CONSTRUINDO ROBÔS AUTÔNOMOS PARA PARTIDAS DE FUTEBOL: O TIME GUARANÁ

Anna Helena Reali Costa
LTI-PCS-EPUSP
Av. Prof. Luciano Gualberto, 158 Travessa3
CEP 05508-900 – São Paulo SP

Renê Pegoraro
DCo-UNESP
Rua Luiz Edmundo Carrijo Coube S/N
CEP 17033-360 – Bauru SP

Resumo Futebol de robôs tem sido adotado internacionalmente como um problema padrão, uma vez que possibilita a avaliação de várias teorias, algoritmos, arquiteturas e desempenhos, onde uma grande variedade de tecnologias podem ser integradas e analisadas, propiciando desenvolvimentos ligados à Inteligência Artificial e Robótica Inteligente. Este artigo descreve nossa experiência na construção de time de futebol de robôs, focando especialmente nos algoritmos usados na implementação do software do time GUARANÁ, um dos representantes do Brasil nas competições da FIRA 98 - Federation of International Robot-soccer Association.

Inicialmente apresentamos o sistema de visão, composto por uma fase preliminar de calibração e uma fase de identificação e rastreamento em tempo real dos objetos de interesse: robôs e bola. A seguir, descrevemos o módulo responsável pela decisão da estratégia de jogo, que se utiliza das informações fornecidas pelo sistema de visão e conhecimentos sobre o domínio (futebol) para determinar as ações que os robôs devem executar, expressas em direção e velocidade de navegação. Finalmente, a plataforma de hardware utilizado é apresentada de forma sucinta. Nas competições da FIRA 98, realizadas na França, o time GUARANÁ teve um ótimo desempenho, conquistando o vice-campeonato na categoria MIROSOT - MIcro-RObot SOccer Tournament.

Palavras Chaves: Visão Computacional, Inteligência Artificial, Robótica, Futebol de Robôs.

Abstract: Robotic soccer has been adopted as a standard problem so that various theories, algorithms, models, performance, and architectures can be evaluated and a wide range of technologies needs to be integrated, promoting both robotics and AI research. This paper describes our experience in these aspects, providing detailed description of the software modules adopted in the construction of the GUARANÁ robot-soccer team, one of the Brazilian teams at FIRA'98 Games - Federation of International Robot-soccer Association.

First we describe the vision layer, composed of an off-line calibration phase and an on-line real-time identification and tracking phase. Then, the strategic layer is detailed, which is based on the information passed from the vision layer and the stored information about the soccer domain. It decides the robot actions to be executed, given in terms of velocity and navigation direction. Finally, we give an overall description of the hardware architecture used. At FIRA'98 Games, held in Paris, France, The GUARANÁ team performed well, and won the worldwide vice-championship in the MIROSOT category (Micro-Robot Soccer Tournament).

Keywords: Computational Vision, Artificial Intelligence, Robotics, Robotic Soccer.

1 INTRODUÇÃO

Futebol de robôs é uma iniciativa internacional voltada à pesquisa e educação, visando promover desenvolvimentos ligados às áreas de Inteligência Artificial e Robótica Inteligente (Kitano et alii, 1997). Através da adoção deste problema padrão - futebol de robôs - pode-se fazer a avaliação de várias teorias, algoritmos, desempenhos e arquiteturas, onde uma grande variedade de tecnologias podem ser integradas e analisadas. Uma das vantagens oferecidas pela adoção de um problema padrão é que a avaliação do progresso de uma pesquisa pode ser claramente definida e acompanhada.

O domínio de futebol de robôs é dinâmico e imprevisível, resultando num domínio bastante complexo, que exige o uso de sistemas com alto grau de autonomia atuando em malha fechada e em tempo real para solucioná-lo. Diversos tópicos específicos de pesquisa são oferecidos por este domínio, incluindo, entre outros: *(i)* completa integração entre percepção, ação e cognição num time de múltiplos agentes robóticos; *(ii)* definição de um conjunto de comportamentos reativos robustos para cada agente, isto é, cada agente deve ser capaz de realizar tanto ações individuais quanto colaborativas;

Artigo Submetido em 03/03/1999
1a. Revisão em 14/06/1999; 2a. Revisão em 27/04/2000
Aceito sob recomendação do Ed. Consultor Prof. Dr. Ricardo Tanscheit

(iii) percepção em tempo real, robusta e confiável, incluindo rastreamento de múltiplos objetos em movimento.

Uma das categorias existentes de futebol de robôs é a MIROSOT - MIcro-RObot SOccer Tournament, promovida pela FIRA. Esta categoria é disputada entre dois times, cada um composto por três robôs de dimensões máximas de 7.5 cm x 7.5 cm x 7.5 cm, num campo verde retangular de madeira de 130 cm x 90 cm. Uma câmera instalada a uma altura mínima de 2 metros acima do campo captura imagens do jogo, que são transmitidas a um computador (*off-board*) responsável por um ciclo de controle bem definido: primeiramente, as imagens são interpretadas, reconhecendo bola e jogadores; a seguir, são decididas táticas e estratégias de jogos e as ações no domínio são definidas em termos de comandos de movimento enviados aos robôs, via comunicação sem fio (

Figura 1). Cada time é identificado por etiquetas coloridas de 3.5 cm x 3.5 cm colocadas sobre os robôs, sendo que um time deve possuir etiquetas amarelas e outro, azuis. A bola utilizada é uma bola de golf laranja.

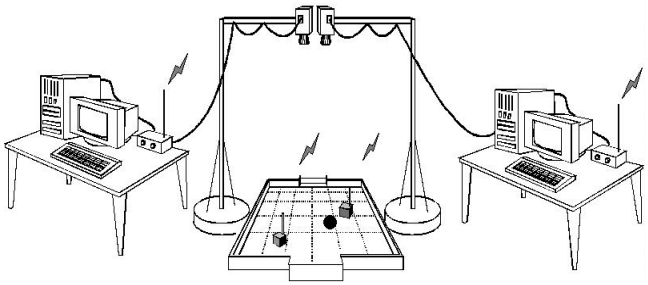

Figura 1 - Campo e configuração para MIROSOT (extraída de www.mirosot.org).

Este artigo tem por finalidade descrever a arquitetura geral de software do time GUARANÁ de futebol de robôs, composta basicamente pelos módulos de percepção visual em tempo real (visão computacional), estimativa futura de movimentação, estratégia de jogo, determinação e comunicação dos comandos a serem enviados ao motores dos robôs, conforme apresentado na Figura 2.

A principal filosofia por nós adotada na construção do nosso time foi a de usar o mínimo de sofisticação de modo a construir um time de robôs tão integrado e robusto quanto possível. Nas competições da FIRA'98, realizadas na França em 1998, o time GUARANÁ teve um ótimo desempenho, conquistando o vice-campeonato na categoria Mirosot.

