



**Universidade do Minho**  
Escola de Engenharia

Sérgio Filipe Veloso Silva

**Controlo e monitorização da equipa de robôs  
futebolistas MINHO TEAM**

Sérgio Filipe Veloso Silva **Controlo e monitorização da equipa de robôs futebolistas MINHO TEAM**

UMinho | 2011

Outubro de 2011



**Universidade do Minho**  
Escola de Engenharia

Sérgio Filipe Veloso Silva

## **Controlo e monitorização da equipa de robôs futebolistas MINHO TEAM**

Mestrado Integrado em Engenharia Electrónica  
Industrial e Computadores

Trabalho realizado sob a orientação do  
**Professor Doutor António Fernando Macedo Ribeiro**



“A grandeza não consiste em receber honras, mas em  
merecê-las.”  
(Aristóteles)



## **AGRADECIMENTOS**

Gostaria de expressar o meu reconhecimento e gratidão a algumas pessoas, cujo apoio tornou possível a realização deste trabalho.

Em primeiro lugar, aos meus familiares em especial aos meus pais Joaquim Costa e Elvira Veloso, por todo o esforço e apoio que me deram ao longo de todos estes anos como estudante.

Quero agradecer ao meu orientador, Doutor Fernando Ribeiro, por todo o apoio e orientação dada, e pela confiança depositada nas minhas capacidades.

A toda a equipa actual de futebol robótico MINHO TEAM, bem como a todos os colegas que frequentam o Laboratório de Automação e Robótica.



## RESUMO

O controlo e a análise do comportamento de robôs em tempo real pode ser uma tarefa difícil, sobretudo quando se trata de robôs cooperantes. Neste contexto a utilização de um software de monitorização e controlo torna-se uma ferramenta essencial no desenvolvimento de uma equipa de robôs cooperativos.

MINHO TEAM é a equipa de futebol robótico da Universidade do Minho que participa na liga de futebol robótico médio no RoboCup. A equipa foi criada pelo Laboratório de Automação e Robótica (LAR) que está inserido no Departamento de Electrónica Industrial (DEI).

O objectivo desta dissertação é criar um software de monitorização e controlo para uma equipa de futebol robótico. O software deve funcionar autonomamente e exibir num ecrã informações sobre o estado dos robôs (como localização em campo, capacidade das baterias, detecção da bola e obstáculos), assim como gerir as posições e papéis atribuídos aos robôs fazendo com que joguem em equipa. O software deve conter outras funcionalidades relacionadas com as necessidades da equipa MINHO TEAM, como criar e editar tácticas para as situações de bola parada, reproduzir os jogos com base em registos ou processar os comandos recebidos da *Referee Box*.

O software criado foi usado no desenvolvimento da nova geração de robôs futebolistas - no auxílio da produção e teste dos algoritmos de localização e de controlo, e permitiu a participação da equipa MINHO TEAM na prova do futebol robótico médio do RoboCup 2011 realizada em Istambul.

Palavras-chave: RoboCup, futebol robótico, controlo, monitorização, sistemas autónomos, cooperação, coordenação, estratégia.





## **ABSTRACT**

The robot behavior control and analysis in real time can be a hard task, particularly what concerns cooperative robots. In this context, the use of control and monitoring software becomes an essential tool for the development of cooperative robots team.

MINHO TEAM is a robotics soccer team from University of Minho that participates in RoboCup Middle Size League. The team was developed on the Laboratory of Automation and Robotics which belongs to the Department of Industrial Electronics.

The purpose of this work consists of developing a monitoring and control software for a robotic soccer team. The software should run autonomously and show information regarding the state of the robots (such as localization on the field, batteries capacity, detection of ball and obstacles), as well as manage positions and roles assigned to robots, making them to play as a team. The software should also include other features required by MINHO TEAM, such as creating and editing tactics for set pieces, replay games based on previous logs or process commands received by the Referee Box.

The engineered software was written for the next generation of soccer robots, to help the development and testing of localization and control algorithms, and allowed the participation of MINHO TEAM on RoboCup 2011 Middle Size League held in Istanbul.

Keywords: RoboCup, robotic soccer, control, monitoring, autonomous systems, cooperation, coordination, strategy.



## ÍNDICE

Agradecimentos.....	V
Resumo.....	VII
Abstract.....	IX
Índice de Figuras.....	XV
Índice de Tabelas.....	XIX
Lista de Acrónimos.....	XXI
1. Introdução.....	1
1.1. O Problema.....	1
1.2. Solução Proposta.....	1
1.3. Objectivos.....	2
1.4. Robótica.....	2
1.4.1. RoboCup.....	3
1.5. Estrutura da Tese.....	6
2. Estado da Arte.....	7
2.1. Water.....	7
2.1.1. Water Coach.....	8
2.2. Tech United Eindhoven.....	9
2.2.1. Turtle Remote Control.....	9
2.3. CMBADA.....	11
2.3.1. CMBADA Basestation.....	11
2.3.2. Posicionamentos.....	12
2.4. 5DPO.....	13
2.4.1. Coach.....	13
2.5. Conclusões.....	14

---

3. MINHO TEAM .....	17
3.1. Descrição do hardware .....	17
3.2. Descrição do software .....	19
3.2.1. Algoritmo de Localização.....	19
3.2.2. Máquina de estados.....	19
4. Arquitectura da MINHO BaseStation.....	21
4.1. Requisitos .....	21
4.2. Visão Geral.....	23
4.3. Protocolo de Comunicações.....	26
4.3.1. Comunicação entre MINHO BaseStation e <i>RefBox</i> .....	27
4.3.2. Comunicação entre MINHO BaseStation e robôs .....	28
4.4. Modo de Jogo.....	29
4.4.1. Estrutura do Modo de Jogo .....	30
4.4.2. Variáveis .....	32
4.4.3. Máquina de Estados.....	34
4.4.4. Comunicação.....	40
4.5. Modo de Edição .....	41
4.5.1. Estrutura do Modo de Edição .....	42
4.5.2. Controlo Manual do Robô .....	44
4.5.3. Editor de Bolas Paradas .....	44
4.5.4. Editor de Papéis.....	49
4.5.5. Localização.....	50
4.5.6. Parâmetros do Jogo .....	51
4.5.7. Parâmetros do Robô .....	51
4.5.8. Comunicação.....	51
4.6. Modo de Análise.....	52

4.6.1.	Estrutura do Modo de Análise.....	53
4.6.2.	Reprodução de jogos.....	54
4.7.	Configurações .....	55
4.7.1.	Estrutura das Configurações.....	56
4.7.2.	Campo e Agentes.....	57
4.7.3.	Rede.....	58
4.7.4.	Verificar Ficheiros .....	58
4.7.5.	Transferir Ficheiros .....	59
4.7.6.	Comunicação.....	59
4.8.	Preferências .....	60
4.8.1.	Estrutura das Preferências .....	61
5.	Resultados.....	63
5.1.	Modo de Jogo.....	63
5.1.1.	Cálculo da posição da bola.....	69
5.1.2.	Atribuição de papéis.....	71
5.2.	Modo de Edição .....	73
5.2.1.	Controlo Manual do Robô.....	73
5.2.2.	Editor de Bolas Paradas .....	74
5.2.3.	Editor de Papéis.....	78
5.2.4.	Localização.....	79
5.2.5.	Parâmetros do Jogo .....	81
5.2.6.	Parâmetros do Robô .....	82
5.3.	Modo de Análise.....	82
5.4.	Configurações .....	85
5.4.1.	Campo e Agentes.....	85
5.4.2.	Rede.....	86

---

5.4.3. Gerir Ficheiros .....	87
5.4.4. Transferir Ficheiros .....	88
5.5. Preferências .....	90
6. Conclusão e Trabalho Futuro .....	93
6.1. Conclusões .....	93
6.2. Trabalho Futuro.....	94
7. Bibliografia .....	95
Apêndice A.....	99
Apêndice B.....	100

## ÍNDICE DE FIGURAS

Figura 1 - Foto de um jogo da liga de futebol robótico médio do RoboCup 2011.....	4
Figura 2 - Representação do campo de jogo com as dimensões oficiais. ....	5
Figura 3 - Imagem do programa de arbitragem <i>Referee Box</i> .....	6
Figura 4 - Water Coach [15]. ....	8
Figura 5 - Esquema de comunicação da equipa Water [16]......	9
Figura 6 – Turtle Remote Control [19]. ....	10
Figura 7 - Diagrama do simulador da equipa Tech United Eindhoven [19]. ....	11
Figura 8 - CAMBADA Basestation [22]. ....	12
Figura 9 - Ferramenta desenvolvida pela equipa CAMBADA para definir posicionamentos [20]. ....	13
Figura 10 - 5DPO Coach [24]. ....	14
Figura 11 - Foto da nova geração de robôs da equipa MINHO TEAM. ....	17
Figura 12 - Esquema das conexões de hardware de cada robô. ....	18
Figura 13 - Esquerda: Pesquisa radial de linhas. Direita: Campo binário. ....	19
Figura 14 - Diagrama da máquina de estados implementada no robô. ....	20
Figura 15 - Esquema da MINHO BaseStation do ponto de vista do utilizador. ....	23
Figura 16 - Diagrama de classes da MINHO BaseStation. ....	24
Figura 17- Diagrama de objectos da MINHO BaseStation. ....	24
Figura 18 - Diagrama de classes da classe CConfigs. ....	25
Figura 19 - Topologias de ligações de rede. ....	27
Figura 20 - Ligações de rede implementadas na MINHO BaseStation.....	28
Figura 21 - Diagrama de classes do Modo de Jogo. ....	30
Figura 22 - Estrutura da base de dados implementada. ....	33
Figura 23 - Diagrama de transição de estados do Modo de Jogo. ....	34
Figura 24 - Fluxograma do estado Processa_Dados. ....	35
Figura 26 - Formato dos dados da <i>RefBox</i> guardados em ficheiro. ....	37
Figura 27 - Formato dos dados dos robôs guardados em ficheiro.....	37
Figura 28 - Formato dos dados do sistema guardados em ficheiro. ....	37
Figura 29 - Fluxograma do estado Actualiza.....	38
Figura 30 – Comunicação realizada no Modo de Jogo. ....	40
Figura 31 - Diagrama de classes do Modo de Edição. ....	42
Figura 32 - Sistema de eixos adoptado. ....	44
Figura 33 - Constituição de cada nó no ficheiro das tácticas. ....	45



---

Figura 34 - Exemplo da obtenção de posições num lançamento.....	46
Figura 35 - Ilustração do cálculo de posições nas faltas. ....	48
Figura 36 - Comunicação realizada no Modo de Edição. ....	52
Figura 37 - Diagrama de classes do Modo de Análise. ....	53
Figura 38 - Fluxograma da reprodução de jogos. ....	54
Figura 39 - Diagrama de classes das Configurações. ....	56
Figura 40 - Comunicação efectuada em Configurações.....	60
Figura 41 - Diagrama de classes das Preferências. ....	61
Figura 42 - Menu da MINHO BaseStation. ....	63
Figura 43 - Interface gráfico do Modo de Jogo. ....	64
Figura 44 - Histórico da <i>RefBox</i> , tempo e resultado de jogo.....	64
Figura 45 - Campo de jogo. ....	65
Figura 46 - Painel informativo de um robô. ....	65
Figura 47 - Barra de menu detalhada. ....	66
Figura 48 - Alterar parâmetros de jogo. ....	66
Figura 49 - <i>RefBox</i> virtual. ....	67
Figura 50 - Ver tácticas atribuídas. ....	67
Figura 51 - Ver informação da base de dados.....	68
Figura 52 - Ver papéis atribuídos. ....	68
Figura 53 - Configurar conexão com os robôs. ....	69
Figura 54 - Configurar conexão com a <i>RefBox</i> .....	69
Figura 55 - Posições reais dos robôs e da bola. ....	70
Figura 56 - Observação e cálculo da posição da bola. ....	70
Figura 57 - Observação e cálculo da posição da bola com um robô não localizado. ....	71
Figura 58 - Atribuição de papéis durante um jogo. ....	72
Figura 59 - Troca de papéis.....	72
Figura 60 - Controlo manual do robô. ....	73
Figura 61 - Painel para controlar manualmente cada robô. ....	74
Figura 62 - Editor de bolas paradas. ....	74
Figura 63 - Pontapé de saída.....	75
Figura 64 - Pontapé de baliza. ....	75
Figura 65 - Marcação de cantos. ....	76
Figura 66 - Penáلتi. ....	76
Figura 67 - Estacionar. ....	76
Figura 68 - Definição de lançamentos.....	77

Figura 69 - Definição de faltas.....	78
Figura 70 - Editor de papéis. ....	78
Figura 71 - Desenho dos pontos detectados pelos robôs.....	79
Figura 72 - Representação dos pontos e distâncias às linhas. ....	80
Figura 73 - Visualização do ficheiro das distâncias calculado por cada robô. ....	80
Figura 74 - Parte do campo binário gerado.....	81
Figura 75 - Escolher parâmetros do jogo. ....	81
Figura 76 - Alterar parâmetros do robô.....	82
Figura 77 - Modo de Análise.....	83
Figura 78 - Escolha de ficheiros para reprodução. ....	83
Figura 79 - Tempo total e actual da reprodução de um jogo.....	84
Figura 80 - Reprodução de um jogo no Modo de Análise.....	84
Figura 81 - Estatísticas de um jogo realizado. ....	85
Figura 82 - Separador campo e agentes. ....	86
Figura 83 - Configurações de rede.....	86
Figura 84 - Lista dos ficheiros existentes nos robôs.....	87
Figura 85 - Comparação dos ficheiros. ....	88
Figura 86 - Relatório de erros. ....	88
Figura 87 - Transferência de ficheiros para os robôs.....	89
Figura 88 - Progresso da transferência de ficheiros.....	89
Figura 89 - Interface gráfica das Preferências. ....	90
Figura 90 - Criação de um novo ficheiro de rede.....	91



## ÍNDICE DE TABELAS

Tabela 1 - Comparação das características dos softwares das equipas analisadas.....	15
Tabela 2 - Lista de eventos fornecida pela <i>RefBox</i> virtual.....	32
Tabela 3 - Descodificação do protocolo2009 da <i>RefBox</i> . ....	99



## **LISTA DE ACRÓNIMOS**

RefBox – *Referee Box*

MSL – *Middle Size League*

IDE - *Integrated Development Environment*

USB – *Universal Serial Bus*

LCD – *Liquid Crystal Display*

UTP – *Unshielded Twisted Pair*

XML – *Extensible Markup Language*

TCP/IP – *Transmission Control Protocol/Internet Protocol*

UDP – *User Datagram Protocol*

FIFO – *First In First Out*

UML – *Unified Modeling Language*



## 1. INTRODUÇÃO

O RoboCup [1] tem como função promover o desenvolvimento da Inteligência Artificial e da robótica, através de um conjunto de desafios propostos. MINHO TEAM [2] é uma equipa da Universidade do Minho e participa na liga de futebol robótico médio do RoboCup.

Os jogos da liga de futebol robótico médio são disputados por duas equipas com cinco robôs cada totalmente autónomos. Cada robô tem uma largura máxima de 52 cm, uma altura máxima de 80 cm e um peso máximo de 40 kg. O campo de jogo tem 18 metros de comprimento por 12 metros de largura, cada baliza mede 1 metro de altura e tem 2 metros entre postes [3].

### 1.1. O PROBLEMA

No futebol robótico, para a realização de jogos as equipas precisam de possuir além dos robôs uma aplicação de software externa totalmente autónoma (conhecida por *BaseStation*), cujo único requisito consiste em enviar os comandos recebidos pela *RefBox* (*Referee Box* [4]) para os robôs.

A *BaseStation* tem também um papel fundamental relativamente ao jogo colectivo e à análise do jogo, porque troca dados com os robôs, atribui posições e papéis aos robôs durante os jogos e exibe num monitor o estado dos robôs (como a sua localização em campo, por exemplo).

É também recomendada a existência de uma ferramenta gráfica capaz de criar e editar tácticas para todas as situações de bolas paradas de uma forma rápida e simples, sem necessitar de recompilar nenhum código.

### 1.2. SOLUÇÃO PROPOSTA

Como resposta ao problema proposto em 1.1, foi desenvolvida uma aplicação de software com vários módulos independentes. Um dos módulos será o Modo de Jogo, onde a aplicação terá um comportamento autónomo e durante a realização de jogos além de monitorizar o estado dos robôs irá também controlar qual o seu papel em campo. No Modo de

---

<sup>1</sup> A *RefBox* consiste na tecnologia usada para auxílio da arbitragem de jogos, nomeadamente para comunicar as decisões do árbitro aos robôs e registar todos os eventos ocorridos.



Edição será possível criar e editar tácticas de forma simples e rápida, sem a recompilação de código. Outros módulos serão incluídos na aplicação caso seja necessário satisfazer outras necessidades da equipa MINHO TEAM.

### **1.3. OBJECTIVOS**

O grande desafio deste projecto consiste em desenvolver uma aplicação de software capaz de monitorizar e controlar a equipa de robôs futebolistas MINHO TEAM. A aplicação deve executar autonomamente durante os jogos e cumprir o requisito mínimo de enviar os comandos do árbitro para os robôs.

Os objectivos propostos consistem em:

- Desenvolver uma aplicação em modo gráfico para a definição de tácticas e estratégias de jogo sem necessidade de recompilar código;
- Definir uma estrutura, elementos e formato para a base de dados;
- Criar um módulo de comunicação para a troca de informações com os robôs a fim de monitorizar o jogo;
- Desenvolver um módulo de análise dentro do modo gráfico que possibilite reproduzir jogos com base em registos previamente guardados.

### **1.4. ROBÓTICA**

A robótica é uma ciência e tecnologia que constrói, programa e utiliza robôs, e requer um conhecimento básico sobre as áreas de electrónica, mecânica e programação. Um robô é um dispositivo mecânico capaz de executar tarefas, de forma autónoma, pré-programada ou controlada por humanos. Karel Capek foi a primeira pessoa a utilizar o termo Robô em 1921 [5], que significava “trabalho forçado” no seu idioma.

A evolução dos robôs ao longo dos anos têm sido notória, sendo que hoje em dia existem vários tipos de robôs a operar em várias áreas. Um dos exemplos é a robótica industrial, onde actualmente existem milhares de robôs em ambientes industriais a desempenharem tarefas, variando entre automáticos e controlados. Existem também os robôs de serviço, onde robôs de forma semi ou completamente autónoma executam tarefas úteis para o bem-estar de humanos e equipamentos. Entre os vários robôs de serviços existem robôs de reparação, manutenção, limpeza, entretenimento ou ainda robôs fisioterapeutas.

Para proporcionar a investigação e o desenvolvimento da robótica existem várias competições que são realizadas anualmente, como o Festival Nacional de Robótica [6] realizado em Portugal ou a competição mundial RoboCup. Apesar de se tratarem apenas de competições, as equipas procuram resolver os desafios com a criação de soluções à medida de cada problema. No entanto, já é comum o uso dessas soluções para o desenvolvimento e comercialização de novos produtos, como por exemplo a cadeira de rodas eléctrica omnidireccional desenvolvida pela empresa SAR (Soluções de Automação e Robótica) [7], que começou com uma ideia originária de uma equipa de futebol robótico.

### **1.4.1. ROBOCUP**

O RoboCup é uma competição realizada a nível mundial, com o objectivo de promover o estudo e desenvolvimento da Inteligência Artificial e da robótica, através de desafios e problemas. As edições do RoboCup são divididas em competições e num simpósio, sendo que a primeira realizou-se no ano de 1997 em Nagoya, no Japão. Desde aí, foi realizado em várias cidades por todo o mundo como em Lisboa, Portugal, no ano de 2004 [8].

O grande objectivo do RoboCup é desenvolver uma equipa de robôs humanóides autónomos que consigam derrotar a equipa campeã mundial de futebol humano, em 2050.

Actualmente, o RoboCup é composto pelas seguintes ligas:

- RoboCup Soccer
  - Humanoid
  - Simulation
  - Small Size League
  - Midle Size League
  - Standard Plataform
- Robocup@Home
- Robocup Junior
  - Junior Dance
  - Junior Soccer
  - Junior Rescue
- RoboCup Rescue
  - Rescue Simulation
  - Rescue Robot

### 1.4.1.1. **MIDDLE SIZE LEAGUE**

*Middle Size League* (MSL), que em português significa liga média, é uma prova onde autonomamente os robôs jogam futebol – ver Figura 1. O futebol é um desporto muito popular em todo o mundo, e apresenta um conjunto de desafios científicos aos investigadores:

- Cooperação entre robôs, uma vez que se trata de um jogo colectivo;
- Comportamentos individuais, pois cada robô necessita de identificar objectos, localizar, driblar, entre outros;
- Coordenação entre robôs, necessária para realização de passes, atribuição de papéis e execução de movimentos em equipa (trabalho defensivo);
- Ambiente dinâmico com alterações constantes, como o movimento da bola, dos robôs da equipa e dos adversários.



Figura 1 - Foto de um jogo da liga de futebol robótico médio do RoboCup 2011.

As regras da MSL são baseadas nas leis oficiais da FIFA, sendo efectuadas algumas alterações para permitir que os robôs consigam jogar. As mudanças mais importantes são as seguintes:

- Equipas: São compostas no máximo por 6 robôs, um dos quais é o guarda-redes e é o único robô que pode entrar dentro da pequena área;
- Robô: Os robôs têm como dimensões máximas permitidas 52 cm de largura e 80 cm de altura, e o peso máximo admitido é de 40 kg. Os robôs devem jogar autonomamente e sem qualquer interferência humana;
- Bola: A bola oficial dos jogos é a bola oficial da FIFA tamanho 5;

- Campo: O campo é verde e as linhas são brancas. Tem 18 metros de comprimento e 12 metros de largura e é acrescido por uma distância de segurança de 1 metro a toda à volta. As balizas têm 2 metros de largura e 1 metro de altura. As medidas do campo estão detalhadas na Figura 2, e a unidade de medida é em metros;

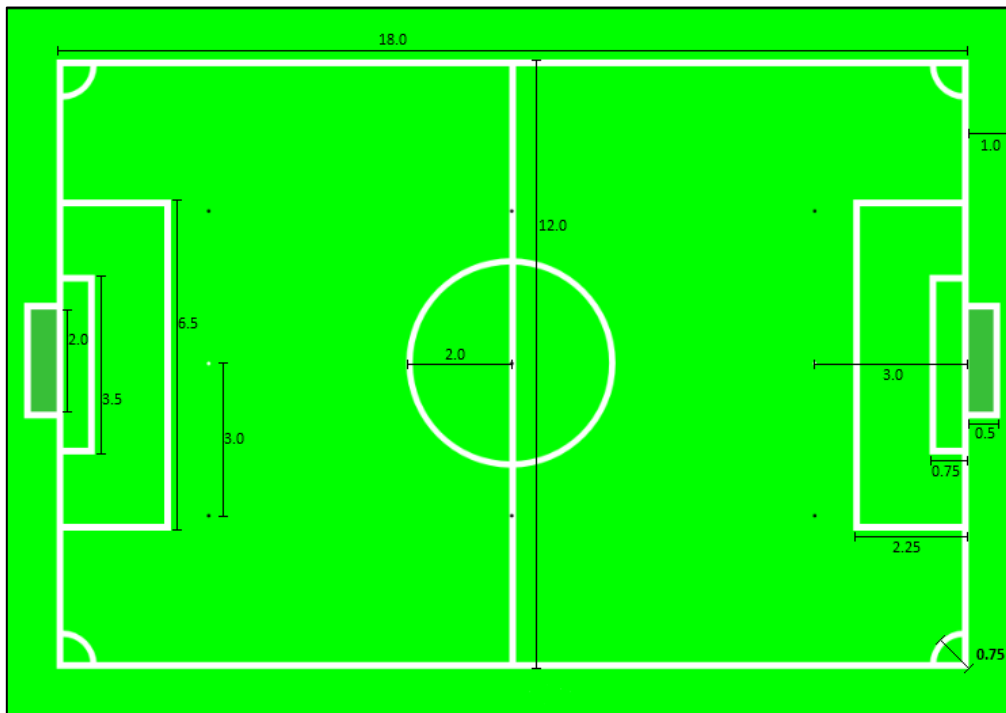


Figura 2 - Representação do campo de jogo com as dimensões oficiais.

- Comunicações: Os robôs podem comunicar com a *BaseStation* usando a tecnologia WLAN e respeitando a especificação IEEE 802.11a ou IEEE 802.11b;
- Cor equipa: Uma das equipas joga com a cor ciano, enquanto a outra com a cor magenta;
- Golos: Só é permitido marcar golo se o robô se encontrar no meio campo do adversário;
- Jogo: Cada jogo é dividido em duas partes de 15 minutos cada, e é arbitrado por um árbitro em conjunto com um assistente que opera a *RefBox*, a qual envia comandos para a *BaseStation* de cada equipa, como *Start*, *Stop* e outros de acordo com as decisões do árbitro. Uma imagem do programa *RefBox* é exibida na Figura abaixo.



Figura 3 - Imagem do programa de arbitragem *Referee Box*.

## 1.5. ESTRUTURA DA TESE

Para além da introdução, onde se enquadra o leitor no universo de aplicação de controlo e monitorização da equipa MINHO TEAM, o futebol robótico, mais concretamente o RoboCup e as regras oficiais da competição, esta dissertação contém mais 5 capítulos e 2 apêndices.

No segundo capítulo é realizado um levantamento dos programas de monitorização e controlo usados por quatro equipas da liga de futebol robótico médio, sendo no terceiro capítulo descrito o hardware e software actual da equipa MINHO TEAM.

No quarto capítulo apresenta-se a arquitectura da MINHO BaseStation, assim como todo o processo de implementação.

No quinto capítulo, Resultados, descreve-se a aplicação desenvolvida, nomeadamente as interfaces gráficas, as funcionalidades e os problemas.

As conclusões constituem o último capítulo, juntamente com propostas de ideias para serem implementadas no futuro.

## **2. ESTADO DA ARTE**

Presentemente, terminada a competição do RoboCup 2011 realizada em Istambul, Turquia, é possível constatar que a liga média de futebol robótico contou com a participação de doze equipas [9]. Um número que já foi superior em edições anteriores, mas devido às exigências sobretudo financeiras que implica construir e manter a equipa de robôs, bem como todos os custos necessários para a participação das equipas no RoboCup, faz com que algumas equipas desistam da participação neste evento mundial.

Estas doze equipas apresentam diferentes robôs desenvolvidos de acordo com as suas ideias. No entanto, já se encontram várias semelhanças em alguns aspectos dos robôs, tais como a velocidade máxima que conseguem atingir, o mecanismo usado para segurar a bola, a potência do chute ou o algoritmo de localização.

Uma das formas das equipas se diferenciarem dos adversários é a forma como os seus robôs jogam em equipa, seja em situações de ataque em que os robôs fazem passes longos procurando fugir à marcação dos adversários, como em situações defensivas, em que as equipas defendem com todos os robôs impedindo os remates dos adversários. Para garantir um trabalho em equipa dos robôs, as equipas desenvolvem um software de monitorização e controlo, que além das suas tarefas básicas gere os papéis atribuídos aos robôs e garante que todos têm conhecimento do estado dos colegas (posição em campo, papel atribuído), fazendo com que joguem em equipa.

A seguir é analisado o software de monitorização e controlo utilizado por quatro equipas que participaram na edição do RoboCup de 2011, nomeadamente a bicampeã Water [10], os vice campeões nas últimas quatro edições Tech United Eindhoven [11], e as equipas portuguesas CMBADA [12] e 5DPO [13].

### **2.1. WATER**

Water é uma equipa da Universidade da Ciência de Informação e Tecnologia de Beijing, China, que foi fundada no ano de 2003. Desde 2006 que participa na competição China Open, tendo vencido no ano de 2010 e ficando em segundo lugar em 2008 e 2009. Quanto ao RoboCup, a equipa participou nas últimas três edições, tendo vencido a prova em 2010 e 2011, o que a torna actualmente bicampeã mundial de futebol robótico médio.

### 2.1.1. WATER COACH

Water Coach, apresentado na Figura 4, é o nome do programa usado pela equipa Water para monitorizar e controlar os seus robôs. Foi implementado numa plataforma em Windows, e desenvolvido num IDE da Microsoft Visual Studio [14] em C++ [15].

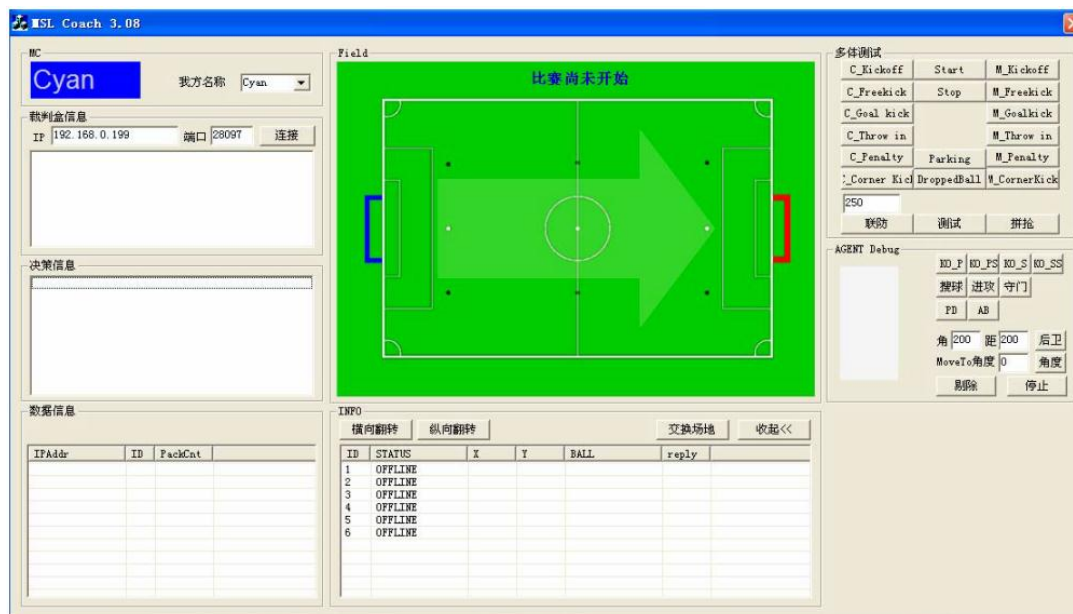


Figura 4 - Water Coach [15].

É utilizado para enviar os comandos recebidos da *RefBox* para os robôs, analisar o estado do jogo e controlar os robôs. No centro do programa está desenhado um campo de futebol robótico à escala (de cor verde), onde são desenhados os robôs da equipa de acordo com as suas posições reais no campo. Um dos vários painéis do programa é utilizado para exibir todas as mensagens recebidas da *RefBox*. Contém também uma versão local da *RefBox*, bem como um painel dedicado a mostrar o estado das comunicações com os robôs, nomeadamente o tamanho dos pacotes de dados enviados em cada instante. Por fim, é apresentado o estado dos robôs, sobretudo as coordenadas do robô no campo, o ângulo de desfasamento entre o robô e o eixo de referência, se o robô consegue ver a bola e ainda o que cada robô está a executar (mover-se para um ponto, ir atrás da bola, permanecer imobilizado, entre outros). Uma das tarefas do programa consiste em atribuir papéis aos robôs automaticamente, gerindo a lista de papéis a atribuir de acordo com o número de robôs disponíveis.

Relativamente ao protocolo de comunicações entre o Water Coach e os robôs, a equipa Water utiliza a estrutura Cliente / Servidor, implementando o protocolo de comunicações TCP/IP. O Water Coach funciona como servidor, aguardando conexões por parte dos robôs. O esquema completo de ligações da equipa Water pode ser consultado na Figura 5.

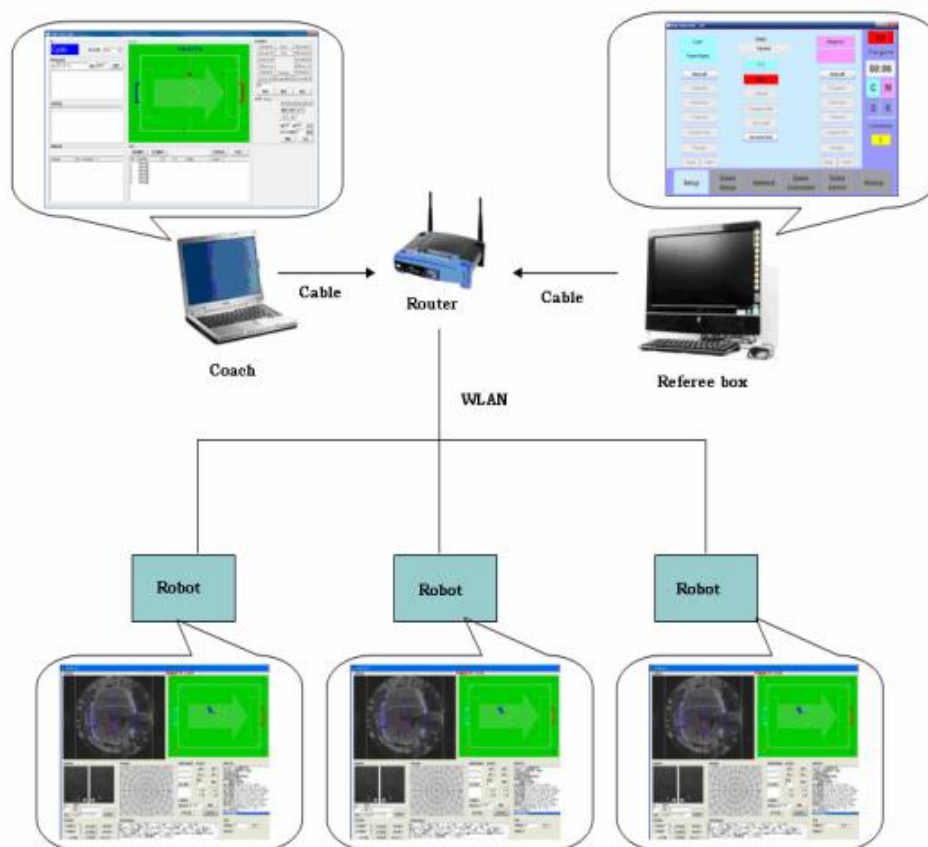


Figura 5 - Esquema de comunicação da equipa Water [16].

