Integração de Informação na equipa de Futebol Robótico CAMBADA

Paulo Bartolomeu, Luis Seabra Lopes, Nuno Lau, Armando Pinho, Luis Almeida

Abstract – **This article describes the software architecture of a *CAMBADA* autonomous agent, focusing on the information integration mechanisms. The system requirements and the developed solutions for data structures and information fusion are also discussed. Additionally, some error detection and correction mechanisms used in the *CAMBADA* robots are presented.**

Keywords – **Mobile Robots, Multi-Agent Systems, Sensor Fusion, Cooperative Sensing, Robotic Soccer**

Resumo – **Este artigo descreve a arquitectura de software de um agente autónomo *CAMBADA* focando os mecanismos de integração de informação. Discutem-se em particular os requisitos do sistema e as soluções encontradas ao nível das estruturas de dados e da fusão de informação. Adicionalmente apresentam-se alguns dos mecanismos de detecção e correcção de erros usados nos robôs *CAMBADA*.**

Palavras chave – **Robótica Móvel, Sistemas Multi-agente, Fusão Sensorial, Percepção em Equipa, Futebol Robótico**

I. Introdução

O projecto CAMBADA (*Cooperative Autonomous Mobile roBots with Advanced Distributed Architecture*) [1] [1] tem como objectivo a construção de uma equipa de futebol robótico F-2000 (*Middle-Size League*) para participação no *RoboCup* [2]. Este projecto é actualmente o mais importante da Actividade Transversal em Robótica Inteligente (ATRI) [3] do IEETA [4] envolvendo a quase totalidade dos seus elementos. De entre as inúmeras áreas científicas que contribuíram para a sua realização, a Inteligência Artificial assume neste artigo especial relevo.

O *RoboCup* é uma iniciativa internacional que tem por objectivo promover o desenvolvimento de uma equipa de futebol robótico que em 2050 seja capaz de vencer um jogo à equipa humana campeã do mundo sob as regras da FIFA [5]. Esta iniciativa possui várias classes de competição, entre elas a *Middle-Size League* (MSL). Esta classe é caracterizada por robôs de tamanho médio (30cm x 30cm $\leq$ área ocupada $\leq$ 50cm x 50cm, 40cm $\leq$ altura $\leq$ 80cm e peso $\leq$ 40kg) com sensores incorporados que jogam futebol num campo de dimensões entre 8m x 6m e 16m x 12m. Algumas convenções possibilitam a identificação de objectos pelo padrão de cor. O terreno de jogo é verde, as linhas marcadas a branco, e as balizas possuem cores distintas, uma amarela e outra azul. Os postes possuem o padrão de cor correspondente ao lado do campo em que se encontram (por exemplo, do lado da baliza amarela os postes são 2/3 amarelos e 1/3 azuis, sendo que este 1/3 azul deve corresponder à faixa central do poste). A bola é cor de laranja.

Cada jogo é dividido em dois períodos de 10 minutos com um intervalo de também 10 minutos. Cada equipa pode jogar com um máximo de 6 robôs (guarda-redes incluído) sendo estes genericamente de cor preta exceptuado a faixa colorida identificativa da equipa (magenta ou azul claro). Os robôs devem jogar em equipa podendo utilizar a tecnologias sem fios, nomeadamente Wi-Fi [7], para comunicarem entre si.

Através de uma estação de controlo electrónica (*referee box*), a equipa de arbitragem comunica com as estações base de cada uma das equipas para dar instruções acerca do estado do jogo.

A estação base é um agente de software da equipa mas que não está instalado em nenhum robô. Este agente pode comunicar com os restantes informado-os das alterações do estado do jogo recebidas da *referee box* e adicionalmente pode desempenhar o papel de "treinador" dando instruções ao nível da estratégia de jogo da sua equipa. No decorrer de uma partida existem diversos cenários em que a estação base pode intervir em resposta a um comando da estação de controlo. Os cenários mais comuns são: inicio ou fim de jogo, retirada ou re-entrada de um jogador, marcação de uma grande penalidade ou canto, etc.

As características da liga MSL relativamente às condições do jogo, do campo e dos jogadores condicionaram a arquitectura de informação dos agentes da equipa CAMBADA.

No resto do artigo, começa-se por apresentar sucintamente a equipa CAMBADA, quer a nível de hardware, quer a nível da arquitectura de software. Em seguida, na secção III, apresenta-se a estratégia de partilha de informação entre agentes e entre processos no mesmo agente, a qual se baseia numa Base de Dados de Tempo-Real. A secção IV apresenta e detalha as estruturas de dados utilizadas. A secção V constitui o tema central deste artigo. Nesta secção discutem-se os problemas identificados ao nível da integração de informação de um agente e apresentam-se as soluções desenvolvidas para os ultrapassar. Finalmente, a última secção conclui este artigo apresentando algumas linhas de trabalho futuro.

II. A equipa CAMBADA

A. Estrutura Mecânica e Sensorial dos Robôs

A estrutura mecânica de um robô da equipa CAMBADA é modular, organizando-se em três camadas. A camada inferior (Figura 1) inclui as rodas, o *kicker*, os motores e as baterias. Tudo isto está montado numa placa de alumínio de formato circular, com 440*mm* de diâmetro, e com cortes que permitem o direccionamento da bola (corte semi-circular

[1] Projecto POSI/ROBO/43908/2002 financiado pela FCT, Fundação para a Ciência e a Tecnologia, e pelo FEDER.

no eixo Y) e a fixação das rodas.

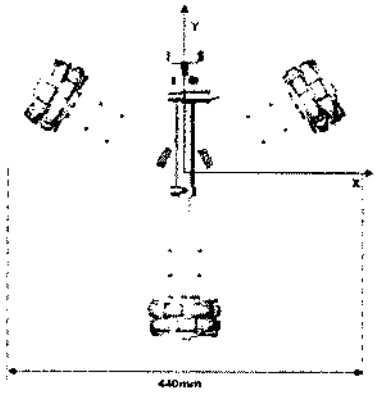

Figure 1 - Base do robô CAMBADA

Os robôs podem-se mover em todas as direcções, ou seja, possuem movimento holonómico. Este movimento é suportado pela utilização de rodas especiais (*OmniWheels*) que permitem movimentos segundo dois eixos em simultâneo. As rodas são dispostas de forma a fazerem um ângulo de 120 graus entre si. Os motores possuem *encoders* permitindo obter o número de rotações que cada roda realiza. A odometria é realizada no baixo-nível (placas de microcontrolador) recorrendo a leituras dos *encoders* e posterior processamento.

A segunda camada agrega toda a electrónica de controlo motor, a qual consiste num conjunto de microcontroladores interligados por uma rede CAN. Dado o desenho modular dos robôs, esta camada pode facilmente ser substituída por outra igual.

A terceira camada inclui um computador pessoal, onde é executado todo o software de controlo de alto nível e integração de informação, bem como as câmaras (*webcam*) que compõem o sistema de visão.