O artigo está organizado do seguinte modo: a Seção 2 descreve o módulo de visão computacional, onde cada objeto de interesse é reconhecido e localizado no campo, em tempo real. Na Seção 3 é descrito o processo de estimativa da movimentação de alguns objetos de interesse, informação esta que é transmitida ao módulo responsável pela definição da estratégia de jogo e decisão dos comandos a serem transmitidos aos robôs, descrito na Seção 4. O protocolo e a forma de comunicação entre o módulo de estratégias e os robôs são descritos na Seção 5. Informações referentes às plataformas de *hardware* e *software* adotadas na implementação do time são descritas na Seção 6. Finalmente, a Seção 7 apresenta nossas conclusões.

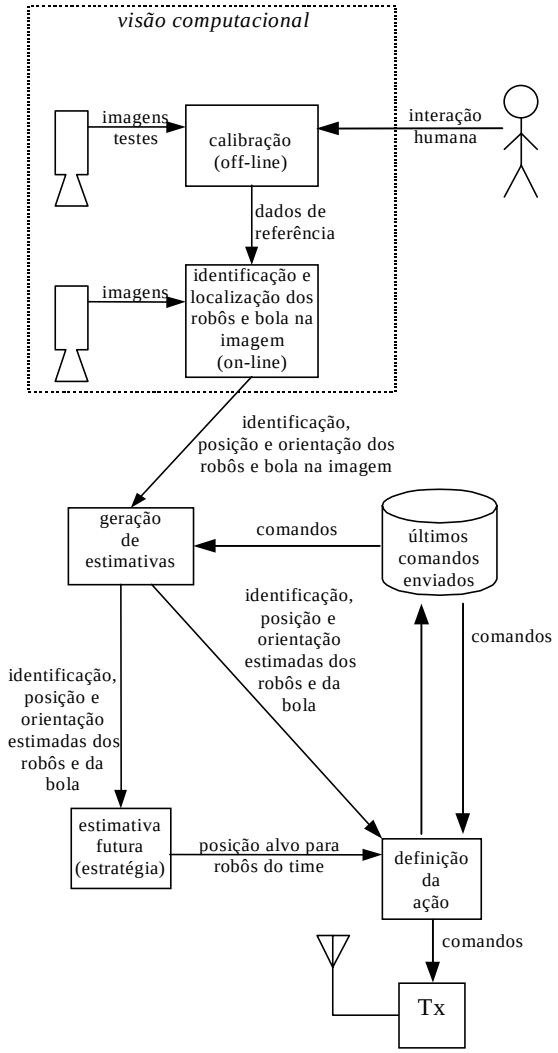

Figura 2 – Módulos e fluxo de dados do time Guaraná.

2 VISÃO COMPUTACIONAL EM TEMPO REAL

O reconhecimento dos objetos de interesse no campo - jogadores e bola - é baseado nas cores da imagem capturada pela câmera, sendo que um conjunto de *pixels adjacentes* de cor laranja identifica a bola; de cor azul, os robôs de um time e de cor amarela, os robôs de outro time. Para completar a identificação, localização e determinação da orientação dos robôs do time GUARANÁ, além da etiqueta da cor do time, exigida pelas regras, uma segunda etiqueta de cor rosa foi utilizada, ambas margeadas por uma moldura preta, conforme apresentado na Figura 3. A cor rosa foi escolhida por não ser uma das cores reservadas na categoria MIROSOT (verde escuro, branca, preta, laranja, amarela e azul) e por apresentar razoável separabilidade no espaço de cores restantes.

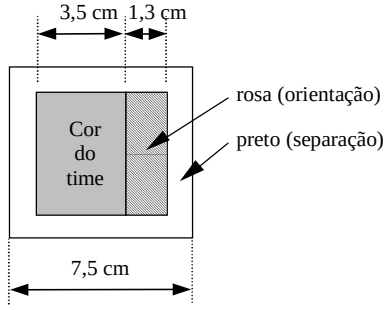

Figura 3 - Etiqueta de identificação dos robôs.

A finalidade da borda preta usada nas etiquetas dos robôs do time GUARANÁ (Figura 3) é a de evitar que duas etiquetas de uma mesma cor, porém colocadas em dois robôs distintos que estejam em contato, possam ser identificadas como um único elemento, evitando um indesejado alinhamento acidental.

O ciclo total de controle do sistema foi definido pela taxa de aquisição de imagens, que, no time GUARANÁ, é de 30 quadros por segundo, ou seja, um quadro a cada 33 ms. Assim, em períodos de 33 ms uma nova imagem refletindo o estado atual do campo torna-se disponível ao computador para processamento.

O módulo de visão computacional possui uma fase inicial *(off-line*) de calibração, executada previamente ao início da partida. Durante a partida (fase *on-line*), a visão identifica, localiza e rastreia a bola e os jogadores no campo.

2.1 Calibração

Nos 20 minutos que antecedem o início de uma partida, as equipes devem preparar os times, instalando equipamentos e calibrando o módulo de visão computacional. Nesta fase de calibração, a visão computacional é preparada para adaptar-se à iluminação do ambiente, reconhecer cores, localizar objetos, relacionar o espaço-imagem (pixels) com o espaço-campo (centímetros), definir cores do time e do adversário, identificar campos de ataque e de defesa, etc. Duas destas tarefas são de extrema importância e serão detalhadas a seguir: definição dos valores limites para classificação das cores e construção de um modelo do campo vazio.

2.1.1 Classificação das cores

A definição dos valores limites para a classificação das cores utilizadas no jogo - azul, amarela, rosa e laranja - é feita através de interação direta com um operador humano.

Numa interface apresentada ao operador, deve-se inicialmente selecionar qual cor será calibrada. A seguir, imagens do campo são continuamente apresentadas, onde a cor de interesse deve ser selecionada pelo operador através do *mouse*. Para cada seleção, ou seja, para cada pixel indicado pelo mouse, são calculadas as relações R/G e G/B do valor RGB do pixel. Esses valores são mostrados em duas faixas correspondentes, abaixo da imagem do campo, para acompanhamento visual pelo operador. Após diversas seleções, os limites superior e inferior das relações R/G e G/B são estabelecidos como valores limites para serem usados na classificação da cor calibrada. Este processo deve ser repetido para cada cor de interesse.

Para maior robustez na classificação, os elementos contendo a cor que estiver sendo calibrada devem ser dispostos em diferentes posições no campo, para que os valores calculados reflitam a não homogeneidade normalmente existente na iluminação.

O operador deve repetir a calibração - ou redefinir limites - caso observe superposição dos limites de aceitação para diferentes cores.