## 2.2. TECH UNITED EINDHOVEN

Tech United Eindhoven é uma equipa proveniente da Universidade da Tecnologia de Eindhoven, Holanda. A sua origem começou em 2005, tendo a equipa participado em competições de futebol robótico médio desde o ano de 2006. Relativamente às suas participações no RoboCup, os Tech United Eindhoven ficaram em segundo lugar nas últimas quatro edições realizadas – 2008, 2009, 2010 e 2011. Durante o ano de 2011, a equipa começou também a participar na prova do RoboCup conhecida como RoboCup@Home [17].

### 2.2.1. TURTLE REMOTE CONTROL

O Turtle Remote Control, ou TRC, consiste no software desenvolvido pela equipa Tech United Eindhoven – ver Figura 6. Foi escrito na linguagem de programação C usando o Glade/GTK+ [18], e funciona no sistema operativo Linux.

Ele é formado por vários módulos, tais como um campo para monitorização dos robôs durante os jogos, um *Joystick*, um simulador, uma ferramenta para ajuste de parâmetros em tempo real e ainda uma versão simplificada da *RefBox*.



O módulo de monitorização disponibiliza ao utilizador informações sobre os três módulos de software que são executados nos robôs (visão, locomoção e modelo do mundo), além da posição da bola, com a particularidade de ser em três dimensões. É igualmente exibida a capacidade das baterias, o estado do CPU e o papel actual de cada robô. O módulo *Joystick* permite o controlo dos robôs de uma forma fácil. A versão da *RefBox* implementada além dos habituais comandos como *Start*, *Stop*, pontapé de saída, entre outros, possui uma opção para a realização de testes e demonstrações, o que permite uma comutação entre o modo de jogo e o modo de testes / demonstrações bastante simples.



Figura 6 – Turtle Remote Control [19].

Quanto ao simulador desenvolvido, o seu esquema pode ser visto na Figura 7. Cada bloco representa um robô, e é composto pelos três módulos de software que constituem os robôs. Uma das particularidades deste simulador é o facto de executar o mesmo código que os robôs, não necessitando assim de nenhuma alteração entre o desenvolvimento, simulação e execução real do código. Tanto os sensores como os actuadores presentes no robô foram desenvolvidos para serem simulados através dos seus modelos físicos, com excepção do módulo de processamento de imagem, que foi removido da actual versão.

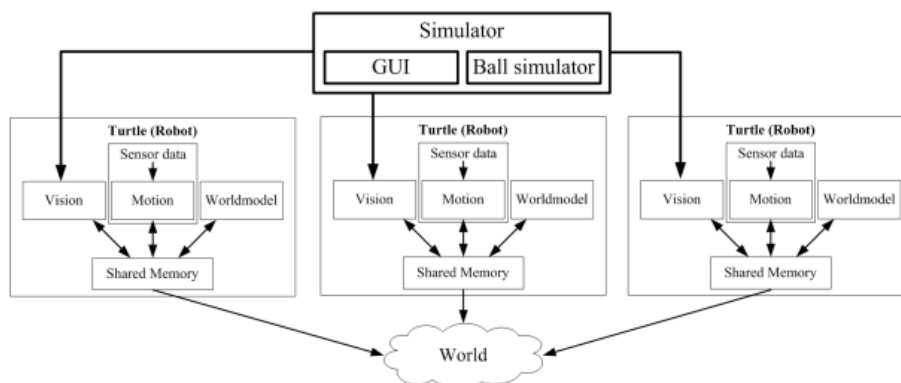


Figura 7 - Diagrama do simulador da equipa Tech United Eindhoven [19].

O protocolo de comunicação que foi implementado entre o Turtle Remote Control e os robôs é o *User Datagram Protocol* (UDP).

## 2.3. CAMBADA

CAMBADA - é um acrónimo de *Cooperative Autonomous Mobile robots with Advanced Distributed Architecture*, é o nome da equipa de futebol robótico médio da Universidade de Aveiro. A equipa teve origem no ano de 2003, participando em várias competições nacionais e internacionais. Nas suas participações no RoboCup, a melhor classificação que obteve foi o primeiro lugar em 2008, tendo nos três anos seguintes ficando no terceiro posto. Relativamente ao Festival Nacional de Robótica, realizado em Portugal, a equipa CAMBADA é pentacampeã, começando o seu percurso de vitórias no ano de 2007 [20].

### 2.3.1. CAMBADA BASESTATION

CAMBADA Basestation (ver Figura 8) consiste numa aplicação de software, desenvolvida pela equipa CAMBADA para controlar e monitorizar o estado interno dos robôs, assim como processar automaticamente a ocorrência de eventos durante os jogos.

A aplicação CAMBADA Basestation foi implementada numa plataforma em Linux e o seu desenvolvimento feito numa *framework* de Qt [21] utilizando a linguagem de programação C++. O protocolo de comunicação utilizado para realizar a comunicação entre os robôs e a CAMBADA Basestation é o *User Datagram Protocol*.

É composta por vários módulos. Um deles consiste num painel para cada robô, que contém alguns botões para enviar comandos (*Start* e *Stop*), várias caixas de texto que exibem

informações sobre o robô, como a sua posição em campo, a capacidade das baterias, e ainda qual o papel e o comportamento atribuído ao robô.



Figura 8 - CAMBADA Basestation [22].

Outro módulo existente consiste no desenho de um campo em duas dimensões e à escala relativamente às medidas oficiais de um campo de futebol robótico médio. Além de desenhar os robôs e a bola de acordo com as suas posições reais, é também possível mover o robô usando o rato. Outro dos módulos implementa uma *RefBox* virtual, o qual fornece uma versão local para testes e ainda gere a informação recebida pela *RefBox* oficial durante os jogos. Por último, o módulo chamado de RobotsInfo mostra num painel toda a informação relacionada com os robôs e que se encontra guardada na base de dados do programa.

### 2.3.2. POSICIONAMENTOS

O modelo de coordenação da equipa CAMBADA é baseado em alguns conceitos como posicionamento estratégico, papel e formações [20], onde formações são conjuntos de posicionamentos estratégicos, cada um relacionado com o movimento de um robô. A atribuição de papéis e posições aos robôs é feita dinamicamente de acordo com algumas regras definidas previamente.

Actualmente, nos lances de bola parada, o cálculo das posições a atribuir a cada robô é feito a partir de um conjunto de posições de bola previamente definidas. Depois é feita uma interpolação dessas posições usando o método de triangulação de Delaunay [23] e os pontos que formam o triângulo no qual a bola está inserida. Esta definição de posições é feita utilizando uma ferramenta gráfica desenvolvida pela equipa CAMBADA e pode ser vista na Figura 9.

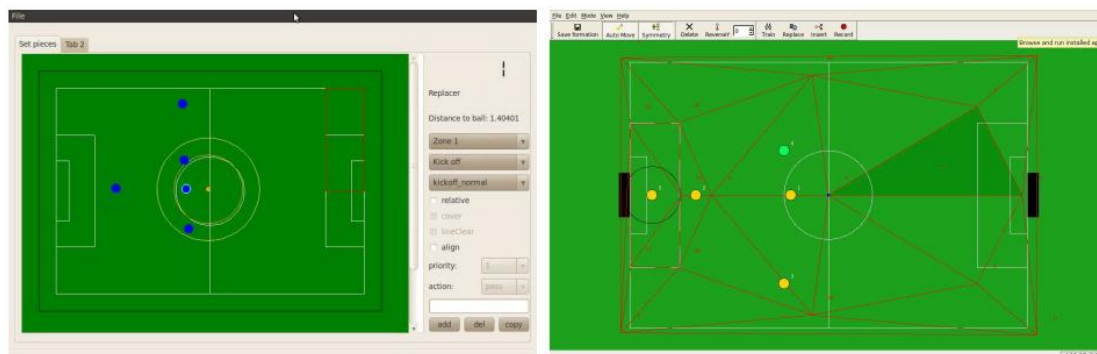


Figura 9 - Ferramenta desenvolvida pela equipa CAMBADA para definir posicionamentos [20].

## 2.4. 5DPO

5DPO é a equipa da liga de futebol robótico médio da Universidade do Porto. A equipa começou no ano de 1998 e competia apenas na liga de futebol robótico pequeno. Em 2001, foi criada a equipa de robôs para participar na liga de futebol robótico médio, e logo nesse ano obteve a classificação de terceiro lugar na competição German Open. Em 2009 e 2010, ficou classificada na segunda posição na competição portuguesa Festival Nacional de Robótica.

A equipa é constituída por elementos que trabalham em várias áreas, desde a estrutura mecânica do robô, arquitectura de hardware e controladores, bem como desenvolvimento de software em áreas de processamento e análise de imagem, de modelação e de controlo [24].

### 2.4.1. COACH

Coach é o nome do software de gestão da equipa 5DPO, e pode ser visto na Figura 10. Foi desenvolvido para funcionar em Linux, usando a linguagem de programação Pascal e o IDE Lazarus [25]. Os robôs comunicam com o Coach usando o protocolo de comunicações UDP.

Uma das suas funções consiste em receber os comandos da *RefBox* e actualizar a máquina de estados de acordo com o tipo de comando recebido. É também o responsável por atribuir papéis a cada robô, uma vez que é nele que as tácticas a usar para as situações de bola parada foram implementadas.

O Coach assume um papel importante durante os jogos, pois além de controlar os robôs – ao enviar comandos do árbitro, atribuir papéis e posições para permitir o jogo em equipa, também serve como ferramenta de análise. Para isso foi desenhado um campo onde são colocados os robôs e a bola de acordo com as suas posições reais no campo, o que torna mais fácil analisar quando os robôs executam comportamentos inesperados. É ainda possível observar

quando os robôs estão activos ou deixam de comunicar com o Coach, através de um sistema de cores que foi implementado.



Figura 10 - 5DPO Coach [24].

## 2.5. CONCLUSÕES

Terminada a análise efectuada ao software utilizado por quatro equipas da liga de futebol robótico médio para o controlo e monitorização dos seus jogadores, verificam-se algumas semelhanças entre as equipas, particularmente na forma como a informação é exibida – interface gráfica, e no tipo de módulos com que o software é equipado. Todas as equipas apresentam um campo desenhado no seu software, onde os robôs são representados de acordo com a sua posição em campo. É comum entre as equipas, mostrar o estado dos robôs através da capacidade das baterias, do seu papel e comportamento, da sua posição e do alerta da ocorrência de erros. Todas as equipas têm incluído uma versão local da *RefBox* para a realização de testes. A inclusão de um simulador é algo que é usado apenas por duas equipas, pois não é considerado como imprescindível para a realização de jogos.

Sobre o tipo de ferramentas escolhidas para o desenvolvimento do software, cada equipa escolheu uma plataforma / ambiente de desenvolvimento integrado (IDE) diferente, pois existem actualmente várias ofertas para esse fim [26,27]. O protocolo de comunicação mais utilizado foi o *User Datagram Protocol*, e o sistema operativo eleito pela maioria das equipas para o desenvolvimento do software foi o Linux. De seguida é apresentada uma tabela comparativa (Tabela 1) das principais características do software das equipas.

Tabela 1 - Comparação das características dos softwares das equipas analisadas.

	Water	Tech United Eindhoven	CAMBADA	5DPO
Sistema operativo	Windows	Linux	Linux	Linux
Linguagem	C++	C	C++	Pascal
Framework / IDE	Microsoft Visual Studio	Glade/GTK+	Qt	Lazarus
Protocolo comunicações	TCP/IP	UDP	UDP	UDP
Simulador		Sim	Sim	
<i>RefBox</i> virtual	Sim	Sim	Sim	Sim
Tipo de campo desenhado	2D	3D	2D	2D



### 3. MINHO TEAM

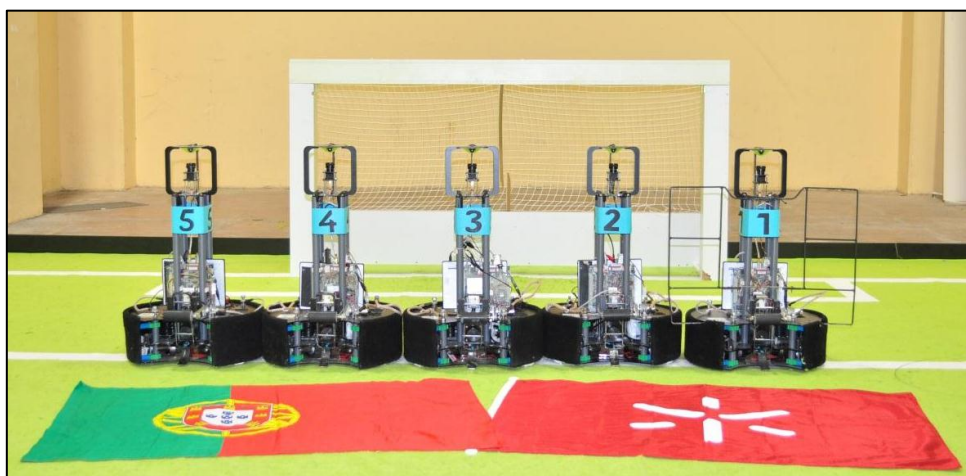


Figura 11 - Foto da nova geração de robôs da equipa MINHO TEAM.

MINHO TEAM é a equipa de robôs futebolistas desenvolvida na Universidade do Minho, que participa na liga de futebol robótico médio. A equipa iniciou a sua participação em competições no ano de 1997 com uma geração de robôs, mas devido a problemas internos e pessoais esteve alguns anos sem competir. No ano de 2010 foi construída uma nova geração de robôs (Figura 11), com o objectivo de os tornar mais leves e rápidos, melhorar a visão do robô (mais alcance e maior resolução) e ainda desenvolver novos algoritmos [28].

#### 3.1. DESCRIÇÃO DO HARDWARE

De seguida é descrito o hardware que constitui os robôs, sendo este exibido na Figura 12.

Começando pelo alto nível, a *BaseStation* envia dados para o robô via *wireless*. Os robôs comunicam usando um adaptador de internet da Asus, através da especificação IEEE 802.11b. O adaptador de internet é necessário, pois os computadores usados não contêm placa de rede sem fios. Os computadores são constituídos por uma motherboard MB890 Mini-ITX e um processador de 1.7 GHz. A câmara usada é uma Point Grey Flea 2 FireWire, e depois de capturar imagens envia-as para o computador através do barramento FireWire. Depois de processada a imagem e os dados recebidos por rede, o computador envia dados para controlar o baixo nível através da conexão USB.





## 3.2. DESCRIÇÃO DO SOFTWARE

### 3.2.1. ALGORITMO DE LOCALIZAÇÃO

O algoritmo de localização implementado utiliza linhas radiais para procurar as linhas brancas do campo, com um desfasamento entre elas de 5 graus e até um máximo de 216 pontos por linha. Sempre que o número de linhas detectadas é inferior a 20, é realizada uma pesquisa axial para se obter mais pontos e consequentemente mais precisão na localização.

Um mapa do campo de jogo é guardado num ficheiro binário, onde 0 significa o verde e 1 as linhas brancas do campo. O algoritmo percorre o ficheiro e para cada posição guarda o valor da distância à linha mais próxima numa matriz com uma resolução de 1dm<sup>2</sup>. Os pontos resultantes das pesquisas axiais e radiais são convertidos para distâncias reais, e de seguida é somado o valor do erro de cada ponto de acordo com a matriz anterior. O valor mais pequeno da matriz representa a localização do robô no campo.

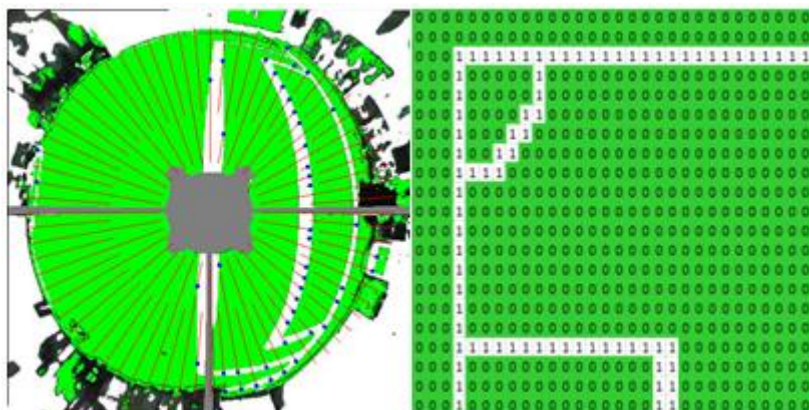


Figura 13 - Esquerda: Pesquisa radial de linhas. Direita: Campo binário.

### 3.2.2. MÁQUINA DE ESTADOS

A máquina de estados implementada no robô é composta por quatro estados principais, designados por Idle, Positioning, Playing e Searching. Para ocorrer uma transição de estados é necessário o robô receber comandos do árbitro, pois a máquina de estados foi projectada para funcionar da seguinte forma: o robô quando se encontra parado corresponde ao estado Idle. Depois, sempre que receber comandos do árbitro como pontapé de saída, lançamento ou falta entra no Positioning. Aí, reposiciona-se em campo de acordo com o tipo de comando recebido e da posição da bola. Quando for dado o sinal de *Start* pelo árbitro o robô comuta para o estado Playing e permanece aí até receber um comando de Stop que o faça mudar para o Idle. Sempre

que nenhum dos robôs observe a bola, um dos robôs é designado para procurar percorrer o todo o campo, correspondendo esse comportamento ao estado Searching.

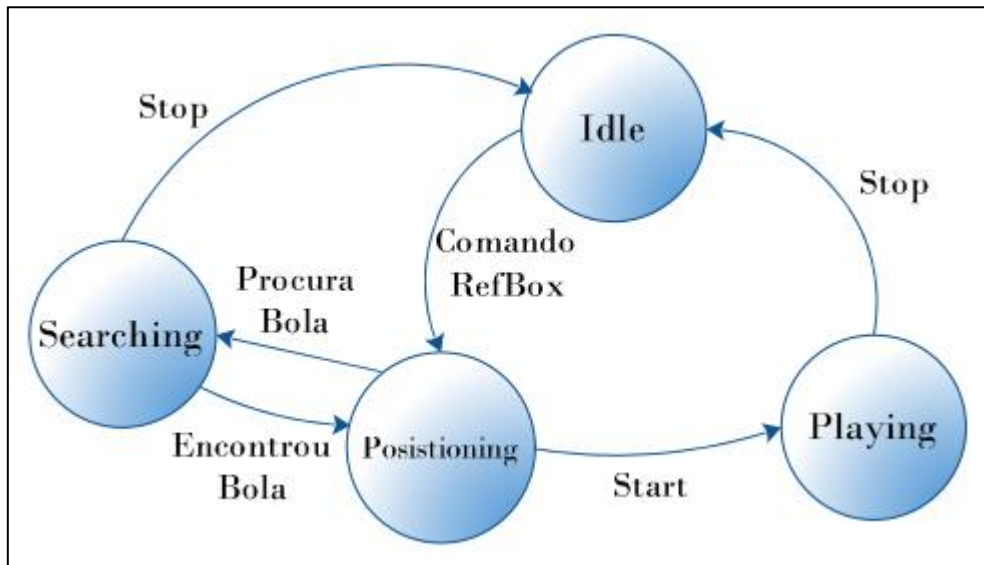


Figura 14 - Diagrama da máquina de estados implementada no robô.

## 4. ARQUITECTURA DA MINHO BASESTATION

Neste capítulo é apresentado o projecto e a implementação da MINHO BaseStation, designação atribuída à aplicação desenvolvida para o controlo e monitorização da equipa MINHO TEAM. Foi implementada na linguagem de programação C++, utilizando o ambiente de desenvolvimento Qt Creator [21] (versão 4.7) no sistema operativo Linux – sendo usada a distribuição Ubuntu (versão 9.10). O computador utilizado contém um processador Dual Core 2.00 GHz, uma memória RAM de 2.00 GB e um sistema operativo de 32 bits.

Inicialmente é apresentada a análise de requisitos (secção 4.1). A seguir é descrito o sistema como um todo (secção 4.2), descrevendo a estrutura em rede e o protocolo de comunicações implementado com os robôs e *RefBox* (secção 4.3). Por último, é apresentada a arquitectura interna de cada módulo que constitui a MINHO BaseStation.

### 4.1. REQUISITOS

Os requisitos da MINHO BaseStation resultam da análise às equipas concorrentes, das exigências da equipa MINHO TEAM, do testemunho de várias pessoas com experiência em jogos de futebol robótico e da necessidade do cumprimento das regras estabelecidas pela organização do RoboCup.

Os requisitos mais importantes que a MINHO BaseStation deve cumprir são os seguintes:

- Estabilidade/Fiabilidade: durante os jogos não é permitida nenhuma interacção humana, portanto tem que ser uma aplicação robusta;
- Simplicidade: as informações exibidas no monitor durante os jogos devem ser de fácil interpretação, permitindo uma rápida detecção da ocorrência de anomalias nos robôs ou no funcionamento do próprio Modo de Jogo. A definição de novas tácticas deve ser simples e sem necessidade de compilação de código;
- Facilidade de operação: a MINHO BaseStation deve ser fácil de utilizar, e permitir de forma intuitiva a definição de elementos do jogo (como por exemplo a cor da equipa para cada jogo ou o endereço IP da *RefBox*);
- Eficiência: o tempo que demora a executar tarefas é uma questão crucial, especialmente durante os jogos e quando há necessidade de parar os robôs;

- Adaptabilidade: uma equipa de futebol robótico está constantemente em desenvolvimento, logo é importante que alterações ou melhorias sejam efectuadas rápida e facilmente;
- Inteligência: aquando o executar autónomo, é essencial um controlo inteligente da equipa, nomeadamente, na atribuição de tácticas e papéis. Não podem ocorrer erros ao, por exemplo, atribuir o papel de avançado a mais do que um robô, ou então no caso de o guarda-redes ser removido do campo, não colocar imediatamente um robô nesse papel.

O cumprimento dos requisitos acima descritos leva à implementação das seguintes condições:

- Construção por módulos / classes: cada funcionalidade deve ser implementada num módulo, permitindo o teste de cada módulo em separado e o desenvolvimento de novos módulos de uma forma independente dos já implementados. Sobre classes, um robô, por exemplo, deve ser uma entidade instanciada da classe Robo, o que permite a existência de mais do que um robô e ainda um nível de abstracção no sistema.
- Sistema de comunicações robusto: um correcto funcionamento da equipa durante um jogo é possível devido às comunicações com os robôs e a *RefBox*. É então necessário um mecanismo de comunicações que garanta o pleno envio de pacotes de dados entre todos.
- *RefBox* virtual: a implementação de uma *RefBox* virtual é extremamente necessária para a simulação da ocorrência de eventos de um jogo durante a fase de desenvolvimento.
- Gravação do jogo: o registo de toda a informação que ocorre durante a realização de jogos torna possível a análise pós jogo de cada robô individualmente, e assim corrigir eventuais erros encontrados.
- Atribuição manual de papéis: durante o período de desenvolvimento e realização de testes, deve ser possível atribuir papéis específicos a cada robô manualmente.

Os requisitos foram definidos para todas as funcionalidades da MINHO BaseStation, no entanto é o Modo de Jogo que requer mais atenção uma vez que é o mais delicado e importante de todos, pois é o único que executa autonomamente e um possível bloqueio durante a realização de um jogo traria consequências bastante graves a toda a equipa MINHO TEAM.

## 4.2. VISÃO GERAL

Durante o desenvolvimento da arquitectura da MINHO BaseStation, a principal preocupação foi assegurar o cumprimento de todos os requisitos do projecto. Dessa forma, o tipo de arquitectura escolhida e implementada pode ser observada na Figura 15.

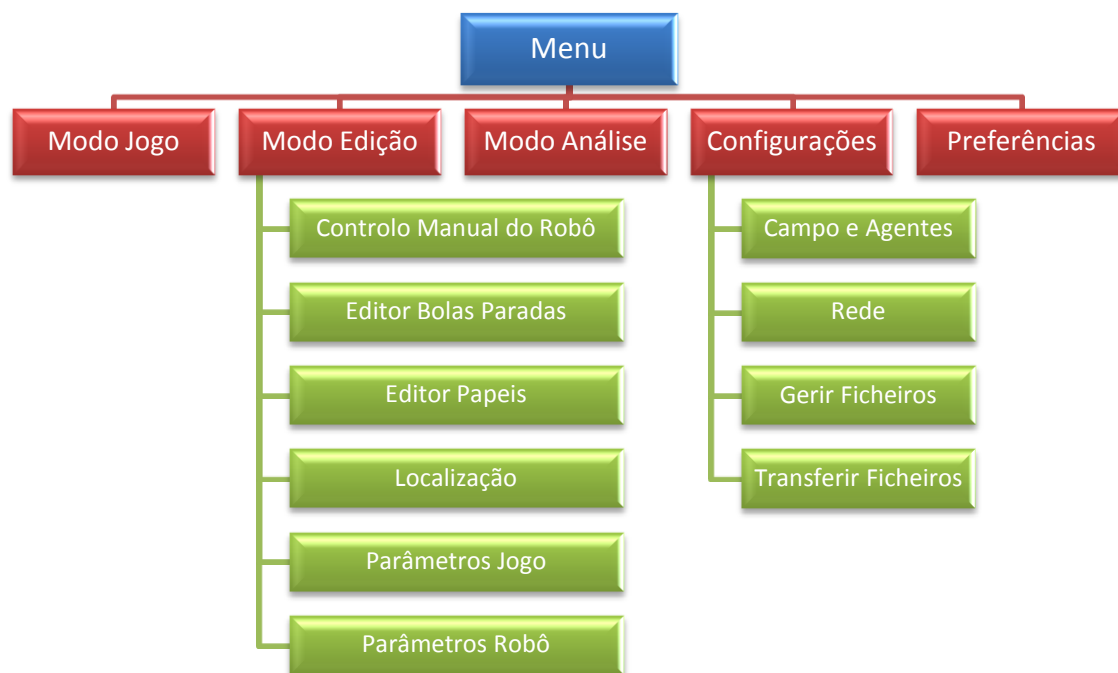


Figura 15 - Esquema da MINHO BaseStation do ponto de vista do utilizador.

Trata-se de uma solução onde a aplicação é composta por um ou vários módulos independentes. A grande vantagem deste tipo de arquitectura é o nível de abstracção fornecido ao utilizador e a facilidade de desenvolvimento de novos módulos, pois não é necessário existir qualquer conhecimento sobre os módulos já existentes, e ainda possibilita que novos módulos sejam desenvolvidos e testados à parte, e só depois adicionados à aplicação.

Quanto ao funcionamento da MINHO BaseStation, depois de esta ser iniciada o utilizador depara-se com um menu composto por cinco módulos. O conceito de exclusividade está presente, pois apenas pode ser escolhida uma opção de cada vez. Cada módulo pode ser formado por vários sub-módulos, como o Modo de Edição ou as Configurações, o que permite que módulos grandes sejam desenvolvidos e testados por partes.

A estrutura de classes e objectos simplificada da MINHO BaseStation é apresentada na Figura 16 e na Figura 17, respectivamente.

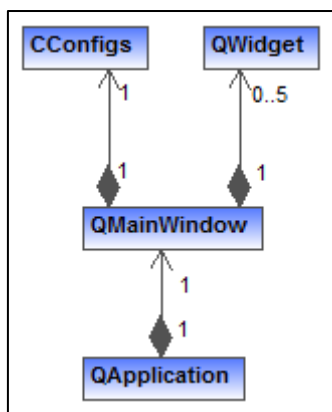


Figura 16 - Diagrama de classes da MINHO BaseStation.

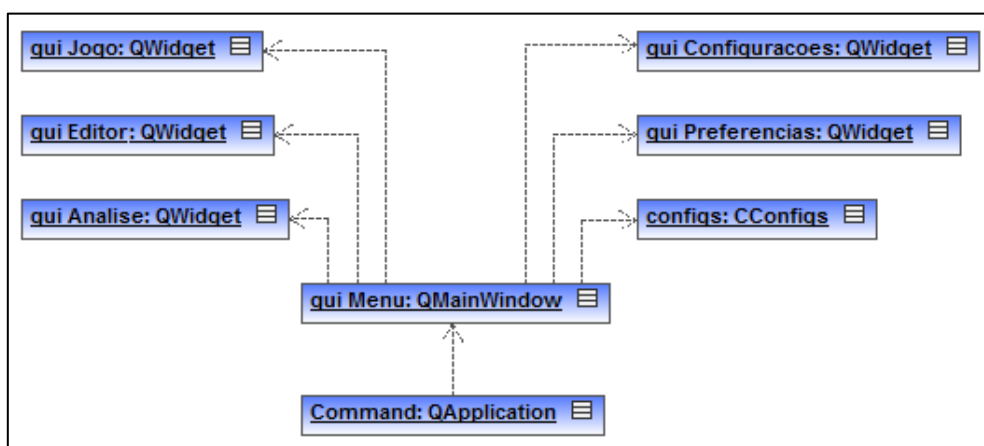


Figura 17- Diagrama de objectos da MINHO BaseStation.

Uma vez que a aplicação é puramente gráfica, o objecto (Command) da aplicação da classe QApplication instancia o interface gráfico (gui) Menu da classe QMainWindow. O objecto Menu, por sua vez, instancia uma série de outros objectos, sendo os mais importantes os cinco objectos da classe QWidget e o objecto configs da classe CConfigs.

O objecto Menu trata-se do interface gráfico principal do programa, e é ele que gere a comunicação com todos os outros objectos através do mecanismo de *signals* e *slots* implementados pelo Qt. Cada um dos cinco objectos instanciados da classe QWidget está associado a um módulo do programa, sendo o Menu que mediante a escolha do utilizador se encarrega de exibir ou ocultar o objecto correspondente. Os objectos instanciados a partir da classe QWidget também contêm uma interface gráfica, no entanto cada um deles será individualmente detalhado na secção correspondente.

A classe CConfigs é instanciada aquando da criação do objecto Menu por se tratar de uma classe global. Isto é, contém atributos que são necessários aos objectos da classe QWidget, pelo que o objecto configs é criado e os seus atributos são consultados ou editados pelos cinco

objectos da classe QWidget sempre que necessário. A classe CConfigs é explicada de seguida mais detalhadamente.

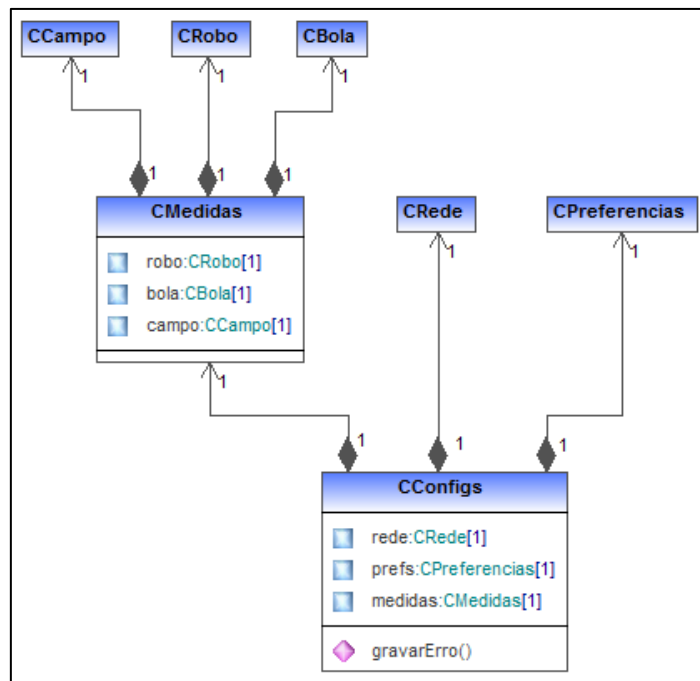


Figura 18 - Diagrama de classes da classe CConfigs.

### CConfigs

Esta classe é uma das mais importantes, uma vez que a informação dos seus atributos está relacionada com o correcto funcionamento do programa. Ela instancia objectos das classes CMedidas, CRede e CPreferencias, sendo estas classes responsáveis por armazenar as dimensões do campo, as configurações de rede, e as preferências sobre quais os ficheiros a usar em todos os modos do programa.