O sistema de visão artificial é composto por duas câmaras. A câmara omnidireccional, direccionada para baixo, é usada para possibilitar a visão de objectos em qualquer posição próxima do robô. A câmara frontal previligia a visão numa determinada direcção, permitindo ver mais longe nessa direcção. Esta câmara está orientada segundo o eixo Y do robô, ficando esta a ser a frente do robô. Com o mesmo alinhamento foi colocado o *kicker*.

A Figura 2 mostra um robô da equipa CAMBADA inteiramente funcional.

Observe-se ainda na Figura 2 o *laptop* que incorpora o robô. Este dispositivo computacional é responsável pelo processamento da informação de alto-nível e pelo comportamento global do robô.

B. Fluxo de Informação Global

A Figura 3 apresenta um diagrama ilustrativo do fluxo de informação de alto nível num jogador CAMBADA. As setas a cheio indicam fluxo de controlo e as setas a tracejado indicam fluxo de informação.

A informação produzida pelos processos das câmaras (*Visão Omnidireccional* e *Visão Frontal*) inicia o ciclo de processamento. Esta informação descreve as posições relativas (ao próprio robô) dos objectos identificáveis pelas

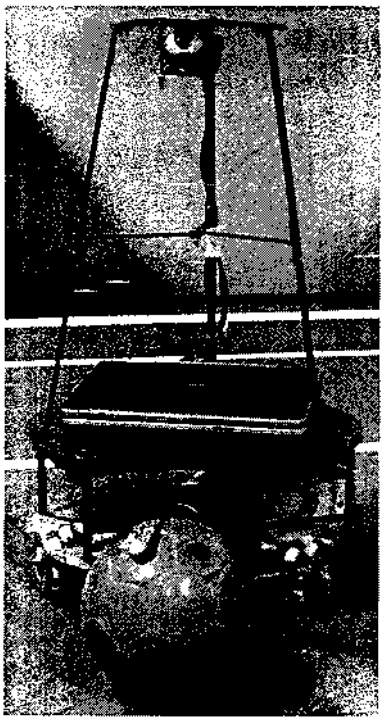

Figure 2 - Robô CAMBADA

câmaras. Os processos da visão disponibilizam esta informação para o processo principal escrevendo-a numa área de memória designada de *Área Local*. Esta área de memória é usada como mecanismo de comunicação entre os processos residentes no robô (processos da visão e processo principal).

O processo principal lê informação da área de memória local e usa-a para calcular, entre outros parâmetros, a sua posição absoluta no campo, a posição da bola, etc. Este procedimento designa-se por *Integração de Informação* e é caracterizado pela fusão de dados sensoriais que resulta em informação de natureza posicional.

A informação obtida da integração é armazenada (*Actualização do Estado do Mundo*) na área de memória partilhada reservada para o próprio agente. Esta área é ilustrada na Figura 3 por duas zonas de memória designadas de *Estado do Mundo*. O *Estado do Mundo* representado a cheio refere-se ao próprio agente enquanto que o *Estado do Mundo* representado a tracejado representa os estados do mundo dos outros agentes. A existência da área de memória partilhada visa possibilitar a comunicação entre agentes. Como resultado, esta área é acessível a todos os agentes para leitura e apenas ao próprio para escrita no *Estado do Mundo* correspondente.

Em seguida, o processo principal realiza a *Integração de Informação da Equipa*. Este procedimento é responsável pela fusão das diferentes visões de jogo dos robôs. O processo de fusão pretende produzir informação (localização dos robôs, da bola e das referências do campo) consistente com as diversas visões de jogo.

Finalmente e com toda a informação disponível, o processo principal decide qual o comportamento a realizar (*Decisão*), executando-o (*Execução de Comportamentos*). O algoritmo de decisão, que tem por base uma máquina de

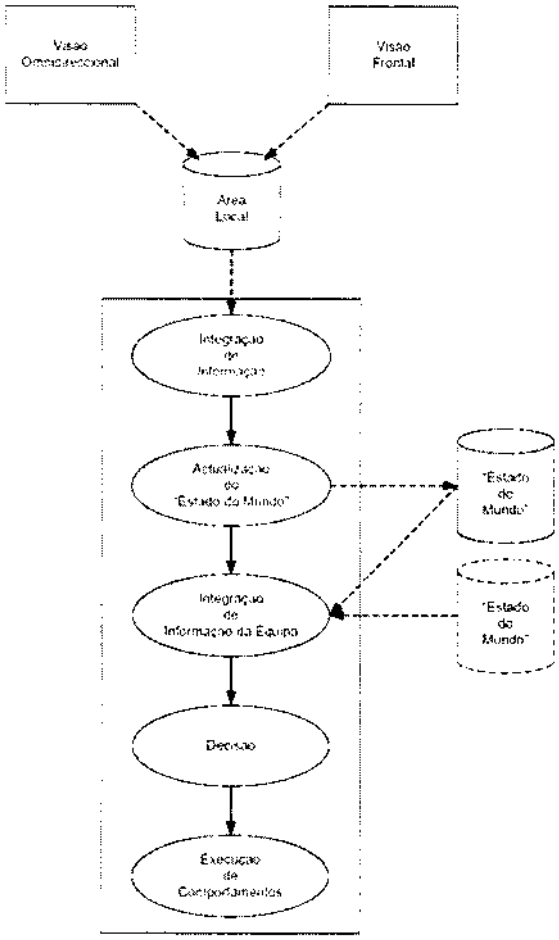

Figure 3 - Fluxo de informação Global

estados, toma em consideração não só informação proveniente das integrações prévias como também considera outros factores importantes, nomeadamente a função do robô na equipa e o estado do jogo.

Os procedimentos apresentados são executados ciclicamente e de acordo com o ritmo a que as câmaras produzem informação.

III. PARTILHA DE INFORMAÇÃO ENTRE PROCESSOS E AGENTES

Do ponto de vista da arquitectura de software, a equipa *CAMBADA* é um conjunto de agentes (os jogadores e a estação base) e o comportamento de cada agente resulta da execução de um conjunto de processos de software, nomeadamente processos percepcionais, decisórios e de controlo. Assim, é importante ter formas de partilhar informação entre processos, no mesmo agente, bem como entre os vários agentes.

O mecanismo de partilha de informação desenvolvido baseia-se numa base de dados de tempo-real distribuída (RTDB) [8], [9]. Esta base de dados reside em todos os robôs e a informação presente em cada uma delas é sincronizada com as restantes por intermédio de um algoritmo próprio.

A organização global das estruturas de dados e a sua organização individual numa base de dados constitui um factor de qualidade que tem reflexos directos na eficiência do armazenamento e na manipulação da informação.

A organização global da base de dados tempo-real desenvolvida é apresentada na Figura 4.

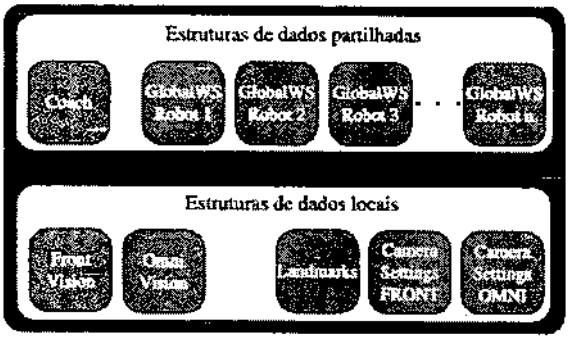

Figure 4 - Organização global das estruturas de dados

Como se pode observar existe uma separação clara entre a informação que deverá ser partilhada com os restantes agentes e a informação local do agente robótico.

