ambas em mesmo sentido), então o robô irá fazer uma curva à esquerda e o contrário também é válido. Nas Figuras 2 e 3 é mostrada uma visão geral da mecânica.

Foram utilizados dois motores *PM080-SS05* da ActionMotors [Mot], os quais são alimentados por corrente contínua. As especificações deles podem ser observadas na Tabela II

Também foram utilizadas caixas de redução, para aumentar o torque do motor. Estas características estão descritas na Tabela II.

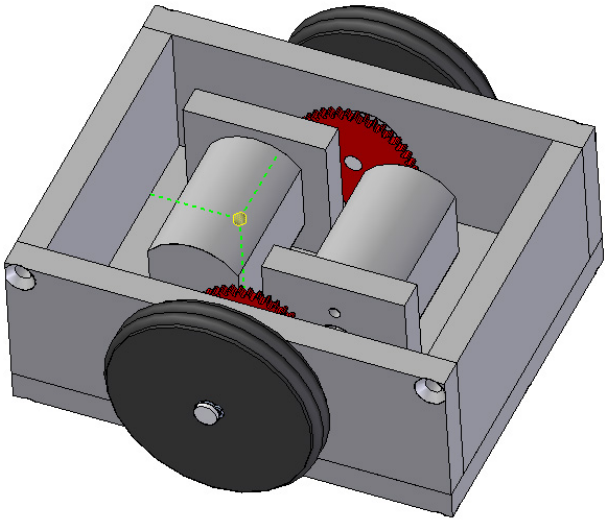

Fig. 3. Visão isométrica da mecânica do robô.

Voltagem	Torque	RPM	Potência	Rendimento
5V	14g.cm	5450	0,77W	55,4

TAB I
ESPECIFICAÇÕES DOS MOTORES

III. ELETRÔNICA

Neste módulo, estão presentes os seguintes componentes: rádio, microprocessador, duas baterias, motores, ponte-H, dentre outros. A Figura 4 detalha o esquema eletrônico dos robôs.

Foram utilizadas baterias diferentes para o PIC e para o restante do circuito. Este modo de alimentação foi escolhido para evitar que o consumo do motor reinicie o PIC toda vez que tente sair da inércia. Isto acontece porque um motor de corrente contínua precisa de muita potência para sair do estado de inércia, o que implica em maior consumo de corrente elétrica.

Como os dois motores utilizados são de corrente contínua, foi utilizada uma ponte-H (LM 298) para que os motores possam funcionar nos dois sentidos de rotação. Também foram utilizados reguladores de tensão *7805* e *78L05* para que as tensões sejam ajustadas em 5V, pois as baterias utilizadas fornecem tensões superiores às desejadas.

O módulo da eletrônica se comunica com dois outros: *Mecânica* e *Controle*. O controle transmite, via rádio, o vetor ***$A,B,C,D,E,F**** para o rádio da eletrônica. O rádio passa o comando para o PIC através da interface *RS-232* de comunicação serial. O microprocessador começa a ler o sinal ***$*** e percebe que está chegando uma nova mensagem. Então, ele irá verificar se ***A*** é o seu ID, portanto, se deve executar a ação descrita no resto da mensagem. Também irá associar ***B*** ao sentido da roda esquerda, ***C*** à velocidade desejada para a mesma, ***D*** ao sentido da roda direita, ***E*** à velocidade desejada para a mesma e ***F*** é um *check-sum*, para verificar se a mensagem não está corrompida. O caracter * indica final de mensagem.

Redução	Roda	Torque Final	RPM Final
1/4	50mm	204 g.cm	1362

TAB II
ESPECIFICAÇÕES DA CAIXA DE REDUÇÃO

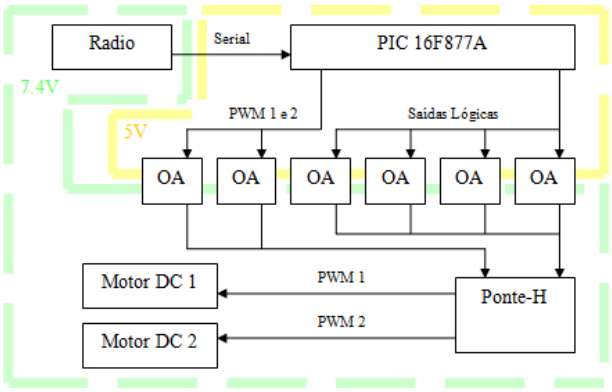

Fig. 4. Esquema do Hardware Robótico

IV. CONTROLE

Foi programado um controle PID da posição do robô. Essa proposta pode ser analisada em Vieira [VMAAJ04], que propôs um mecanismo de controle para robôs de duas rodas. Esse controle é baseado na Figura 10, e consiste na teoria de que se o erro angular α e o erro linear forem minimizados, então o robô terá se deslocado do ponto (x, y) para o ponto (x_{ref}, y_{ref}), como na figura IV.