Esta classe contém ainda um método responsável por registar a ocorrência de todos os erros no ficheiro de erros, para dar conhecimento ao utilizador de incongruências no programa.

### CMedidas

A classe CMedidas no momento em que é criada instancia três classes, nomeadamente a CBola, CCampo e CRobo. A função da CMedidas consiste em armazenar as dimensões do campo – no caso da CCampo, e o diâmetro do robô e da bola de jogo no caso da CRobo e CBola respectivamente.



### **CRede**

Esta classe guarda todas as variáveis de rede necessárias para a comunicação com a *RefBox* e os robôs, como os endereços IP dos robôs e portas usadas para a comunicação ou o endereço IP da *RefBox* e as portas para os vários protocolos.

### **CPreferencias**

A classe *CPreferencias* guarda as preferências do utilizador, nomeadamente a cor da equipa, o protocolo a utilizar para a ligação à *RefBox*, e quais os vários tipos de ficheiros a usar. Isto é, dependendo do local onde a equipa efectua os jogos, as dimensões do campo podem não ser as oficiais. Então, existem vários ficheiros com as dimensões do campo, e o utilizador selecciona qual o ficheiro a usar de acordo com o local onde se encontra. O mesmo acontece com os ficheiros que contêm as variáveis de rede, que dependendo do local onde a equipa MINHO TEAM se encontra, as configurações de rede a usar variam. Todas estas escolhas são armazenadas nos atributos da classe *CPreferencias*.

## **4.3. PROTOCOLO DE COMUNICAÇÕES**

A comunicação é uma das partes mais importantes numa equipa de robôs futebolistas, uma vez que para o correcto funcionamento da equipa nos jogos de futebol é necessário garantir que a MINHO BaseStation envia para os robôs os comandos recebidos da *RefBox*, a modelação do mundo ou ainda instruções para se posicionarem nas bolas paradas.

A implementação das comunicações necessita de obedecer a um conjunto de regras impostas pela organização do RoboCup, para assim garantir que todas as equipas competem ao mesmo nível. A topologia de rede imposta pela organização encontra-se na Figura 19. Toda a comunicação *wireless* efectuada pelas equipas deve passar pelos *Access Points* fornecidos pela organização. São implementados dois *Access Points*, uma para a especificação IEEE 802.11a e outro para a especificação IEEE 802.11b. Só é permitido o uso do modo de comunicações *unicast* e *multicast*, e cada equipa não deve exceder 20% da largura de banda. A cada equipa que participa no RoboCup é atribuída uma gama de endereços IP para o modo *unicast* e *multicast*, evitando assim a possibilidade de ocorrência de conflitos de endereços IP.

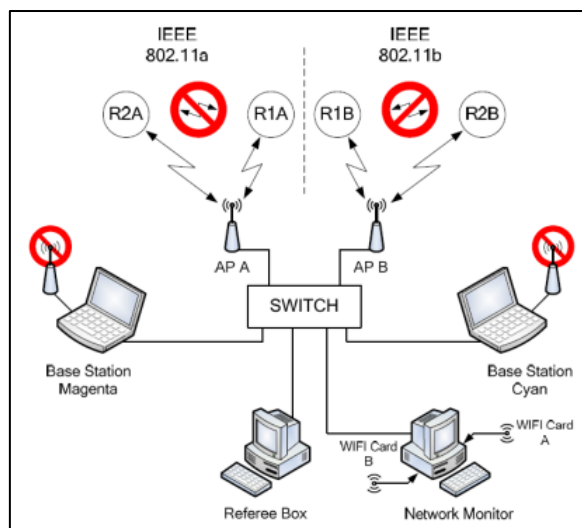


Figura 19 - Topologias de ligações de rede.

Na MINHO BaseStation as comunicações foram divididas em duas partes, nomeadamente, a comunicação com a RefBox e a comunicação com os robôs.

#### 4.3.1. COMUNICAÇÃO ENTRE MINHO BASESTATION E *REFBOX*

A comunicação com a *RefBox* é realizada usando um cabo UTP (*Unshielded Twisted Pair*). Neste caso, a comunicação via *wireless* foi preterida para evitar atrasos na recepção das mensagens.

A versão da *RefBox* utilizada durante o RoboCup 2011 disponibiliza às equipas dois tipos de protocolos de dados, denominados de protocolo2009 e protocolo2010. A diferença entre eles consiste na estrutura das mensagens que são enviadas, uma vez que o protocolo2009 envia um caractere associado a cada evento, enquanto o protocolo2010 envia mensagens no formato XML (*Extensible Markup Language*), onde cada mensagem contém o tipo de evento, o tempo de jogo ou ainda o número da mensagem.

O protocolo2009 para ser usado é necessário utilizar o protocolo de comunicação TCP/IP, enquanto o protocolo2010 pode ser utilizado com o protocolo de comunicação TCP/IP ou UDP. A MINHO BaseStation foi implementada para operar com qualquer protocolo da *RefBox*, independentemente do protocolo de rede utilizado.

No entanto, durante os jogos é utilizado o protocolo2010 da *RefBox* juntamente com o protocolo de comunicação TCP/IP. A escolha recaiu sobre o protocolo2010 porque disponibiliza um conjunto de informação extra relativamente ao protocolo2009. Quanto ao TCP/IP, foi o

protocolo de comunicação escolhido uma vez que as ligações estabelecidas são exibidas na *RefBox* e assim existe garantia que a ligação foi efectuada correctamente, garante a entrega dos pacotes de dados e ainda verifica a integridade dos dados transmitidos [29].

### 4.3.2. COMUNICAÇÃO ENTRE MINHO BASESTATION E ROBÔS

Para a comunicação entre MINHO BaseStation e os robôs foi escolhido o protocolo de comunicações UDP. Uma vez que a comunicação com os robôs é feita várias vezes por segundo, não é muito relevante para o correcto funcionamento do programa que alguns pacotes de dados não sejam entregues aos robôs. Dessa forma, a preocupação foi escolher um protocolo de comunicação que entregasse mensagens o mais rápido possível.

Devido a restrições relacionadas com os adaptadores de internet dos robôs, o modo de comunicação adoptado foi o *unicast*.

A próxima figura simboliza as ligações usadas pela MINHO BaseStation para comunicar com a *RefBox* e com os robôs. As setas de cor verde correspondem ao protocolo de comunicação utilizado.

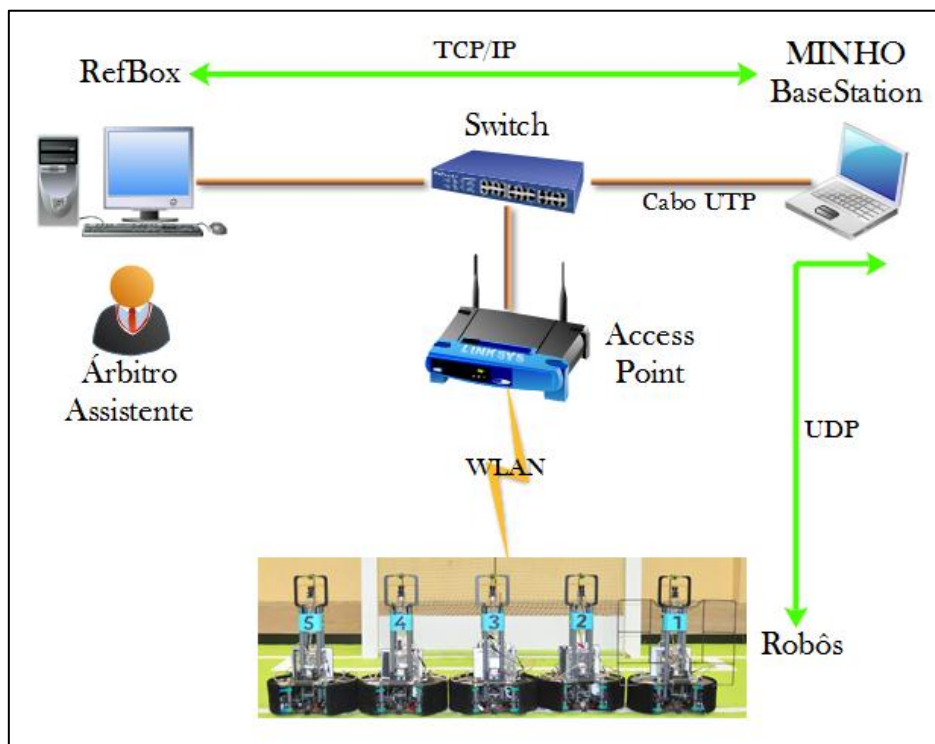


Figura 20 - Ligações de rede implementadas na MINHO BaseStation.

## 4.4. MODO DE JOGO

O Modo de Jogo foi desenvolvido para a realização de jogos de futebol robótico, quer sejam oficiais ou testes. Quando se verificar a realização de jogos oficiais, o modo de jogo deve executar de forma autónoma durante todo o jogo, uma vez que não é permitida nenhuma interferência humana.

Durante a sua execução, deve concretizar um conjunto de tarefas. Depois de ser conectado à *RefBox*, tem que receber, exibir e processar os comandos recebidos, e informar os robôs das alterações do estado do jogo. No caso do tipo de comando recebido corresponder às situações de bola parada, o Modo de Jogo calcula e transmite as posições em campo desejadas para cada robô. Outra das tarefas consiste em receber e exibir informações dos robôs, como a posição e orientação em campo, a posição e a velocidade da bola, a capacidade actual das baterias, o estado dos infravermelhos que detectam a presença da bola na frente do robô, a ocorrência de erros no hardware do robô e o valor de variáveis do programa que executa em cada robô. A informação correspondente à bola que é enviada pelos robôs é fundida e reenviada de volta, o que permite que todos os robôs tenham conhecimento da posição da bola mesmo que o seu sistema de visão não a encontre. Por último, todas as mensagens recebidas da *RefBox*, assim como todos os dados recebidos dos robôs são guardados em ficheiro para posterior análise. Para a realização de todas as tarefas referidas anteriormente de uma forma rápida e eficaz, foi implementada uma máquina de estados finita.

O Modo de Jogo é constituído por uma interface gráfica, que pode ser separada em três partes: o campo de jogo, a informação dos robôs e informação do jogo. O campo de jogo consiste num campo desenhado à escala de acordo com as dimensões do campo real, onde são desenhados os robôs e a bola de acordo com a sua posição em campo. Para a exibir as informações sobre os robôs é designado um painel para cada robô, e dentro dele são colocados os dados relativos a cada robô. O tempo de jogo, resultado e o histórico de mensagens da *RefBox* compõe as informações sobre o jogo, e também eles são mostrados na interface gráfica. Opcionalmente, podem também ser exibidas um conjunto de opções gráficas, como uma *RefBox* virtual – útil na realização de jogos de teste, as informações da base de dados do programa, os papéis atribuídos a cada robô, as posições calculadas pelo Modo de Jogo para situações de bola parada e os parâmetros de jogo – onde é possível escolher a cor da equipa, o estado do jogo, o resultado e o nome do adversário.

#### 4.4.1. ESTRUTURA DO MODO DE JOGO

O diagrama de classes com apenas os atributos e métodos mais importantes do Modo de Jogo encontra-se na Figura 21. As classes fornecidas pelo Qt não foram representadas no diagrama, no entanto são usados os recursos gráficos e de rede do Qt, nomeadamente as classes `QUdpSocket` e `QTcpSocket` que fornecem um *socket* UDP e um *socket* TCP, respectivamente. Para o desenho do campo de jogo, dos robôs e das bolas foram utilizadas as classes `QGraphicsView` e `QGraphicsScene`.

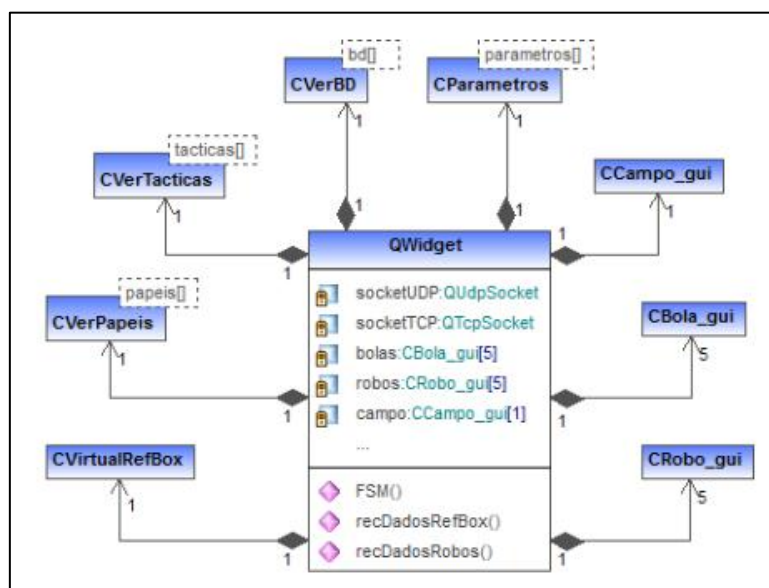


Figura 21 - Diagrama de classes do Modo de Jogo.

O método `FSM` é um dos mais importantes, pois é nele que se encontra a implementação da máquina de estados. A sua descrição pormenorizada encontra-se no subcapítulo 4.4.3. O `recDadosRefBox` é executado sempre que a `RefBox` envia alguma mensagem, através dos mecanismos de *signals* e *slots* do Qt. As mensagens recebidas são colocadas num array de mensagens, para depois serem processadas na máquina de estados. O `recDadosRobos` é semelhante ao método anterior, mas neste caso as mensagens recebidas provêm dos robôs. As classes do Modo de Jogo são detalhadas a seguir.

#### **CRobo\_gui, CBola\_gui e CCampo\_gui**

Estas classes desenham um robô, uma bola e o campo de jogo à escala, respectivamente. São instanciados cinco objectos da classe `CRobo_gui` e `CBola_gui`, uma vez que é representada a bola que cada robô observa.

### **CParametros**

O objectivo desta classe é fornecer ao utilizador a opção de alterar o estado do jogo, o resultado, o nome do adversário e a cor da equipa. Para isso, quando a classe é criada recebe como parâmetro os valores actuais das variáveis, e exibe-as ao utilizador numa interface gráfica.

### **CVerDB**

Esta classe é constituída por uma interface gráfica, a qual mostra no ecrã a informação em formato de texto recebida dos robôs. A sua visualização é opcional, uma vez que toda essa informação já é exibida no ecrã, a maior parte na forma de desenho.

### **CVerTacticas**

A classe CVerTacticas é composta por uma interface gráfica, a qual desenha um campo à escala e cinco robôs. Sempre que a MINHO BaseStation calcula e envia posições para os robôs, o objecto desta classe recebe essas posições e desenha os robôs nas posições correspondentes. Com esta classe é possível verificar se as posições estão a ser calculadas correctamente, e se os robôs estão a ir para o local desejado.

### **CVerPapeis**

A classe CVerPapeis também é formada por uma interface gráfica. Ela apresenta no ecrã o papel que é atribuído a cada robô em cada instante. Apesar de a sua visualização ser facultativa, esta classe é bastante importante no decorrer os jogos, uma vez que possibilita saber qual o papel que cada robô está a desempenhar, se os papéis estão a ser atribuídos correctamente e se os robôs estão a receber os novos papéis.

### **CVirtualRefBox**

Durante a realização de testes, a *RefBox* virtual é sem dúvida uma mais-valia, pois apresenta os mesmos comandos que a *RefBox* oficial e não necessita que esta esteja a executar em nenhum computador nem seja efectuada nenhuma conexão. Esta classe é composta por uma interface gráfica, onde cada botão corresponde a um tipo de evento do árbitro. Uma vez que para operar a *RefBox* virtual é exigida a presença de um utilizador junto do computador, foi incorporada a opção de controlo através do controlo Wii Remote da Nintendo [30]. O Wii Remote

comunica com a aplicação através do Bluetooth, e permite a actuação da *RefBox* virtual à distância.

A lista de eventos disponibilizada pela classe *CVirtualRefBox* é a seguinte:

Tabela 2 - Lista de eventos fornecida pela *RefBox* virtual.

	Pontapé de saída: equipa ciano	Pontapé de saída: equipa magenta
<i>Start</i>	Lançamento: equipa ciano	Lançamento: equipa magenta
<i>Stop</i>	Falta: equipa ciano	Falta: equipa magenta
Bola ao solo	Canto: equipa ciano	Canto: equipa magenta
Estacionar	Pontapé de baliza: equipa ciano	Pontapé de baliza: equipa magenta
	Penákti: equipa ciano	Penákti: equipa magenta

#### 4.4.2. VARIÁVEIS

Controlar uma equipa de robôs futebolistas durante um jogo de forma dinâmica é um processo complicado, devido ao número de variáveis a controlar e ao número de tipos de situações que podem ocorrer. Então, foram criadas algumas definições para facilitar a identificação de situações, e criadas estruturas de memória para armazenar informação.

##### Papéis

Os papéis consistem em determinados comportamentos que os robôs devem adoptar. Foram criados para introduzir inteligência à equipa e possibilitar o jogo em equipa por parte dos robôs.

- Guarda-redes – É o papel atribuído ao robô guarda-redes. A sua execução consiste em ficar na pequena área e defender bolas que vão na direcção da baliza.
- Defesa – Este papel é o de defesa, e tem como objectivos proteger a baliza de remates e recuperar bolas ao adversário.
- Avançado – O Avançado é atribuído ao robô mais perto da bola. Este deve avançar na direcção da baliza adversária, e chutar à baliza caso esteja dentro do meio campo adversário.
- Marcador – Este papel é imputado nas situações de bola parada a favor da equipa MINHO TEAM, ao robô que passa a bola.

- Receptor – O papel Receptor é executado pelo robô que recebe a bola caso ocorra uma situação de bola parada a favor da equipa MINHO TEAM.

### Estados do Robô

Os estados do robô consistem nas acções que os robôs realizam, e estão de acordo com a aplicação que executa nos robôs durante os jogos.

- Idle – Sempre que o robô está parado.
- Positioning – Se ocorrer uma situação de bola parada, e o robô recebe coordenadas para se posicionar.
- Playing – Quando o árbitro envia o sinal *Start*, o robô entra no estado Playing, onde lhe é atribuído um papel.
- Searching – Caso nenhum dos robôs consiga visualizar a bola, um dos robôs executa o estado Searching, o que significa que procura o campo de forma ordenada até algum dos robôs detectar a bola.

### Base de Dados

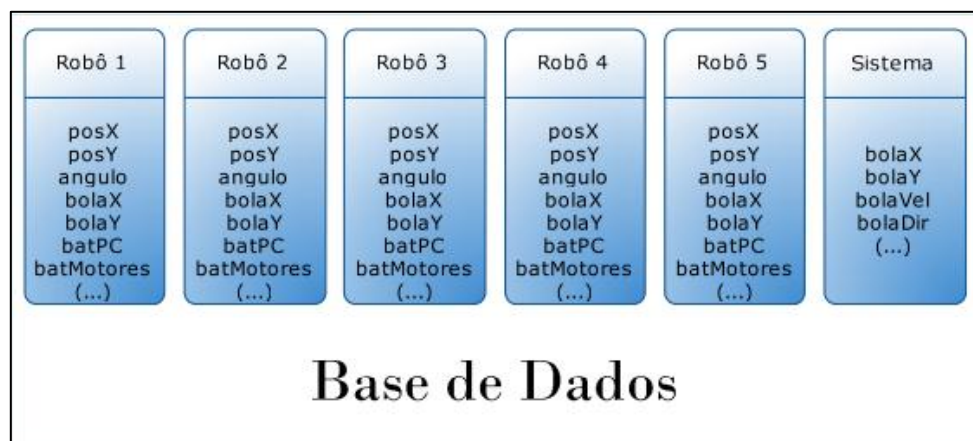


Figura 22 - Estrutura da base de dados implementada.

A base de dados consiste numa estrutura onde é armazenada a informação recente recebida dos robôs. Sempre que chegam dados provenientes dos robôs, esses dados são guardados no lugar correspondente ao robô responsável pelo envio. Depois, são utilizados para calcular a posição da bola, para verificar se algum robô deixou de comunicar, e para exibir a informação no modo gráfico. É ainda armazenada na base de dados a posição e velocidade actual da bola do sistema – que é obtida através dos dados de cada robô. A estrutura da base de dados é a da Figura 22.



## Táticas

A definição de táticas para as situações de bola parada é realizada no Modo de Edição. Lá, depois de as táticas serem escolhidas são guardadas num ficheiro. No Modo de Jogo, as táticas são lidas e carregadas para uma estrutura na memória no seu início, e depois são consultadas quando necessário. A estrutura consiste numa matriz com quatro dimensões, onde cada posição contém o valor da coordenada x, da coordenada y e da orientação. A definição e o funcionamento da estrutura são detalhados no capítulo 4.5.3.

## Array de Mensagens

O *array* de mensagens usado foi implementado segundo a filosofia FIFO (*First In First Out*). Sempre que chegam mensagens provenientes da *RefBox* ou dos robôs são colocadas no *array*, para depois serem processadas. No final, são removidas do *array*.

## Contador da Máquina de Estados

Para a elaboração da máquina de estados – secção abaixo, foi necessário implementar um temporizador. Para isso, foi utilizada a classe *QTimer* disponibilizada pelo Qt, a qual emite um *signal* sempre que a contagem chegou ao fim.

### 4.4.3. MÁQUINA DE ESTADOS

A máquina de estados implementada é responsável por todo o processamento de dados. É constituída por três estados e algumas condições para ocorrer a transição de estados. O seu diagrama de transição de estados UML encontra-se disponível na Figura 23.

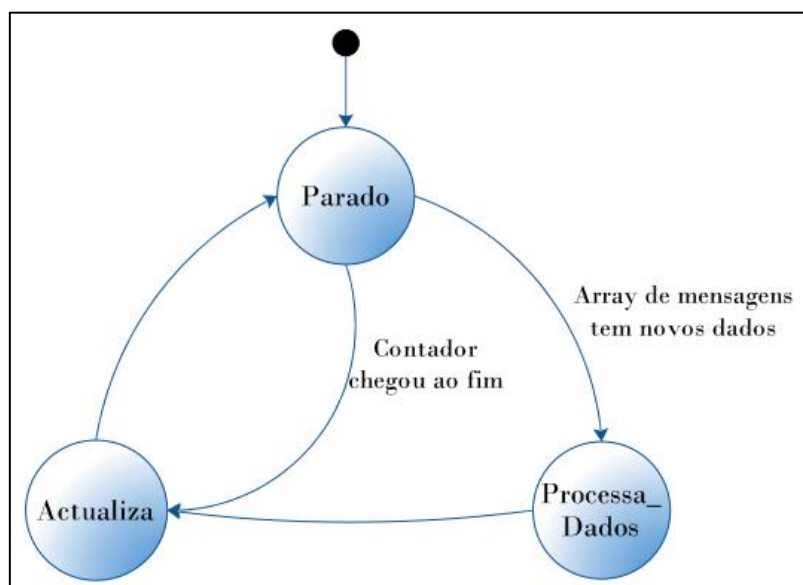


Figura 23 - Diagrama de transição de estados do Modo de Jogo.

Quando a máquina de estados é iniciada, o estado inicial é designado de Parado – a única acção realizada consiste em começar a contagem de tempo. Quando passarem 50 milissegundos (tempo escolhido para o contador) ocorre uma transição de estados, sendo o estado Actualiza o próximo. No Actualiza são realizadas várias tarefas, como por exemplo actualizar o display que indica o tempo de jogo na interface gráfica. Terminadas as tarefas do Actualiza, a máquina de estados volta a transitar para o estado Parado, onde o processo é reiniciado. Caso o Modo de Jogo receba dados (da *RefBox* ou dos robôs), então o estado Processa\_Dados é executado, e dentro deste é realizado o tratamento da informação recebida. Quando acabar o processamento dos dados é executado o estado Actualiza, e depois o estado Parado.

Nos próximos subcapítulos são descritos detalhadamente os estados Processa\_Dados e Actualiza.

#### 4.4.3.1. ESTADO PROCESSA\_DADOS

O estado Processa\_Dados pode ser representado pelo fluxograma da Figura 24.

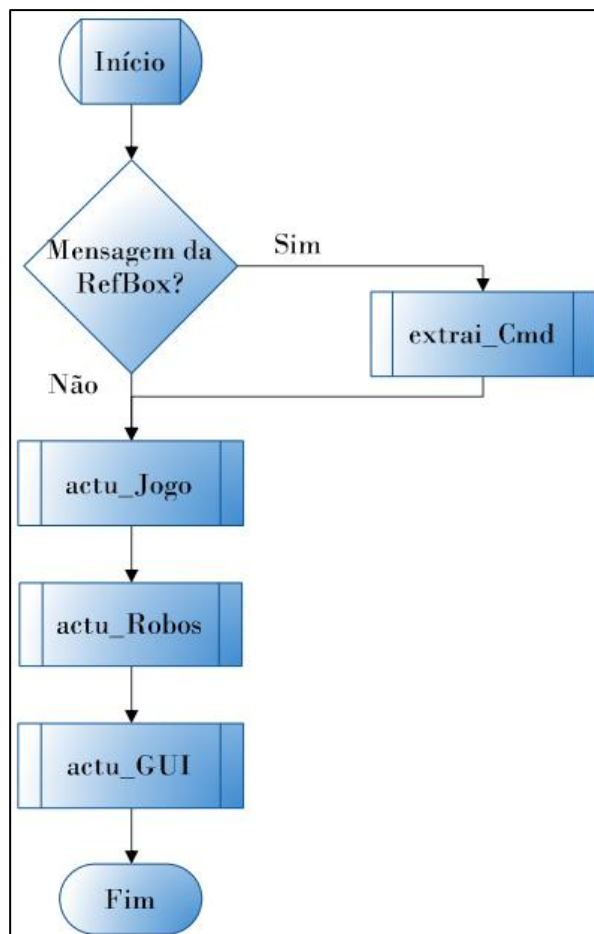


Figura 24 - Fluxograma do estado Processa\_Dados.

Este estado é executado sempre que a *RefBox* ou os robôs enviam dados para a MINHO BaseStation durante os jogos. Depois de iniciado, é efectuado um teste a comparar o tipo de dados a processar. Caso seja uma mensagem proveniente da *RefBox*, é convertida a mensagem num estado ou numa situação de jogo correspondente. A actualização do jogo, robôs e GUI é realizada independentemente da origem da mensagem. Cada rotina do estado Processa\_Dados é explicada no próximo texto.

### **extrai\_Cmd**

Esta função é responsável por converter as mensagens da *RefBox* num estado de jogo ou situação de jogo, e está preparada para funcionar com o protocolo2009 e o protocolo2010.

### **actu\_Jogo**

No actu\_Jogo é feita a actualização do estado do jogo conforme a origem da mensagem.

Se a mensagem consistir em dados enviados pelos robôs, esses dados são colocados na base de dados. De seguida, é calculada a posição e a velocidade da bola através dos valores de todos os robôs, obtendo-se assim uma única estimativa para a posição e velocidade da bola em campo. A fórmula usada para esse cálculo consistiu na média aritmética simples, mas posteriormente foi substituída pela média aritmética ponderada, uma vez o sistema de visão do robô perde resolução à medida que é aumentada a distância, e ao atribuir uma maior importância à medida do robô que está mais próximo da bola foram obtidos melhores resultados [31] (ver capítulo 5.1.1).

Caso a mensagem seja oriunda da *RefBox* e o seu conteúdo coincida com as situações de jogo para bola parada, é calculada e conferida uma posição e um papel a cada robô. De acordo com o tipo de situação de jogo, é pesquisado na estrutura que armazena as táticas qual a posição definida para os cinco robôs. Depois, é atribuído a cada robô um papel e uma posição. A atribuição é realizada em função do número de robôs em campo, sendo o Guarda-redes o primeiro papel a ser atribuído. A seguir, é atribuído o papel de Marcador e Receptor no caso de situações de bola parada a favor da equipa, ou Avançado caso contrário. Aos restantes robôs é conferido o papel de Defesa.

Por último, é realizada a gravação da informação recebida em ficheiro. Foram criadas três estruturas diferentes – uma para os robôs, uma para a *RefBox* e uma para o sistema, onde cada uma cria um ficheiro no arranque do programa com a data e a hora como nome (para cada robô é criado um ficheiro próprio). Para o formato dos ficheiros foi escolhido o XML pela sua

simplicidade e legibilidade, e por permitir criar formatos de dados como listas, registos ou árvores [32]. Então, as três estruturas criadas são:

- *RefBox*: todas as mensagens recebidas da *RefBox* são guardadas no ficheiro. Para isso, foram criados os seguintes formatos:

tempoS	tipo	tempo	id		
tempoS	tipo	tempo	id	comando	equipa

Figura 25 - Formato dos dados da *RefBox* guardados em ficheiro.

O tempoS consiste no tempo do sistema, e é necessário para garantir que no Modo de Análise os ficheiros são reproduzidos em sincronia. O tipo pode ser Gmi (*Game Info*) ou Ref (*Referee*).

- Robôs: O formato do robô é o mais longo de todos, pois são várias as informações que são guardadas. A estrutura é a seguinte:

tempoS	num	px	py	dir	bx	by	bv	bd	estado	bats	vars
--------	-----	----	----	-----	----	----	----	----	--------	------	------

Figura 26 - Formato dos dados dos robôs guardados em ficheiro.

O num indica o número do robô, e são guardadas a posição e orientação do robô em campo, a informação da bola e um conjunto de variáveis do robô.

- Sistema: No ficheiro sistema são armazenadas o resto das informações relevantes, como a posição e velocidade da bola calculada ou o nome do adversário e resultado de jogo. Foram criados os dois seguintes formatos:

tempoS	tipo	msg			
tempoS	tipo	bx	By	bv	bd

Figura 27 - Formato dos dados do sistema guardados em ficheiro.

### **actu\_Robos**

O actu\_Robos tem a incumbência de enviar dados para os robôs, caso se trate de uma mensagem da *RefBox* com uma situação de jogo. É aqui que é enviada a posição e o papel que o robô deve executar nas situações de bolas paradas, ou simples comandos de jogo como *Start* ou *Stop*.

Como o protocolo de comunicação estabelecido com os robôs é UDP, não existe garantia que a mensagem enviada chega ao destino. Uma vez que todas as mensagens importantes (principalmente a mensagem de *Stop*) têm obrigatoriamente que chegar aos robôs, foi criado um mecanismo que garante que as mensagens são recebidas pelos robôs. O mecanismo consiste em enviar um número juntamente com a mensagem, e receber de volta esse número do robô. Em cada mensagem o número enviado é diferente. É também definido um tempo máximo para receber o número do robô, e caso esse tempo seja atingido sem se obter resposta a mensagem é reenviada para o robô. O valor escolhido foi de 0,5 segundos.

### **actu\_GUI**

Esta função é responsável por actualizar os dados exibidos no ecrã. São actualizados os robôs e a bola desenhados no campo, os painéis com informação relativos a cada robô e o painel que mostra o histórico de mensagens recebidas da *RefBox*. Também é actualizado o resultado do jogo caso ocorra algum golo, e se alguma das interfaces gráficas opcionais estiver seleccionada também ela vê os seus valores actualizados.

#### **4.4.3.2. ESTADO ACTUALIZA**

O estado Actualiza é executado várias vezes por segundo e é responsável por executar várias tarefas, como actualizar o tempo de jogo na interface gráfica, verificar as conexões com os robôs ou atribuir novos papéis. O fluxograma que representa o funcionamento do estado Actualiza é o da figura seguinte.

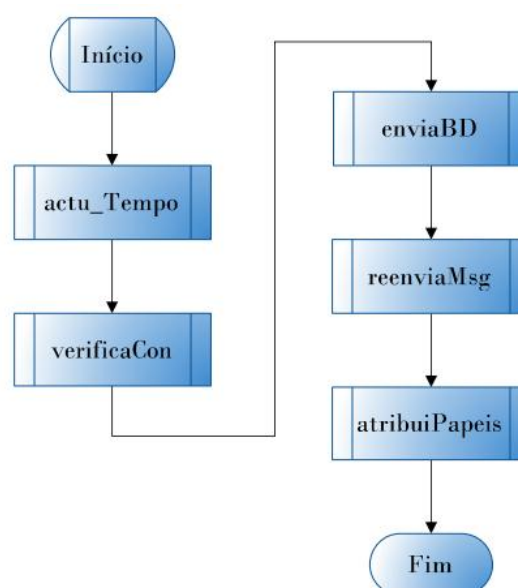


Figura 28 - Fluxograma do estado Actualiza.

Cada uma das rotinas que compõe o estado Atualiza tem os seguintes objectivos:

### **actu\_Tempo**

A função actu\_Tempo actualiza o tempo actual de jogo que é exibido no ecrã, e ainda se o jogo se encontra na primeira parte, segunda parte, intervalo ou fim de jogo.

### **verificaCon**

Esta função é responsável por verificar se os robôs estão aptos para jogar. Ela analisa a conexão com os robôs, e quando estes ficarem sem enviar qualquer mensagem durante um determinado tempo são considerados como não aptos para jogar. Sempre que é alterado o estado de algum dos robôs, essa situação é sinalizada graficamente no painel de cada robô. Para o correcto funcionamento da equipa MINHO TEAM é importante a MINHO BaseStation ter conhecimento de quais são os robôs que estão realmente em jogo durante a atribuição de papéis, para que não ocorram situações do tipo atribuir o papel de guarda-redes a um robô que está inapto.