A. *Área Local*

A área local da RTDB destina-se apenas a blocos de dados que não sejam passíveis de serem partilhados com os restantes robôs. Neste contexto, encontram-se os parâmetros das câmaras (*Camera Settings FRONT* e *Camera Settings OMNI*), as posições de marcas fixas no campo (*Landmarks*) e finalmente os dados obtidos a partir das câmaras (*Front Vision* e *Omni Vision*).

Os parâmetros das câmaras e a posição de marcas fixas no campo residem na área Local da RTDB para facilitar a sua manipulação. Uma vez que estes parâmetros e estas posições são inicializadas no primeiro ciclo de funcionamento de cada robô permanecendo posteriormente inalterados, não foi considerada relevante a sua difusão pelos restantes robôs. Adicionalmente, os parâmetros das câmaras são locais por definição uma vez que modelam características físicas das câmaras montadas no robô.

A informação produzida pelas câmaras é local uma vez que apenas o processo principal possui informação adicional que possibilita a sua integração. Os processos da visão comunicam com o processo principal através da área local da RTDB. A utilização da RTDB neste contexto permite uniformizar a transferência de informação entre processos locais.

B. *Área Partilhada*

A área partilhada da RTDB destina-se a informação não só relevante para o próprio robô mas também para a sua equipa. A informação constante desta área é difundida periodicamente por todos os elementos da equipa estabelecendo uma plataforma de conhecimento comum do seu "mundo".

Quando começa, ou no decorrer de um encontro de futebol robótico, a estação base instrui os elementos da equipa para assumirem um determinado comportamento. A comunicação das instruções é realizada ao nível da RTDB recorrendo à estrutura *Coach* aí residente.

Esta estrutura apenas pode ser escrita pela estação base mas pode ser lida por todos os elementos da equipa.

A informação constante da classe *GlobalWS* caracteriza a localização e o estado de um robô. A partilha da informação potencia a tomada de decisão de forma mais eficaz por qualquer elemento da equipa. Desta forma, os robôs dispõem também das visões individuais de cada um dos robôs em jogo podendo "decidir" de forma mais "esclarecida".

IV. ESTRUTURAS DE DADOS

Esta secção apresenta as estruturas desenvolvidas, referindo as motivações e os requisitos de cada uma. Para maior clareza, as classes desenvolvidas são apresentadas em notação UML (*Unified Modeling Language*) [10]-[12].

A. O "Estado do Mundo"

O "Estado do Mundo" é uma estrutura de dados que modela a organização da informação essencial a um agente robótico. A organização desta informação foi concebida tomando como objectivos a separação semântica dos diversos elementos que constituem o "Estado do Mundo", a reutilização de estruturas de dados genéricas (Vectores, etc.) e a adequação do modelo ao problema.

Após um estudo preliminar identificaram-se os seguintes elementos como essenciais ao "Estado do Mundo":

- localização, orientação e velocidade do robô e dos colegas de equipa no campo;
- localização, orientação e velocidade dos adversários no campo;
- localização e velocidade da bola no campo;
- parâmetros específicos do robô (por exemplo a sua função na equipa);
- e finalmente, parâmetros específicos da equipa (por exemplo a cor da equipa).

A localização das balizas, dos postes ou de quaisquer outros elementos estáticos no campo foi considerada irrelevante para o "Estado do Mundo" visto que os mesmos são conhecidos *a priori* e não se alteram no decorrer de uma partida.

Os elementos identificados resultaram em classes que integram a classe *GlobalWS* representada na Figura 5. O "Estado do Mundo" de um robô é assim constituído por um conjunto de objectos (Robôs, Bola e parâmetros adicionais do robô e da equipa) que modelam separadamente objectos físicos e parâmetros funcionais.

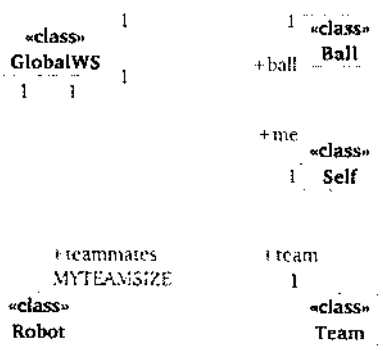

Figure 5 - "Estado do Mundo"

O número de atributos da classe *Robot* que uma dada classe *GlobalWS* possui é dado pelo somatório dos elementos de cada equipa[2]. Cada "Estado do Mundo" possui ainda informação relativa aos parâmetros adicionais para o próprio robô (*Self*), aos parâmetros da equipa (*Team*) e informação relativa à bola (*Ball*).

As subsecções seguintes descrevem em detalhe as classes que compõem a classe *GlobalWS*.

A.1 A classe Object e suas derivadas

A classe *Object* foi desenvolvida com o objectivo de modelar as características de um objecto real, para que outros objectos reais como robôs e bola pudessem possuir estruturas de base semelhantes diferindo apenas na sua especificidade.

A classe *Object* é constituída por diversos elementos de informação que permitem caracterizar parcialmente um objecto genérico, nomeadamente:

- posição absoluta do objecto;
- posição relativa do objecto (ao próprio robô);
- o objecto encontra-se visível?
- o objecto encontra-se visível com índice de confiança elevado?
- tempo de vida da informação.

Conforme ilustrado na Figura 6, as classes *Ball* e *Robot* derivam da classe *MobileObject*, que por sua vez deriva da classe *Object*. Ou seja, um objecto móvel (*MobileObject*) é um objecto que pode ter uma determinada velocidade. A bola é um objecto móvel. Um robô é um objecto móvel com uma orientação.

Observam-se ainda na mesma figura duas variáveis booleanas *see* e *seeXY*. Estas variáveis permitem determinar se o objecto se encontra visível com confiança elevada, com confiança moderada ou não se encontra visível de todo.

A existência de duas variáveis booleanas *see* e *seeXY* deve-se a que a visão dos robôs não apresenta os mesmos índices de confiança em todas as direcções. As lentes usadas no sistema de visão possuem um elevado nível de distorção que é parcialmente suprimido usando correcção por *software*. Esta correcção conduz a uma compressão da imagem que resulta em duas áreas distintas: área corrigida e área não-corrigida. Os objectos visíveis nesta última apresentam um grau de confiança diminuto por oposição aos objectos visíveis na área corrigida. Assim, a variável booleana *seeXY* assinala a visibilidade do objecto na área corrigida e a variável booleana *see* assinala a visibilidade do objecto na imagem (independentemente da área onde este se encontra).

A Figura 6 mostra adicionalmente os atributos que compõem as classes *Rotation* e *Vector*.

[2]A equipa do robô possui *MYTEAMSIZE* elementos (incluíndo ele próprio) e a equipa adversária *THEIRTEAMSIZE* elementos.