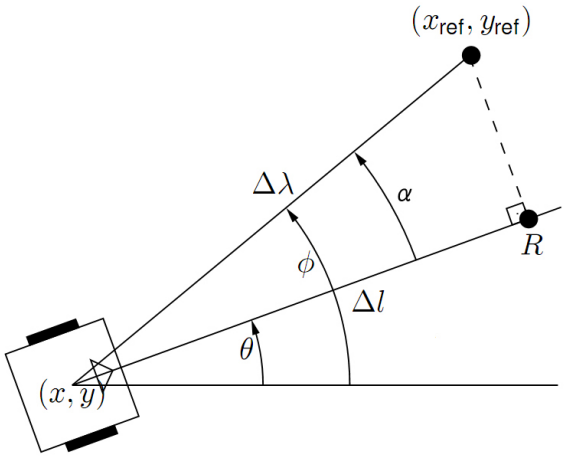

Fig. 5. Referenciais do Controle. [VMAAJ04]

O controle de posição foi programado apenas de maneira não recursiva, com intervalo das medições de $33ms$. Embora as abordagens recursivas e não recursivas sejam diferentes, elas geram resultados muito parecidos [Oga]. Foi escolhido pela não recursiva, portanto, devido à sua maior clareza.

Não foi necessário usar filtro de dados, já que a detecção da posição dos robôs é uma informação razoavelmente precisa.

No entanto a estratégia teve que ser adaptada, de forma a gerar esse ponto de referência, uma vez que os Campos Potenciais trabalham com velocidades. A solução encontrada foi determinar para quais pontos o vetor da estratégia aponta e qual deles o robô atingiria em um segundo. Ao contrário do controle por velocidade, no qual o erro pode ser muito significativo para valores baixos de velocidade, no controle por posição o erro varia pouco em função do número de pixels que o campo ocupa na imagem da visão. O resultado da convergência do controle está ilustrado na Figura 6.

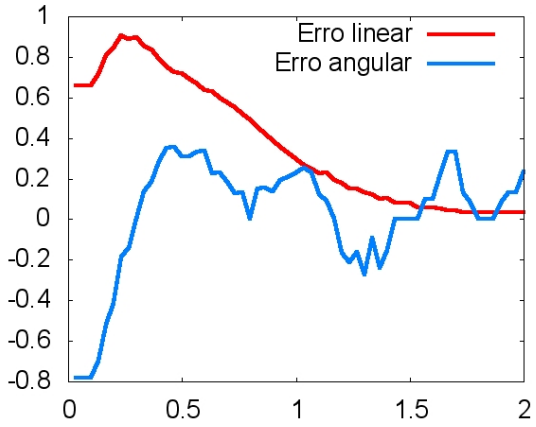

Fig. 6. Convergência do Controle.

V. ESTRATÉGIA

A estratégia utilizada é a de Campos Potenciais Localmente Orientados (CPLO). Esta estratéiga foi proposta por Faria [referência]. O CPLO envolve Campos Potenciais Harmônicos (CPH) e Campos Potenciais Orientados (CPO). Os quais serão detalhados adiante.

A. Campos Potenciais Harmônicos

As funções harmônicas foram primeiramente utilizadas para controle de robôs por Connolly [Con92]. De acordo com Connolly, uma função $p : \Omega \rightarrow R$, com $\Omega \subset R^n$, é uma função Harmônica, se a Equação 1 for setisfeita.

$$\nabla^2 p = \sum_{i=1}^{n} \frac{\partial^2 p}{\partial x_i^2} = 0 \quad (1)$$

Embora os robôs sejam tridimensionais, eles se movem em um plano. Dessa forma $\Omega \subset R^2$. A Equação 1 pode ser reescrita na forma da Equação 2.

$$\frac{\partial^2 p}{\partial x^2} + \frac{\partial^2 p}{\partial y^2} = 0 \quad (2)$$

Conforme descrito por Faria [Far06], para resolver equações diferenciais parciais existem métodos numéricos como Gauss-Seidel (GS), Sucessiva Sobre-Relaxações (SOR *Successive Over Relaxion*) e o método de Jacobi. Aplicar um método de relaxação em uma malha resolverá o sistema numericamente.