### **enviaBD**

Nesta rotina é enviada a informação da base de dados para os robôs, e foi definida uma frequência de envio de 10 vezes por segundo. No enviaBD é analisado o tempo que passou desde o último envio de dados, e sempre que o tempo for igual ou superior a 0,1 segundos a informação é enviada de novo. Quanto ao conteúdo das mensagens, é enviado para cada robô a posição e velocidade da bola calculada no estado Processa\_Dados. A informação enviada permite que os robôs tenham informação da bola mesmo que a sua visão não a visualize.

Neste envio de informação não foi implementado nenhum mecanismo para garantir que os robôs recebem todas as mensagens enviadas, uma vez que não é crítico a perda de algumas mensagens na rede. Já o atraso na entrega de mensagens é algo que pode afectar o comportamento dos robôs.

### **reenviaMsg**

Realiza o reenvio de mensagens classificadas como críticas. Em primeiro, é verificado se existem mensagens à espera de confirmação por parte dos robôs como foram recebidas. Caso existam, se o tempo estimado para a recepção da resposta for inferior ao tempo que passou desde o envio, a mensagem é reenviada para o robô e a contagem de tempo é reiniciada.

### atribuiPapeis

O atribuiPapeis apenas é efectuado quando o jogo não se encontra parado, e tem como função atribuir aos robôs novos papéis e posições durante os jogos. Uma das atribuições acontece depois de situações de bola parada a favor da equipa MINHO TEAM, em que os robôs com o papel Marcador e Receptor recebem novos papéis. Os robôs com o papel de Defesa são enviados para uma posição de defesa, onde devem bloquear os adversários e impedir remates à sua baliza. É também efectuada a troca do papel de Avançado entre robôs, caso algum dos robôs se encontre mais próximo da bola do que o actual avançado. Nessa situação, os papéis dos dois robôs em questão são trocados.

#### 4.4.4. COMUNICAÇÃO

No Modo de Jogo é realizada a comunicação com a *RefBox* e com os robôs. É possível observar, na Figura 29, o esquema de ligações e as mensagens trocadas.

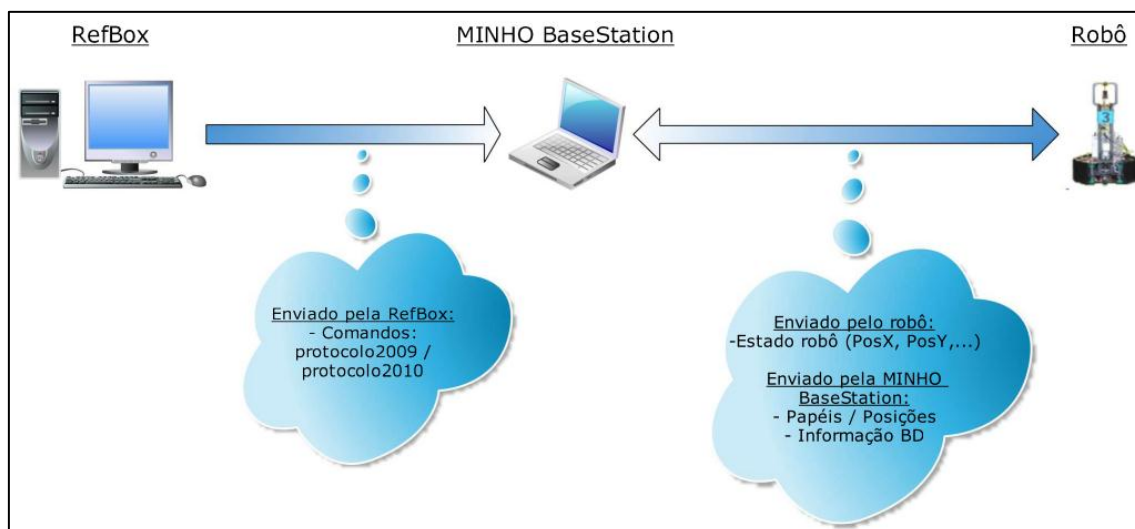


Figura 29 – Comunicação realizada no Modo de Jogo.

A *RefBox* envia comandos de jogo para a MINHO BaseStation, utilizando o protocolo2009 ou o protocolo2010. No protocolo2009, é enviado um caractere em cada mensagem. Existe um total de vinte e três mensagens possíveis, as quais se encontram descritas no Apêndice A. Quanto ao protocolo2010, as mensagens chegam no formato XML. Todas as mensagens contêm o tempo de jogo, o id (número) da mensagem e o estado do jogo. A lista com um exemplo de cada tipo de mensagem pode ser consultada no Apêndice B.

Quanto aos robôs, eles enviam o seu estado, nomeadamente a posição e orientação do robô, posição e velocidade da bola, tensão das baterias e um conjunto de variáveis internas (como estado, papel, comportamentos, entre outros).

A MINHO BaseStation envia para os robôs dois tipos de mensagens: atribui papéis / posições para os robôs (inclui comandos como *Start* e *Stop*), e envia a informação da base de dados (é enviada várias vezes por segundo, e permite que o robô tenha conhecimento da posição da bola mesmo que não a veja através do seu sistema de visão).

## **4.5. MODO DE EDIÇÃO**

O Modo de Edição foi projectado como uma ferramenta de suporte ao desenvolvimento da equipa MINHO TEAM, nomeadamente na criação de táticas, no auxílio do teste dos algoritmos de visão, na verificação do hardware dos robôs, na definição de papéis e na alteração de parâmetros do robô remotamente.

Encontra-se dividido em seis partes, onde cada uma contém várias funcionalidades. Uma das partes é designada de Controlo Manual do Robô, e permite ao utilizador além de observar qual a posição do robô em campo enviar o robô para posições desejadas. Permite ainda controlar os dispositivos de hardware do robô, como o chute horizontal e vertical, e ainda atribuir manualmente papéis aos robôs. O Editor de Bolas Paradas é onde é possível definir posições para os robôs para as situações de bola parada. Existe também o Editor de Papéis, onde são definidos e escolhidos papéis e posições para os robôs durante os jogos. Outra das partes consiste na Localização, a qual além de mostrar a localização dos robôs em campo também exhibe os pontos detectados pelo algoritmo de localização dos robôs. O Parâmetros de Jogo abrange todos os valores de parâmetros do Modo de Jogo que podem ser alterados – como o tempo de envio de dados para os robôs ou o tempo máximo estabelecido para obter confirmação de mensagens recebidas por parte dos robôs. Por fim, a parte chamada de Parâmetros do Robô possibilita a alteração remota de parâmetros do robô.

Para a realização das funcionalidades descritas anteriormente, foi implementado um painel constituído por separadores, o que obriga à exibição de um separador de cada vez. No separador correspondente ao Controlo Manual do Robô foi introduzido um campo de jogo e os robôs desenhado à escala. Sempre que é desejado mover o robô para uma posição específica, basta arrastar o robô pretendido com o rato para a posição desejada. No separador existe



também um painel para cada robô, onde é possível atribuir papéis, ver informações do robô ou controlar o hardware. Para a definição de táticas para situações de bola parada, existe no separador correspondente um campo desenhado à escala onde os robôs são colocados nas posições desejadas pelo utilizador.

### 4.5.1. ESTRUTURA DO MODO DE EDIÇÃO

Na construção da interface gráfica do Modo de Edição foi utilizado um painel de separadores, recorrendo à classe `QTabWidget` fornecida pelo Qt. A comunicação realizada com os robôs é concretizada utilizando um *socket* UDP fornecido pela classe `QUdpSocket`. Sempre que os robôs enviam mensagens o método `recDadosRobos` é executado, sendo depois os dados lidos do *socket*. O diagrama de classes com os atributos e métodos mais importantes do Modo de Edição está representado na Figura 30.

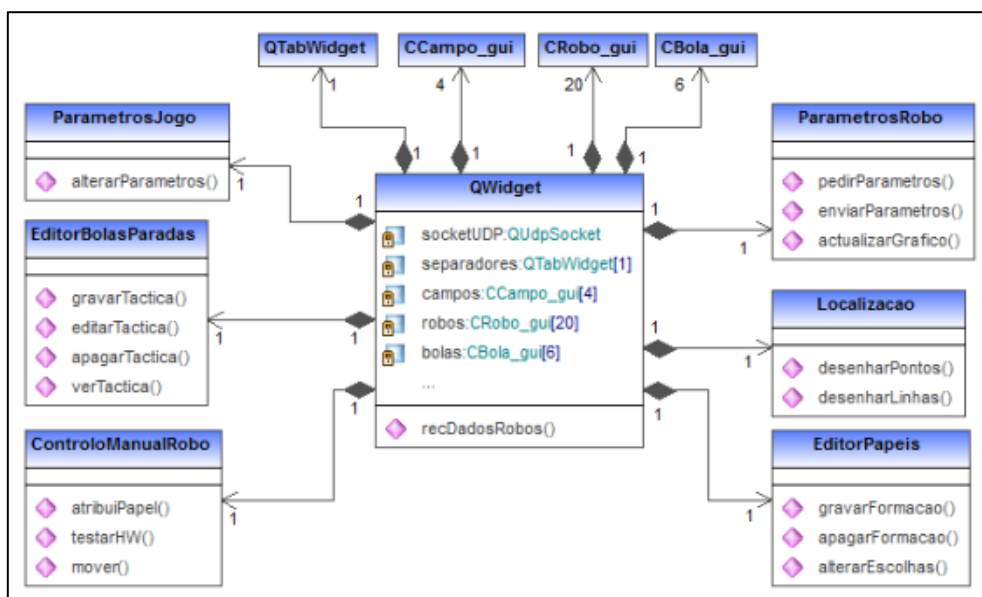


Figura 30 - Diagrama de classes do Modo de Edição.

De seguida é descrita a função de cada classe do Modo de Edição.

#### QTabWidget

Esta é uma classe do Qt e fornece um conjunto de painéis, implementando cada um deles num separador. É a classe que gere a comutação entre separadores e a exibição do painel correspondente ao separador escolhido.

### **CRobo\_gui, CBola\_gui e CCampo\_gui**

Estas classes desenham um robô, uma bola e o campo de jogo à escala, respectivamente.

### **ControloManualRobo**

No ControloManualRobo existem todos os atributos e métodos necessários para a execução das tarefas do Controlo Manual do Robô, como os métodos atribuirPapel, testarHW ou mover.

### **EditorBolasParadas**

Esta classe fornece todos os recursos necessários para criar, apagar, editar ou ver as tácticas definidas para todas as situações de bola parada.

### **EditorPapeis**

O EditorPapeis é a classe responsável por criar e gerir os papéis e as formações que os robôs devem adoptar durante os jogos.

### **Localizacao**

A classe Localizacao possui as variáveis essenciais para a exibição de informação, e o método designado para desenhar os pontos recebidos do robô na posição correcta no campo de jogo exibido no ecrã.

### **ParametrosJogo**

À semelhança das anteriores, esta classe auxilia a alteração dos valores dos parâmetros necessários para a realização de jogos.

### **ParametrosRobo**

Esta classe é responsável por obter o valor dos parâmetros do robô, enviar os novos valores e ainda actualizar os gráficos de acordo com os valores escolhidos.

De seguida será explicado detalhadamente cada um dos separadores do Modo de Edição.

### 4.5.2. CONTROLO MANUAL DO ROBÔ

Esta funcionalidade foi implementada na MINHO BaseStation para auxiliar o desenvolvimento da equipa MINHO TEAM. Permite controlar todo o hardware do robô, testar a localização, a visão do robô e um conjunto de comportamentos. É composto por um campo de jogo, robôs e bolas, todos desenhados à escala, e um painel de informações para cada robô.

Sempre que cada robô envia dados, eles são exibidos na interface gráfica. O robô é desenhado no campo de acordo com a informação recebida – posição e orientação, assim como a bola. As restantes informações – como tensão das baterias, valor da bússola ou estados dos infravermelhos, são colocadas no painel correspondente.

Em cada um dos painéis com informação sobre os robôs, além de ser exibido o estado de alguns componentes de hardware também é possível efectuar o controlo individual de todo o hardware do robô. Para isso, existe um conjunto de botões que permite chutar horizontal e vertical, regular a intensidade do chuto, actuar o mecanismo de chuto, ligar/desligar o sistema de drible e mover o robô em quatro direcções pré-definidas (frente, trás, esquerda e direita).

Quanto à realização de testes sobre o software, a aplicação permite atribuir manualmente papéis a cada robô e iniciar ou parar a execução desses papéis. Além disso, é possível mover o robô para posições específicas arrastando o robô desenhado no campo.

### 4.5.3. EDITOR DE BOLAS PARADAS

O Editor de Bolas Paradas é utilizado para ver, criar, editar ou apagar posições para os robôs nas situações de bola parada, a favor ou contra a equipa MINHO TEAM.

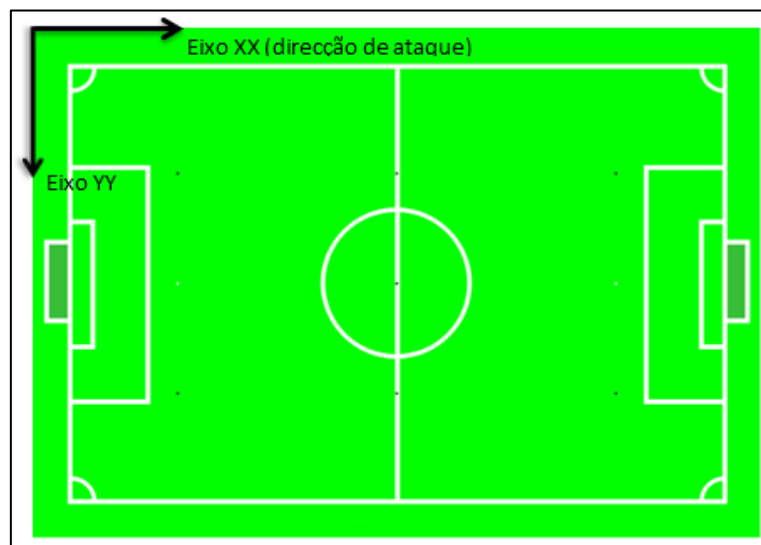


Figura 31 - Sistema de eixos adoptado.

Isto é, sempre que durante os jogos ocorre uma situação de bola parada, a posição e orientação que os robôs devem ocupar em campo na reposição do jogo é previamente definida no Editor de Bolas Paradas.

As situações de bola parada podem ser divididas em dois grupos<sup>2</sup> - a favor e contra, sendo cada um desses grupos constituídos pelas situações: Pontapé de saída (*KickOff*), Lançamento (*ThrowIn*), Canto (*Corner*), Pontapé de baliza (*GoalKick*), Falta (*FreeKick*), Penáلتi (*Penalty*), Bola ao solo (*Dropped ball*) e Estacionar (*Parking*).

Todas as tácticas definidas são guardadas num ficheiro XML. Cada nó da estrutura XML implementada consiste no seguinte:

BolaX	X1	X2	X3	X4	X5
BolaY	Y1	Y2	Y3	Y4	Y5
	Dir1	Dir2	Dir3	Dir4	Dir5

Figura 32 - Constituição de cada nó no ficheiro das tácticas.

Para cada táctica criada é gravado no ficheiro a posição da bola, a posição e a orientação dos cinco robôs da equipa.

Sempre que a aplicação é iniciada, as tácticas são lidas do ficheiro para uma estrutura em memória. O mesmo acontece no Modo de Jogo, em que uma estrutura igual é utilizada para armazenar a posição e orientação para os robôs. A estrutura é constituída por uma matriz com quatro dimensões, onde em cada posição existe uma coordenada X e Y e uma orientação. A primeira dimensão tem o tamanho de dois – indica qual a origem do lance, a segunda dimensão apresenta um tamanho de oito – corresponde ao tipo de situação, e o tamanho da quarta dimensão é seis – na primeira é guardada a posição da bola e nos cinco seguintes a posição e orientação de cada robô. A terceira dimensão está directamente relacionada com o tipo de situação.

Cada uma das situações possíveis encontra-se descrita a seguir, nomeadamente como é gravada e consultada a informação.

#### **4.5.3.1. PONTAPÉ DE SAÍDA**

Nos lances de pontapé de saída a bola é colocada no centro do campo. Depois, os robôs da equipa que vai beneficiar do lance aproximam-se da bola para executarem o passe, enquanto a outra equipa aguarda no seu meio campo e a uma distância mínima da bola de três metros.

---

<sup>2</sup> À excepção de Bola ao solo e Estacionar, que não são atribuídos a nenhuma das equipas.

Para a definição de posições para os pontapés de saída, é definida uma tática para situações a favor e outra para situações contra. A bola é colocada no centro do campo e os robôs são colocados nas posições desejadas.

Durante os jogos quando ocorrer este tipo de situações, as posições enviadas para os robôs consiste nas únicas definidas para cada robô. Como a bola pode não se encontrar exactamente no centro do campo, todas as posições enviadas aos robôs são corrigidas tendo em conta a posição real da bola quando é executado o lance.

#### 4.5.3.2. LANÇAMENTO

A reposição de lançamentos é efectuada ao longo das linhas laterais do campo. Sempre que a bola ultrapassa na sua totalidade as linhas laterais do campo, é assinalado lançamento e o árbitro coloca a bola no sítio onde ela saiu do campo - em cima da linha.

Para a escolha de táticas para os lançamentos, é seleccionado um conjunto de pontos para a bola ao longo da linha lateral e é definida a posição e orientação desejada para cada robô, uma vez que é inviável definir táticas ao longo de toda a linha.

Quando são marcados lançamentos, o procedimento para obter a posição e a orientação dos robôs é o seguinte:

- Através da posição da bola, é procurada a tática mais próxima definida à esquerda e a tática mais próxima definida à direita.
- É utilizado o método denominado de interpolação linear [33] para calcular a posição de cada robô, em função da posição desse robô na tática à direita e à esquerda.
- No final, a posição resultante é reajustada (na coordenada Y) de acordo com a real posição da bola – por vezes o árbitro coloca a bola afastada alguns decímetros da linha.

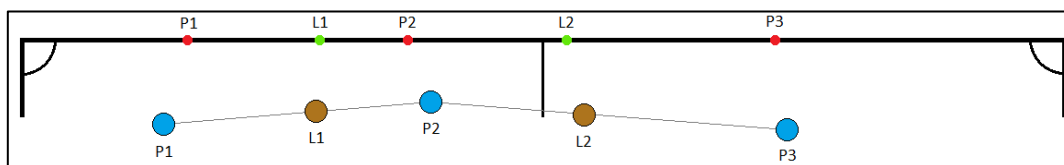


Figura 33 - Exemplo da obtenção de posições num lançamento.

A Figura 33 ilustra o cálculo da posição para um robô. Foram definidos três pontos para a bola (a vermelho), e o robô (a azul) foi colocado em cada um deles de acordo com a

preferência do utilizador. Para dois possíveis lançamentos (a bola a verde), os robôs com a cor castanho indicam a posição calculada para cada um dos lançamentos. Para o lançamento 1, foram usados os pontos 1 e 2, enquanto para o lançamento 2 foram utilizados os pontos 2 e 3. A linha de cor cinzenta que une os robôs de cor azul corresponde às posições atribuídas ao robô caso o lançamento ocorra entre o ponto 1 e o ponto 3.

#### **4.5.3.3. CANTO**

Sempre que ocorre um canto, existem dois sítios onde o árbitro pode colocar a bola para a reposição do jogo: à direita ou à esquerda da baliza, no lugar designado para os cantos. Então, na definição de tácticas para cantos foram escolhidas um total de quatro situações: duas para lances defensivos (contra a equipa) e duas para lances ofensivos (a favor da equipa), e cada lance foi definido no lugar à esquerda e no lugar à direita da baliza.

Depois, no decorrer dos jogos, quando é marcado um canto é analisado se o árbitro colocou a bola à esquerda ou à direita da baliza. Em função desse resultado, é procurada a táctica que corresponde a essa situação. A posição e a orientação guardada na táctica é enviada para cada robô, depois de ser reajustada em função da posição real da bola e da posição definida para a bola.

#### **4.5.3.4. PONTAPÉ DE BALIZA**

A definição dos lances de pontapé de baliza é análoga à dos cantos. A diferença reside no lugar onde os lances são recomeçados, uma vez que para os pontapés de baliza o lugar estipulado para a reposição da bola são duas marcas situadas em frente à grande área.

#### **4.5.3.5. FALTA**

A marcação de faltas pode ocorrer em qualquer lugar do campo. A bola é colocada pelo árbitro no lugar onde ocorreu a falta, excepto quando ocorrem dentro da grande área. Nessa situação, a bola é posta nas marcas em frente à grande área, que também são usadas para a marcação de pontapés de baliza.

Para a definição de tácticas, é colocada a bola num ponto do campo e os robôs na posição desejada – formando assim uma nova táctica. Quando são criadas as primeiras três tácticas é formado um triângulo, que é composto pelos pontos onde a bola foi colocada. Sempre que é adicionada uma nova táctica é criado um novo triângulo, formado pelo novo ponto e pelos

dois mais próximos. Para a criação de triângulos através dos pontos definidos foi utilizado o método de Delaunay. Os triângulos formados nas faltas a favor são diferentes dos das faltas contra, uma vez que as posições pretendidas para os robôs são diferentes.

O processo realizado durante os jogos sempre que são assinaladas faltas consiste em:

- É procurado qual o triângulo a que a posição actual da bola pertence.
- Para cada robô é calculada a posição pretendida, utilizando para isso as três posições definidas para as três situações do triângulo.
- Para o cálculo das posições é utilizado o método geométrico das coordenadas baricêntricas [34].

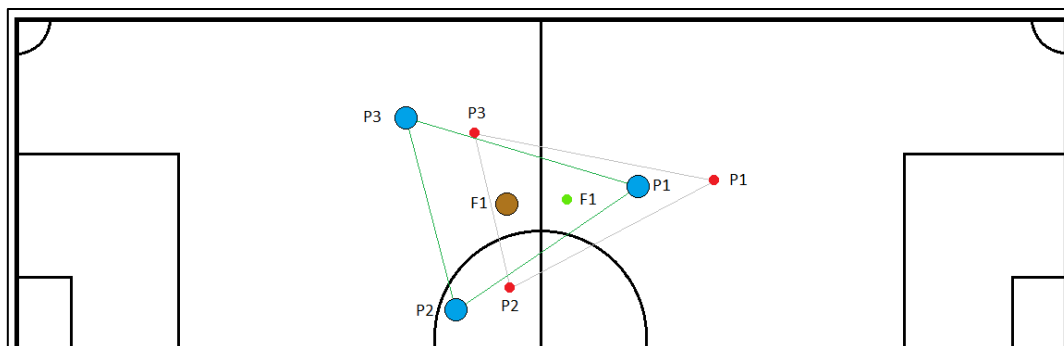


Figura 34 - Ilustração do cálculo de posições nas faltas.

A Figura 34 apresenta a marcação de uma falta e a posição atribuída a um robô. O local onde a falta foi assinalada está representado por uma bola verde, e corresponde ao triângulo formado pelos pontos a vermelho (as posições da bola em cada situação). De cor azul encontram-se os robôs para cada um dos pontos definidos, e a cor castanho indica a posição atribuída ao robô na marcação da falta. Caso a falta seja marcada dentro do triângulo formado pelos pontos a vermelho, a posição resultante para o robô será dentro do triângulo formado pelos pontos azuis.

#### 4.5.3.6. PENÁLTI

O procedimento para a escolha de táticas nos penáltis é idêntico ao realizado para os pontapés de saída, uma vez que na marcação de penáltis a bola é sempre colocada na marca de penálti.

#### **4.5.3.7. BOLA AO SOLO**

A criação de táticas para a ocorrência de situações de bola ao solo é equivalente ao procedimento da marcação de faltas, uma vez que a reposição de bola neste tipo de ocorrência pode ser assinalada em qualquer lugar do campo<sup>3</sup>.

#### **4.5.3.8. ESTACIONAR**

Quando é recebido o comando de estacionar é pretendido que os robôs se desloquem para junto da linha lateral, de uma forma ordenada, e fiquem alinhados. Então, é escolhida a posição e a orientação para cada robô, e sempre que é recebido o comando para estacionar a posição e orientação enviadas a cada robô correspondem exactamente às definidas como Estacionar.

#### **4.5.4. EDITOR DE PAPÉIS**

O Editor de Papéis oferece ao utilizador a possibilidade de criar formações para os defesas durante os jogos, e ainda escolher que formações utilizar durante os períodos do jogo.

A equipa MINHO TEAM na realização de jogos de futebol robótico utiliza um dos robôs como guarda-redes, outro como avançado e os restantes como defesas. Uma vez que o guarda-redes posiciona-se dentro da pequena área e o avançado disputa a bola onde quer que esta se encontre, é necessário atribuir posições aos defesas durante os jogos. Então, através da criação de um campo desenhado à escala, é possível mover três robôs para três posições desejadas e guardar uma nova formação. A ordem de colocação dos robôs nas posições pretendidas é importante, uma vez durante os jogos a atribuição aos robôs é feita nessa ordem.

Relativamente a qual formação utilizar durante os jogos, a definição dessas escolhas é realizada no Editor de Papéis. Depois, no Modo de Jogo, a formação é seleccionada de acordo com os critérios:

- Utilizar a mesma formação para todo o jogo: se esta opção for activada, a formação escolhida será a utilizada durante todo o jogo.
- Quando o jogo estiver empatado: sempre que o resultado do jogo se traduza num empate, a formação atribuída é a escolhida nesta opção.
- Sempre que o número de jogadores for menor do que 4: todas as vezes que a equipa possuir menos do que quatro robôs, é seleccionada esta formação.

---

<sup>3</sup> Excluindo o interior da grande área de cada equipa.



- Quando estiver em vantagem: nas situações em que a equipa MINHO TEAM esteja em vantagem no marcador, é usada a tática definida.
- Se estiver em desvantagem: caso a equipa MINHO TEAM se encontre a perder, a formação atribuída aos robôs será a seleccionada nesta opção.

A introdução das preferências descritas acima permite que a equipa MINHO TEAM apresente vários comportamentos posicionais durante o jogo, e reaja a alterações no marcador ou à perda de jogadores de campo.

#### **4.5.5. LOCALIZAÇÃO**

O separador Localização foi implementado para auxiliar o desenvolvimento e teste dos algoritmos de localização dos robôs.

Uma das suas funcionalidades consiste em receber e desenhar a posição dos robôs e os pontos detectados pelo algoritmo de visão. Para isso, no separador foi desenhado um campo à escala, onde cada robô é colocado de acordo com a posição recebida. Caso o algoritmo do robô detecte pontos (transições verde-branco e branco-verde no campo), as coordenadas deles são enviadas e cada um dos pontos recebidos é desenhado no campo. Isto permite saber em qualquer altura quais os pontos que o robô usa para estimar a sua posição em campo, assim como se a distância calculada para cada ponto está correcta.

Outra das funções consiste em verificar se o ficheiro das distâncias usado pelo software do robô está correcto. Ou seja, um dos algoritmos do robô necessita de um ficheiro que, para cada posição do campo (com resolução de 1dm<sup>2</sup>) informa qual a distância à linha mais próxima a toda à volta desse ponto (num total de 36 direcções). Para a validação dos dados desse ficheiro, os valores são carregados para uma estrutura em memória. No campo desenhado na interface gráfica, movendo o rato ao longo do campo é possível observar o desenho de linhas à sua volta, e caso os valores do ficheiro se encontrem correctos as linhas desenhadas vão desde o rato até à linha do campo mais próxima.

É também da responsabilidade do separador Localização gerar um ficheiro, composto por um campo de zeros e uns à escala do campo de jogo. A resolução para gerar o ficheiro é configurável e escolhida pelo utilizador. O ficheiro produzido é necessário para o algoritmo de localização dos robôs.

#### **4.5.6. PARÂMETROS DO JOGO**

O separador Parâmetros do Jogo é empregado para modificar os valores de um conjunto de variáveis, sendo grande parte utilizadas durante a realização de jogos. O valor dos parâmetros sempre que é alterado é guardado num ficheiro XML.

Do conjunto de variáveis fazem parte os parâmetros do jogo (contador da máquina de estados, tempo de verificação de conexões, entre outro), o endereço MAC do comando Wii usado na *RefBox* virtual, e a directoria onde os *logs* são gravados.

#### **4.5.7. PARÂMETROS DO ROBÔ**

O Parâmetros do Robô é o último separador apresentado pelo Modo de Edição na interface gráfica do programa.

Uma das suas funções consiste em exibir e alterar os valores de um conjunto de parâmetros utilizados no software do robô. Para isso, os parâmetros são enviados pelos robôs e mostrados no ecrã ao utilizador. No fim, os valores de todos os parâmetros são enviados para os robôs.

Outra das funcionalidades comporta receber e desenhar a posição da bola relativamente ao robô. A localização do robô não é necessária, e com a posição da bola em relação ao robô consegue-se observar o alcance máximo do sistema de visão, se o mapa das distâncias usado pelo robô está correcto e se o sistema de visão (espelho, câmara ou estrutura de ferro) está alinhado com o eixo vertical o robô.

#### **4.5.8. COMUNICAÇÃO**

No Modo de Edição é realizada a comunicação com os robôs, nomeadamente nos separadores Controlo Manual do Robô, Localização e Parâmetros do Robô. A próxima Figura exemplifica o tipo de mensagens enviadas.

No separador Controlo Manual do Robô, os robôs apenas enviam o seu estado (posição e orientação, capacidade das baterias, estado infravermelhos, etc.). A MINHO BaseStation envia mensagens de hardware (para testar o chute, o drible e o solenóide de chute), e atribui papéis e posições aos robôs.

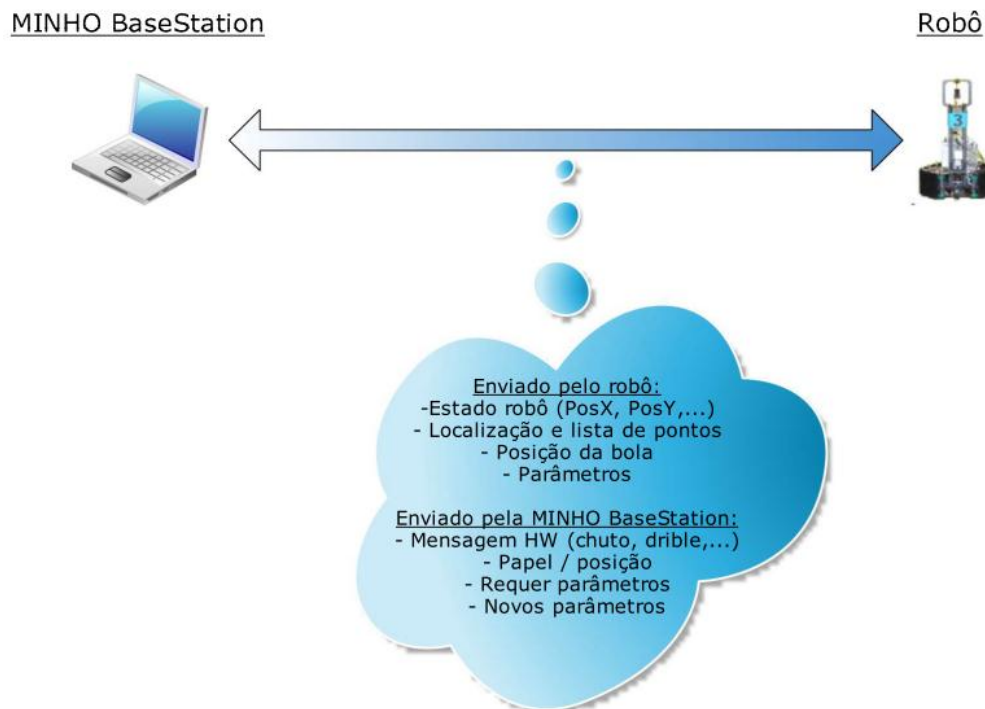


Figura 35 - Comunicação realizada no Modo de Edição.

No separador da Localização só os robôs enviam dados. Cada mensagem contém a posição e orientação do robô, o número de pontos detectados pelo robô e a posição de cada um desses pontos.

Quanto ao separador Parâmetros do Robô, a MINHO BaseStation efectua o pedido do valor dos parâmetros e envia os novos valores escolhidos pelo utilizador. Os robôs enviam continuamente a posição da bola para a modelação local, e sempre que solicitados, enviam também o valor actual dos parâmetros usados.

## 4.6. MODO DE ANÁLISE

O Modo de Análise foi desenvolvido para possibilitar a análise dos jogos realizados. A partir da informação recolhida e gravada em ficheiros durante os jogos, o modo de análise realiza a reprodução exacta de jogos da equipa MINHO TEAM.

Durante a realização de jogos, o Modo de Jogo regista todas as informações importantes nos ficheiros correspondentes – informações sobre a *RefBox*, cada robô e a MINHO BaseStation. O formato utilizado para os ficheiros é o XML. No Modo de Análise, o utilizador começa por seleccionar quais os ficheiros que deseja reproduzir – pode optar pelos ficheiros da *RefBox*, de cada robô ou do sistema. Depois, pode escolher reproduzir o jogo, parar a reprodução, alterar a velocidade de reprodução ou ainda avançar uma *frame* de cada vez.