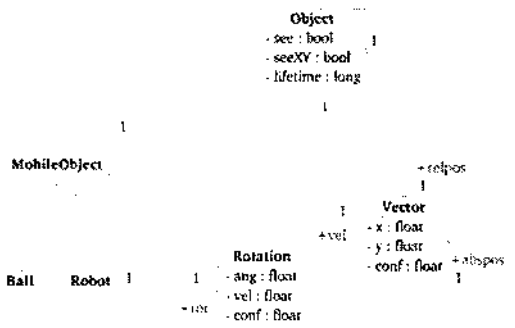

Figure 6 - Classe *Object* e suas derivadas

A.2 A classe Self

A classe *Self* (Figura 7) permite armazenar informação adicional do robô que não seja adequada para constar da classe *Robot*. Um robô possui um número que o distingue dos restantes e um "papel" na equipa que tipicamente está associado ao seu comportamento em campo. Esta classe integra dois atributos das classes *See* e *Distance* e uma variável para contabilizar a "idade" da informação. A classe *See* é apenas um conjunto de variáveis booleanas indicando a visibilidade das balizas e a classe *Distance* armazena a distância total que o robô percorreu desde que foi ligado.

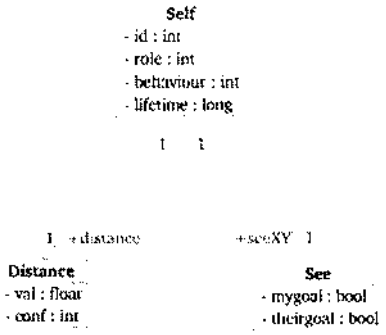

Figure 7 - Classe *Self*

A.3 A classe Team

A classe *Team* (Figura 8) armazena apenas informação relativa à equipa. Nestas circunstâncias identificou-se apenas a cor da equipa como atributo. Em fases de desenvolvimento posteriores outros atributos poderão surgir.

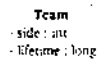

Figure 8 - Classe *Team*

A.4 A classe GlobalWS

A classe *GlobalWS* caracteriza o estado do mundo e agrega as classes anteriormente apresentadas. Esta classe possui adicionalmente informação sobre o número de ciclos de controlo executados, número de inconsistências ocorridas[3], estado do jogo no ciclo de controlo actual, estado do jogo no ciclo de controlo anterior e várias variáveis booleanas que são descritas em seguida:

- *base_station* - Indica se este agente é a estação base (verdadeiro/falso);
- *station_active* - Indica o estado da estação base (verdadeiro/falso);
- *ball_seen* - Indica se a bola foi vista no actual ciclo de controlo (verdadeiro/falso);
- *running* - Indica se "este" robô está em operação (verdadeiro/falso);
- *behaviour_changed* - Indica se o comportamento "deste" robô foi alterado no ciclo de controlo actual (verdadeiro/falso);
- *teammates_active[MYTEAMSIZE]* - *Array* de variáveis booleanas que assinala os agentes que estão activos. Entende-se por agentes activos aqueles cuja variável correspondente do *array* seja verdadeira.

A Figura 9 ilustra a classe *GlobalWS*.

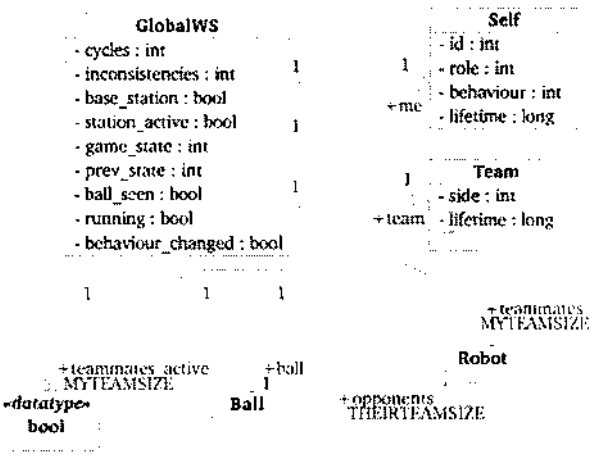

Figure 9 - Classe *GlobalWS* em detalhe

A informação que a classe *GlobalWS* agrega permite caracterizar a visão do mundo que um dado agente robótico possui. Esta informação é posteriormente usada no processo de decisão comportamental do robô.

B. O Treinador - Coach

A classe *Coach* é a estrutura de dados que modela a organização da informação correspondente ao treinador. Esta classe é usada para "transportar" informação de estado de jogo e parâmetros de início (ou re-ínicio) da partida.

Como se pode observar pela Figura 10 a classe *Coach* possui adicionalmente informação sobre a cor da equipa, estado da estação base (activa/inactiva), estado (activo/inactivo) e "papel" de cada um dos robôs que participam na partida.

[3]Consideram-se como inconsistências situações anómalas, i.e., situações originadas por medidas sensoriais inconsistentes. Um exemplo de uma situação destas é o robô determinar que a bola se encontra fora do terreno de jogo a uma distância de 100m deste.

Figure 10 - Classe *Coach*

C. O Sistema de Visão

As classes *FRNTVision* e *OMNIVision* foram desenvolvidas para responder à necessidade de comunicação entre os processos da visão e o processo de controlo.

A visão possui dois processos independentes que correspondem às duas câmaras existentes em cada robô: frontal e omnidireccional. A informação reunida por cada uma sobre os objectos ao seu alcance é escrita na classe correspondente; *FRNTVision* ou *OMNIVision*.

Apresentam-se em seguida as hierarquias das classes *FRNTVision* (Figura 11) e *OMNIVision* (Figura 12).

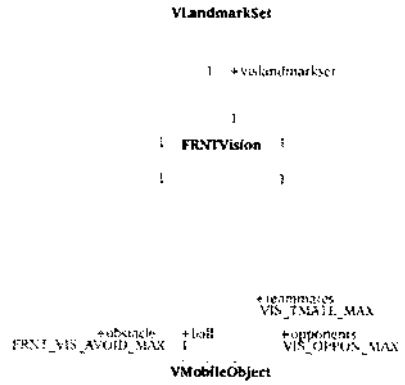

Figure 11 - Classe *FRNTVision*

Como se pode observar, estas classes são compostas por várias classes que modelam objectos físicos existentes no campo. A classe *FRNTVision* armazena informação relativa a objectos passíveis de serem detectados na câmara frontal (bola, obstáculos, adversários, colegas de equipa e referências do campo como balizas ou postes). De forma semelhante, a classe *OMNIVision* armazena informação relativa a objectos passíveis de serem detectados na câmara omnidireccional tais como: bola, obstáculos, linhas do campo, etc. A informação contida em ambas as classes é posteriormente integrada (pelo processo de controlo) no "estado do mundo".

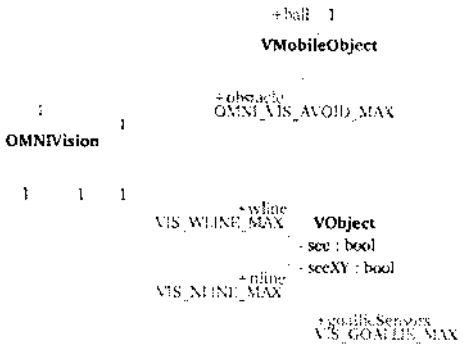

Figure 12 - Classe *OMNIVision*

As classes *FRNTVision* e *OMNIVision* usam as classes do tipo *VLandmark* e *VMobileObject* (Figura 13).