2.1.2 Campo Vazio

Durante uma partida, para acelerar o processamento de imagens, é realizada uma fase preliminar de subtração da imagem atual do campo com uma imagem modelo do campo vazio - sem jogadores e sem a bola. Esta subtração é realizada somente nas bandas R e G da imagem. A definição da imagem modelo do campo vazio também é realizada na fase de calibração do sistema. São adquiridas N imagens do campo vazio, a um intervalo ΔT entre as aquisições, para melhor refletir as oscilações de luminosidade devidas tanto à iluminação não homogênea quanto aos ajustes automáticos de algumas câmaras utilizadas. No time GUARANÁ, os valores de N e ΔT foram definidos empiricamente como 5 imagens e 1 segundo, respectivamente. Para determinar a imagem modelo do campo vazio, calcula-se a média entre as N imagens:

$$\overline{p}(x,y) = \frac{\sum_{i=1}^{N} p_i(x,y)}{N}; \quad x = xi,...,xf; \quad y = yi,...,yf \tag{1}$$

onde $\overline{p}(x,y)$ é o valor do pixel na coordenada (x, y) da imagem modelo do campo vazio; $p_i(x, y)$ é o valor do pixel na coordenada (x, y) da i-ésima imagem do campo vazio; N é o número de imagens consideradas; (xi, yi), (xf, yf) são as coordenadas superior esquerda e inferior direita que definem os limites do campo na imagem, refletindo a área de interesse na imagem. Esses limites dependem da fixação da câmera em relação ao campo, a qual deve ser feita de forma que todo o campo possa ser visualizado na imagem e que longitudinalmente o campo esteja alinhado com o eixo horizontal da imagem.

A partir da imagem modelo, calcula-se os desvios dos valores dos pixels em relação às outras N imagens, definindo limites de variação em R e G:

$$\begin{aligned} &LR = \max(\left| r(\overline{p}(x,y)) - r(p_i(x,y)) \right|); \\ &i = 1...N; x = x_i...x_f; y = y_i...y_f \\ &LG = \max(\left| g(\overline{p}(x,y)) - g(p_i(x,y)) \right|); \\ &i = 1...N; x = x_i...x_f; y = y_i...y_f \end{aligned} \tag{2}$$

onde LR, LG são os limites dos componentes vermelho e verde da imagem, respectivamente, que serão utilizados na subtração das imagens, durante a fase *on-line* do sistema; $r(p)$, $g(p)$ são os componentes vermelho e verde do pixel p, respectivamente; $\overline{p}(x,y)$ é o valor do pixel na coordenada (x, y) da imagem modelo do campo vazio; $p_i(x, y)$ é o valor do pixel na coordenada (x, y) da i-ésima imagem do campo vazio; (xi, yi), (xf, yf) são as coordenadas superior esquerda e inferior direita que definem os limites do campo na imagem; N é o número de imagens consideradas.

2.2 Identificação e Rastreamento

Para enviar um comando a um determinado robô, o computador central deve ser capaz de identificar este robô, além de determinar sua posição, orientação e velocidade atual. No entanto, no time GUARANÁ, as etiquetas são idênticas e, portanto, não podem ser utilizadas para distinguir um determinado robô dos outros membros do time. Desta forma, é necessário o uso de um processo de rastreamento dos robôs, a partir de uma identificação realizada por um operador humano no início do jogo.

2.2.1 Subtração inicial de imagens

Quando uma imagem é capturada, gera-se inicialmente a imagem diferença $p_d(x, y)$ da subtração desta imagem atual $p_a(x, y)$ com a imagem modelo do campo vazio $\overline{p}(x, y)$:

$$r(p_d(x,y)) = \left| r(\overline{p}(x,y)) - r(p_a(x,y)) \right|;$$
$$g(p_d(x,y)) = \left| g(\overline{p}(x,y)) - g(p_a(x,y)) \right|; \quad (3)$$
$$x = xi,...,xf; \quad y = yi,...,yf$$

onde $p_d(x, y)$ é o módulo da diferença do valor do pixel na coordenada (x, y) da imagem atual com a imagem modelo do campo vazio; $p_a(x, y)$ é o valor do pixel na coordenada (x, y) da imagem atual – última imagem capturada; $\overline{p}(x, y)$ é o valor do pixel na coordenada (x, y) da imagem modelo do campo vazio; $r(p)$, $g(p)$ são os componentes vermelho e verde, respectivamente, do pixel p; (xi, yi), (xf, yf) são as coordenadas superior esquerda e inferior direita que definem os limites do campo na imagem.

Se $r(p) > LR$ e $g(p) > LG$, o pixel p corresponderá a elementos de interesse na imagem ou a ruídos. Na imagem diferença são determinados conjuntos de pixels *conectados* de valores *similares*. A medida de similaridade é definida pelos valores limites das relações R/G e G/B das cores de interesse (veja em 2.1.1), determinadas na fase de calibração.

2.2.2 Determinação do Centróide

Pixels adjacentes conectados com cores similares são considerados componentes de um mesmo elemento, desde que o número de pixels conectados esteja na faixa de limiares previamente estabelecidos na fase de calibração do sistema (em função do tamanho do elemento naquela respectiva cor, da resolução da imagem e da posição da câmera). Os que estiverem fora dos limiares são considerados ruídos e são desprezados. Para localizar estes elementos na imagem, a coordenada (x, y) do centróide (centro de massa) de cada elemento é calculada (Han et alii, 1996):

$$x = \frac{\sum_{i=1}^{n} x_i}{n} \quad \text{e} \quad y = \frac{\sum_{i=1}^{n} y_i}{n} \quad (4)$$

onde (x_i, y_i) são as coordenadas dos pixels e n é número de pixels próximos que compõem o elemento.

2.2.3 Rastreamento

Quando uma partida é iniciada, cada robô é identificado por um operador humano. A partir deste instante, a identificação se dá por intermédio de um algoritmo de rastreamento, que usa um método de otimização exaustiva para encontrar a solução de mínimo custo do seguinte sistema:

$$s(i, j, r) = d(e_{time}(i), e_{orient}(j)) + d(e_{time}(i), e_{ant}(r)); \quad (5)$$

sujeito a:
$$li_{time} \le ne_{time}(i) \le ls_{time};$$
$$li_{orient} \le ne_{orient}(j) \le ls_{orient};$$
$$0cm \le d(e_{time}(i), e_{ant}(r)) \le 14cm;$$
$$4cm \le d(e_{time}(i), e_{orient}(j)) \le 7cm.$$

Na expressão anterior r = 1, 2, 3 é o número do robô considerado; i = 1, 2, ..., n é número de um dos n elementos da cor do time encontrado na imagem; j = 1, 2, ..., m é número de um dos m elementos da cor rosa na imagem que definem orientação; $s(i, j, r)$ é a função objetivo a ser minimizada; $d(e_1, e_2)$ é a distância cartesiana entre os elementos e_1 e e_2 no sistema de coordenadas (em cm); $e_{time}(i)$ é o centróide do i-ésimo elemento com a cor do time; $e_{orient}(j)$ é o centróide do j-ésimo elemento de orientação – etiqueta de cor rosa; $e_{ant}(r)$ é o centróide do elemento da cor do time do r-ésimo robô identificado na imagem anterior; $ne_{time}(i)$ é o número de pixels no i-ésimo elemento da cor do time; $ne_{orient}(j)$ é o número de pixels no j-ésimo elemento de orientação (cor rosa); li_{time} e ls_{time} são os limites inferior e superior de número de pixels, que definem a faixa válida do tamanho do elemento (etiqueta) da cor do time. li_{orient} e ls_{orient} são os limites inferior e superior de número de pixels, que definem a faixa válida do tamanho do elemento (etiqueta) de orientação (cor rosa).