Para a reprodução de jogos, foi implementada uma interface gráfica semelhante à do Modo de Jogo. Foi desenhado um campo, robôs e bolas, todos à escala, onde os robôs e as bolas são desenhados de acordo com os dados do ficheiro de cada um. Um dos painéis existentes é designado para registar os comandos recebidos da *RefBox* durante o jogo em análise. Para cada robô é atribuído um painel onde são colocadas informações sobre o estado das baterias, dos infravermelhos, o estado e o papel actual do robô.

#### 4.6.1. ESTRUTURA DO MODO DE ANÁLISE

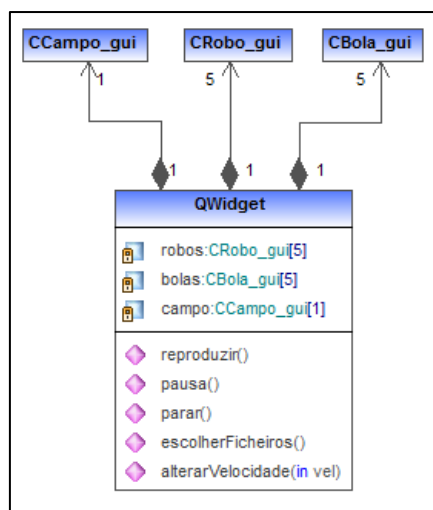


Figura 36 - Diagrama de classes do Modo de Análise.

A representação da estrutura do Modo de Análise através de um diagrama de classes simplificado pode ser vista na Figura 36. São utilizadas três classes para a representação visual do campo, dos robôs e da bola. São também usadas algumas classes facultadas pelo Qt, sobretudo para a construção e actualização da interface gráfica e para a leitura dos ficheiros no formato XML. Os métodos mais importantes foram representados no diagrama de classes, e são o reproduzir, o pausa e o parar – responsáveis pela reprodução ou pausa dos jogos, o escolherFicheiros que é utilizado para seleccionar quais os ficheiros a reproduzir, e o alterarVelocidade, que serve para mudar a velocidade de reprodução. A função das classes usadas é explicada de seguida.

#### **CRobo\_gui, CBola\_gui e CCampo\_gui**

Estas classes desenhavam um robô, uma bola e o campo de jogo à escala, respectivamente. São instanciados cinco objectos da classe CRobo\_gui e CBola\_gui, uma vez que é representada a bola que cada robô observa. Apesar disso, é possível que o utilizador não

escolha nenhum robô para análise (caso opte por analisar as mensagens recebidas da *RefBox* ou a informação processada pela MINHO BaseStation).

#### 4.6.2. REPRODUÇÃO DE JOGOS

Para reproduzir jogos foi utilizado um contador de tempo, implementado através da classe QTimer do Qt. O tempo configurado inicialmente é igual ao tempo de registo da informação no Modo de Jogo (50 milissegundos), mas pode ser alterado pelo utilizador.

Para cada ficheiro escolhido para reprodução é utilizado um analisador XML que devolve a informação contida em cada nó<sup>4</sup>. No caso de ser escolhido mais do que um ficheiro, é possível que tenham informação com o tempo máximo diferente (tempo de registo dos últimos dados no ficheiro). Nessa situação, a reprodução do jogo só é concluída quando a reprodução de todos ficheiros tiver terminado.

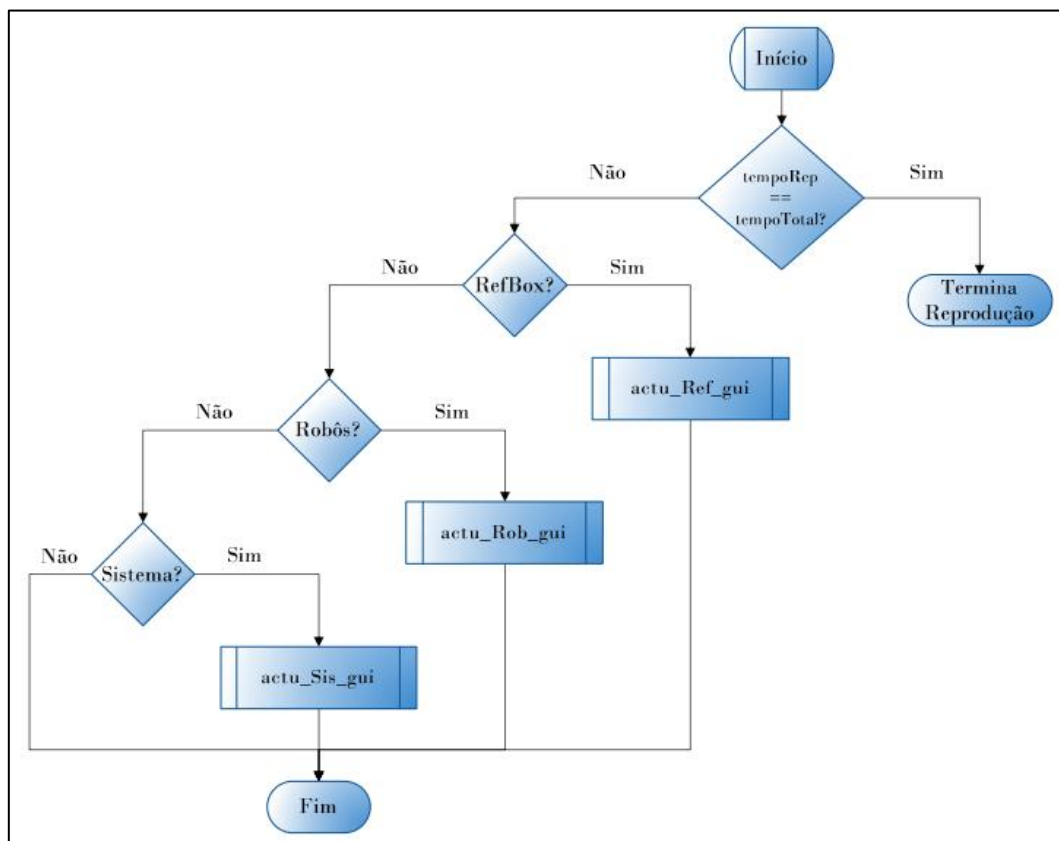


Figura 37 - Fluxograma da reprodução de jogos.

Cada vez que o contador atinge o valor definido, é executado para cada ficheiro o algoritmo representado na Figura 37. É realizada a verificação se o tempo actual de reprodução é igual ao tempo total do ficheiro. Se esse teste for verdade, então a análise do ficheiro está

<sup>4</sup> O nó é uma linha do ficheiro XML, constituído por informação como o tempo de registo e o tipo de dados.

terminada. Se for falso, significa que o ficheiro ainda tem informação para ser analisada. De seguida, é averiguado que tipo de informação contém o ficheiro. Se forem dados sobre a *RefBox*, então o método `actu_Ref_gui` efectua a actualização desta nova informação na interface gráfica. Caso sejam informações sobre os robôs, o `actu_Rob_gui` é responsável por actualizar o campo de jogo e o painel de informações dos robôs com a nova informação. Por último, se o tipo de informação em análise consistir em dados do sistema, o método `actu_Sis_gui` é invocado para exibir no ecrã a informação no local adequado. Terminada a actualização da interface gráfica, é aguardado que o contador recomece a contagem para o processo ser repetido.

## 4.7. CONFIGURAÇÕES

A funcionalidade denominada de Configurações permite ao utilizador introduzir as dimensões do campo de jogo, testar as comunicações com robôs e *RefBox*, enviar ficheiros para os robôs e verificar se os robôs têm os ficheiros necessários para realizarem jogos correctamente.

Está dividido em 4 partes. A primeira é designada de Campo e Agentes, e oferece a possibilidade de alterar as medidas do campo, o diâmetro dos robôs e da bola de jogo. Outra das partes é chamada de Rede e é lá onde são escolhidos todos os endereços IP e portas utilizadas para a comunicação da MINHO BaseStation. Outra das tarefas possíveis de executar na Rede consiste em testar as comunicações realizadas com os robôs e com a *RefBox* usando os vários tipos de protocolos possíveis. A terceira parte serve para gerir os ficheiros dos robôs. Isto é, é necessário que durante os jogos os robôs possuam um determinado conjunto de ficheiros para executarem as suas funções correctamente. Então, o modo Gerir Ficheiros obtém os ficheiros de cada robô e depois de comparar com os necessários elabora um relatório onde indica quais os ficheiros que estão incorrectos – podem não existir, ter o tamanho diferente ou a data e hora diferente. A última parte é conhecida por Transferir Ficheiros, e tem a função de enviar qualquer tipo de ficheiros para os robôs.

A interface gráfica é composta por um painel constituído por separadores, cada um associado a uma funcionalidade. No primeiro separador é desenhado um campo de jogo e são colocadas todas as medidas do campo. No segundo separador é possível testar individualmente a conexão com cada robô, assim como ver quais as mensagens recebidas da *RefBox*. Para a verificação de ficheiros, os ficheiros existentes em cada robô são catalogados em listas e identificados por cores caso estejam incorrectos. No envio de ficheiros, todos os ficheiros a

enviar para os robôs são listados, e depois do comando enviar ser executado são identificados os ficheiros enviados com sucesso e os ficheiros que não foi possível enviar.

#### 4.7.1. ESTRUTURA DAS CONFIGURAÇÕES

A interface gráfica desenvolvida consiste num painel de separadores, que foi implementado recorrendo à classe `QTabWidget` fornecida pelo Qt. Cada um dos separadores é preenchido de acordo com a sua funcionalidade. A título de exemplo, ao primeiro separador designado de Campo e Agentes é associado um objecto da classe `CCampo_gui`, que desenha um campo à escala. A comunicação realizada com os robôs e com a *RefBox* implementa um *socket* UDP e um *socket* TCP fornecido pelas classes do Qt `QUdpSocket` e `QTcpSocket`, respectivamente. A recepção de mensagens é efectuada executando os métodos `recDadosRefBox` para as mensagens da *RefBox*, e `recDadosRobos` para mensagens dos robôs. O diagrama de classes com os atributos e métodos mais relevantes do módulo Configurações pode ser consultado na Figura 38.

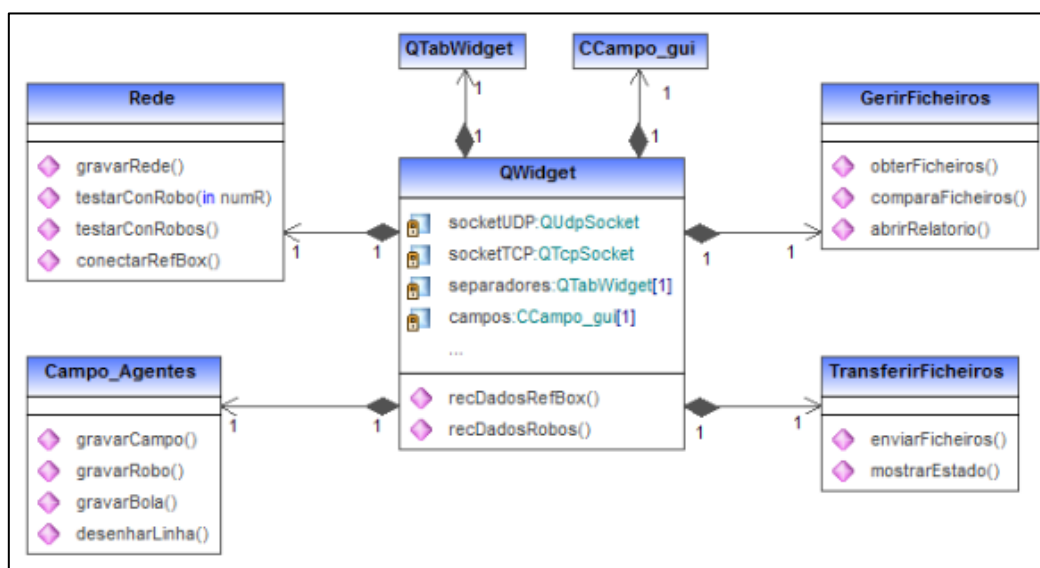


Figura 38 - Diagrama de classes das Configurações.

A descrição das classes constituintes das Configurações traduz-se em:

#### QTabWidget

Esta é uma classe do Qt e fornece um conjunto de painéis, implementado cada um deles num separador. É a classe que gere a comutação entre separadores e a exibição do painel correspondente ao separador escolhido.

### **CCampo\_gui**

A classe CCampo\_gui é utilizada para desenhar um campo de jogo. Todas as linhas são desenhadas à escala de acordo com as medidas reais.

### **Campo\_Agentes**

No Campo\_Agentes existem todos os atributos e métodos necessários para a alteração das dimensões do campo e dos agentes, como os métodos gravarCampo, gravaRobo, gravarBola ou desenharLinha.

### **Rede**

Esta classe fornece todos os recursos necessários para alterar os endereços IP e as portas para a comunicação, e ainda testar todos os protocolos de comunicação implementados.

### **GerirFicheiros**

O GerirFicheiros é a classe responsável por obter e comparar os ficheiros existentes nos robôs, necessários durante a realização de jogos.

### **TransferirFicheiros**

Esta classe é responsável por enviar para os robôs os ficheiros escolhidos pelo utilizador, e indicar se foi possível transferir os ficheiros ou se ocorrem erros na transferência.

A seguir será explicado detalhadamente cada um dos separadores que constituem o módulo Configurações.

## **4.7.2. CAMPO E AGENTES**

O separador Campo e Agentes é usado para modificar o valor das dimensões do campo, do diâmetro dos robôs e da bola. É desenhado um campo à escala de acordo com as medidas representadas.

Sempre que uma medida é seleccionada para alteração do seu valor, é desenhado no campo uma linha vermelha que indica ao utilizador o que corresponde essa medida na realidade. É ainda efectuado um zoom do campo, relativamente à posição do rato, para o utilizador observar melhor onde começa e termina cada medida.



Quando o utilizador decida gravar as novas medidas, os métodos `gravarCampo`, `gravarRobo` e `gravarBola` são executados mediante as medidas alteradas. Estes métodos têm como objectivo actualizar o valor dos atributos das classes `CCampo`, `CRobo` e `CBola` (apresentadas no capítulo 4.2), registar no ficheiro correspondente os novos valores e redesenhar o campo da interface gráfica de acordo com as novas dimensões escolhidas.

### **4.7.3. REDE**

É no separador Rede que são tratadas todas as questões relacionadas com a comunicação, nomeadamente a alteração dos endereços IP e das portas usadas para comunicar, bem como o teste das conexões com os robôs e com a *RefBox*.

Os endereços IP e as portas usadas na comunicação com os robôs por *unicast* e *multicast*, bem como o endereço IP e as portas da *RefBox* para os vários tipos de protocolos podem ser definidos no separador Rede. Quando o utilizador decide gravar as alterações, os atributos da classe `CRede` (secção 4.2) são actualizados, e os novos valores são registados no ficheiro correspondente.

O teste da comunicação com os robôs pode ser realizado individualmente ou globalmente. Para cada robô é enviado uma mensagem que, caso o robô a receba ele envia uma mensagem de volta para a MINHO BaseStation. Quanto à *RefBox*, é possível escolher qual dos protocolos é pretendido testar. As opções consistem no protocolo2009 usando o protocolo de rede TCP/IP, ou o protocolo2010 com os protocolos de rede TCP/IP ou UDP. Escolhido um protocolo e estabelecendo uma conexão, é visível num painel o conteúdo das mensagens enviadas pela *RefBox*.

### **4.7.4. VERIFICAR FICHEIROS**

A verificação dos ficheiros existentes em cada robô consiste na função deste separador. Para isso, é realizado um pedido de ficheiros aos robôs. Recebidos os ficheiros, são comparados e é elaborado um relatório que indica quais os ficheiros em falta.

Para o pedido de ficheiros, é enviada uma mensagem aos robôs. Depois, cada robô usando a classe `QProcess` do Qt executa um script que devolve uma lista com o nome, tamanho, data e hora de todos os ficheiros existentes. Essa lista é posteriormente enviada pelos robôs para a MINHO BaseStation.

Quando a lista de ficheiros é recebida, todos os ficheiros são listados no painel correspondente a cada robô. Os ficheiros de cada robô são comparados com os ficheiros existentes numa pasta local à aplicação MINHO BaseStation. No caso de os robôs não possuírem os mesmos ficheiros que a referida pasta, é assinalado na interface gráfico o robô e são criados erros. A comparação de ficheiros é feita inicialmente usando o nome e o tamanho dos ficheiros, mas pode ser adicionada a comparação por data e hora da última modificação.

Cada erro que ocorre é guardado, e no fim é gerado um relatório com todos os erros que ocorreram. É especificado o ficheiro, o motivo da ocorrência e o robô que possui o ficheiro.

### **4.7.5. TRANSFERIR FICHEIROS**

O separador Transferir Ficheiros tem como função enviar para os robôs qualquer tipo de ficheiros (ficheiros com parâmetros, código para executar no robô, máscaras da cabeça do robô, entre outros).

Os ficheiros a ser enviados são colocados numa pasta para esse fim, que se encontra no mesmo computador que a aplicação MINHO BaseStation. Quando os ficheiros são colocados na pasta, eles são listados automaticamente na interface gráfica.

Sempre que o utilizador decide enviar os ficheiros para os robôs, é enviado inicialmente o nome e o tamanho de cada ficheiro. Só é enviado o próximo ficheiro depois de recebida a confirmação do robô em como recebeu o ficheiro anterior.

No caso de o ficheiro recebido pelo robô ter um tamanho diferente do enviado inicialmente, o robô comunica essa situação à MINHO BaseStation. Na interface gráfica essa situação é registada, identificando o ficheiro como enviado irregularmente.

### **4.7.6. COMUNICAÇÃO**

A Figura 39 ilustra a comunicação realizada em Configurações, designadamente ente a *RefBox*, a MINHO BaseStation e os robôs.

O separador de Rede executa ligações à *RefBox*, onde ambos os protocolos podem ser seleccionados e testados. A comunicação com os robôs também é testada, com a MINHO BaseStation a enviar uma mensagem ACK e os robôs a reenviarem a mensagem de volta.

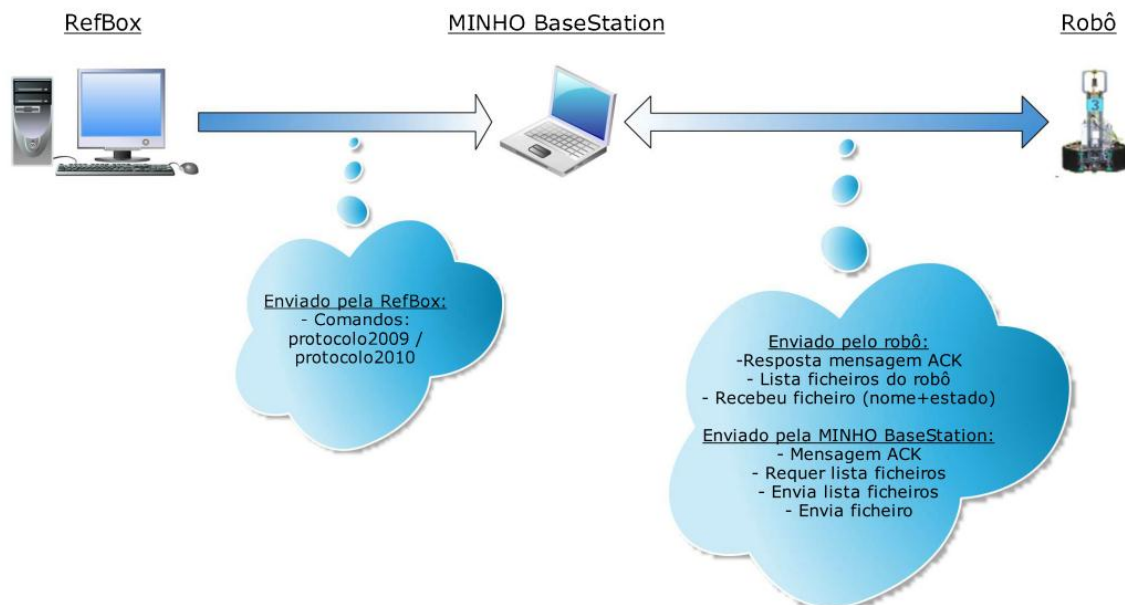


Figura 39 - Comunicação efectuada em Configurações.

No separador Verificar Ficheiros, os robôs sempre que solicitados enviam à MINHO BaseStation a lista com o nome, tamanho, data e hora dos ficheiros existentes nas suas pastas.

Na transferência de ficheiros, é enviado para os robôs o nome e o tamanho dos ficheiros a transferir, e cada um dos ficheiros individualmente. Os robôs, sempre que recebem um ficheiro, informam a MINHO BaseStation da recepção e da integridade dos dados do ficheiro.

## 4.8. PREFERÊNCIAS

A funcionalidade Preferências foi implementada para o utilizador escolher um conjunto de opções directamente relacionadas com o funcionamento da MINHO BaseStation.

É possível escolher quais os ficheiros a usar – nomeadamente ficheiros com as dimensões do campo, com as configurações de rede e com as táticas, escolher que tipo de protocolo utilizar para comunicar com a *RefBox* e definir qual a cor da equipa MINHO TEAM durante os jogos. Existe ainda a opção de serem criados novos ficheiros para qualquer um dos tipos anteriormente identificados.

É apresentada ao utilizador sobre a forma de uma interface gráfica, onde para cada tipo de ficheiros (Campo e Agentes, Rede e Táticas) é associada uma lista que apresenta todos os ficheiros existentes. Os ficheiros usados actualmente pela MINHO BaseStation são seleccionados automaticamente, sendo que para o utilizador trocar qualquer um deles apenas necessita de clicar no nome do ficheiro pretendido. O mesmo sucede com a escolha do tipo de comunicação

com a *RefBox* e a cor da equipa, em que para mudar de opção apenas é necessário clicar na nova escolha.

#### 4.8.1. ESTRUTURA DAS PREFERÊNCIAS

O diagrama de classes do modo Preferência encontra-se na Figura 40. Para cumprir as suas funcionalidades, foram implementados os métodos que permitem alterar e criar ficheiros, e modificar as preferências para o jogo.

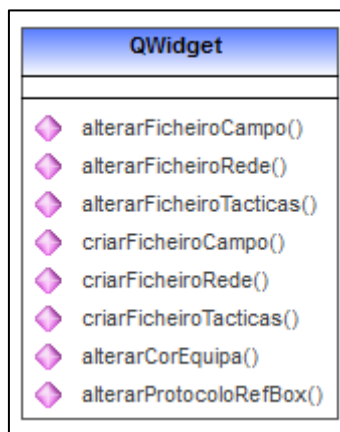


Figura 40 - Diagrama de classes das Preferências.

As opções apresentadas em Preferências são as seguintes:

- Campo: Todas as dimensões do campo de jogo, diâmetro do robô e da bola são guardadas num ficheiro XML. As dimensões admitidas estão estabelecidas nas regras da competição, contudo nem todos os campos de jogo as cumprem – como por exemplo o campo usado pela equipa MINHO TEAM. Então, para cada campo pode ser criado um ficheiro, e acedendo ao separador Campo e Agentes em Configurações, as dimensões podem ser modificadas. Todos os ficheiros com dimensões de campos são listados na interface gráfica, e a troca de ficheiros é realizada clicando no nome do ficheiro pretendido.
- Rede: As variáveis para a comunicação de rede, como endereços IP e portas usadas, são guardadas num ficheiro XML. A existência de vários ficheiros de rede é necessária, porque durante um jogo é imposto o uso dos endereços IP e do router destacados pela organização, mas no laboratório da equipa MINHO TEAM as configurações de rede usadas são totalmente diferentes.

- Tácticas: Também é possível existirem vários ficheiros de tácticas. Uma vez que as tácticas são criadas antes da realização de jogos, podem ser criadas várias abordagens, e depois seleccionada a que mais se adequa ao adversário.
- Protocolo de comunicação com a *RefBox*: Como já foi referido no capítulo 4.3.1, a *RefBox* disponibiliza às equipas dois tipos de protocolos: o protocolo2009 e o protocolo2010. O protocolo2009 é utilizado juntamente com o protocolo de comunicação TCP/IP, enquanto o protocolo2010 fornece a opção TCP/IP e UDP. A escolha do protocolo a usar para a ligação à *RefBox* é realizada seleccionando o nome do protocolo pretendido na lista de protocolos implementada na interface gráfica.
- Cor da equipa: Permite alterar a cor escolhida para a equipa durante os jogos – magenta ou ciano.

## 5. RESULTADOS

Neste capítulo apresentam-se as interfaces, as funcionalidades e os problemas da aplicação MINHO BaseStation.

Sempre que a MINHO BaseStation é executada é exibido o menu, apresentado na figura abaixo.

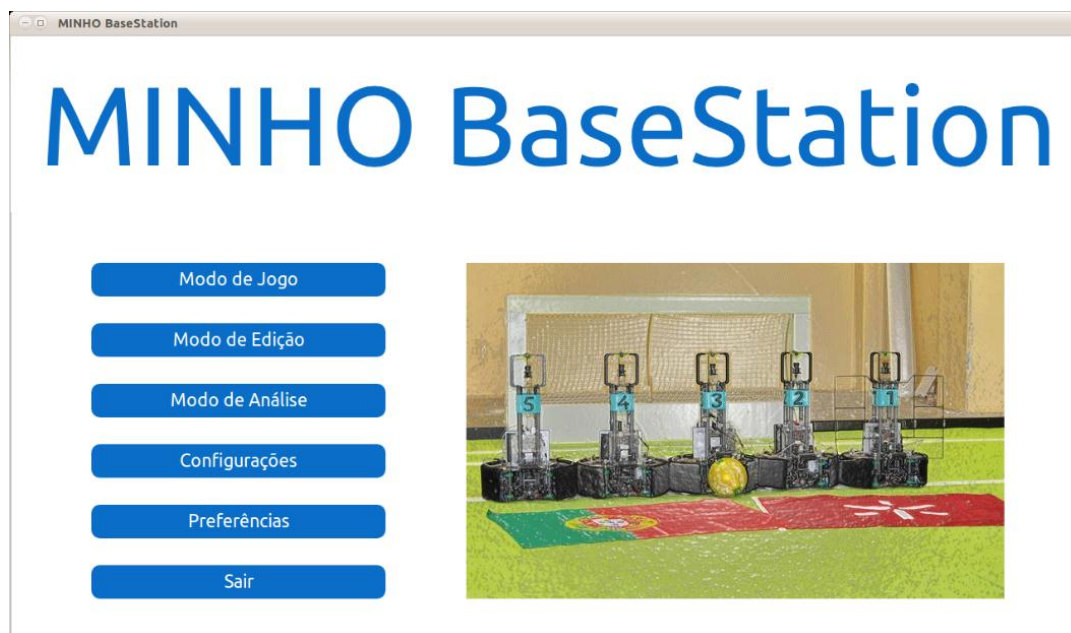


Figura 41 - Menu da MINHO BaseStation.

O menu apresenta ao utilizador a hipótese de escolher uma das cinco opções principais (Modo de Jogo, Modo de Edição, Modo de Análise, Configurações ou Preferências), sendo que só pode ser seleccionada uma de cada vez. O botão Sair, quando premido, termina a execução do programa. A aplicação necessita de ser acompanhada pela pasta de ficheiros configs, que contém as dimensões dos campos de jogo, as configurações de rede, as táticas ou os *logs* gravados. A sua ausência impede o programa de funcionar correctamente.

Nas próximas secções é apresentado cada um dos modos da MINHO BaseStation.

### 5.1. MODO DE JOGO

O Modo de Jogo é usado para a realização de jogos da equipa MINHO TEAM, oficiais ou de testes. Nos jogos oficiais, a aplicação executa autonomamente uma vez que não é permitido nenhum tipo de interferência humana.

Sempre que o Modo de Jogo é escolhido, a interface gráfica mostrada ao utilizador consiste na imagem da Figura abaixo.



Figura 42 - Interface gráfico do Modo de Jogo.

A partir do momento em que é iniciado, o Modo de Jogo encontra-se preparado para realizar jogos. A única exceção ocorre quando o protocolo de comunicação escolhido para a conexão com a *RefBox* consiste no TCP/IP, em que é necessário estabelecer a conexão com a *RefBox* antes do início dos jogos.

Quanto à interface gráfica, a sua estrutura pode ser dividida em quatro partes. Uma das partes consiste em exibir as mensagens recebidas da *RefBox*, o tempo e resultado do jogo, o nome da equipa adversária, as opções escolhidas (cor da equipa, ficheiro de campo, rede e tácticas) e o estado do PC. A informação descrita anteriormente é visível na próxima Figura.



Figura 43 - Histórico da *RefBox*, tempo e resultado de jogo.

O campo de jogo constitui outra das partes, onde robôs e bolas são desenhados de acordo com as suas posições reais em campo. As dimensões do campo, do robô e da bola encontram-se representadas à escala. Cada robô é identificado com um número e com uma cor diferente. As bolas são pintadas usando o mesmo esquema de cores que os robôs, o que torna mais fácil a análise das posições das bolas e dos robôs. Na parte inferior do campo é desenhada uma seta que indica qual a direcção de ataque da equipa MINHO TEAM.

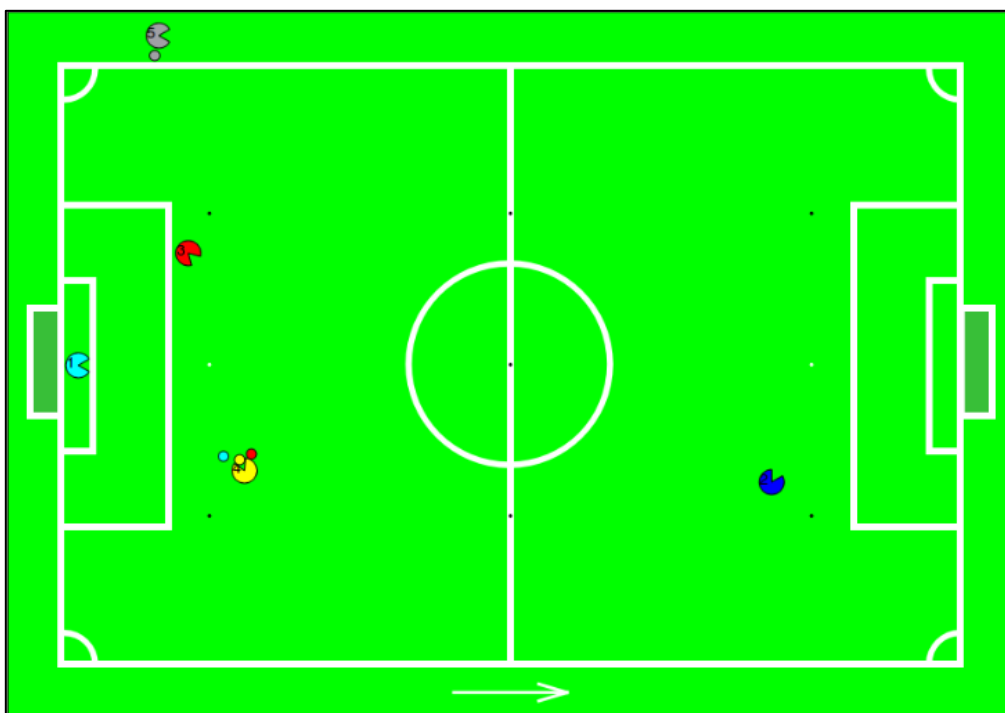


Figura 44 - Campo de jogo.

O painel com informações sobre cada robô representa a terceira parte do Modo de Jogo. Na Figura 45 é ilustrado o painel de um dos robôs. É representado do lado esquerdo a capacidade actual das baterias, sendo que a cor varia entre o verde e o vermelho consoante o valor da capacidade. É igualmente exibido o valor da bússola, se os infravermelhos (da frente e de trás) detectam a bola e se os motores se encontram alimentados.

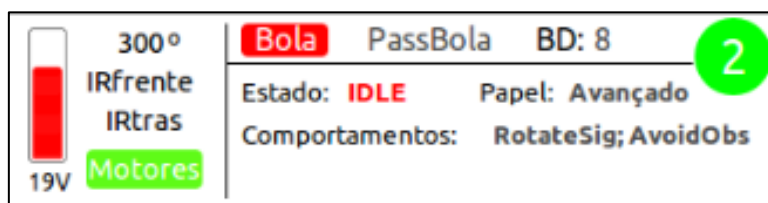


Figura 45 - Painel informativo de um robô.

No lado direito é mostrado o estado, o papel e o comportamento do robô, bem como se o robô se encontra a comunicar ou detecta a bola.



A última parte é composta pela barra de menu, e tem como objectivo abrir as interfaces gráficas opcionais (como ver as táticas ou os papéis atribuídos), trocar a direcção de ataque da equipa MINHO TEAM ou sair do Modo de Jogo.