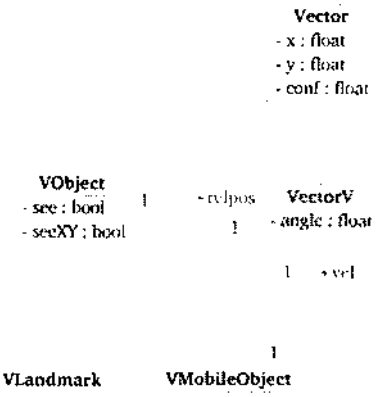

Figure 13 - Classe *VObject* e suas derivadas

Assim *VMobileObject* apenas difere de *VLandmark* por possuir uma velocidade, ou seja, objectos da classe *VMobileObject* são móveis.

Na classe *Vector* e na sua derivada *VectorV*, usada pela generalidade das classes associadas à interface com o sistema de visão artificial, os atributos x e y armazenam as coordenadas cartesianas do objecto visto na área corrigida da imagem. Se o objecto for visto fora da área corrigida, o sistema de visão fornece apenas o ângulo relativo a que o mesmo se encontra. Em ambos os casos existe uma confiança associada à medida.

Finalmente, a classe *VLandmarkSet* acomoda todas as *landmarks* (exceptuando as linhas) que a câmara frontal pode detectar (ou seja, balizas e postes).

C.1 *Os parâmetros das Câmaras* - Camera Settings

Verificou-se que cada câmara possui parâmetros de calibração específicos. Para armazenar estes pârametros desenvolveu-se a classe *CameraSettings* ilustrada na Figura 15.

Como se pode observar, esta classe possui atributos que permitem guardar toda a informação que caracteriza a

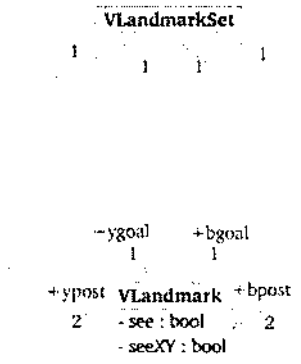

Figure 14 - Classe *VLandmarkSet*

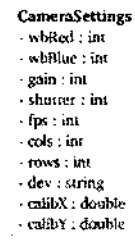

Figure 15 - Classe *CameraSettings*

calibração de uma câmara (balanceamento de "brancos", ganho, geometria da imagem, etc.).

D. Os Pontos de Referência do Campo - Landmarks

Uma vez que um campo de futebol robótico possui um conjunto de pontos de referência estáticos que podem auxiliar os robôs a localizarem-se no terreno de jogo, desenvolveu-se uma classe que permite guardar a localização real dos mesmas, i.e., a sua posição real no campo e não a percepção do robô (Figura 16).

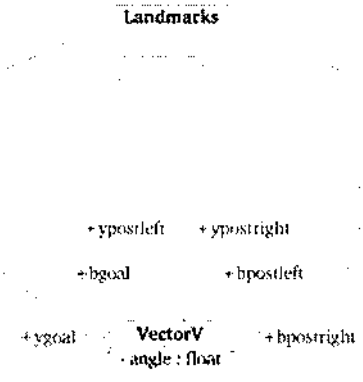

Figure 16 - Classe *Landmarks*

A informação recolhida pela visão em *FRNTVision* pode ser validada com auxílio da informação existente em *Landmarks*.

Convém notar que, embora a classe *VectorV* seja usada no âmbito da estrutura *Landmarks*, o atributo *angle* não é usado uma vez que se conhece com confiança máxima (1) a localização da marca no campo.

V. INTEGRAÇÃO DE INFORMAÇÃO

A integração de informação pretende construir um modelo interno do mundo tão preciso quanto possível, baseando-se nos dados sensoriais próprios e partilhados e nas acções executadas. Esta integração é realizada ao longo do tempo, minimizando o efeito do ruído normal dos sensores e a ausência momentânea de informação. Os dados que constam do modelo interno do mundo foram já referidos na secção anterior.

O posicionamento do robô no campo constitui o tema central desta secção. Como é facilmente perceptível, o conhecimento desse dado assume uma importância crucial numa equipa de futebol robótico. Os motivos que contribuem para a importância do conhecimento da posição absoluta descrevem-se em seguida:

- estimar as posições relativas das *landmarks* (balizas, postes, limites do campo, etc.);
- estimar as posições relativas dos seus *teammates* por forma a poder interagir com os mesmos (executar passes, etc.);
- aumentar o nível de coordenação da equipa, ao tornar possível a utilização de formações e tornar mais facilmente reconhecível qual a tarefa a desempenhar;
- permitir o planeamento de trajectórias;
- evitar colisões entre robôs e entre robôs e elementos do campo;
- decidir sobre qual a acção a executar.

O mecanismo de *Integração de Informação* foi desenvolvido tendo por base *know-how* proveniente da equipa "FC Portugal" [13]-[16] e de robótica móvel [17], [18].

As seguintes subsecções descrevem as limitações observadas na odometria e os mecanismos de correcção desenvolvidos para as ultrapassar.

A. Odometria

O sistema de odometria oferece tipicamente um mecanismo de localização simples, contudo, não suficientemente robusto para fornecer uma estimativa consistente da posição de um robô no decorrer de uma partida completa de futebol.

Foi levado a cabo um conjunto de experiências para avaliar o erro posicional proveniente do uso exclusivo de odometria na localização de um robô da equipa CAMBADA. Este conjunto de experiências foi realizado utilizando um robô que desempenha o papel de atacante até percorrer uma distância pré-definida. Durante a experiência um humano está constantemente a desviar a bola do robô obrigando-o a perseguir a bola. Ao fim de cada 10 metros são registadas a posição real e a posição indicada pela odometria. Em testes preliminares verificou-se que quando o robô é forçado a arranques e paragens súbitas e frequentes, o erro associado à odometria do robô aumenta muito rapidamente. Assim os testes foram efectuados de forma a evitar este tipo de arranques e paragens. Ao todo, foram realizados 10 percursos de

100 metros.

A.1 Erro em X e Y

As figuras 17 e 18 mostram a dependência do erro médio associado às componentes X e Y da posição absoluta em função da distância percorrida pelo robô. As barras verticais indicam o desvio padrão associado a cada valor médio de erro.