Os três pontos de mínimo (combinação de i, j e r que resultam nos três menores valores da função s) encontrados através deste sistema representam o relacionamento da etiqueta da cor do time identificado por i com a etiqueta de orientação (rosa) identificada por j, referentes ao robô r.

Os valores de li_{time}, ls_{time}, li_{orient} e ls_{orient} mudam de acordo com o tamanho da etiqueta, da resolução da imagem usada e da posição da câmara e receberam 60, 80, 20 e 30, respectivamente. Esses valores foram obtidos pela variação observada experimentalmente durante algumas sessões de teste. O número de pixels esperados para a etiqueta do time, considerando a área da etiqueta de 3,5*5 cm^2 e a área aproximada de cada pixel no campo de 0,25 cm^2, é:

$$\frac{3,5 * 5}{0,25} = 70\, pixels$$

Já para a etiqueta rosa onde a área é aproximadamente 1,3*5 cm^2, o número esperado é:

$$\frac{1,3 * 5}{0,25} = 26\, pixels$$

O sistema utilizado tem se mostrado bastante eficaz na prática. Não se observou perdas ou trocas de identificação no rastreamento dos robôs em situações normais de jogo.

2.3 Orientação e Posição

A orientação do robô é definida pelo ângulo α medido do eixo x do campo até o vetor (c_p, c_t), com $c_p=(y_p, x_p)$ sendo o centróide rosa p e $c_t=(y_t, x_t)$, o centróide da cor do time t, conforme Figura 4[1].

$$\alpha = tg^{-1} \frac{y_t - y_p}{x_t - x_p} \quad (6)$$

Para as dimensões das etiquetas usadas pelo robô e a resolução da imagem usada, foi observado um desvio máximo de ±10 graus. Esse desvio foi medido experimentalmente comparando

[1] Na implementação, por motivos de desempenho, usamos uma tabela de tangentes (look up table), onde o índice é uma composição do Δx e Δy inteiros. O valor armazenado corresponde a ângulos entre 0 e 359. Utilizando esta tabela implementamos uma função similar ao atan2 do C, porém muito mais rápida.

o resultado apresentado pelo sistema de visão e o valor real medido na orientação do robô.

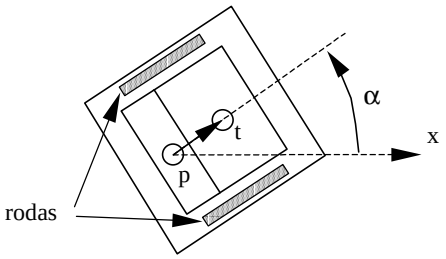

Figura 4 - Orientação do robô a partir das etiquetas

A posição (x, y) de cada robô do time resulta da média ponderada das posições dos centróides da etiqueta do time e da etiqueta rosa:

$$x = \frac{c_1 x_t + c_2 x_p}{c_1 + c_2} \quad \text{e} \quad y = \frac{c_1 y_t + c_2 y_p}{c_1 + c_2} \qquad (7)$$

onde (x_t, y_t) são as coordenadas do centróide do elemento da cor do time; (x_p, y_p) são as coordenadas da centróide do elemento rosa e c_1 e c_2 são fatores dependentes do tamanho das etiquetas. Nesta implementação usou-se c_1=3 e c_2=1, uma vez que a largura da etiqueta do time é aproximadamente três vezes maior que a largura da etiqueta rosa de orientação.

A posição dos robôs adversários e da bola são definidos pelas coordenadas dos centróides dos elementos das cores do time adversário e da bola, respectivamente, Suas orientações são inferidas pela direção do movimento, dada pela diferença de coordenadas entre a posição dos elementos na imagem atual e na anterior, uma vez que não existe outra informação disponível sobre os adversários.

2.4 Sistema de coordenadas

Após identificar cada objeto na imagem, o sistema de coordenadas da imagem (em pixels) é transformado em sistema cartesiano do campo (com unidades de 0.5cm), como mostrado na Figura 5. Todas as fórmulas serão apresentadas considerando o campo de defesa no lado esquerdo do campo (origem de x).

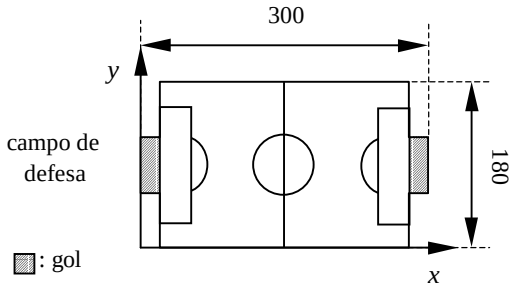

Figura 5 – Sistema de coordenadas globais obtidas do sistema de coordenadas da câmera.

O módulo de visão implementado é bastante rápido (7ms por quadro) e eficaz, permitindo algumas variações nas condições de iluminação e nas cores do campo, etiquetas e bola. A fase de calibração, nas condições normais de jogo, é sempre rápida e simples de ser executada graças à interface amigável entre homem-máquina desenvolvida. Esta interface possibilita facilidades tanto na captura das imagens do campo vazio quanto na identificação das faixas válidas para as cores.

3 MÓDULO DE ESTIMATIVA

Os robôs e a bola estão em constante movimento e podem alcançar altas velocidades (da ordem de 2m/s). Desta forma, a última imagem capturada não representa, devido aos atrasos decorrentes do processamento, a situação dos objetos no instante futuro em que os robôs receberão os comandos, isto é, um comando c é enviado para o robô r baseado na situação (posição, orientação e velocidade) de todos objetos (robôs e bola) observados na imagem no instante t. Entre os instantes t e t', quando o robô efetivamente executa o comando c, existe um período de atraso Δt. Este período Δt é decorrente de diversos fatores, tais como: aquisição e digitalização da imagem, processamento, taxa de transmissão e execução dos comandos pelos robôs. Para que o comando c possa ser calculado e executado corretamente é necessário que o sistema consiga estimar a situação futura dos objetos no campo em t'.