1) Método de Jacobi: O método de Jacobi é extremamente parecido com o método de Gauss-seidel. Matematicamente, a diferença entre os dois métodos está na equação de relaxação. A Equação 3 mostra a relaxação para o método de Jacobi. É possível observar que, para executar esté método serão necessárias duas matrizes. A primeira guarda a malha no estado atual e a segunda é utilizada para guardar o resultado da iteração seguinte.

$$p_{i,j}^{t+1} = \frac{1}{4} \times (p_{i-1,j}^{t} + p_{i,j-1}^{t} + p_{i,j+1}^{t} + p_{i+1,j}^{t}) \quad (3)$$

A implementação paralela foi feita através da biblioteca OpenMP [ope]. Com o intuito de obter o melhor desempenho possível é necessário levar em consideração as características da arquitetura do computador utilizado. O processador é um Intel Core 2 Quad modelo Q6600 2.4GHz com 8MB de memória Cache.

Cada célula da malha ocupa 28 bytes. Isso faz com que toda a malha necessite de aproximadamente 150KB. Logo, o grid pode ser facilmente alocado na memória cache, o que provê um considerável ganho de desempenho.

A biblioteca OpenMP é composta por uma interface que permite gerenciar o paralelismo da aplicação de modo simples. Após implementar a versão paralela da relaxação, utilizando o método de Jacobi, o tempo de processamento necesário por iteração desceu para $0,45ms$ aproximadamente. Este tempo consiste em um speedup absoluto de $3,6$ e desempenho de 0.9. Mais informações sobre o cálculo de speedup e rendimento de uma aplicação paralela podem ser encontradas em [AGK03].

Utilizando os resultados obtidos é possível manipular um CPH com o custo computacional de $4,5ms$, a cada quadro. O tempo de $4,5ms$ é considerado baixo, entretanto é necessário alocar um CPH para cada jogador, o que resultaria em $13,5ms$ por quadro, que é um tempo alto. Uma vez que a visão computacional consome $20ms$, a estratégia e o controle possuem, no máximo, $13ms$ para serem executados.

B. Campo Potencial Orientado

A técnica CPO foi proposta por Prestes em [Pre03]. Está técnica consiste em adicionar um vetor de influência sobre a malha com o intuito de direcionar o movimento do robô. As Figuras 7, 8, 9 e 10 ilustram a influência provocada por diferentes vetores. É possível observar que o vetor de influência força o caminho alcançar a meta com um ângulo próximo ao ângulo do vetor de influência.

A diferença matemática entre CPH e CPO está na equação do relaxamento das células. A Equação 4 representa o relaxamento das células no CPO, utilizando o método de Jacobi. O CPO adiciona um termo a equação do CPH, este termo é responsável por produzir o efeito de orientação na malha.

$$p_{i,j}^{t+1} = \frac{1}{4}(p_{i-1,j}^t + p_{i,j-1}^t + p_{i,j+1}^t + p_{i+1,j}^t) + \frac{1}{8}[(p_{i+1,j}^t - p_{i-1,j}^t)vx + (p^t i, j+1 - p^t i, j-1 vy)] \quad (4)$$

O CPO é muito vantajoso, por exemplo, para fazer com que o jogador atacante direcione o seu ataque para o gol adversário ou também para impedir que o goleiro abandone a área do gol. Entretanto, o CPO só consegue mapear um tipo de comportamento por malha. Ainda com o CPO é necessário utilizar três malhas para o controle dos três jogadores. O CPLO provê uma solução para controle de multiplos robôs em uma única malha.

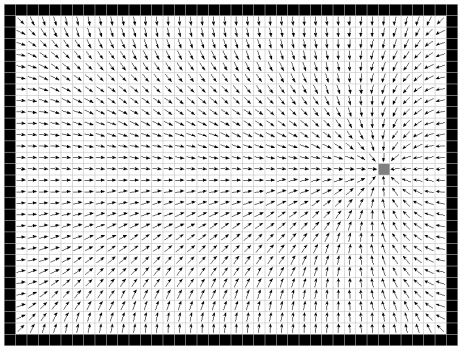

Fig. 7. CPH (sem influência do vetor)