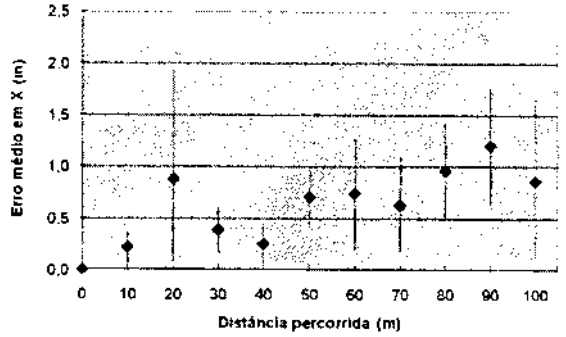

Figure 17 - Erro na componente X da posição absoluta

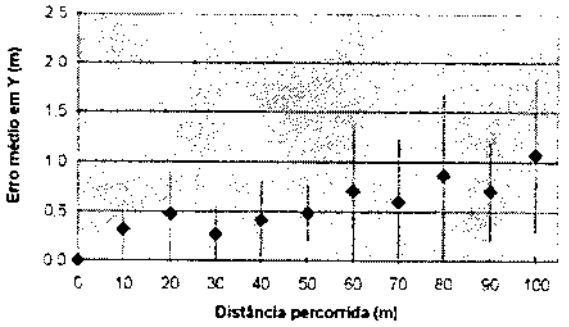

Figure 18 - Erro na componente Y da posição absoluta

Observa-se que, em média, o erro em ambas as componentes da posição absoluta tende a aumentar com a distância percorrida.

A.2 Erro posicional (módulo)

A Figura 19 mostra o valor médio do erro posicional (em módulo) e respectivo desvio padrão em função da distância percorrida. A figura inclui uma linha de tendência de evolução do valor do erro com a distância.

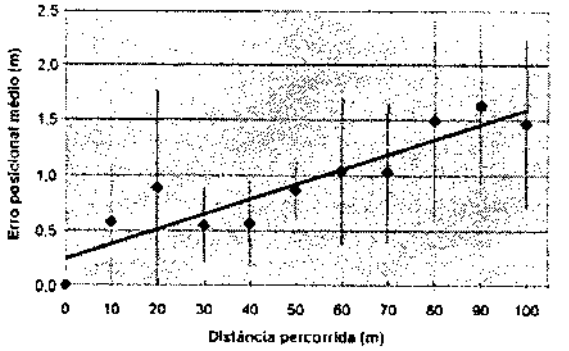

Figure 19 - Erro Posicional *versus* Distância

Conclui-se das experiências realizadas que, para distâncias percorridas até a $100m$, o erro posicional médio é uma função aproximadamente linear da distância percorrida, tendendo para cerca de 1.5% da distância percorrida. Verifica-se que para o conjunto dos 10 percursos realizados, e considerando todos os momentos de observação, o erro médio global situa-se em torno de 1 metro. Este valor pode ser considerado aceitável, dado que o campo de futebol tem habitualmente a dimensão de 12 metros de comprimento por 8 metros de largura.

Convém, no entanto, notar, como já foi referido, que estas experiências foram realizadas com o cuidado de evitar situações que reconhecidamente aumentam o erro de forma significativa. Além disso, mesmo considerando o pior caso dado pela figura 19 (média + desvio padrão), já se obtem níveis de erro menos bons ($\sim 1.6m$). Mais, ao aproximar-se dos 100 m, o pior caso do erro atinge cerca de 2.5 metros, valor que já se aproxima de uma situação de incerteza causadora de problemas ao nível, por exemplo, da coordenação da equipa.

Por estas razões, é necessário desenvolver mecanismos paralelos à odometria que forneçam uma melhor estimativa da posição do robô no decorrer de uma partida completa. Estes mecanismos são apresentados nas subsecções seguintes.

B. Calibração visual da posição absoluta do robô

Desenvolveram-se dois mecanismos distintos de calibração visual da posição absoluta. O primeiro mecanismo de calibração baseia-se no facto de que é possível realizar uma melhoria da estimativa da posição absoluta com base no conhecimento da posição relativa de uma *landmark* (na presente circunstância um poste). O segundo mecanismo de calibração é baseado no conhecimento da posição relativa de duas *landmarks* (dois postes) permitindo fazer a triangulação com o robô e determinar a sua posição absoluta com maior confiança. Nenhum dos mecanismos é totalmente infalível dado que ambos se baseiam em posições obtidas a partir do sistema de visão (que possui erros associados), no entanto, verificaram-se melhorias de desempenho assinaláveis com o seu uso. As secções seguintes abordam mais detalhadamente cada um dos mecanismos.

É importante notar que, tal como acontece com o sistema de odometria, também o sistema de visão fornece informação afectada por erros. Esses erros resultam de várias circunstâncias. À partida, as câmaras deverão estar posicionadas nos robôs exactamente na posição definida na fase de projecto. Para o garantir, é necessário calibrar a configuração física do sistema com alguma regularidade. Entretanto, mesmo nesse pressuposto, pequenas irregularidades no campo, ou até as próprias oscilações resultantes do movimento do robô, levam a alterações pontuais da posição das câmaras relativamente ao campo. Essas alterações podem levar o robô a ver objectos, não nas suas reais posições, mas sim a alguns (ou mesmo muitos) metros de distância. Na fig. 20, apresenta-se, para distâncias entre 0 e 15m, a curva do erro na distância dada pelo sistema de visão (em m), quando a inclinação da câmara frontal é superior à inclinação projectada por uma diferença de 1°. Como se

vê, o erro é tolerável apenas até 6m de distância. A partir dos 8m, a utilização da informação do sistema de visão exige grande cuidado. Esta é uma restrição importante dado que um campo tem normalmente 8 m por 12 m.

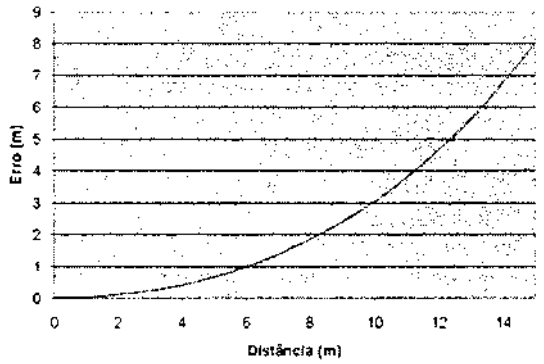

Figure 20 - Erro Posicional resultante de uma diferença de 1° na inclinação da câmara.

B.1 Calibração baseada num único poste

Como referido, o mecanismo de calibração baseado na posição de um único poste permite apenas realizar uma melhoria à estimativa da posição absoluta e não uma correcção completa do seu erro. Este facto pode ser observado na figura 21. Note-se que a direcção do robô é assinalada com uma seta.

Na posição 1, a frente do robô faz um ângulo ø com o poste e está a uma distância d do mesmo. Na posição 2 repetem-se as circunstâncias, ou seja, a frente do robô faz um ângulo ø com o poste e está a uma distância *d* do mesmo. Verifica-se claramente que não é possível calibrar a posição do robô com base em apenas um *landmark* (poste) dado que estando o robô em posições distintas poderá "ver" o poste na mesma posição relativa.

Assim, usando este mecanismo de calibração é apenas possível melhorar a estimativa da posição absoluta. A figura 22 ilustra o conceito. Este mecanismo de calibração parcial baseia-se em três considerações:

- Assume-se que erros na estimativa actual da posição absoluta do robô não são significativamente elevados e portanto esta pode ser usada no cálculo da nova estimativa.
- Com base na distância relativa ao poste, é possível traçar uma circunferência que indica as posições absolutas em que o robô poderia estar com base na distância relativa ao poste. Se se traçar uma recta entre o poste e a posição estimada[4] do robô intersecta-se a circunferência num ponto que seria o ponto de calibração. Contudo, este ponto não é usado uma vez que os erros associados ao sistema de visão levaram-nos a pesar com igual peso a estimativa existente e o valor da distância obtido através da visão.
- Cruzando a informação anterior (realizando uma média pesada entre as posições actual e induzida a partir da visão) obtém-se a estimativa final também assinalada na figura 22.