As posições, orientações e velocidades estimadas da bola e dos adversários são extraídas das imagens atual e anterior, assumindo velocidade constante. As estimativas referentes aos robôs do time são calculadas a partir da situação atual e dos últimos comandos enviados, de acordo com o seguinte modelo:

$$s_{r_{t'}} = f_r(s_{r_t}, c_t, c_{t-1}, \ldots, c_{t-n}) \qquad (8)$$

onde $s_{r_{t'}}$ é a situação estimada, após um atraso Δt; s_{r_t} é a situação no instante t; r=1, 2, 3 é o robô considerado; $c_t, c_{t-1}, \ldots, c_{t-n}$ são os n+1 últimos comandos considerados e $f_r(\,)$ é uma função ajustada empiricamente.

4 MÓDULO DE ESTRATÉGIA

Costuma-se dizer que cada brasileiro é um técnico de futebol. Porém, qualquer pessoa que já tenha jogado futebol sabe que um time melhor preparado fisicamente normalmente vence um outro com menor preparo, mesmo que este último tenha uma técnica muito melhor. Acreditando que as experiências no futebol tradicional são válidas também para os robôs, formulou-se algumas regras estratégias básicas que foram seguidas:

- Se o adversário for mais rápido, defenda-se melhor para sofrer um número menor de gols.
- Se o adversário for mais lento, ataque mais. Certamente existirá tempo para recuar os atacantes para a defesa.

O módulo de estratégia define, para cada robô, uma posição alvo - posição a ser atingida no campo - e respectivas velocidade e orientação associadas para que estes dados possam ser traduzidos em termos de velocidades e sentido de rotação dos motores que comandam as rodas dos robôs. A definição da posição alvo depende do estado do jogo e dos comportamentos associados a cada robô.

4.1 Comportamentos

Foram utilizados três comportamentos estratégicos principais, que cada robô pode assumir de acordo com sua posição: goleiro, defensor e atacante. Um robô fixo é designado para ser o goleiro, devendo sempre se manter neste comportamento. Já

os outros dois robôs alternam de comportamento, dependendo do estado do jogo; no entanto, sempre um deles é o defensor e outro, atacante - atualmente, não ocorre o caso de ter dois atacantes ou dois defensores.

4.1.1 Goleiro

Basicamente, o comportamento do goleiro está descrito no diagrama de transições de estados da Figura 6, o qual consiste de três estados: *estado* A_{gol} - posicionar goleiro no centro do gol sobre a linha defensiva; *estado* B_{gol} - posicionar goleiro na interseção da direção da bola com a linha defensiva do gol e *estado* C_{gol} - posicionar goleiro alinhado à coordenada y_b da bola na linha defensiva[2]. A Figura 6 representa o comportamento nas diversas situações de jogo, sempre considerando a posição da bola.

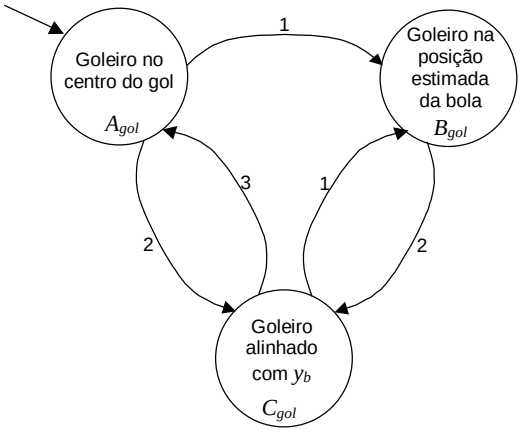

Figura 6 – Diagrama de transições de estados do goleiro

As transições representadas na Figura 6 ocorrem segundo as seguintes condições:

Transição 1: xb < d1 e Δxb < 0
Transição 2: xb < d2 e Δxb ≥ 0
Transição 3: xb ≥ d2

onde (x_b, y_b) coordenada da bola; Δx_b deslocamento da bola nos últimos 33ms ($\Delta x_b > 0$, indica bola se dirigindo ao campo do adversário); d_1 limite estratégico a partir do qual o robô goleiro pode chegar à posição estimada da bola para efetuar a defesa, valor utilizado: 40cm; d_2 limite estratégico a partir do qual se considera que não existir perigo iminente, valor utilizado: 70cm.

O estado B_{gol} é o mais importante - é ele que efetua as defesas. Nesse estado, o robô é colocado na linha de defensiva (coordenada x_{gol}) e y_{inters} (estimativa de interseção da linha de direção da bola e da linha x_{gol}). O y_{inters} é calculado pela equação 9.

$$y_{inters} = y_b - \frac{\Delta y_b.(x_b - x_{gol})}{\Delta x_b} \qquad (9)$$

sendo (x_b, y_b) a coordenada da bola; $(\Delta x_b, \Delta y_b)$ os deslocamentos da bola em x e y, observados nos últimos 33ms (últimos dois quadros); x_{gol} a linha x onde o robô goleiro deve permanecer para não sair da área nem invadir o gol (linha defensiva).

Nos três estados (A_{gol}, B_{gol} e C_{gol}) a abscissa do robô é sempre x_{gol}. Já a ordenada nos estados B_{gol} e C_{gol} é calculada, mas sempre respeitando os limites do gol, pela expressão (10).

$$y_{g\,min} \le y_{inters} \le y_{g\,max} \qquad (10)$$

onde $y_{g\,min}$ e $y_{g\,max}$ são os limites inferior e superior do gol, definidos pelas dimensões do campo e do gol.

Para que o robô goleiro possa se posicionar mais rapidamente, ele como qualquer outro robô, pode se deslocar igualmente para frente e para trás, mantendo portanto sempre sua orientação em 90º ou 270º em relação ao eixo x do campo (Figura 7).

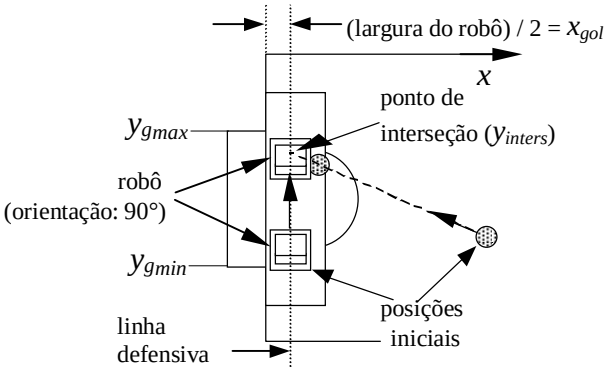

Figura 7 – Comportamento do goleiro

4.1.2 Defensor

A posição alvo do defensor situa-se na coordenada y da bola, em uma faixa de defesa distanciada de 30cm a 60cm do gol, bloqueando a bola para impedi-la de chegar ao gol ou obstruindo um possível chute do adversário. Porém, ele sai do caminho da bola para permitir que o atacante do mesmo time leve a bola para longe da área de defesa. Quando o defensor e a bola estão em uma situação favorável com respeito ao gol adversário, ocorre uma permuta de comportamentos do defensor com o do atacante, simplificando e agilizando uma situação de ataque ao gol adversário. As situações favoráveis acontecem quando o defensor é o robô mais próximo à bola e, além disso, que ele se encontre na área de transição, como mostrado na Figura 8.

