FACULDADE DE ENGENHARIA DA UNIVERSIDADE DO PORTO

Robôs Móveis com Visão Omnidirecional

Telmo Miguel Silva Costa

Mestrado Integrado em Engenharia Eletrotécnica e de Computadores

Orientador: Paulo José Cerqueira Gomes da Costa (Prof. Dr.)

Co-orientador: António Paulo Gomes Mendes Moreira (Prof. Dr.)

25 de Julho de 2018

Resumo

Atualmente, na área da robótica, existe uma grande variedade de situações que implicam o deslocamento dos robôs. Assim, é necessário o ajuste da posição do robô para diferentes locais, pelo que os sistemas de visão atuam, não só na determinação do posicionamento global do robô relativamente ao meio onde se encontra, mas também em reconhecimento de padrões, identificação de objetos e extração de caraterísticas das imagens recolhidas. Os sistemas de visão omnidirecionais permitem obter imagens com um ângulo aumentado do sistema.

Nesse sentido, os sistemas catadióptricos, que consistem, tipicamente, numa câmara convencional apontada a um ou mais espelhos, normalmente de forma hiperbólica, permitem aumentar o campo de visão, embora com uma taxa de distorção elevada. Neste trabalho pretende-se abordar estes assuntos, com particular incidência na calibração do sistema de visão catadióptrico, utilizando como plataformas de testes os robôs do futebol robótico MSL da equipa 5DPO.

No futebol robótico, é essencial a informação extraída a partir do sistema de visão, quer para a deteção da bola, dos limites do campo, obstáculos e balizas, quer em termos de localização do robô em campo, através da fusão de dados com a odometria. Deste modo, é de maior relevância a calibração do sistema de visão, para garantir que os dados são corretos, ou pelo menos, fidedignos, com o erro dentro de limites admissíveis.

Abstract

Nowadays, in the area of robotics, there are a major variety of situations involving the displacement of robots. Thus, it is necessary to adjust the position of the robot to different locations, whereat vision systems act, not only in determining the overall positioning of the robot in the environment in which it is located, but also in pattern recognition, object identification and features extraction from the obtained images. Omnidirectional vision systems allow to obtain images with an increased angle of the system.

In this regard, catadioptric systems, which typically consist of a conventional camera pointed at one or several mirrors, usually in a hyperbolic shape, allow to increase the field of view, albeit with a high distortion rate. This work intends to address these issues, with particular emphasis on the calibration of the catadioptric vision system, using as test platforms the MSL robotic soccer robots of the 5DPO team.

In robotic soccer, information extracted from the vision system is essential, whether for detecting the ball, field boundaries, obstacles and goal, or in terms of robot location in the field, by merging data with odometry. For that reason, the calibration of the vision system is of greater relevance, to ensure that the data is correct, or at least reliable, with the error within permissible limits.

Agradecimentos

Em primeiro lugar, queria agradecer ao Professor Doutor Paulo Costa e ao Professor Doutor António Paulo Moreira, meu orientador e co-orientador, respetivamente, por todos os conselhos, apoio e oportunidades que me proporcionaram. A participação na equipa 5DPO e no Festival de Robótica foram experiências que me fizeram crescer e que recordo com satisfação.

Obrigado à equipa de futebol robótico 5DPO, em particular ao Pedro Relvas, por me ter recebido no grupo e acompanhado nos primeiros passos desta dissertação, ao Tiago Raúl, por todas as discussões, sugestões e disponibilidade, ao Eurico pela ajuda e prestabilidade, sem esquecer o Henrique, o Sandro, o Francisco, o Héber e o Paulo. Foram realmente um exemplo de entreajuda e companheirismo.

Um abraço aqueles que ao longo deste curso tive como colegas e que agora levo como amigos, assim como a todos os companheiros além do contexto académico, pelo vosso incentivo, amizade e bons momentos partilhados.

Por fim, uma palavra de apreço aos meus pais, irmão e a toda a minha família, pelo vosso carinho, por serem o meu suporte, pelos valores transmitidos, e por assegurarem a minha educação e ensino.

Telmo Costa

"We are increasingly becoming cyborg-like beings.
We are becoming literally what we create.
Biology, physics, and technology are evolving towards one and the same thing."

Gray Scott

Conteúdo

Lista de Figuras

Lista de Tabelas

Abreviaturas e Símbolos

2D	Espaço Bidimensional
3D	Espaço Tridimensional
CAMBADA	Cooperative Autonomous Mobile roBots with Advanced Distributed Architecture
CRIIS	Centre for Robotics in Industry and Intelligent Systems
DEEC	Departamento de Engenharia Electrotécnica e de Computadores
GUI	Graphical User Interface
IDE	Integrated Development Environment
FEUP	Faculdade de Engenharia da Universidade do Porto
FIFA	Federação Internacional de Futebol
HSV	Hue, Saturation, Value
IDE	Integrated Development Environment
INESC TEC	Instituto de Engenharia de Sistemas e Computadores – Tecnologia e Ciência
LM	Levenberg-Marquardt
MAE	Mean Absolute Error
MSL	Middle Size League
RMSE	Root-Mean-Square Error
RoboCup	Robot World Cup Initiative
SVP	Single View-point

Capítulo 1

Introdução

Neste primeiro capítulo é realizada uma abordagem introdutória à dissertação "Robôs Móveis com Visão Omnidirecional". Inicialmente, é efetuado um enquadramento do tema, sucedido pela motivação para a sua realização. Seguidamente, são apresentados os principais objetivos desta dissertação. Por fim, é explicada a estrutura do documento.