[4]Posição estimada com base na informação existente no "Estado do Mundo" do robô.

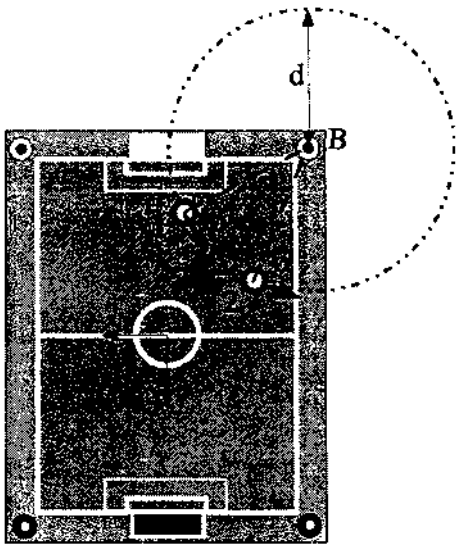

Figure 21 - Calibração posicional baseada num poste - o problema

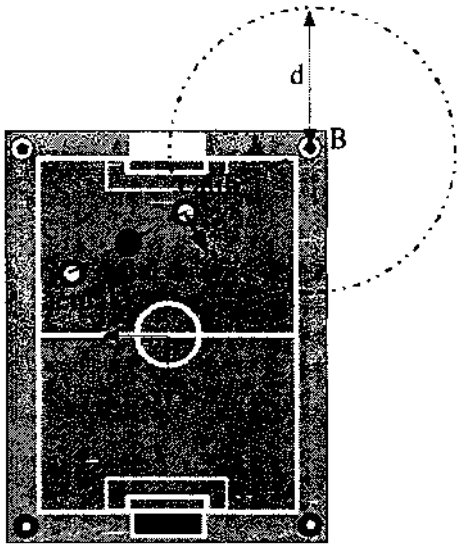

Figure 22 - Calibração posicional baseada num poste - a solução

B.2 Calibração baseada em triangulação

A correcção da posição absoluta baseada num único poste mostrou-se insuficiente para que o robô possuísse uma estimativa consistente da mesma. A solução descrita de seguida baseia-se no facto de que é possível corrigir integralmente a posição absoluta de um robô a partir de duas *landmarks* (nestas circunstâncias, dois postes).

A Figura 23 identifica um *set-up* de demonstração da teoria associada. Nesta figura, observa-se um triângulo constituído por dois postes e pelo robô. As letras maiúsculas referem-se aos ângulos e as minúsculas ao comprimento dos segmentos de recta. Considerou-se ainda um sistema de coordenadas cartesiano com as direcções assinaladas na figura.

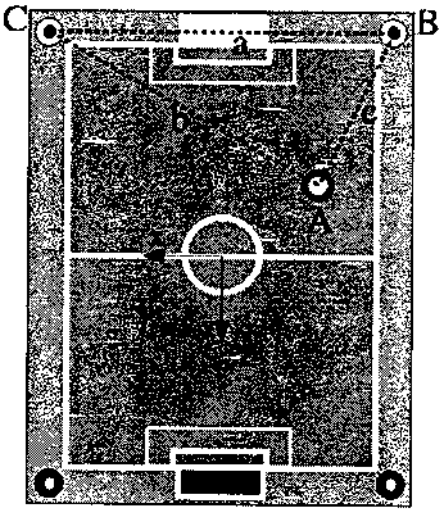

Figure 23 - Calibração posicional baseada em triangulação

A equação 1 permite determinar o ângulo entre o segmento de recta que une os dois postes e o segmento de recta que une o poste esquerdo ao robô (na figura 23 - ângulo C) a partir apenas das distâncias entre os três objectos.

$$C = \arccos\left(\frac{a^2 + b^2 - c^2}{2.a.b}\right) \quad (1)$$

A equação 2 (que varia conforme os eixos considerados) permite realizar a calibração da posição absoluta do robô com base no ângulo anteriormente determinado, nas distâncias entre os três elementos e na posição absoluta do poste esquerdo.

$$Robot(x,y) = \Big(Ypostleft_x - b.\cos C, \\ Ypostleft_y + b.\sin C\Big) \quad (2)$$

Verificou-se experimentalmente que apenas em ocasiões muito raras o robô "vê" os dois postes em simultâneo. Tal deve-se sobretudo à baixa resolução das imagens do sistema de visão e ao facto de que este sistema não possui visão omnidireccional a mais do que 1m.

Para ultrapassar esta limitação, desenvolveu-se um mecanismo de correcção activa da posição absoluta. Este mecanismo caracteriza-se por um comportamento especial através do qual o robô calibra a sua posição, mas que implica o abandono do seu envolvimento directo no jogo durante algum tempo. Em seguida descreve-se sucintamente este comportamento.

1. Rotação do robô de 360 graus pesquisando a existência de postes. Para cada poste detectado armazena-se o ângulo e a distância correspondentes.
2. Após a conclusão da rotação verifica-se se foram encontrados 2 ou mais postes:
   (a) Se não – o procedimento de calibração termina sem a realizar.
   (b) Se sim – os dados recolhidos são filtrados de forma a eliminar possíveis inconsistências. Após esta filtragem verifica se existem postes em posições consistentes e em número suficiente (2 no mesmo meio-campo) para realizar uma calibração completa:
      i. Se não – o mecanismo de correcção termina sem a realizar.
      ii. Se sim – usa as equações apresentadas anteriormente para realizar uma calibração completa da posição absoluta do robô.

Como se pode observar pela descrição anterior, nesta fase não se considerou a possibilidade dos postes pertencerem a lados distintos do campo. Optou-se por esta solução dado que quando o robô está de um dos lados do campo apenas consegue detectar os postes existentes na sua proximidade, logo, os postes que se encontram do lado oposto do campo dificilmente serão visíveis, e se o forem, o erro associado às suas distâncias faz com que sejam "eliminados" no processo de filtragem descrito.

C. *Outros mecanismos de tratamento de erros*

Para além dos mecanismos de calibração visual da posição absoluta descritos acima, outros mecanismos de tratamento de erros foram desenvolvidos, o quais se passa a descrever.

C.1 *Calibração da rotação*

Em alguns testes controlados (apenas rotação, apenas translação, etc.), verificou-se que o sistema de odometria fornecia valores de rotação afectados de erros sistemáticos, os quais se revelaram difíceis de anular na camada de *software* de baixo nível. Este problema acaba também por se reflectir nas componentes X e Y da posição absoluta.

Para corrigir este erro, mede-se o incremento de rotação em cada iteração do ciclo de controlo, multiplicando-o por um factor de calibração. Tipicamente existem dois factores de calibração de rotação, um para o movimento no sentido horário dos ponteiros do relógio e outro para o sentido oposto. É ainda de salientar que os parâmetros de calibração são específicos de cada robô dado que cada um deles possui uma geometria física própria.