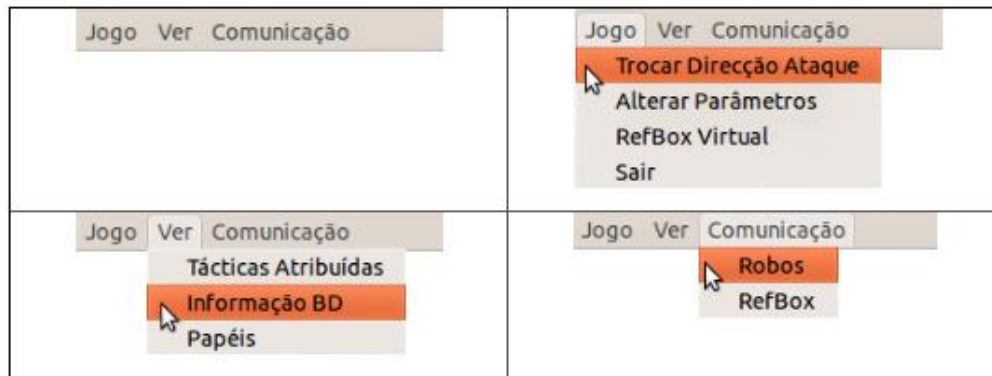


Figura 46 - Barra de menu detalhada.

Relativamente às opções da barra de menu, cada uma delas (excepto a trocar direcção de ataque) tem um interface gráfico associado, e encontra-se descrito nos próximos parágrafos.

- Alterar Parâmetros

A alteração dos parâmetros do jogo é efectuada na interface gráfica seguinte:

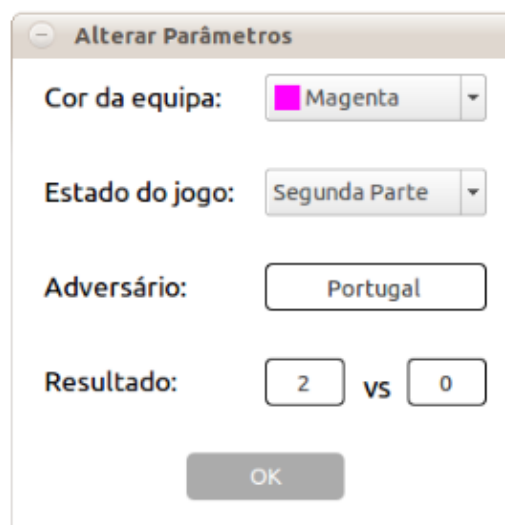


Figura 47 - Alterar parâmetros de jogo.

É possível o utilizador escolher a cor da equipa MINHO TEAM (magenta ou ciano), o período em que o jogo se encontra (antes do jogo, primeira parte, intervalo, segunda parte ou fim do jogo), o nome do adversário e o resultado do jogo.

- RefBox Virtual

A *RefBox* virtual é uma ferramenta bastante útil na realização de testes, pois fornece os mesmos tipos de eventos que a *RefBox* oficial. Foi desenvolvida com a particularidade de permitir o seu controlo à distância através de um comando da Wii Mote.

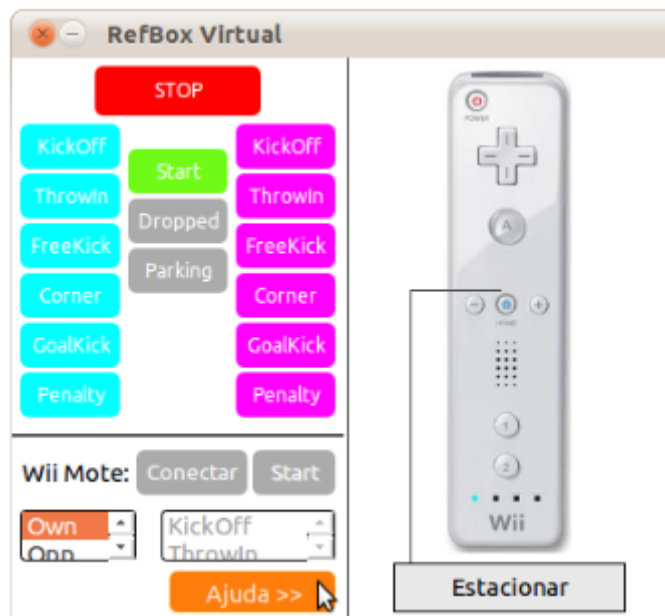


Figura 48 - *RefBox* virtual.

Para a utilização do comando da Wii foi criada uma secção de ajuda que explica como operar a *RefBox* virtual através do comando. Na Figura 48, é exibida uma imagem da *RefBox* virtual, onde a secção de ajuda indica que o botão HOME corresponde ao evento Estacionar.

- Ver Tácticas Atribuídas

A interface gráfica exibida para esta opção pode ser consultada na Figura abaixo.

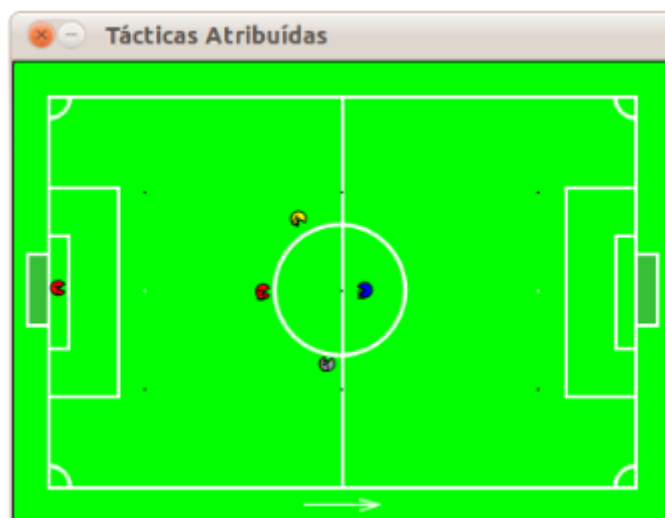
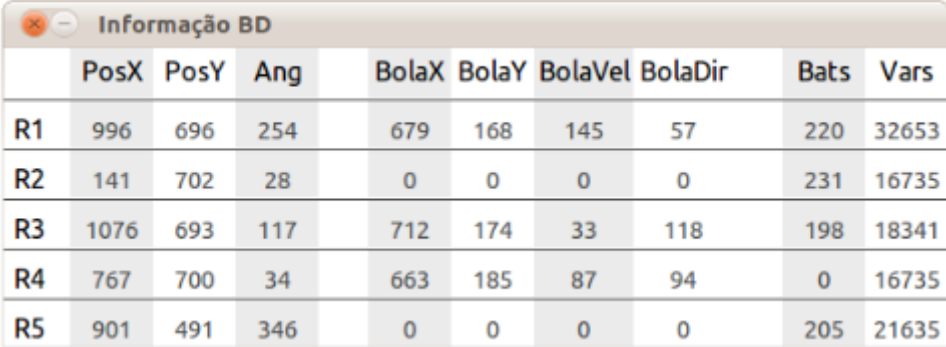


Figura 49 - Ver tácticas atribuídas.

Sempre que a MINHO BaseStation calcula e atribui posições aos robôs, esta janela desenha os robôs nessas posições. Assim, é possível determinar se o cálculo das posições está a ser correctamente realizado, e se os robôs estão a ir para o local pretendido.

- Ver Informação BD

Aqui é exibida a informação armazenada na base dados – correspondente à informação mais actual de cada robô, mas no formato de texto. É mostrada a posição e orientação dos robôs, a posição e a velocidade que cada robô observa a bola, bem como o nível de tensão das baterias e o valor das variáveis dos robôs.

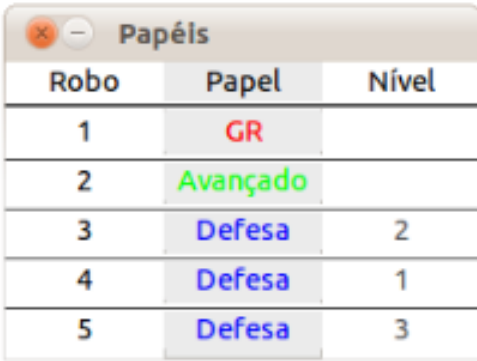


	PosX	PosY	Ang	BolaX	BolaY	BolaVel	BolaDir	Bats	Vars
R1	996	696	254	679	168	145	57	220	32653
R2	141	702	28	0	0	0	0	231	16735
R3	1076	693	117	712	174	33	118	198	18341
R4	767	700	34	663	185	87	94	0	16735
R5	901	491	346	0	0	0	0	205	21635

Figura 50 - Ver informação da base de dados.

- Ver Papéis

Caso seja pretendido, pode ser apresentado no ecrã qual o papel atribuído a cada robô. Para isso, basta abrir a interface gráfica da Figura 51, que mostra qual o papel e o nível (no caso dos defesas) conferido aos robôs. Com isto, é possível analisar se a atribuição de papéis pela MINHO BaseStation está a decorrer correctamente, e se os robôs estão a desempenhar o papel correcto.



Robo	Papel	Nível
1	GR	
2	Avançado	
3	Defesa	2
4	Defesa	1
5	Defesa	3

Figura 51 - Ver papéis atribuídos.

- Conexão com os robôs

Esta funcionalidade foi implementada para ser utilizada durante a realização de jogos de teste. Exibe qual o endereço IP de cada robô, e de forma simples permite a alteração da ordem dos robôs ou colocar algum como offline. É extremamente útil na realização de testes, pois a atribuição de papéis é realizada de forma dinâmica do robô 1 para o robô 5, e os papéis são atribuídos de acordo com a sua importância (por exemplo, o guarda-redes é sempre o primeiro papel a ser atribuído).



Figura 52 - Configurar conexão com os robôs.

- Conexão com a RefBox

No caso de o protocolo de comunicação com a *RefBox* consistir no TCP/IP, é necessário estabelecer uma conexão com a *RefBox* antes do início dos jogos. Para esse fim, foi criada a interface gráfica representada na Figura 53, onde além de permitir a desconexão da *RefBox* informa se a MINHO BaseStation se encontra conectada ou desconectada.

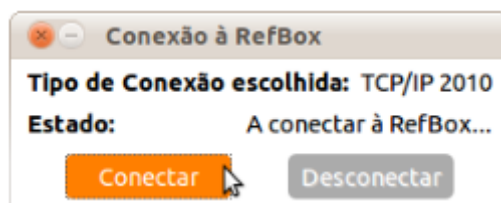


Figura 53 - Configurar conexão com a *RefBox*.

### 5.1.1. CÁLCULO DA POSIÇÃO DA BOLA

O cálculo de uma única posição da bola é realizado utilizando os valores de cada robô. Inicialmente, a fórmula empregada no cálculo consistiu na média aritmética simples, mas foi substituída pela média aritmética ponderada. O motivo consistiu no facto de o sistema de visão do robô perder resolução à medida que a distância aumenta, e assim será atribuído um maior peso à medida do robô mais próximo da bola. O facto de por vezes a localização do robô estar errada é outro dos motivos, e com o uso da média aritmética ponderada o valor calculado é mais próximo do valor real.

A próxima Figura apresenta as posições reais dos robôs e a posição real da bola (sinalizada de cor vermelha).

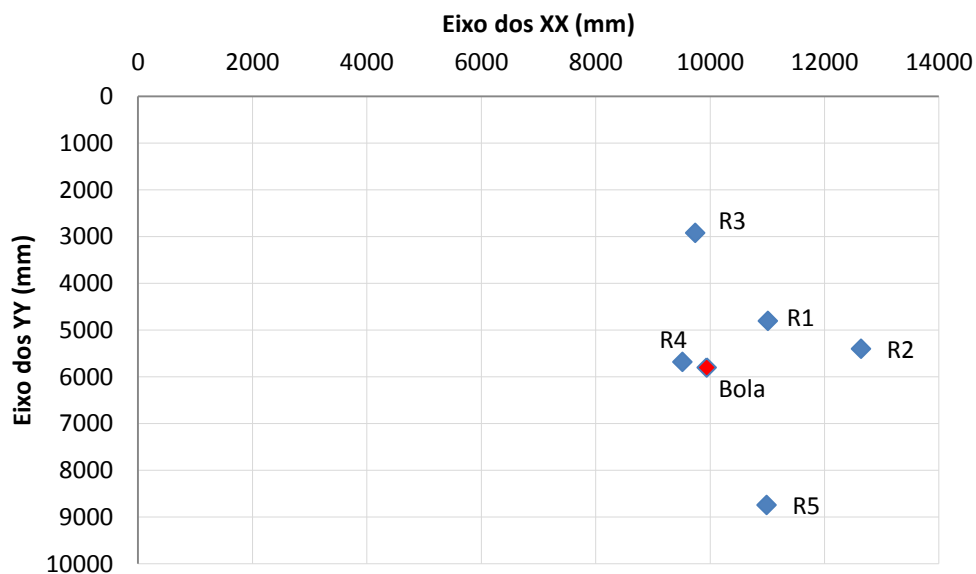


Figura 54 - Posições reais dos robôs e da bola.

Cada robô, posicionado nas coordenadas da Figura 54, observou a bola nas posições indicadas na Figura abaixo. De cor vermelha, é indicada a posição real da bola. O ponto de cor verde corresponde ao cálculo da média aritmética normal, e o amarelo à média aritmética ponderada.

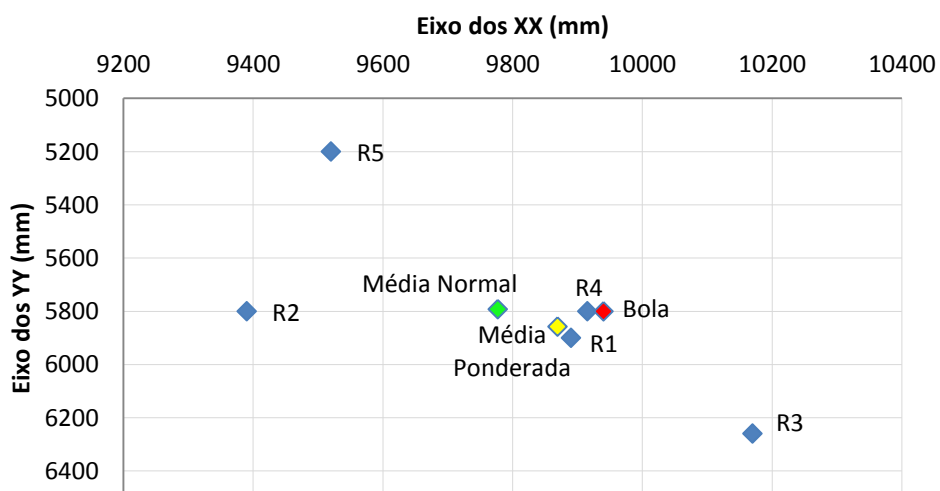


Figura 55 - Observação e cálculo da posição da bola.

A próxima Figura, à semelhança da Figura anterior, também indica qual a posição que cada robô observou a bola. A diferença reside no robô 5, que nesta situação encontra-se com o

valor da localização errado, e dessa forma a posição calculada para a bola (R5) também está errada.

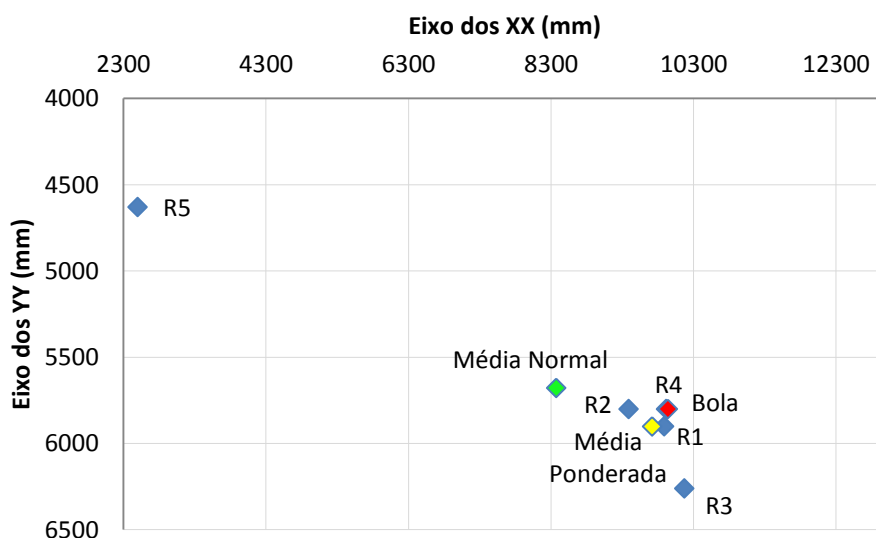


Figura 56 - Observação e cálculo da posição da bola com um robô não localizado.

Analisando as Figura 55 e Figura 56, é possível concluir que a média aritmética ponderada apresenta melhores resultados para o cálculo da posição global da bola que a média aritmética simples, e em situações que os robôs podem não se encontram localizados correctamente.

### 5.1.2. ATRIBUIÇÃO DE PAPÉIS

A atribuição de papéis aos robôs é realizada pela MINHO BaseStation em três situações. A primeira verifica-se sempre que a *RefBox* envia comandos de jogo (como pontapé de baliza, lançamento ou falta). A segunda situação ocorre após os robôs passarem a bola, na marcação de um lance de bola parada a favor da equipa MINHO TEAM. Os robôs com os papéis de marcador e receptor recebem novos papéis, nomeadamente avançado e defesa. A última situação acontece durante o jogo, sempre que algum robô se encontra mais perto da bola que o avançado. Nessa situação, os papéis dos dois robôs envolvidos são trocados.

Quanto aos papéis, o guarda-redes é sempre o primeiro a ser atribuído e nunca a mais do que um robô. O marcador e receptor são utilizados na reposição de lances de bola parada, e o avançado apenas pode ser conferido a um robô de cada vez.

A próxima Figura mostra os papéis atribuídos aos robôs no início de um jogo.

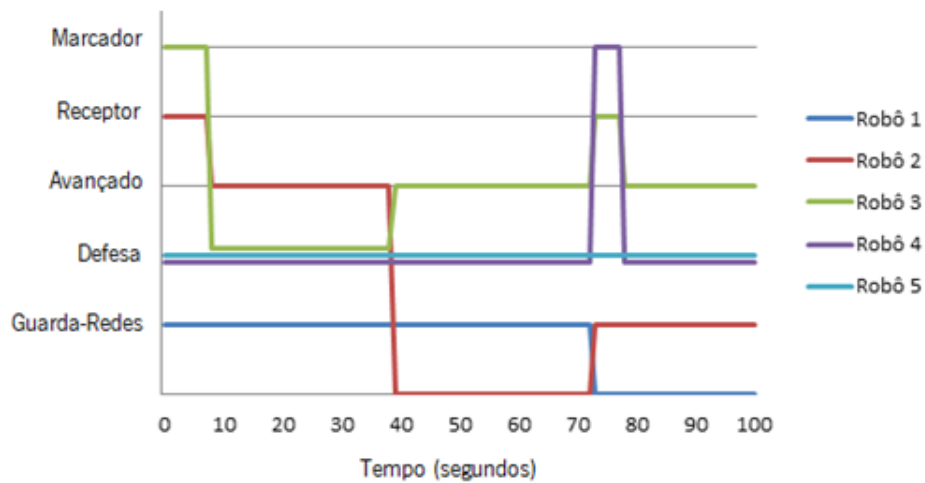
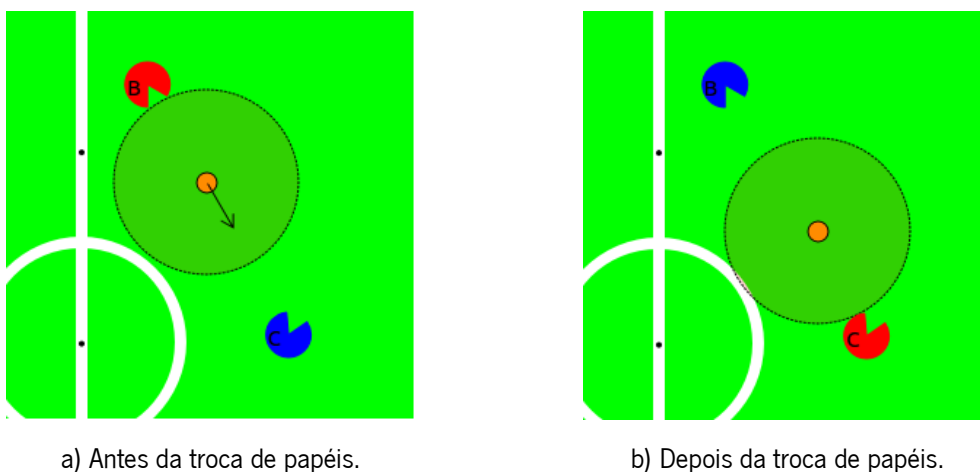


Figura 57 - Atribuição de papéis durante um jogo.

Foi assinalado um pontapé de saída para a equipa MINHO TEAM, e foram atribuídos papéis aos cinco robôs. Depois de ser efectuado o passe da bola (cerca de 9 segundos), a equipa jogava com 3 Defesas e um Avançado. Aos 38 segundos, foi marcado um lançamento para a equipa adversária. O robô 2 foi removido do campo, tendo o robô 3 actuado como Avançado. Por volta dos 72 segundos, o árbitro assinalou uma falta. Nesse momento, o robô 1 foi retirado de campo, e o robô 2 foi recolocado em campo. O Guarda-redes passou a ser o robô 2, os robôs 4 e 5 estavam a jogar como Defesas e o robô 3 como Avançado.

Analisando o gráfico da Figura 57, é possível observar que o papel de Guarda-redes foi atribuído apenas a um robô de cada vez. O papel de Avançado também foi desempenhado apenas por um robô ao mesmo tempo, atribuído directamente pela MINHO BaseStation ou atribuído ao Receptor depois de a bola ser passada pelo Marcador.



a) Antes da troca de papéis.

b) Depois da troca de papéis.

Figura 58 - Troca de papéis.

Quanto à troca do papel de Avançado durante o jogo, a Figura anterior descreve como esse processo é realizado. Na imagem da esquerda, o robô B é o Avançado (vermelho), e o robô C é um dos Defesas (azul). A bola encontra-se em movimento – na direcção do robô C, e quando o robô C fica mais próximo da bola passa a ser o Avançado, e o robô B passa a Defesa.

## 5.2. MODO DE EDIÇÃO

O Modo de edição fornece opções como controlar o robô manualmente, editar lances de bolas paradas, editar os papéis para os jogos, auxiliar o desenvolvimento do algoritmo da localização, alterar os parâmetros do jogo e configurar os parâmetros dos robôs. De seguida é efectuada a descrição de cada uma das funcionalidades.

### 5.2.1. CONTROLO MANUAL DO ROBÔ

O separador que implementa o controlo manual do robô é constituído por um campo de jogo onde os robôs são desenhados de acordo com as suas posições reais. Do lado direito, existe um conjunto de painéis – um para cada robô, onde são exibidas algumas informações. Para controlar o robô, um grupo de botões constituem cada painel, onde é possível chutar, ligar / desligar o drible ou accionar o mecanismo de chuto. Além disso, a atribuição manual de papéis pode ser efectuada, sendo que o papel designado de IrPara permite que, ao arrastar com o rato um robô para uma posição, o robô execute esse movimento.

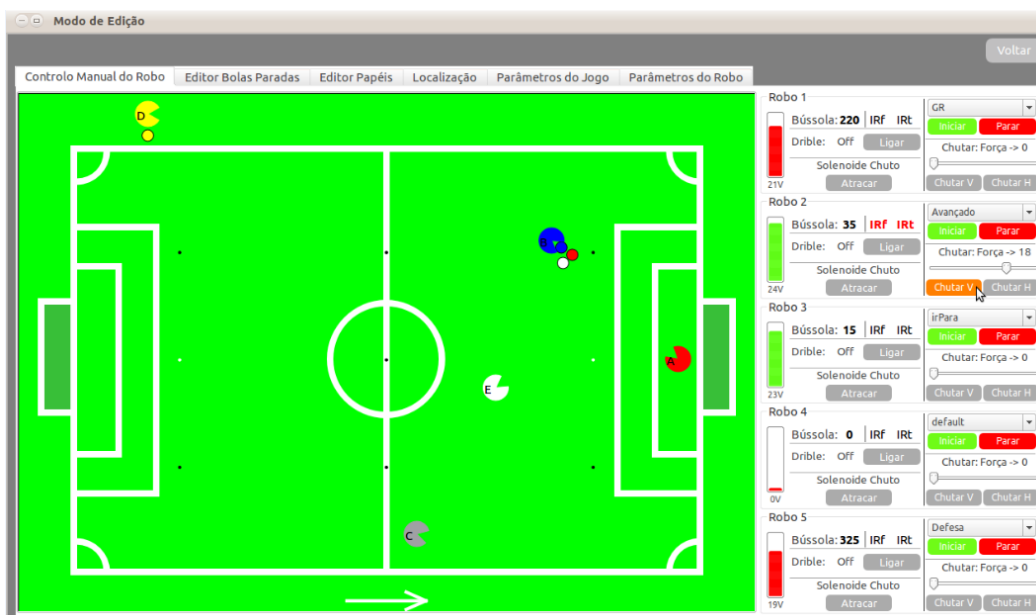


Figura 59 - Controlo manual do robô.



Uma imagem de um dos painéis laterais atribuídos a cada robô pode ser consultada na Figura 60. De uma forma fácil e rápida, é possível monitorizar o estado do hardware do robô - como a capacidade das baterias, o valor da bússola ou o estado dos infravermelhos (na Figura 60 encontram-se a detectar a bola). Quanto ao controlo, cada painel apresenta botões para iniciar ou parar a execução dos papéis, para regular a força do chute e chutar horizontal ou vertical, testar o funcionamento do drible e do solenóide de chute.

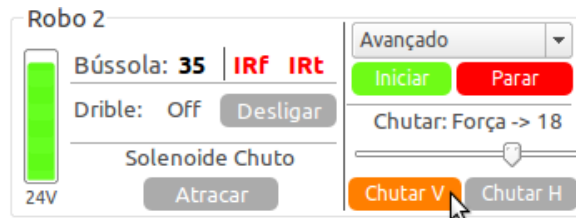


Figura 60 - Painel para controlar manualmente cada robô.

### 5.2.2. EDITOR DE BOLAS PARADAS

Os lances de bola parada são criados, editados ou eliminados neste separador - Figura 61. Todas as tácticas já definidas são listadas de acordo com o tipo de situação escolhido. É possível alinhar os robôs com a bola, e desenhar um círculo à volta da bola com raio variável (1 metro, 2 metros ou 3 metros). O desenho do círculo é importante para o cumprimento das regras na colocação dos robôs, pois consoante o tipo de situação os robôs necessitam de estar afastados da bola uma determinada distância para não serem admoestados pelo árbitro.

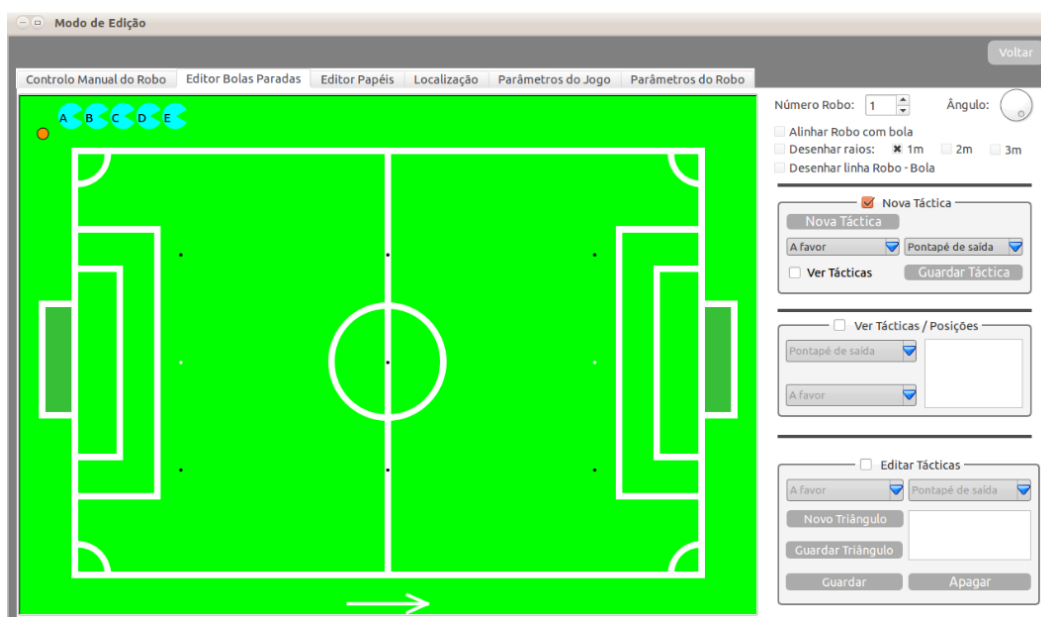


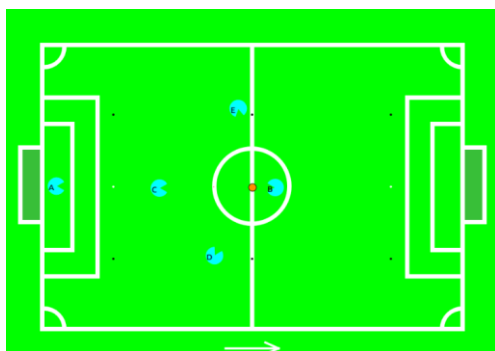
Figura 61 - Editor de bolas paradas.

Antes de as posições serem atribuídas aos robôs são verificadas se cumprem as distâncias impostas pelas regras, e reajustadas caso necessário. Para os lances a favor da equipa MINHO TEAM, o robô que passa e o robô que recebe são alinhados com a bola.

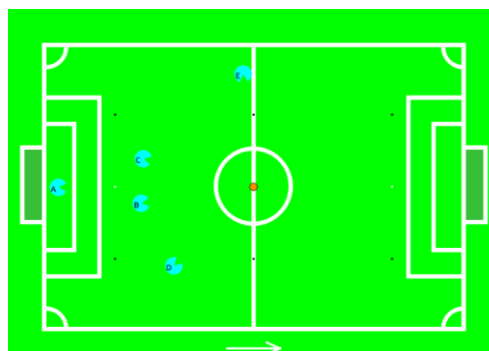
As situações de pontapé de saída, pontapé de saída, canto, penálti e estacionar são definidas de forma semelhante, uma vez que a posição do campo para reposição da bola em jogo é limitado.

A posição dos robôs para cada lance descrito anteriormente é apresentada nas próximas Figuras.

### **Pontapé de saída**



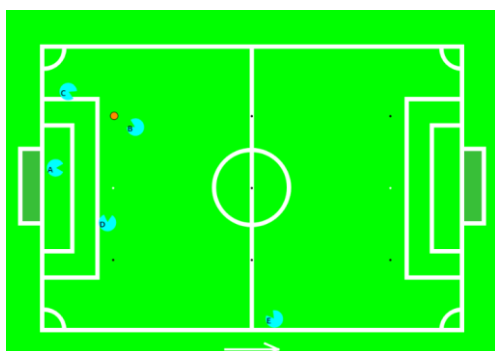
a) A favor da equipa.



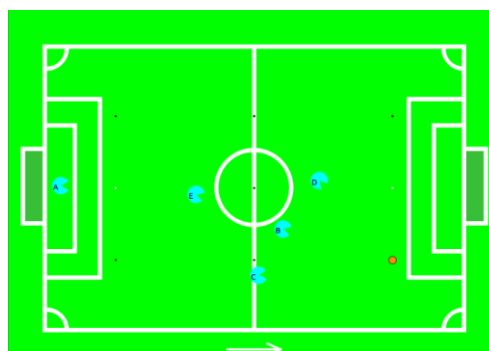
b) Contra a equipa.

Figura 62 - Pontapé de saída.

### **Pontapé de baliza**



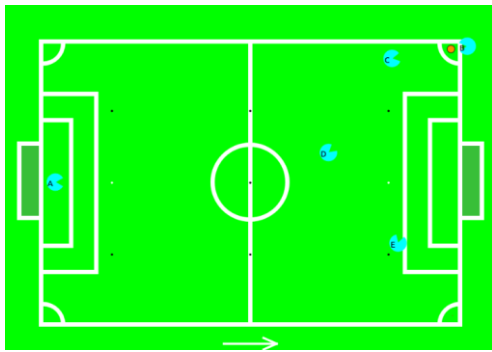
a) A favor da equipa.



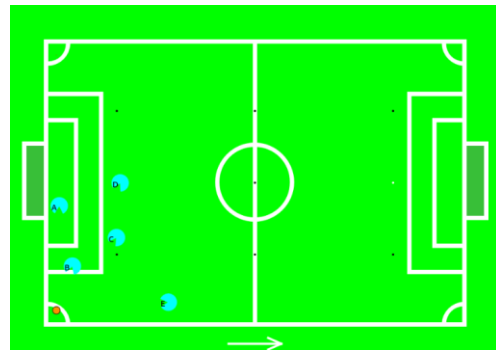
b) Contra a equipa.

Figura 63 - Pontapé de baliza.

**Canto**



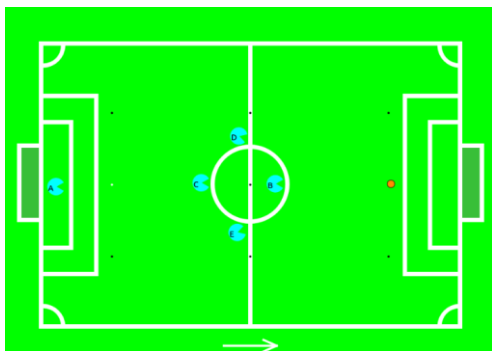
a) A favor da equipa.