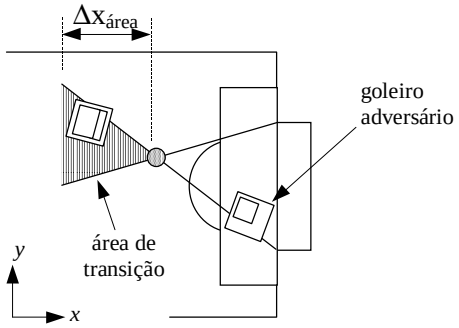

Figura 8 – Área de transição

Para que ocorra a permuta, o defensor deve ter orientação voltada ao gol adversário e estar posicionado na área de transição. A área de transição é definida pelos segmentos das retas que partem dos limites da abertura do gol, cruzando no centro da bola e com extensão definida por $\Delta x_{área}$ = 20cm, da

[2] Todas as coordenadas seguem sempre o sistema de coordenadas apresentado em 2.4.

bola em direção oposta ao gol adversário, conforme mostra do na Figura 8. A medida $\Delta x_{área}$ = 20cm é empírica e baseada nas dimensões do campo e nas velocidades dos robôs e bola.

4.1.3 Atacante

O atacante pode estar em dois estados distintos: modo de posicionamento (E_{atac}), quando o alvo g é colocado atrás da bola numa posição que permita a condução da bola ao gol adversário ou o bloqueio da bola empurrada por um atacante adversário - desta forma, este posicionamento serve também para fortalecer a defesa; modo de condução (D_{atac}), quando o robô se desloca e corrige sua trajetória – alterando o alvo g – para conduzir a bola de forma a alcançar o gol adversário. A Figura 9 mostra os estados possíveis para o defensor e o atacante.

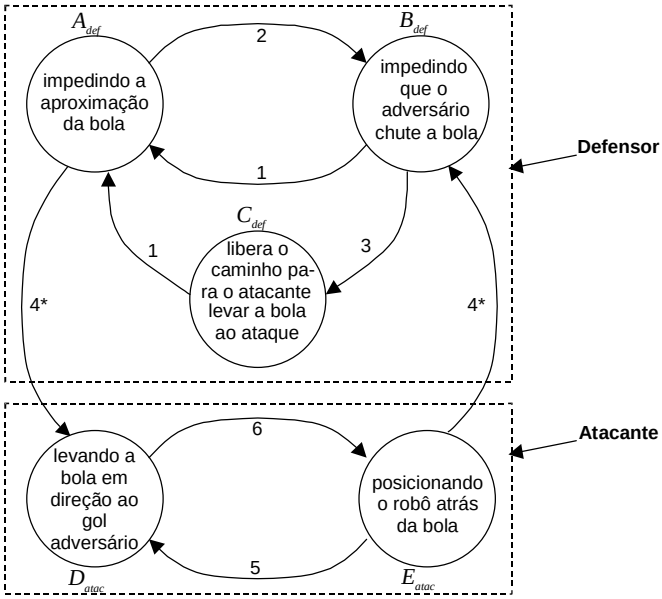

Figura 9 – Diagrama de transições de estado do comportamento do defensor e do atacante

As transições enumeradas na Figura 9 ocorrem segundo:

Transição 1: $x_b - d_1 > x_d$
Transição 2: $x_b + d_1 < x_d$
Transição 3: $x_a < x_b < x_d$ **e** $y_a - y_b < d_2$ e $y_d - y_b < d_2$
*Transição 4**: $x_b - x_d < d_3$ **e**

$$y_{g\min} \le y_b + \frac{(y_d - y_b).(x_b - x_g)}{(x_d - x_b)} \le y_{g\max}$$

* - troca entre o comportamento do atacante e do defensor
Transição 5: $x_b - x_a < d_3$ **e**

$$y_{g\min} \le y_b + \frac{(y_a - y_b).(x_b - x_g)}{(x_a - x_b)} \le y_{g\max}$$

Transição 6: $x_a - x_b > d_4$

sendo (x_b , y_b) a coordenada da bola; (x_a , y_a) a coordenada do robô atacante; (x_d , y_d) a coordenada do robô defensor; d_1, d_2, d_3, d_4 os valores determinados empiricamente (4, 20, 4, 8 cm, respectivamente); $y_{g\min}$ e $y_{g\max}$ os limites inferior e superior do gol (Figura 7) e x_g a coordenada da linha de gol do campo adversário.

4.2 Evitando Obstáculos

Devido à natureza extremamente dinâmica do domínio futebol de robôs, a tarefa de evitar obstáculos e, consequentemente, evitar colisões, torna-se muito importante, uma vez que reduz o número de faltas cometidas contra jogadores adversários e aumenta a possibilidade de alcançar a bola quando bloqueada por outros jogadores. Após a definição da posição alvo pelo módulo de estratégia, o sistema tenta evitar colisões por meio de um planejamento de caminhos entre os obstáculos.

Inicialmente calcula-se a trajetória do robô r da posição atual s até a posição alvo g, utilizando uma reta R_{sg}. Se a trajetória de um obstáculo o cruzar R_{sg} (Figura 10a), calcula-se um alvo intermediário g' para o robô r, como mostrado na Figura 10b, levando-se em consideração o ponto P_i onde as trajetórias de o e r se interceptariam e a posição atual do robô (veja item 4.3).

Nesse processo, a velocidade e direção do obstáculo definem o ponto de interseção P_i e o tempo para a colisão t_c. Se t_c é menor que o tempo t_g que o robô r levaria para alcançar o alvo g (isto é, $0 < t_c < t_g$) então um alvo intermediário g' é definido[3]. g' é localizado a uma distância d do obstáculo o, perpendicularmente a R_{sg}, considerando o caminho s-g'-g mais curto. A distância d é definida empiricamente. As bordas do campo também impõem restrições à escolha do caminho s-g'-g mais curto, uma vez que o robô também deve evitar colisões com as bordas.

Como em Veloso et alii (1997) este processo de evitar colisões é iterativo, sendo aplicado a um obstáculo por vez – aquele que estiver mais próximo do robô r. Se o obstáculo o estiver muito próximo do robô r, o alvo intermediário g' escolhido ainda poderá resultar em colisão.