Os dados experimentais expostos anteriormente foram colhidos tendo já activado este mecanismo de calibração.

C.2 *Tratamento de inconsistências*

Numa situação de funcionamento normal do robô, admite-se que a estimativa da posição do robô é suficientemente fiável, podendo-se derivar as posições relativas dos restantes elementos fixos existentes no terreno de jogo. Quando um determinado elemento imóvel (poste, baliza, etc.) é detectado pelo sistema de visão e transmitido ao sistema de controlo de alto nível, este verifica se o mesmo se encontra numa posição coerente com o seu "estado do mundo". Isto significa que, se a posição do elemento não se aproximar (por uma determinada margem) da estimativa actual da sua posição, a informação recém-adquirida é ignorada, sendo então incrementado um contador de inconsistências. Quando o número de inconsistências atinge um determinado limite[5], o processo de controlo instrui o

[5]O facto de existir um número elevado de inconsistências é indicativo de que a posição absoluta do robô está corrompida. Nestas circunstâncias é imperativo que se realize uma calibração da posição absoluta.

robô para realizar uma calibração activa (ver subsecção *Calibração baseada em triangulação*).

C.3 Limites nas coordenadas X e Y

Quando o robô se aproxima dos limites físicos do campo, um erro posicional, ainda que pequeno, facilmente resulta em coordenadas *X* e *Y* absolutas que excedem esses limites. Esta situação constitui uma oportunidade de calibração. Assim, sempre que uma das coordenadas sai do intervalo de valores possíveis (num campo normal, -7.0m a 7.0m em Y e -5.0m a 5.0m em X), o respectivo valor é corrigido no Estado do Mundo, passando a ser o limite que foi ultrapassado.

C.4 Integração de informação ao longo do tempo

Com vista à construção de um Estado do Mundo mais fiável do que aquele que se obtém através de calibrações pontuais, foi iniciado o desenvolvimento de um módulo de integração de informação ao longo do tempo. Neste caso, em vez de usar informações pontuais sobre a posição dos pontos de referência (objectos fixos como balizas, postes, etc.), mantém-se internamente actualizada a posição relativa de cada um deles. De cada vez que um dado ponto de referência é detectado, a respectiva posição relativa é actualizada através de uma média ponderada com a posição actual, desde que não haja inconsistência clara entre ambas. Entretanto, dado que os pontos de referência têm uma posição relativa (de uns ao outros) imutável, os valores das posições desses pontos de referência relativamente ao robô são corrigidas por forma a aproximarem-se da sua posição correcta no campo.

Este módulo de integração de informação sobre os pontos de referência ainda não está devidamente avaliado.

VI. CONCLUSÃO

Este artigo apresentou a arquitectura de software de um agente robótico da equipa CAMBADA focando essencialmente nas estruturas de dados e nos mecanismos de integração de informação. Para maior clareza, as estruturas de dados são apresentadas em UML, sendo também identificados os problemas associados.

Os mecanismos de integração de informação que foram desenvolvidos e descritos têm por objectivo principal estimar as posições absolutas do robô e dos outros objectos móveis existentes no campo. Nesta fase, integra-se essencialmente informação odométrica e visual. Na continuação dos trabalhos, estão a merecer atenção os seguintes aspectos:

- Desenvolver mecanismos adicionais de calibração da posição do robô.
- Avaliação e eventual utilização do mecanismo de integração de informação ao longo do tempo.
- Desenho e implementação do sistema de gestão de confianças.
- Integração de informação ao nível da equipa.

REFERENCES

[1] CAMBADA Homepage, http://www.ieeta.pt/atri/cambada/, February, 2005

[2] RoboCup Official Web Site, http://www.robocup.org/, February, 2005

[3] Transverse Activity on Intelligent Robotics Homepage, http://www.ieeta.pt/atri/, February, 2005

[4] Institute of Electronics and Telematics Engineering of Aveiro Homepage, http://www.ieeta.pt/, February, 2005

[5] Fédération Internationale de Football Association, http://www.fifa.com/en/index.html, February, 2005

[6] RoboCup, Middle Size Robot League Rules and Regulations for 2004, Draft Version pre-8.3, June, 2004

[7] Wi-Fi Alliance, http://www.wi-fi.org, February, 2005

[8] L. Almeida, F. Santos, T. Facchinetti, P. Pedreiras, V. Silva and L. Seabra Lopes, "Coordinating Distributed Autonomous Agents with a Real-Time Database: The CAMBADA Project", Computer and Information Sciences - ISCIS 2004: 19th International Symposium, Proceedings, Aykanat, Cevdet; Dayar, Tugrul; Korpeoglu, Ibrahim (Eds.), Lecture Notes in Computer Science, Vol. 3280, p. 876-886.

[9] F. Santos, L. Almeida, P. Pedreiras, L. Seabra Lopes, T. Facchinetti, "An Adaptive TDMA Protocol for Soft Real-Time Wireless Communication among Mobile Autonomous Agents", Proc. WACERTS'2004 (em publicação).

[10] Unified Modeling Language Resource Page, http://www.uml.org/, February, 2005

[11] James Rumbaugh, Ivar Jacobson, Grady Booch, "The Unified Modeling Language Reference Manual", Addison-Wesley, 1999

[12] Martin Fowler, Kendall Scott, "UML Distilled Second Edition - A Brief Guide to the Standard Object Modeling Language", Addison-Wesley, 2000

[13] FC Portugal Homepage, http://www.ieeta.pt/robocup/index.htm, February, 2005

[14] L.P. Reis, J.N. Lau, E.C. Oliveira, "Situation Based Strategic Positioning for Coordinating a Team of Homogeneous Agents", Balancing Reactivity and Social Deliberation in Multi-Agent Systems, Markus Hannebauer, Jan Wendler, Enrico Pagello, editors, LNCS 2103, 175-197, Springer Verlag, 2001

[15] L.P. Reis, J.N. Lau, "FC Portugal Team Description: RoboCup 2000 Simulation League Champion", RoboCup-2000: Robot Soccer World Cup IV, Peter Stone, Tucker Balch and Gerhard Kraetzschmar editors, LNAI 2019, 29-40, Springer Verlag, Berlin, 2001

[16] J.N. Lau, L.P. Reis, "FC Portugal 2001 Team Description: Flexible Teamwork and Configurable Strategy", RoboCup-2001: Robot Soccer World Cup V, Andreas Birk, Silvia Coradeshi, Satoshi Tadokoro editors, LNAI, Springer Verlag, Berlin, 2002

[17] L. Seabra Lopes, J.N. Lau, L.P. Reis, "Intelligent Control and Decision-Making demonstrated on a Simple Compass-Guided Robot", Proceedings of SMC'2000, IEEE Int. Conference on Systems, Man and Cybernetics, Nashville, USA, October 2000

[18] L. Seabra Lopes, "Carl: from Situated Activity to Language Level Interaction and Learning", Proc. IEEE Int'l Conf. on Intelligent Robots and Systems (IROS), Lausanne, Switzerland, p. 890-896, 2002