1.1 Contexto

Atualmente, na área da robótica, existe uma grande variedade de situações que implicam o deslocamento e mobilidade dos robôs, dado que os objetos sobre os quais as ações dos robôs incidem correspondem, em certas aplicações, a áreas de vastas e variadas dimensões. Assim, é necessário o ajuste da posição do robô para diferentes posições, pelo que os sistemas de visão atuam, não só na determinação do posicionamento global do robô relativamente ao meio onde se encontram, mas também em reconhecimento de padrões, identificação de objetos e extração de caraterísticas das imagens recolhidas. Os sistemas de visão omnidirecionais permitem obter imagens com um ângulo aumentado do sistema, pelo que a sua utilização é vantajosa nalgumas finalidades como, por exemplo, determinar a posição da bola num jogo de futebol robótico, que se pode encontrar em qualquer ponto do campo.

1.2 Motivação

As câmaras convencionais têm um campo de visão que limita a sua aplicação em sistemas que requerem um ângulo de visão alargado. Assim, os sistemas catadióptricos, que consistem, tipicamente, numa câmara convencional apontada a um ou mais espelhos, normalmente de forma hiperbólica, permitem aumentar o campo de visão, embora com uma taxa de distorção elevada. Assim, o controlo e movimentação de robôs dotados de visão omnidirecional coloca desafios diversos. Desde a correta calibração do conjunto câmara–espelho, passando pela deteção de características na imagem, transmissão e planificação da imagem captada e planeamento de trajetórias, tendo em conta diversos critérios (evitar obstáculos, inspecionar zonas, detetar alvos, localizar-se). Neste

trabalho pretende-se abordar estes assuntos, com particular incidência na calibração do sistema de visão catadióptrico, utilizando como plataformas de testes os robôs do futebol robótico MSL da equipa 5DPO.

1.3 Objetivos

No futebol robótico, é essencial a informação extraída a partir do sistema de visão, quer para a deteção da bola, dos limites do campo, obstáculos ou balizas, quer em termos de localização do robô em campo, através da fusão de dados com a odometria. Deste modo, é de maior relevância a calibração do sistema de visão, para garantir que os dados são corretos, ou pelo menos, viáveis, com o mínimo de erro possível.

Tendo em conta as dificuldades apresentadas anteriormente, pretende-se com a realização desta dissertação apresentar uma proposta de solução que passa pelo estudo e implementação do sistema de calibração do sistema de visão, estudo e implementação do sistema de movimentação e transmissão de imagem, concluindo com o teste e validação do sistema global desenvolvido.

1.4 Estrutura da Dissertação

Para além da introdução, esta dissertação contém mais 5 capítulos.

No capítulo 2, é descrito o estado da arte e são apresentados trabalhos relacionados com a temática abordada, terminando com a definição do problema, para o qual se apresenta uma proposta de solução.

De seguida, no capítulo 3 são explicadas algumas convenções adotadas, estudam-se as relações matemáticas entre as grandezas e detalham-se as tarefas necessárias para levar a cabo a metodologia proposta.

Ao longo do capítulo 4 analisam-se algoritmos de otimização, explica-se o *software* desenvolvido e retiram-se ilações da função de mapeamento.

Na sequência, o capítulo 5 contém resultados de testes efetuados que permitem validar o sucesso do método proposto.

Finalmente, encerra-se com o capítulo 6 fazendo um balanço do trabalho realizado e promovendo algumas sugestões de melhorias.

Capítulo 2

Revisão Bibliográfica

No presente capítulo é feito um levantamento do estado da arte relativo ao tema a ser desenvolvido nesta dissertação. Inicialmente, é abordado o futebol robótico, algumas regras e desafios, e ainda algumas equipas participantes nas competições. De seguida, é explicado o que é um sistema catadióptrico e como é classificado.

Por último, são apresentadas formas de abordar a problemática da calibração dos sistemas catadióptricos, quer com soluções paramétricas, quer com soluções não paramétricas.

2.1 Futebol Robótico

A entidade responsável pela organização de competições de futebol robótico é o *RoboCup* [3], que promove a ciência e a investigação científica, em particular nas áreas de robótica e inteligência artificial. Através das competições que organiza, o objetivo é suscitar o interesse nessas áreas de estudo e, em última instância, potenciar o aparecimento de novas soluções para os desafios inerentes a essas mesmas áreas [6].