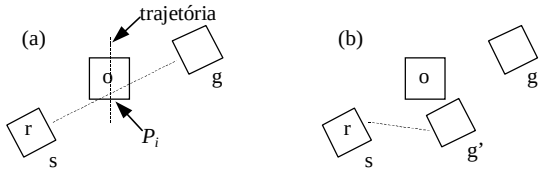

Figura 10 - Evitando Obstáculos

4.3 Interseção de trajetórias

O ponto de interseção $P_i = (x_i, y_i)$ é definido como o ponto onde a trajetória do robô r cruza a trajetória do objeto o (alvo ou obstáculo). Uma aproximação é usada neste cálculo, utilizando o número de quadros n_x que decorre do instante atual até r e o se interceptarem em x e o número de quadros n_y até r e o se interceptarem em y, que são:

$$n_x = \frac{x_r - x_o}{\Delta x_o - \Delta x_r} \quad \text{e} \quad n_y = \frac{y_r - y_o}{\Delta y_o - \Delta y_r}, \qquad (11)$$

onde (x_r, y_r) são as coordenadas do robô; (x_o, y_o) são as coordenadas do objeto e Δx_r, Δy_r, Δx_o e Δy_o são os deslocamentos do objeto e do robô, dados pela diferença entre

[3] t_c e t_g, representam o número de quadros de imagem até a interseção das trajetórias.

a posição obtida da imagem atual e a posição obtida da imagem anterior.

Se $n_x < 0$ ou $n_y < 0$, não existe ponto de intersepção. Quando as velocidades de r e o são constantes, n_x e n_y são iguais e representam o número de quadros de imagem até a interseção.

Esta mesma forma de cálculo é também utilizada no modo de posicionamento do atacante (estado E_{atac}, Figura 9), porém, neste caso, como o robô pode ainda não ter a trajetória desejada (o alvo foi recentemente alterado pela estratégia), n_x e n_y podem ser diferentes. Neste caso o número de quadros até a interseção é calculado por: $n_i = min(n_x, n_y)$. Quando as trajetórias real e a desejada para levar o robô até o alvo não coincidem, n_x e n_y são diferentes e n_i corresponde a um número menor que o número real de quadros até a interseção; entretanto, à medida que as trajetórias real e desejada se aproximam, n_x e n_y convergem para o mesmo valor.

O ponto de interseção (x_i, y_i) é calculado como segue:

$$x_i = n_i.\Delta x_o + x_o \quad \text{e} \quad y_i = n_i.\Delta y_o + y_o \qquad (12)$$

5 MÓDULO DE DEFINIÇÃO E COMUNICAÇÃO DOS COMANDOS

Este módulo é responsável por calcular o próximo comando que melhor se ajusta às situações atuais do robô (posição, orientação e velocidade) de forma a levá-lo à situação alvo indicada pela estratégia. O comando escolhido leva em consideração as características mecânicas dos robôs do time, não as excedendo. Em nossa implementação consideramos que:

- O robô não pode acelerar (ou freiar) bruscamente, para evitar derrapagens;
- O robô deve respeitar velocidades limites quando estiver em curvas, evitando tombamento e derrapagens laterais.

Para efetuar tais operações foram criadas tabelas com valores limites que a partir da velocidade atual indicam as próximas velocidades admissíveis (limites superior e inferior) e o ângulo máximo aceitável de curva para o robô.

O sistema de comunicação que transmite os comandos do computador aos robôs é composto por um transmissor e três receptores utilizando a freqüência de 72,830MHz. Eles foram adaptados de receptores de rádio de controle remoto de servomotores para transmissão de sinais digitais, com uma taxa de 1200bps, usando codificação FSK Manchester (Giozza et alii 1986). O transmissor envia mensagens para todos robôs segundo o seguinte protocolo: 2 bits de sincronismo, 2 bits de preâmbulo, 8 bits para o robô 1, 8 bits para o robô 2, 8 bits para o robô 3 e 4 bits de verificação (check-sum), totalizando uma palavra de 32 bits por transmissão.

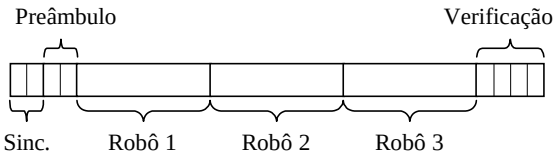

Figura 11 – Protocolo de comunicação

Os 8 bits enviados para cada robô são divididos em 4 bits para cada motor – esquerdo e direito, mostrados na Figura 12. Nesta figura, D é a direção de giro do motor (frente / trás) e V são os três bits para codificar diferentes velocidades.

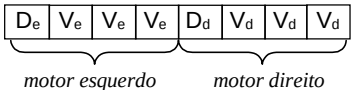

Figura 12 – Palavra de controle enviada para cada robô

Durante o desenvolvimento do time, outros tipos de comandos foram testados. Para simplificar a alteração destes tipos, durante o período de teste, as mensagens aos robôs incluem um preâmbulo que permite a alteração, a qualquer instante, do formato do comando e da mensagem utilizada. Os outros tipos de comandos não serão discutidos aqui, pois foram utilizados apenas no período de desenvolvimento.

6 PLATAFORMAS UTILIZADAS

O software do time GUARANÁ é centralizado e foi executado em um PC Pentium II 233 MHz, usando Windows 95. Na implementação utilizamos C e, em algumas rotinas do módulo de visão, linguagem de montagem. As imagens são capturadas por uma câmara de vídeo colorida JVC com saída em S-VHS e são digitalizadas por uma placa PCI da PixelView PV-Bt-848. A imagem digitalizada usa uma resolução de 320x240 pixels e 24 bits de quantização em RGB.

O processamento da imagem utiliza como interface o Video for Windows® da Microsoft® - VfW (Microsoft, 1993). O uso do VfW fornece versatilidade e compatibilidade, já que possibilita aquisição de imagens digitais oriundas de qualquer placa digitalizadora compatível, desde que seja capaz de operar em pelo menos 30 quadros por segundo com 24 bits RGB.

Os robôs usam pequenos motores DC Mabuchi retirados de máquinas de calcular eletromecânicas. Cada um desses motores apresentam torque de 20gf.pol com 5V. Um estágio único de redução é usado para cada montagem motor/roda, com uma redução de 5:1. Em cada roda existe um reticulado reflexivo estampado, de onde um sensor infravermelho detecta informações de rotação que realimentam a unidade de controle permitindo a regulagem da velocidade.

Os robôs podem desenvolver uma velocidade máxima de aproximadamente 1m/s. A unidade de controle é composta de um microcontrolador (Atmel 89C5021), executando em 14,75MHz de clock. O microcontrolador controla a tensão nos motores através de duas pontes H's completas ligadas aos pinos de E/S, permitindo o controle de velocidade e direção por PWM (pulse width modulation).