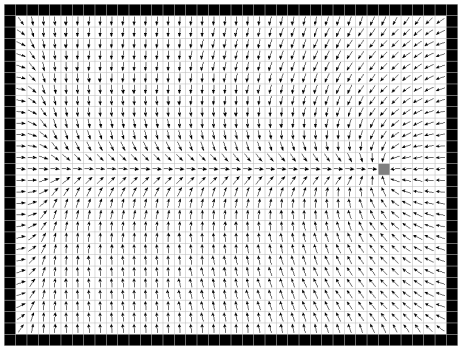

Fig. 8. CPO com vetor $v = (1, 0)$

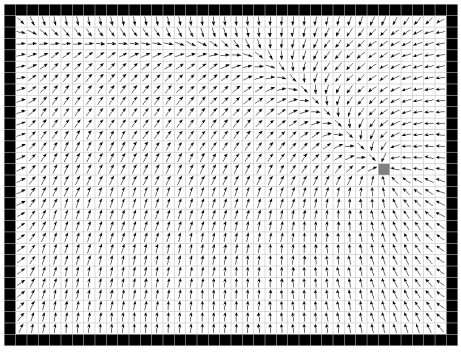

Fig. 9. CPO com vetor $v = (1, 1)$

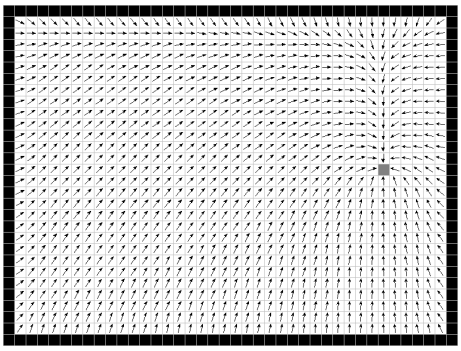

Fig. 10. CPO com vetor $v = (0, 1)$

C. Campo Potencial Localmente Orientado

O método de Campo Potencial Localmente Orientado (ou CPLO), proposta em [FRPI4a] e [FPIR06], é uma modificação do método CPO (em última instância, uma modificação do método CPH) que apresenta uma solução de baixo custo computacional para o controle de múltiplos robôs. O objetivo desta técnica é gerar vários comportamentos (direções do campo gradiente) em um mesmo grid. Desta maneira, vários robôs podem seguir diferentes trajetórias através de diferentes vetores de orientação v_k (onde k indica qual robô está sendo tratado)

O CPLO é basicamente uma junção entre CPH e múltiplos CPOs. A cada robô é associado um vetor e duas constantes λ e ϵ. O algoritmo para relaxar a malha consiste em percorrer todos as células da malha. Para cada célula encontra-se o robô mais próximo. Se a distância até o robô mais próximo for maior do que λ, é utilizada a Equação 2, caso contrário, utilização a Equação 4.

Isto é obtido aplicando a equação 4 somente na região de influência[1] de cada robô e, nas outras regiões, a Equação 2 é utilizada.

Como visto em [SRC08], o método CPLO é formalmente descrito como:

$$\nabla^2 P(x,y) + \epsilon(x,y) \langle \nabla P(x,y), v(x,y) \rangle = 0 \quad (5)$$

onde

$$\epsilon(x,y) = \begin{cases} \epsilon_k & \text{com } \epsilon_k \in \Re, \text{ caso } (x,y) \text{ esteja} \\ & \text{na região de influẽncia do robô } k \\ 0 & \text{caso contrário} \end{cases}$$

e,

$$v(x,y) = \begin{cases} v_k & \text{com } v_k \in \Re^2, \text{ e } ||v_k|| = 1, \text{ caso}(x,y) \\ & \text{esteja na região de influẽncia do robô } k \\ 0 & \text{caso contrário} \end{cases}$$

Os métodos de solução do CPLO são os mesmos apresentados nas sub-seções V-A e V-B.

[1] é assumido que a região de influência é um círculo de raio fixo, com centro na posição do robô

VI. VISÃO

O sistema de visão tem como função a localização de objetos coloridos através da análise de imagens obtidas por uma câmera posicionada perpendicular ao chão. O robô é identificado por marcas coloridas, em sua parte superior, para que seja diferenciado dos demais objetos presentes na cena. Para detectar o robô, foi desenvolvido um sistema de visão computacional baseado em classificação de cores.