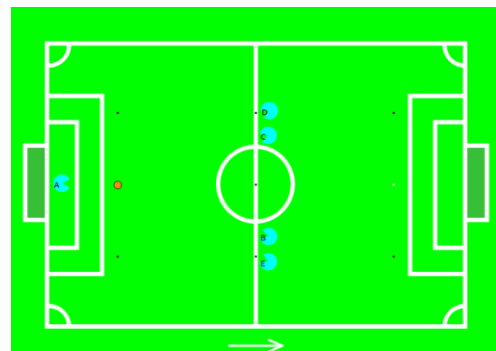
b) Contra a equipa.

Figura 64 - Marcação de cantos.

**Penálti**



a) A favor da equipa.



b) Contra a equipa.

Figura 65 - Penálti.

**Estacionar**

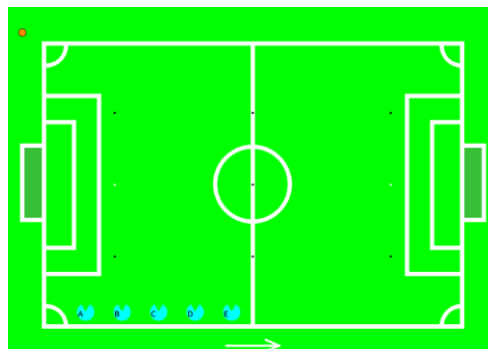


Figura 66 - Estacionar.

### **Lançamento**

Nos lançamentos o processo realizado é diferente, uma vez que a bola pode ser colocada ao longo das linhas laterais do campo. Para a definição dos lançamentos, são gravadas posições para a bola e robôs ao longo das linhas laterais. Quando ocorrer um lançamento, as posições atribuídas os robôs são calculadas efectuando uma interpolação linear. A próxima Figura exhibe a definição de 3 pontos para a bola – a), b) e c), e a posição resultante atribuída aos robôs na ocorrência de um lançamento.

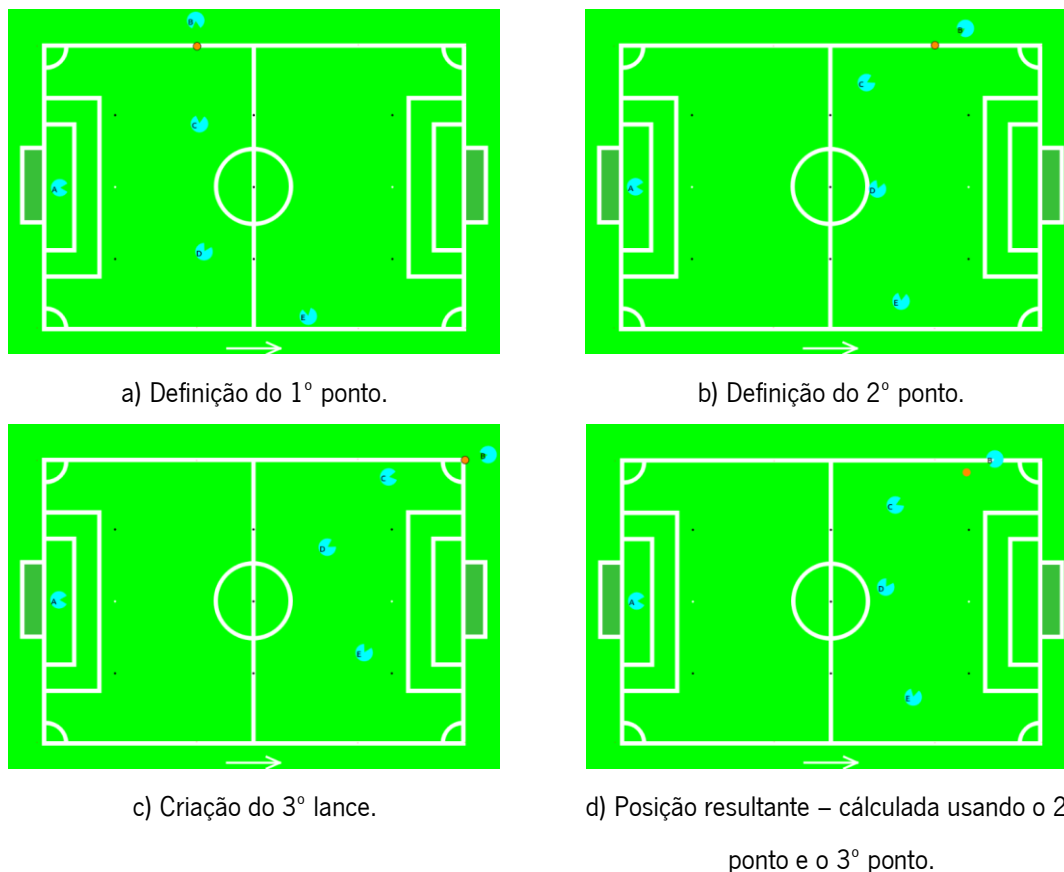


Figura 67 - Definição de lançamentos.

### **Faltas e bola ao solo**

Quanto às faltas e aos lances de bola ao solo, uma vez que podem ser assinalados em todo o campo o seu processo de definição é diferente das situações anteriores. Assim, são definidas as posições desejadas para os robôs em pontos aleatórios do campo, e sempre que o conjunto de pontos definidos ultrapassa os 2 são criados triângulos. A Figura 68 ilustra a formação de um triângulo (definindo 3 pontos), e a posição resultante para os robôs na marcação de uma falta.

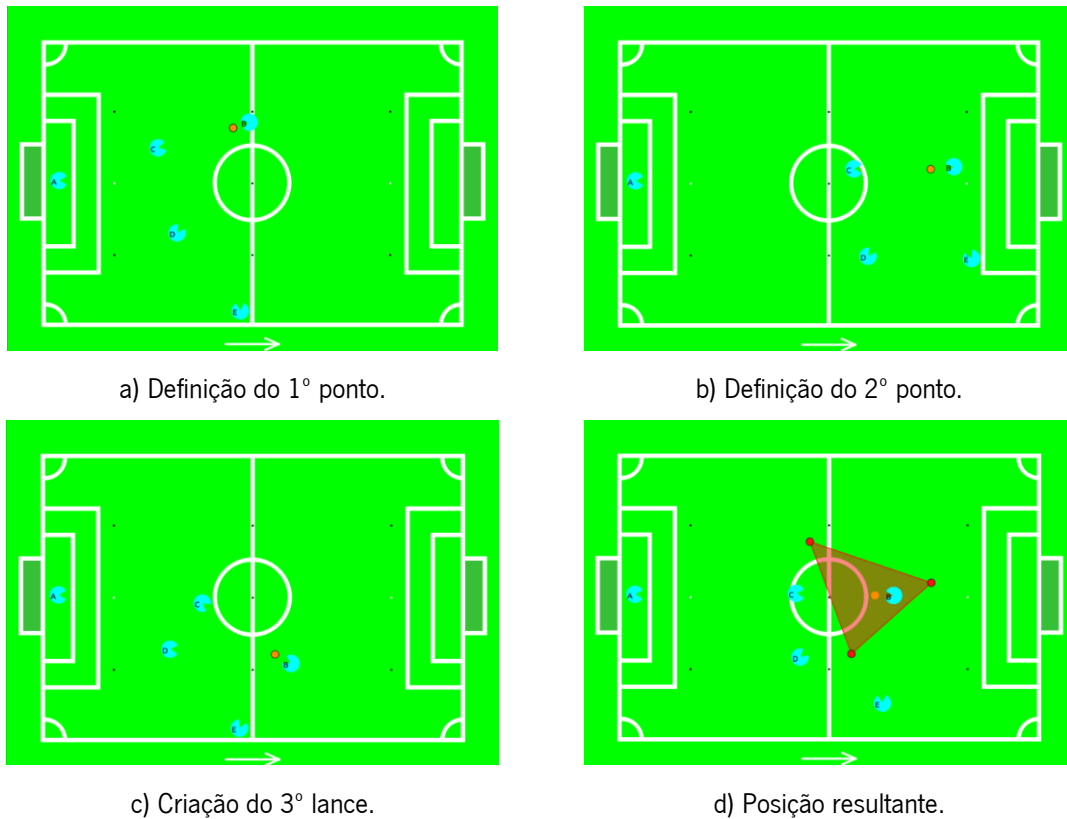


Figura 68 - Definição de faltas.

### 5.2.3. EDITOR DE PAPÉIS

O editor de papéis é usado para criar posições para os defesas durante os jogos. Podem ser criadas vários tipos de formações. No interface gráfico do Editor de Papéis (Figura 69), a parte superior é destinada à criação de formações.

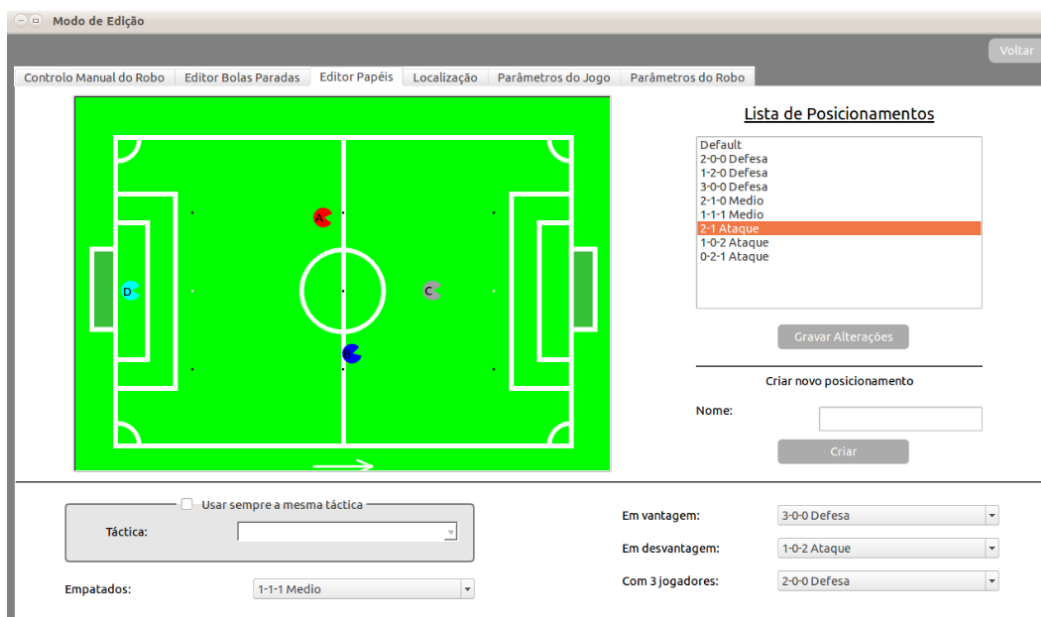


Figura 69 - Editor de papéis.

A parte inferior é usada para o utilizador escolher quando cada formação é seleccionada. Isto é, durante a realização de jogos, caso se verifique alguma das condições (empatados, em vantagem, em desvantagem ou com 3 jogadores em campo), a MINHO BaseStation selecciona automaticamente a formação a usar.

### 5.2.4. LOCALIZAÇÃO

O separador Localização auxilia o desenvolvimento e a análise do algoritmo de localização. Uma das suas tarefas consiste em desenhar os pontos (transições verde – branco e branco - verde) detectados pelo sistema de visão dos robôs. A Figura 71 ilustra o desenho dos pontos detectados e enviados por dois robôs.

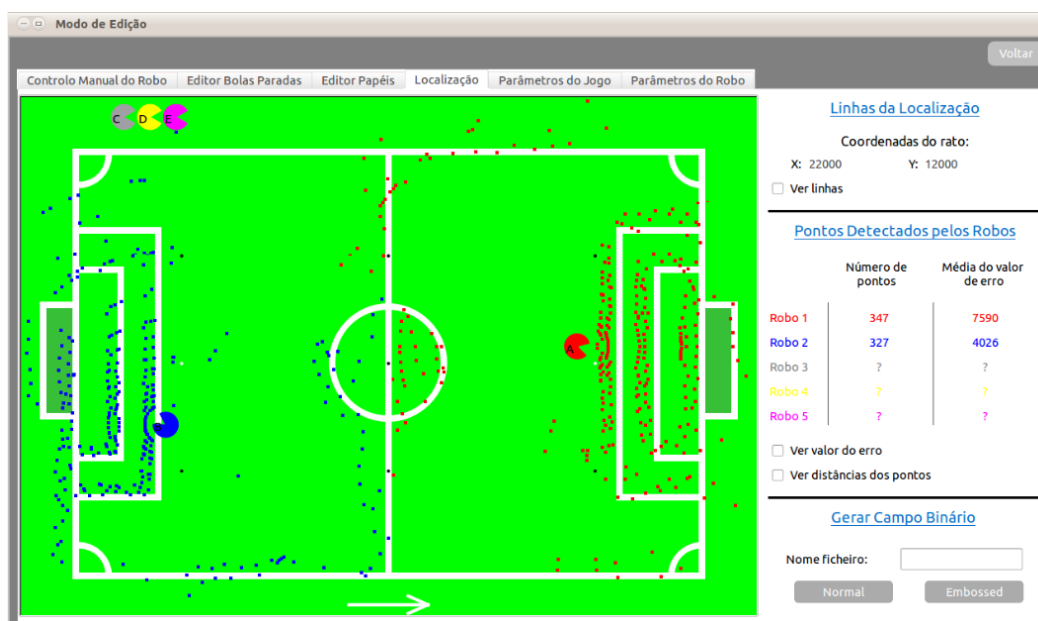


Figura 70 - Desenho dos pontos detectados pelos robôs.

Os pontos detectados são usados para calcular a localização do robô. Para cada ponto encontrado, o robô utiliza o valor da distância do ponto à linha do campo mais próxima.

Então, para analisar se as distâncias utilizadas pelos robôs estão correctas, foi implementada a opção de visualizar os pontos e a distância de cada ponto à linha branca mais próxima. Para isso, os robôs enviam juntamente com as coordenadas de cada ponto a direcção e o valor da distância. A Figura 71 exhibe, para um robô, os pontos detectados e uma linha que liga cada ponto até à linha branca mais próxima.

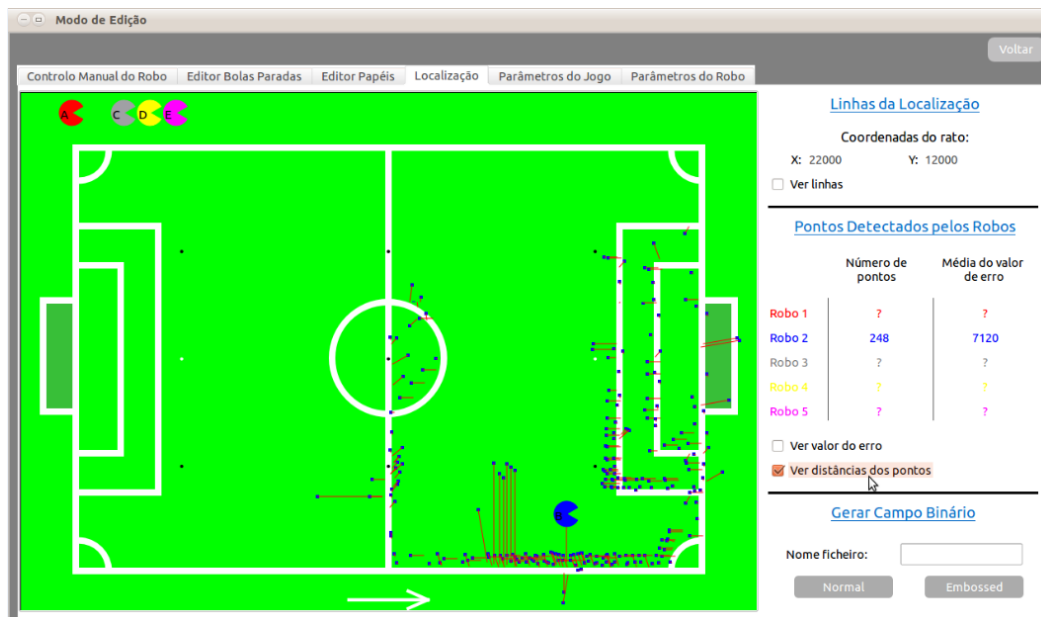


Figura 71 - Representação dos pontos e distâncias às linhas.

É também calculada a distância em cada coordenada do campo à linha mais próxima. O utilizador, para saber qual o valor numa determinada coordenada, apenas necessita de mover o rato até essa coordenada. A próxima Figura mostra as distâncias do ficheiro para a posição actual do rato.



Figura 72 - Visualização do ficheiro das distâncias calculado por cada robô.

Outra das funções do separador Localização consiste em gerar um campo binário, onde as linhas são representadas por uns e o campo por zeros. O campo é desenhado à escala, e é usado pelo robô para o cálculo da posição em campo.

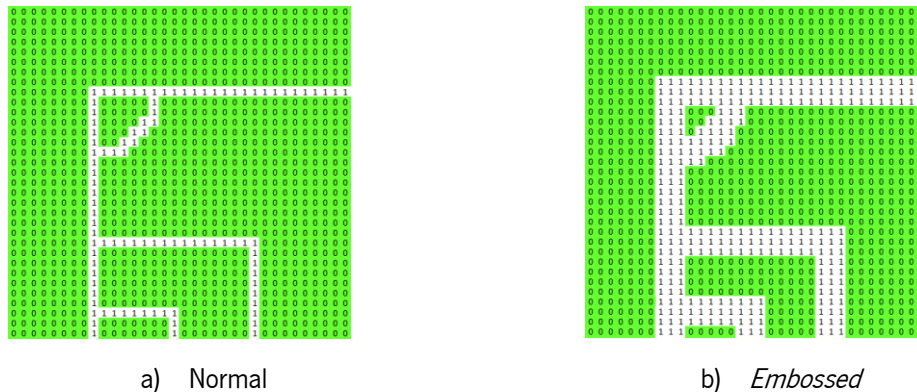


Figura 73 - Parte do campo binário gerado.

Na Figura 73 encontra-se representado duas imagens de uma parte do campo binário criado. O campo da Figura b) foi criado utilizando a opção de engrossar a largura das linhas de campo.

### 5.2.5. PARÂMETROS DO JOGO

A funcionalidade Parâmetros do Jogo permite alterar o valor de vários parâmetros, necessários durante a realização de jogos e na execução de outras funcionalidades da MINHO BaseStation. Entre eles, consta o endereço MAC do comando Wii Mote (utilizado na *RefBox* virtual), a resolução escolhida para o campo binário gerado no separador localização e a directoria onde são gravados os *logs* durante os jogos.

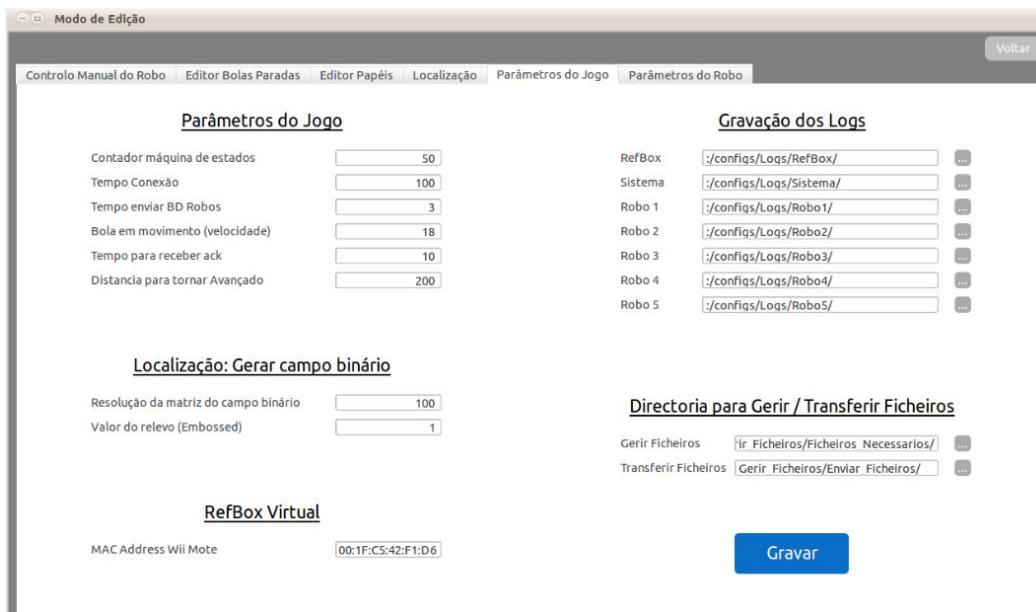


Figura 74 - Escolher parâmetros do jogo.



### 5.2.6. PARÂMETROS DO ROBÔ

O separador designado por Parâmetros do Robô configura remotamente os parâmetros do robô, e exibe a modelação da bola local.

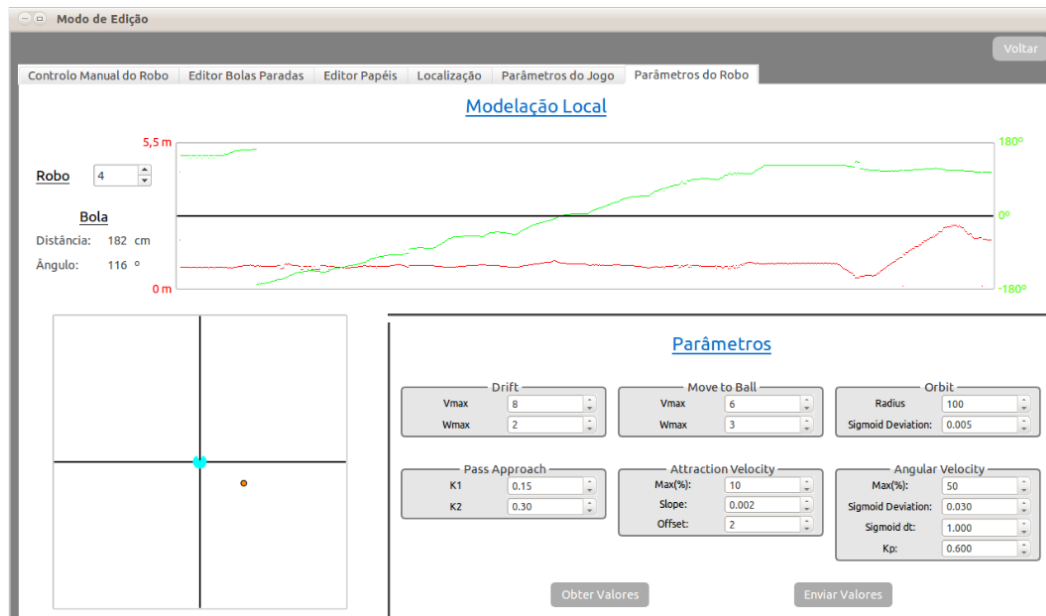


Figura 75 - Alterar parâmetros do robô.

Quanto aos parâmetros, o valor actual de cada parâmetro é actualizado na interface gráfica depois de o utilizador pressionar o botão “Obter Valores”. Para enviar os novos valores para o robô, apenas é necessário pressionar o botão destinado para esse fim.

Relativamente à modelação da bola, é exibida a distância e o ângulo de desfasamento entre a frente do robô e a bola. São ainda desenhados dois gráficos, onde um deles mostra a bola desenhada em relação ao robô de acordo com a sua posição real. O outro gráfico desenha, ao longo do tempo, a distância (de cor vermelha) e o ângulo (de cor verde) da bola. A modelação local realizada à bola permite analisar se a estrutura de visão (cabeça do robô, espelho e lente) está alinhada, e se o mapa das distâncias do robô está correcto.

### 5.3. MODO DE ANÁLISE

No Modo de Análise é realizada a reprodução de jogos. Durante a realização de jogos, o Modo de Jogo grava num ficheiro todos os dados recebidos dos robôs e da *RefBox*, e os dados do sistema. Com base nesses ficheiros, os jogos são reproduzidos de forma igual à sua ocorrência, de modo a permitir uma análise individual ou colectiva ao desempenho dos robôs, e possibilitar assim um desenvolvimento da equipa em jogos futuros.

A interface gráfica construída para o Modo de Análise consiste na Figura 76.



Figura 76 - Modo de Análise.

A estrutura do Modo de Análise pode ser dividida em dois blocos: o bloco de controlo (onde são escolhidos os ficheiros, a velocidade de reprodução e os eventos dos botões – como reproduzir ou parar), e o bloco de exibição - responsável por apresentar a reprodução do jogo.

Antes de ser iniciada a reprodução, é necessário o utilizador escolher os ficheiros a usar. Os ficheiros variam entre ficheiros dos robôs, da *RefBox* e do sistema, e para que a reprodução seja efectuada com sucesso, é necessário o utilizador não escolher mais do que um ficheiro do mesmo tipo. A Figura 77 exemplifica a escolha de um conjunto de ficheiros, do jogo realizado no dia 20 de maio de 2011 às 16h14.

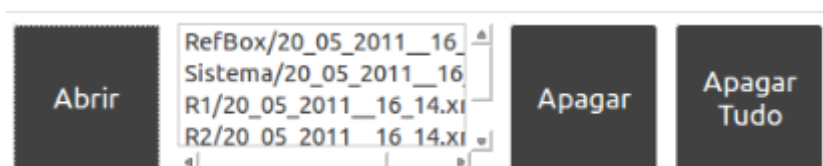


Figura 77 - Escolha de ficheiros para reprodução.

Terminada a selecção dos ficheiros, é possível a reprodução ser iniciada. O utilizador tem ao seu dispor os botões PLAY (reproduzir), PAUSA, STOP (parar), e FRAME (avançar uma *frame* na reprodução). Iniciando a reprodução, é exibido o tempo total do jogo, assim como o tempo actual.



Figura 78 - Tempo total e actual da reprodução de um jogo.

Relativamente à reprodução do jogo, na interface gráfica semelhante ao Modo de Jogo são exibidos os acontecimentos ocorridos durante o jogo, dependendo dos ficheiros escolhidos. O nome da equipa, resultado e período do jogo são mostrados caso o ficheiro do Sistema seja seleccionado. Os comandos recebidos da *RefBox* são listados mediante a escolha do ficheiro da *RefBox*. No campo de jogo são desenhados os robôs e as bolas de acordo com as posições armazenadas no ficheiro de cada robô. Nos painéis laterais são exibidas as informações de estado de cada robô. A Figura abaixo exemplifica a reprodução de um jogo escolhendo os ficheiros do Sistema, da *RefBox* e dos robôs.

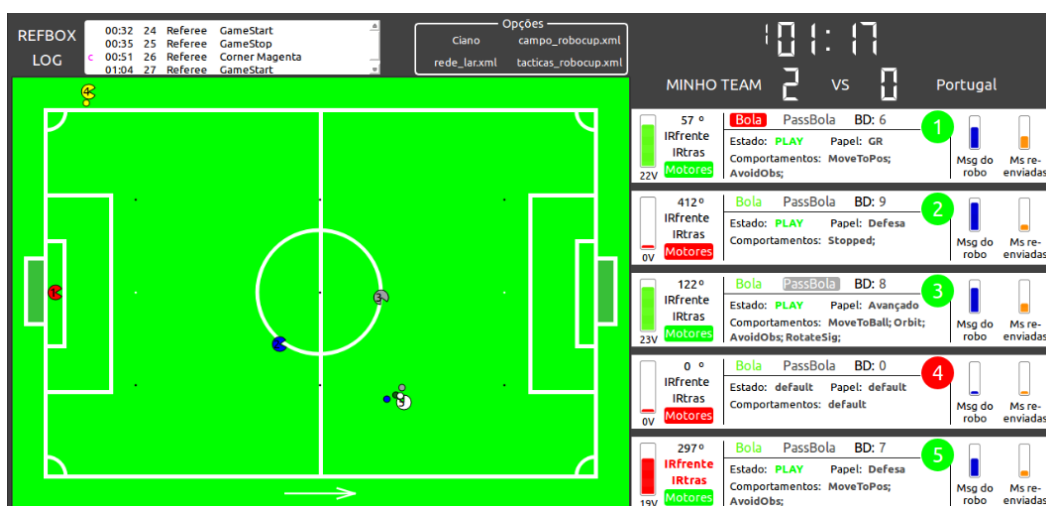


Figura 79 - Reprodução de um jogo no Modo de Análise.

Terminada a análise do jogo, são geradas e apresentadas as estatísticas – consultar Figura 80. Além do número de golos de cada equipa, são exibidos o número de cantos, faltas, pontapés de baliza, cartões amarelos e cartões vermelhos. São ainda criados dois gráficos, que mostram a percentagem de posse de bola da equipa MINHO TEAM, e a percentagem do jogo de cada estado obtive (*Idle*, *Positioning* e *Playing*).

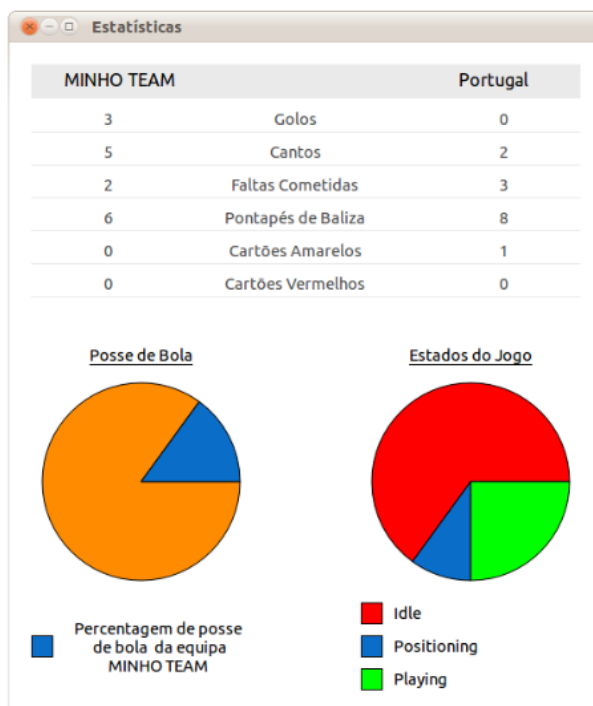


Figura 80 - Estatísticas de um jogo realizado.

Relativamente aos problemas do Modo de Análise, para a correcta reprodução de jogos é necessário que o utilizador seleccione nas Preferências o ficheiro com as dimensões de campo igual ao que foi usado na realização dos jogos, uma vez que o desenho dos robôs e das bolas em campo depende disso. É igualmente importante que não seja escolhido mais do que um ficheiro do mesmo tipo.

## 5.4. CONFIGURAÇÕES

É nas Configurações onde é possível alterar as dimensões do campo, as configurações de rede, gerir e transferir ficheiros para os robôs. Cada funcionalidade das Configurações encontra-se descrita nas secções seguintes.

### 5.4.1. CAMPO E AGENTES

O separador Campo e Agentes foi construído para o utilizador alterar as dimensões do campo de jogo – ver Figura 81. É listado o nome do ficheiro actual, bem como todas as medidas desse ficheiro. Quando o utilizador selecciona alguma das medidas, no campo existente é desenhada uma linha de cor vermelha, que representa o que corresponde essa medida na realidade. No canto inferior direito é realizado o *zoom* do campo de acordo com a posição do

rato, de forma a melhorar as condições de observação da medida – onde começa e onde acaba, se inclui a espessura da linha, entre outros.

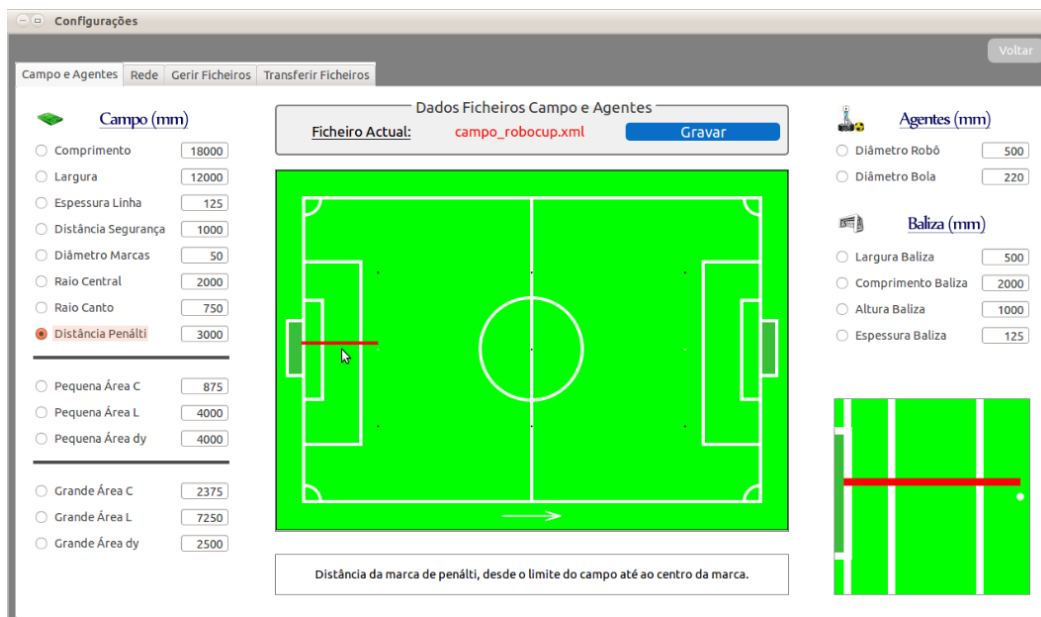


Figura 81 - Separador campo e agentes.