O receptor dos robôs é adaptado de um receptor de rádio controle remoto de servo Futaba. O sinal recebido é enviado ao microcontrolador que é responsável pela recuperação do sincronismo e pela demodulação do FSK Manchester.

A Figura 13 mostra o diagrama de blocos da eletrônica embarcada de cada robô, incluindo receptor, microcontrolador, tacômetro e motores associados às rodas.

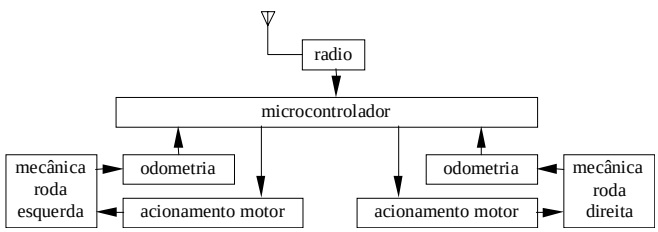

Figura 13 – Diagrama de blocos da eletrônica embarcada

Os corpos dos robôs foram construídos com placas de epoxi-fibra de vidro e canos de plástico PVC de ½ polegada. A potência elétrica é fornecida por 4 pilhas de NiCd fornecendo 800mA.h em 4.8V. Cada robô possui um tampo de cobre, que atua como uma antena receptora (Figura 14). Devido à relativa alta potência de transmissão (100 mW) e aos receptores altamente sensíveis, a faixa de comunicação do rádio alcança mais de 50 metros. A Figura 14 apresenta fotos dos robôs.

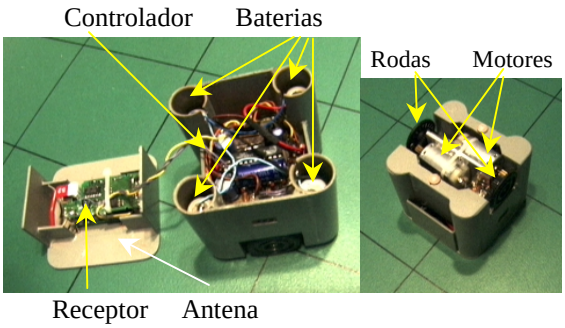

Figura 14 – Detalhes de montagem do robô (vista superior e inferior)

7 CONCLUSÃO

A principal filosofia por nós adotada nesta construção foi a de usar o mínimo de sofisticação de modo a adquirir um time de robôs tão integrado e robusto quanto possível.

Os pontos fortes do time Guaraná são suas táticas simples e eficientes e o sistema de visão computacional: simples, robusto às variações ambientais de iluminação e de fácil ajuste. Acreditamos que o bom desempenho do time é proveniente do razoável equilíbrio deste sistema e de uma boa integração entre os módulos constituintes.

O controle do robô tem se mostrado impreciso, uma vez que não foi conseguido um ajuste eficaz entre os comandos enviados pelo software da estratégia e controle com as reações do hardware dos robôs. Assim o controle pode ser um ponto para grandes melhorias. O uso de motores pequenos, rodas e corpo de plástico resultou num robô leve, robusto e de baixo custo. O pequeno peso do robô apresenta, por um lado, vantagens devido ao baixo consumo e à alta capacidade de aceleração; por outro lado, tem sua velocidade alterada em choques com a bola e é facilmente empurrado pelos adversários. Portanto, para tornar o sistema mais competitivo, os robôs deveriam tornar-se mais pesados e fortes, com maior torque e maior velocidade. Os robôs Coreanos, campeões em 1998 em tal competição, pesam aproximandamente 700g e atingem velocidades de 2m/s.

AGRADECIMENTOS

Gostaríamos de agradecer especialmente ao Prof. Guido Stolfi (PEE/EPUSP) pelo projeto e construção dos robôs e sistema de comunicação, além da preciosa colaboração em diversas fases do projeto e nas competições. Diversos docentes, pesquisadores e alunos de graduação e pós-graduação são co-responsáveis pelo sucesso do time GUARANÁ: Prof. Jaime S. Sichman (PCS/EPUSP), Prof. Humberto Ferasoli Filho (DCo/UNESP-Bauru), Prof. Felipe Pait (PEE/EPUSP), Prof. Tsen Chung Kang (PEE/EPUSP), Prof. Marcelo N. Franchin (FEE/UNESP-Bauru), Reinaldo A. C. Bianchi, Bruno A. Basseto, Renato Mikio Nakagomi, Fernando L. Goulart, Guido M. Chagas, Júlio Monteiro, Ricardo Matsushima Teixeira, Ricardo Pereira, Leonardo Scardua, Roberto Barra, Ricardo Domenecci, Mário A. N. Solis - a eles, nossos agradecimentos.

Gostaríamos ainda de externar nossa gratidão ao Prof. Antônio Marcos de Aguirra Massola, Diretor da Escola Politécnica da Universidade de São Paulo - EPUSP, ao Prof. Geraldo Lino de Campos, Chefe do Departamento de Engenharia de Computação e Sistemas Digitais da EPUSP e ao Prof. Eduardo M. Morgado, Chefe do Departamento de Computação da UNESP de Bauru pelo apoio recebido.

Nossa participação no MIROSOT da FIRA'98 foi patrocinada pela Escola Politécnica da Universidade de São Paulo. Renê Pegoraro é parcialmente financiado pela CAPES/PICD 33020450. Anna Helena Reali Costa conta atualmente com o apoio parcial da FAPESP, Proc. FAPESP N. 98 / 6417-9.

REFERÊNCIAS

Giozza, W. F.; Araújo, J. F. M.; Moura, J. A. B. & Sauvé, J. P. (1986). *Redes Locais de Computadores: Tecnologia e Aplicações*. Mc Graw-Hill, São Paulo, Brasil.

Han, W.-G.; Baek, S.-M. and Kuc T.-Y (1996). Path Planning of Visual-Servoed Multiple Mobile Robots using the Genetic Algorithms. *Proceedings of Micro-Robot World Cup Soccer Tournament*, pp. 57-63.

Kim, S. H.; Kim, J. K.; Choi, J. C. and Kim, B. K. (1996). Development of Cooperation System for Robot Soccer Team. *Proceedings of Micro-Robot World Cup Soccer Tournament*, pp. 90-94.

Kitano, H.; Kuniyoshi, Y.; Noda, I.; Asada, M.; Matsubara, H. and Osawa, H. (1997). RoboCup: A Challenge Problem for AI. *AI Magazine* 18(1), pp.73-85.

Microsoft® Video for Windows Documentation (1993). ftp://ftp.microsoft.com/developer/drg/Multimedia/Jumpstart/VfW11e/DK/VFWDK.

Veloso, M; Stone, P.; Han, K and Achim, S (1998). The CMUnited-97 Small Robot Team. In H. Kitano (Ed.), *RoboCup-97 - Robot Soccer World Cup I.* Springer Verlag, Berlin.