A classificação é necessária, pois um objeto de uma mesma cor possui diversos pixels. Ao olharmos um objeto colorido em uma imagem, assumimos que todos os pixels que o formam são coloridos com a mesma cor, porém computacionalmente, esta cor recebe diferentes valores. O sistema visual humano, ao observar a imagem, é capaz de atribuir a mesma cor mesmo com diferentes valores descrevendo a cor. Entretanto, o computador não é capaz disto e para que isto seja realizado é necessário empregar um classificador de cores. Este classificador recebe um pixel colorido e determina a classe a qual ele pertence.

Conforme descrito por Penharbel [Pen08], foi desenvolvido um upgrade no sistema de visão, que consiste numa rede neural que ajusta esse sistema, detectando as cores mais presentes na imagem e que classifica os objetos em robôs amigos, inimigos ou bola de jogo. Esse sistema é responsável pela detecção de objetos em tempo real em menos de 5,2 milissegundos (quantização e classificação). Segundo exibido na Figura 11, o processo responsável pelo desempenho ótimo do sistema de visão é a construção de uma lookup table, baseada nessa rede neural. Essa tabela mapeia as cores em função dos objetos, e sua construção é realizada num estágio off-line. O estágio on-line se aproveita dessa tabela, apenas acessando o objeto que cada cor corresponde, o que faz o sistema de classificação ser muito rápido.

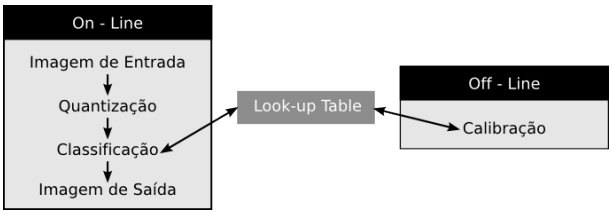

Fig. 11. Diagrama do sistema (on-line e off-line) da visão computacional.

Após classificar os pixels da imagem, é necessário agrupá-los conforme sua classe para que objetos sejam determinados de acordo com os pixels que os compõem. Para tal tarefa, é empregado um algoritmo de segmentação baseado em crescimento de regiões chamado *Blob Coloring*. Este algoritmo é capaz de localizar os centros dos agrupamentos coloridos.

Maiores informações sobre o sistema podem ser encontradas em [Pen08].

VII. CONCLUSÃO

No presente trabalho foi apresentado um sistema integrado que controla o time de futebol de robôs USPDroids. Este time pertence à categoria *Very Small Size*.

Foram descritos os módulos que compõem o sistema bem como os experimentos realizados em ambientes de simulação.

REFERENCES

[AGK03] George Karypis Ananth Grama, Anshul Gupta and Vipin Kumar. *Introduction to Parallel Computing*. Addison Wesley, 2003.

[Con92] Christopher I. Connolly. Aplication of harmonic functions to robotics. *Proceedings of the 1992 IEEE International Symposium on*, 1992.

[Far06] G. Faria. *Uma arquitetura de controle inteligente para múltiplos robôs*. PhD thesis, Instituto de Ciências Matemáticas e de Computação - USP, 2006.

[FPIR06] Gedson Faria, Edson Prestes, Marco A. P. Idiart, and Roseli Aparecida Francelin Romero. Multi robot system based on boundary value problems. In *International Conference on Intelligent Robots and Systems (IROS)*, 2006.

[FRPI4a] G. Faria, R. A. F. Romero, E. Prestes, and M. A. P. Idiart. Comparing harmonic functions and potential fields in the trajectory control of mobile robots. *IEEE Conference on Robotics, Automation and Mechatronics, Singapore*, 2004a.

[Mot] Action Motors.

[Oga] K. Ogata. *Engenharia de Controle Moderno*. Ed. Pearson do Brasil.

[ope] http://www.openmp.org.

[Pen08] Eder Augusto Penharbel. Desenvolvimento de um sistema de classificação de cores em tempo real para aplicações robóticas, 2008.

[Pre03] Edson Prestes. *Navegação Exploratória Baseada em Problemas de Valores de Contorno*. PhD thesis, Universidade Federal do Rio Grande do Sul (UFRGS), Porto Alegre, RS, 2003.

[SRC08] Marcelo O. Silva, Roseli A.P. Romero, and Glauco A. P. Caurin. Analysis of locally oriented potential fields. In *IROBOT*, 2008.

[VMAAJ04] F. C. VIEIRA, A. A. D. MEDEIROS, P. J. ALSINA, and A. P. ARAÚJO JR. Position and orientation control of a two-wheeled differentially driven nonholonomic mobile robot. *International Conference on Informatics in Control, Automation and Robotics, 2004, Setúbal. Portugal. Proceeding of ICINCO 2004, 2004*, 2004.