## 5.4.2. REDE

No separador de Rede é possível alterar as configurações de rede e testar a comunicação com os robôs e com a *RefBox*. É exibido o nome do ficheiro de Rede escolhido e todos os endereços IP e portas relativas à comunicação dos robôs – *unicast* e *multicast*, e com a *RefBox* – protocolo 2009 e 2010. A imagem do interface gráfico encontra-se na Figura abaixo.

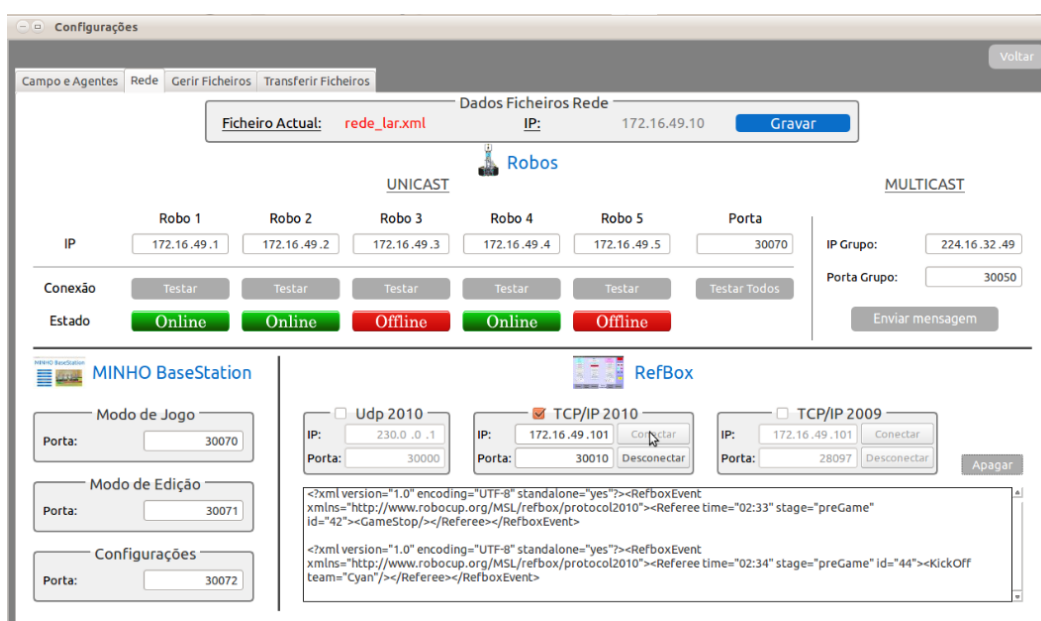


Figura 82 - Configurações de rede.

Outra das funcionalidades oferecidas pelo separador Rede consiste na possibilidade de testar a comunicação. Para os robôs, é enviada uma mensagem e caso o programa do robô esteja a executar ele envia de volta uma resposta. Quanto à *RefBox*, seleccionando um dos protocolos é possível estabelecer conexões e receber dados.

### 5.4.3. GERIR FICHEIROS

Os ficheiros necessários para que cada robô desempenhe correctamente as suas funções podem ser geridos a partir deste separador.

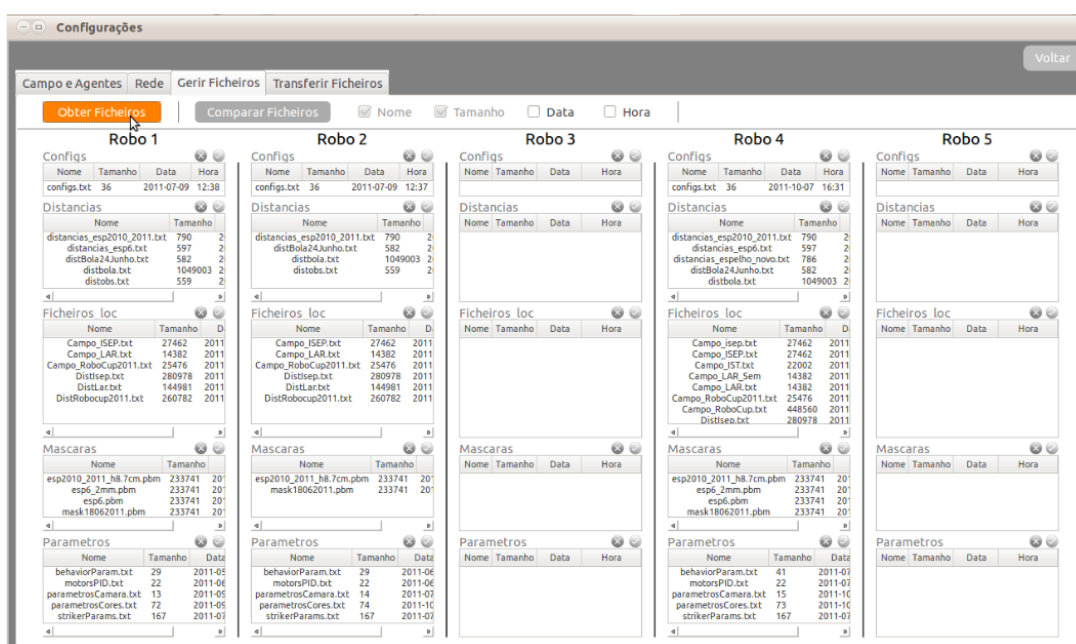


Figura 83 - Lista dos ficheiros existentes nos robôs.

O nome dos ficheiros existentes nos robôs é consultado e exibido nas respectivas listas – Figura 83.

Depois, os ficheiros são comparados com os existentes numa pasta local à MINHO BaseStation, através do tamanho, data e hora. O resultado da comparação pode ser examinado na Figura 84, onde os ficheiros incorrectos estão identificados a vermelho.

Sempre que o processo de comparação dos ficheiros detectar algum erro, é gerado um relatório de erros. A informação disponibilizada pelo relatório indica qual o robô e pasta onde ocorreu o erro, o nome e o problema do ficheiro. Desta forma, torna-se fácil para o utilizador tomar conhecimento de todos os problemas com os ficheiros.



Figura 84 - Comparação dos ficheiros.

Um exemplo de um relatório de erros está presente na Figura 85.

Relatório Erros				
	Robo	Pasta	Ficheiro	Problema
1	2	mascaras	esp6_2mm.pbm	Ficheiro em falta.

Fechar

Figura 85 - Relatório de erros.

#### 5.4.4. TRANSFERIR FICHEIROS

O objectivo deste separador consiste em enviar qualquer tipo de ficheiros para os robôs. Para isso, os ficheiros a enviar devem ser colocados numa pasta para esse efeito local à MINHO BaseStation. Terminada a colocação dos ficheiros na pasta, o nome, o tamanho, a data e a hora dos ficheiros são listadas na interface gráfica (consultar Figura 86). O utilizador pode ainda escolher quais os robôs que pretende enviar os ficheiros.

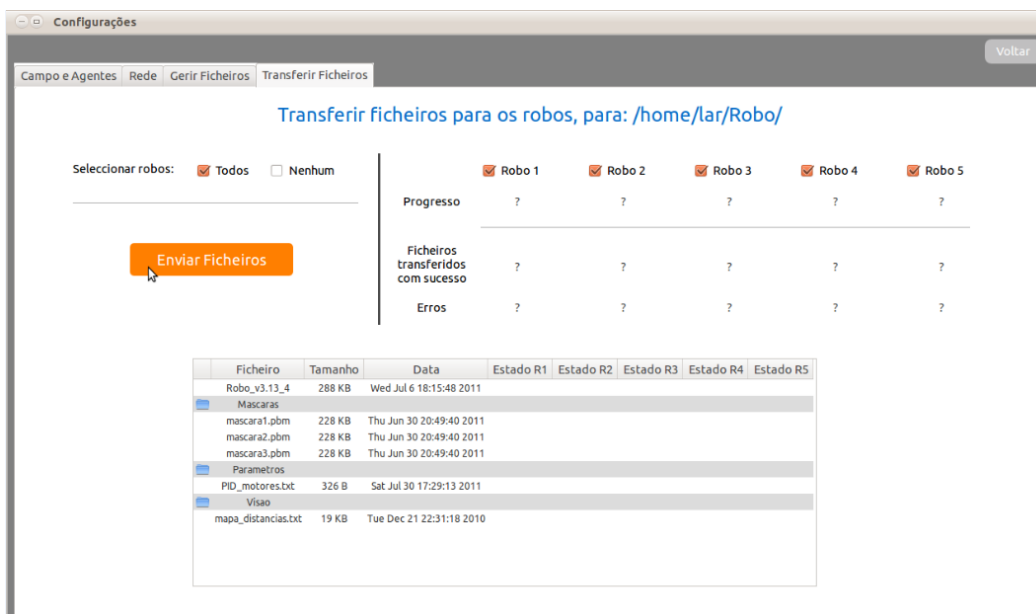


Figura 86 - Transferência de ficheiros para os robôs.

Pressionado o botão “Enviar Ficheiros”, à medida que os ficheiros são recebidos correctamente pelos robôs, são colocados na lista de cor verde.

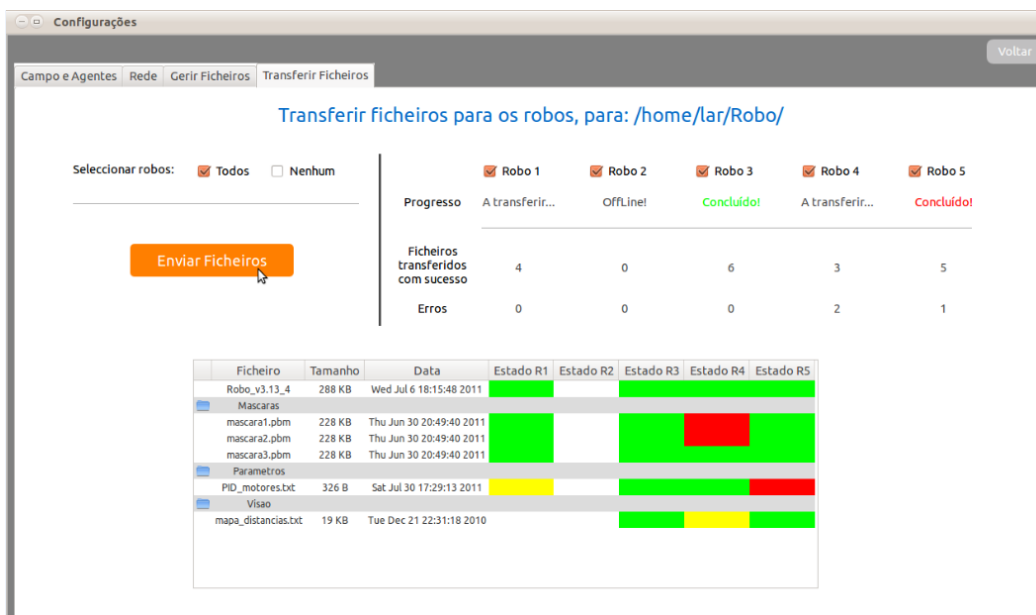


Figura 87 - Progresso da transferência de ficheiros.

Caso algum ficheiro seja recebido corrompido, o robô alerta a MINHO BaseStation e esse ficheiro é automaticamente identificado com a cor vermelha. A cor amarela indica que os ficheiros estão a ser transferidos. Quando a conclusão de ficheiros estiver terminada, o utilizador é informado – Figura 87.



A transferência de ficheiros a partir da MINHO BaseStation tornou-se uma ferramenta bastante útil à equipa MINHO TEAM, pois envia ficheiros para todos os robôs e ainda permite controlar o progresso individual de cada transferência.

## 5.5. PREFERÊNCIAS

É no modo Preferências que o utilizador escolhe os tipos de ficheiros a usar, a cor da equipa e o protocolo a usar para a conexão à *RefBox*. A interface gráfica é exibida na Figura 88.

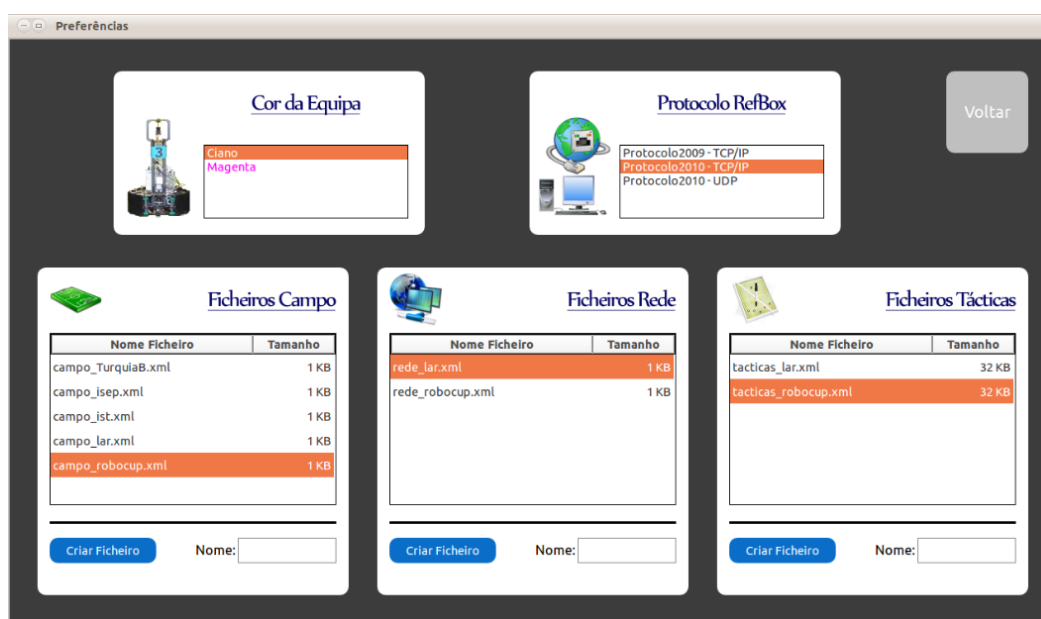


Figura 88 - Interface gráfica das Preferências.

Os tipos de ficheiros consistem nas dimensões do campo de jogo, nas configurações de rede e nas definições das táticas. Para cada tipo, são listados os ficheiros existentes e é automaticamente seleccionado o ficheiro usado actualmente. Caso seja pretendido criar um novo ficheiro, apenas é necessário o utilizador escolher um nome válido e o ficheiro é criado. A Figura abaixo exemplifica a criação de um novo ficheiro de Rede.





## **6. CONCLUSÃO E TRABALHO FUTURO**

### **6.1. CONCLUSÕES**

No âmbito desta dissertação foi desenvolvida uma aplicação para controlo e monitorização de uma equipa de futebol robótico, nomeadamente a equipa da Universidade do Minho denominada de MINHO TEAM.

Todo o processo de desenvolvimento ocorreu com alguma dificuldade, sobretudo devido a novas necessidades que surgiram ao longo do tempo e provocaram sucessivas reformulações na arquitectura da MINHO BaseStation de modo a não comprometer o cumprimento de todos os requisitos.

O objectivo que consiste em definir tácticas e estratégias de jogo foi um dos que ocupou mais tempo de implementação, pois era necessário construir uma interface gráfica simples de usar, que criasse e alterasse tácticas de forma fácil, rápida e sem recompilar nenhum código. Assim, foi criada uma interface gráfica constituída por um campo e robôs desenhados à escala, onde basta arrastar os robôs para as posições pretendidas, podendo a orientação do robô ser alterada através do botão scroll do rato. Todas as tácticas definidas são guardadas e carregadas num ficheiro do tipo XML sempre que há necessidade.

O Modo de Análise criado reproduz os jogos realizados da mesma forma que ocorreram. Permite aumentar ou diminuir a velocidade de reprodução, no entanto não é possível seleccionar um tempo de jogo específico para analisar, pois é realizada a reprodução sequencial de todos os eventos do jogo.

O Modo de Jogo, porque executa autonomamente durante a realização de jogos foi o que requereu mais atenção, nomeadamente no delineamento e implementação da máquina de estados. A realização de testes ocorreu de forma complicada, uma vez que para testar a comunicação e movimentos em equipa (como passes de bola) era necessário utilizar vários robôs.

O desenvolvimento das comunicações, especialmente com os robôs, foi concretizado de acordo com o software existente nos robôs de forma a manter a integridade, a arquitectura e o correcto funcionamento do software dos robôs.

Em conclusão, os objectivos propostos foram alcançados. Foi criada uma aplicação, designada de MINHO BaseStation, que efectua o controlo e monitorização da equipa MINHO TEAM, executando de forma completamente autónoma durante os jogos e fornecendo ferramentas como a criação e edição de tácticas ou a análise de jogos efectuados. A aplicação é constituída por vários módulos independentes, o que facilita o desenvolvimento e a integração de novos módulos com novas funcionalidades. A MINHO BaseStation, além de auxiliar o desenvolvimento de novos algoritmos de localização e controlo, permitiu à equipa MINHO TEAM participar no RoboCup 2011 realizado em Istambul, Turquia, na liga de futebol robótico médio.

## **6.2. TRABALHO FUTURO**

O trabalho desenvolvido nesta tese, uma vez que faz parte do desenvolvimento de uma equipa de robôs futebolistas, nunca se pode classificar como finalizado. Isto porque o desenvolvimento é contínuo, as regras da competição são alteradas, as equipas adversárias progridem e então surge a necessidade de melhorar software e hardware de modo a evoluir a nível científico e tecnológico, e manter os índices competitivos.

Então, é proposto que no futuro seja desenvolvido um simulador para a equipa MINHO TEAM. Pode ser adicionado à MINHO BaseStation, e traria benefícios como poupar as baterias dos robôs, excluir avarias causadas nos robôs, diminuir o tempo de testes, facilidade de testar novos algoritmos, simplicidade de estudar comportamentos cooperativos e eliminar a necessidade de realizar calibrações em vários parâmetros dos robôs.

É igualmente sugerido que seja implementado na MINHO BaseStation a possibilidade de observar a imagem capturada em cada robô. Através da rede de comunicações, as imagens capturadas seriam enviadas e visualizadas num ecrã externo. A implementação deste objectivo iria simplificar a tarefa do programador e reduzir o tempo de análise a todo o processo de visão dos robôs, pois deixaria de ser necessário o transtorno de transportar e alimentar um monitor para junto dos robôs sempre que é desejado analisar a imagem capturada e o processo de detecção de linhas, bola e obstáculos.

A implementação de mais inteligência na MINHO BaseStation, nomeadamente durante a realização de jogos, é algo a considerar no futuro. Através de redes neuronais ou outras técnicas semelhantes, analisando os movimentos dos adversários e adaptando a táctica de jogo, defensiva ou ofensivamente, é algo que pode melhorar o desempenho dos robôs durante o jogo, a nível do jogo colectivo e do número de golos marcados e sofridos.

## 7. BIBLIOGRAFIA

- [1] The Robocup Federation. (2011, Setembro) RoboCup. [Online]. [www.robocup.org](http://www.robocup.org)
- [2] MINHO TEAM. (Setembro, 2010) MINHO TEAM. [Online]. [www.robotica.dei.uminho.pt/robocup/](http://www.robotica.dei.uminho.pt/robocup/)
- [3] RobocupMSL. (2011, Janeiro) Regras RoboCup MSL. [Online]. <http://groups.google.com/group/robocupmsl>
- [4] Geeknet. (2010, Novembro) RoboCup MSL refbox. [Online]. [Geeknet; http://sourceforge.net/projects/msl-refbox/](http://sourceforge.net/projects/msl-refbox/)
- [5] M. Bellis, About.com. (2010, Outubro) The Robot Story. [Online]. <http://inventors.about.com/od/rstartinventions/a/RobotStory.htm>
- [6] Instituto Superior Técnico. (2011, Janeiro) Festival Nacional de Robótica. [Online]. <http://robotica2011.ist.utl.pt/>
- [7] SAR – Soluções de Automação e Robótica. (2010, Outubro) Cadeira de Rodas Omnidireccional. [Online]. [http://industry.sarobotica.pt/index.php?option=com\\_content&task=view&id=67&Itemid=84](http://industry.sarobotica.pt/index.php?option=com_content&task=view&id=67&Itemid=84)
- [8] Wikipedia, the free encyclopedia. (2010, Setembro) RoboCup. [Online]. <http://en.wikipedia.org/wiki/RoboCup#Venues>
- [9] Wikipedia, the free encyclopedia. (2011, Julho) MSL Qualified teams. [Online]. [http://wiki.robocup.org/wiki/Results\\_World\\_Championships\\_2011:\\_Istanbul,\\_Turkey#Qualified\\_teams](http://wiki.robocup.org/wiki/Results_World_Championships_2011:_Istanbul,_Turkey#Qualified_teams)
- [10] Water Team. (2010, Outubro) Water Team. [Online]. [http://jdgcy.bistu.edu.cn/robocup/index\\_robocup.asp](http://jdgcy.bistu.edu.cn/robocup/index_robocup.asp)
- [11] Technische Universiteit Eindhoven University of Technology. (2010, Setembro) Tech United Eindhoven. [Online]. <http://www.techunited.nl/en>
- [12] IEETA Universidade de Aveiro. (2010, Outubro) CAMBADA. [Online]. <http://www.ieeta.pt/atri/cambada/>
- [13] FEUP Universidade do Porto. (2010, Outubro) 5DPO Site. [Online]. <http://paginas.fe.up.pt/~robosoc/en/doku.php>
- [14] Microsoft Corporation. (2010, Outubro) Microsoft Visual Studio. [Online]. <http://www.microsoft.com/visualstudio/en-us>

- [15] M., Huang; S., Hui; X., Wang; Y., Zhao; Z., Wang; Q., Zhang; L., Zhao; X., Xu; W., Zhang; S., Chen; X., Liu;., Zhao; L., Zhang; M., Wang; Y., Lu; Z., Zhu; C., Wang; B., Qin; YinYue; S., Li; B., Huang, *Water Team Description Paper 2010*.
- [16] H., Min; G., Xinsheng; H., Shiyong; W., Xueyan; C., Song; X., Xinxin; Z., Wanjie; L., Ye; L., Xiaoming; Z., Liang; W., Miao; Z., Zhe; W., Chenyu; H., Bin; M., Libo; Q., Biao; Z., Fangyu; W., Changyun, *Water Team Description Paper 2011*.
- [17] Kanters, F.; Hoogendijk, R.; Janssen, R.; Meessen, K.; de Best, J.; Bruijnen, D.; Naus, G.; Aangenent, W.; van der Berg, R.; van de Loo, H.; Heldens, G.; Vugts, R.; Harkema, G.; van Brakel, P.; Bukkums, B.; Soetens, R.; Merry, R.; van de Molengraft, M., *Tech United Eindhoven Team Description 2011*.
- [18] The Glade project. (2010, Outubro) Glade. [Online]. <http://glade.gnome.org/>
- [19] de Best, J.; Bruijnen, D.; Hoogendijk, R.; Janssen, R.; Meessen, J.; Merry, R.; van de Molengraft, M.; Naus, G.; Ronde, M., *Tech United Eindhoven Team Description 2010*.
- [20] Neves, A.; Azevedo, J.; Cunha, B.; Dias, R.; Fonseca, P.; Lau, N.; Silva, J.; Ângelo, T.; Cruz, C.; Corrente, G.; Lopes, L.; Pedreiras, P.; Pereira, A.; Pinho, A.; Rodrigues, J.; Santos, F.; Vieira, J., *CAMBADA'2011: Team Description Paper*.
- [21] Nokia Corporation. (2010, Outubro) Qt. [Online]. <http://qt.nokia.com/products/>
- [22] Neves, A.; Azevedo, J.; Cunha, B.; Lau, N.; Pereira, A.; Corrente, G.; Santos, F.; Martins, D.; Figueiredo, N.; Silva, J.; Cunha, J.; Ribeiro, B.; Sequeira, R.; Almeida, L.; Lopes, L.; Rodrigues, J.; Pinho, A., *CAMBADA'2010: Team Description Paper*.
- [23] Mark de Berg, Otfried Cheong, Marc van Kreveld, and Mark Overmars, *Computational Geometry: Algorithms and Applications*, Third Edition ed.: Springer - Verlag, 2008.
- [24] Nascimento, T.; Pinto, M.; Sobreira, H.; Guedes, F.; Castro, A.; Malheiros, P.; Pinto, A.; Alves, H.; Ferreira, M.; Costa, P.; Costa, P. G.; Souza, A.; Almeida, L.; Reis, L.; Moreira, A., *5DPO'2011: Team Description Paper*.
- [25] TinyPortal. (2010, Outubro) Lazarus. [Online]. <http://www.lazarus.freepascal.org/>
- [26] Wikipédia, a enciclopédia livre. (2010, Novembro) Ambiente de Desenvolvimento Integrado. [Online]. [http://pt.wikipedia.org/wiki/Ambiente\\_de\\_Developolvimento\\_Integrado](http://pt.wikipedia.org/wiki/Ambiente_de_Developolvimento_Integrado)
- [27] Wikipédia, a enciclopédia livre. (2010, Novembro) Comparison of Web application frameworks. [Online]. [http://en.wikipedia.org/wiki/Comparison\\_of\\_Web\\_application\\_frameworks](http://en.wikipedia.org/wiki/Comparison_of_Web_application_frameworks)

- [28] Ribeiro, Fernando; Lopes, Gil; Costa, João; Rodrigues, João P.; Pereira, Bruno; Silva, João; Silva, Sérgio; Ribeiro, Paulo; Trigueiros, Paulo, *Minho MSL - A New Generation of soccer robots*.
- [29] Tanenbaum, Andrew S., *Redes de Computadores*, 4th ed.: Campus, 2003.
- [30] Nintendo. (2011, Fevereiro) Wii Controllers. [Online]. <http://www.nintendo.com/wii/console/controllers>
- [31] Smith, Geoffrey C., *Introductory Mathematics: Algebra and Analysis*, 2nd ed.: Springer, 1998.
- [32] W3C. (2011, Janeiro) Extensible Markup Language (XML). [Online]. <http://www.w3.org/XML/>
- [33] Vera Lúcia Da Rocha Lopes and Márcia A: Gomes Ruggiero, *Cálculo Numérico*, 2nd ed.: Makron Books, 1988.
- [34] Alan F. Beardon, *Algebra and Geometry*, 1st ed.: Cambridge University Press, 2005.





## APÊNDICE A

Mensagens enviadas da *RefBox* usando o protocolo2009.

Tabela 3 - Descodificação do protocolo2009 da *RefBox*.

<b>Comando Recebido</b>	<b>Significado</b>
<b>1</b>	Início da primeira parte do jogo
<b>2</b>	Início da segunda parte do jogo
<b>h</b>	Intervalo
<b>e</b>	Fim do jogo
<b>S</b>	<i>Stop</i>
<b>s</b>	<i>Start</i>
<b>K</b>	Pontapé de saída para a equipa ciano
<b>k</b>	Pontapé de saída para a equipa magenta
<b>F</b>	Falta para a equipa ciano
<b>f</b>	Falta para a equipa magenta
<b>G</b>	Pontapé de baliza para a equipa ciano
<b>g</b>	Pontapé de baliza para a equipa magenta
<b>T</b>	Lançamento para a equipa ciano
<b>t</b>	Lançamento para a equipa magenta
<b>C</b>	Canto para a equipa ciano
<b>c</b>	Canto para a equipa magenta
<b>P</b>	Penákti para a equipa ciano
<b>p</b>	Penákti para a equipa magenta
<b>N</b>	Bola ao solo
<b>X</b>	Anulação do último comando
<b>A</b>	Golo para a equipa ciano
<b>a</b>	Golo para a equipa magenta
<b>L</b>	Estacionamento

## APÊNDICE B

Mensagens enviadas da *RefBox* usando o protocolo2010.

- *Start*

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<RefboxEvent xmlns="http://www.robocup.org/MSL/refbox/protocol2010">
<Referee time="00:52" stage="firstHalf" id="37">
<GameStart/>
</Referee>
</RefboxEvent>
```

- *Stop*

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<RefboxEvent xmlns="http://www.robocup.org/MSL/refbox/protocol2010">
<Referee time="08:04" stage="preGame" id="17">
<GameStop/>
</Referee>
</RefboxEvent>
```

- Pontapé de saída para a equipa magenta

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<RefboxEvent xmlns="http://www.robocup.org/MSL/refbox/protocol2010">
<Referee time="08:23" stage="preGame" id="21">
<KickOff team="Magenta"/>
</Referee>
</RefboxEvent>
```

- Falta para a equipa ciano

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<RefboxEvent xmlns="http://www.robocup.org/MSL/refbox/protocol2010">
<Referee time="00:37" stage="firstHalf" id="27">
<FreeKick team="Cyan"/>
</Referee>
</RefboxEvent>
```

- Pontapé de baliza para a equipa magenta

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<RefboxEvent xmlns="http://www.robocup.org/MSL/refbox/protocol2010">
<Referee time="00:40" stage="firstHalf" id="29">
<GoalKick team="Magenta"/>
</Referee>
</RefboxEvent>
```

- Lançamento para a equipa ciano

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<RefboxEvent xmlns="http://www.robocup.org/MSL/refbox/protocol2010">
<Referee time="00:43" stage="firstHalf" id="31">
<ThrowIn team="Cyan"/>
</Referee>
</RefboxEvent>
```

- Canto para a equipa magenta

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<RefboxEvent xmlns="http://www.robocup.org/MSL/refbox/protocol2010">
<Referee time="00:46" stage="firstHalf" id="33">
<Corner team="Magenta"/>
</Referee>
</RefboxEvent>
```

- Penákti para a equipa ciano

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<RefboxEvent xmlns="http://www.robocup.org/MSL/refbox/protocol2010">
<Referee time="00:48" stage="firstHalf" id="35">
<Penalty team="Cyan"/>
</Referee>
</RefboxEvent>
```

- Bola presa

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<RefboxEvent xmlns="http://www.robocup.org/MSL/refbox/protocol2010">
<Referee time="01:37" stage="secondHalf" id="75">
<DroppedBall/>
</Referee>
</RefboxEvent>
```

- Anulação do último comando

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<RefboxEvent xmlns="http://www.robocup.org/MSL/refbox/protocol2010">
<Referee time="01:25" stage="firstHalf" id="41">
<Cancel/>
</Referee>
</RefboxEvent>
```

- Estacionamento

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<RefboxEvent xmlns="http://www.robocup.org/MSL/refbox/protocol2010">
  <Referee time="08:12" stage="preGame" id="19">
    <Parking/>
  </Referee>
</RefboxEvent>
```

- Alteração do estado do jogo

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<RefboxEvent xmlns="http://www.robocup.org/MSL/refbox/protocol2010">
  <Referee time="00:00" stage="firstHalf" id="22">
    <StageChange newStage="firstHalf"/>
  </Referee>
</RefboxEvent>
```

- Golo para a equipa magenta

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<RefboxEvent xmlns="http://www.robocup.org/MSL/refbox/protocol2010">
  <Referee time="01:36" stage="firstHalf" id="45">
    <GoalAwarded time="01:36" team="Magenta" stage="firstHalf" player="3" own="false"/>
  </Referee>
</RefboxEvent>
```

- Cartão amarelo para a equipa magenta

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<RefboxEvent xmlns="http://www.robocup.org/MSL/refbox/protocol2010">
  <Referee time="02:14" stage="firstHalf" id="49">
    <CardAwarded time="02:14" team="Magenta" stage="firstHalf" player="4" number="1"
color="yellow"/>
  </Referee>
</RefboxEvent>
```

- Cartão vermelho para a equipa ciano

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<RefboxEvent xmlns="http://www.robocup.org/MSL/refbox/protocol2010">
  <Referee time="02:27" stage="firstHalf" id="50">
    <CardAwarded time="02:27" team="Cyan" stage="firstHalf" player="4" number="1"
color="red"/>
  </Referee>
</RefboxEvent>
```

- Substituição na equipa magenta: sai o robô 1 e entra o robô 5

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<RefboxEvent xmlns="http://www.robocup.org/MSL/refbox/protocol2010">
  <Referee time="00:11" stage="secondHalf" id="62">
    <Substitution team="Magenta" playerOut="1" playerIn="5"/>
  </Referee>
</RefboxEvent>
```

- Substituição na equipa ciano: sai o robô 3 para reparação

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<RefboxEvent xmlns="http://www.robocup.org/MSL/refbox/protocol2010">
  <Referee time="00:48" stage="secondHalf" id="66">
    <PlayerOut team="Magenta" reason="outForRepair" player="3"
minimumNrOfSecondsOfField="30"/>
  </Referee>
</RefboxEvent>
```

- Substituição na equipa magenta: entra o robô 2

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<RefboxEvent xmlns="http://www.robocup.org/MSL/refbox/protocol2010">
  <Referee time="01:31" stage="secondHalf" id="72">
    <PlayerIn team="Magenta" reason="inOnRequest" player="2"/>
  </Referee>
</RefboxEvent>
```