Em relação à estrutura da organização, este é organizada em *Trustees* e alguns comités, nomeadamente, executivos, técnicos, organizadores e regionais [7].

A organização estabeleceu como objetivo principal ter "a meio do século XXI, uma equipa de futebol constituída por robôs humanóides, completamente autónomos, capazes de derrotar o vencedor da mais recente edição do Mundial, numa partida jogada com as regras da FIFA" [6], embora a organização reconheça que neste momento as tecnologias utilizadas ainda não permitam perseguir a concretização desse objetivo.

Entretanto, o *RoboCup* promove periodicamente competições e eventos para fomentar o desenvolvimento de novas soluções. Entre as competições organizadas tem-se as ligas *RoboCupSoccer*, que são subdivididas em *Simulation Leagues*, *Small Size*, *Middle Size*, *Standard Platform League* e *Humanoid*. Para além destes, existem ainda outros desafios como o *RoboCupRescue*, *RoboCup@Home*, *RoboCupIndustrial*, *RoboCupJunior* [7].

Figura 2.1: Equipas de futebol robótico da liga *RoboCupSoccer* durante uma partida [1].

A equipa 5DPO, pertencente à FEUP e ao CRIIS - Centro de Robótica Industrial e Sistemas Inteligentes do INESC TEC, e constituída por professores, estudantes e investigadores, enquadra-se na liga *RoboCupSoccer*, em particular, na categoria *Middle Size League* (MSL), embora já tenha participado nas categorias *Small Size* e *Humanoid* [2]. Esta atividade tem como finalidade desenvolver competências de visão artificial, localização, fusão sensorial, controlo em tempo real, decisão e cooperação de equipas de robôs móveis [8].

Figura 2.2: Robôs da equipa de futebol robótico 5DPO. Imagem retirada de [2].

A competição MSL é regrada por regulamentos que todos os anos são atualizados pelo comité

técnico do *Robocup*. Entre as leis que regem os desafios tem-se, por exemplo, normas para as dimensões do campo, bola e robôs, constituição das equipas, tempos de jogo, formatos da competição, sanções, normas de segurança, regras de jogo, entre outras [9].

Para além da equipa 5DPO, existem várias outras que participam na competição MSL promovida pelo *RoboCup*, sendo de destacar as equipas que têm obtido melhores resultados nos últimos anos, mais concretamente, a equipa chinesa *Water*, atual campeã da mais recente edição, e a equipa holandesa *Tech United Eindhoven*, que tem repartido a liderança das mais recentes edições com a *Water* [10].

Também em Portugal existem equipas que participam nas competições, para além da já mencionada 5DPO, como a ISePorto, MINHO, SocRob e CAMBADA. É de realçar que esta última tem apresentado bons desempenhos a nível internacional, tendo alcançado o pódio nas últimas duas edições da prova [10].

Tal como grande parte das equipas de futebol robótico MSL, o sistema de visão dos robôs da equipa 5DPO é catadióptrico, tal como é possível constatar na figura abaixo, onde se verifica a câmara apontada ao espelho situado no topo da estrutura do robô.

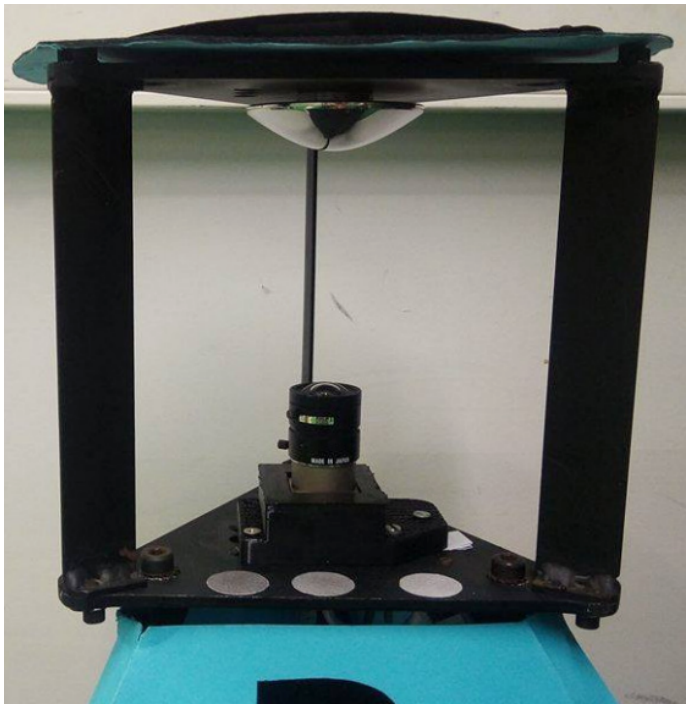


Figura 2.3: Sistema de visão catadióptrico presente nos robôs da equipa 5DPO [2].

Para além do *hardware* que equipa os robôs, também existe *software* implementado, que segue uma arquitetura distribuída que confere autonomia ao conjunto de robôs, tal como está esquematizado na figura seguinte.

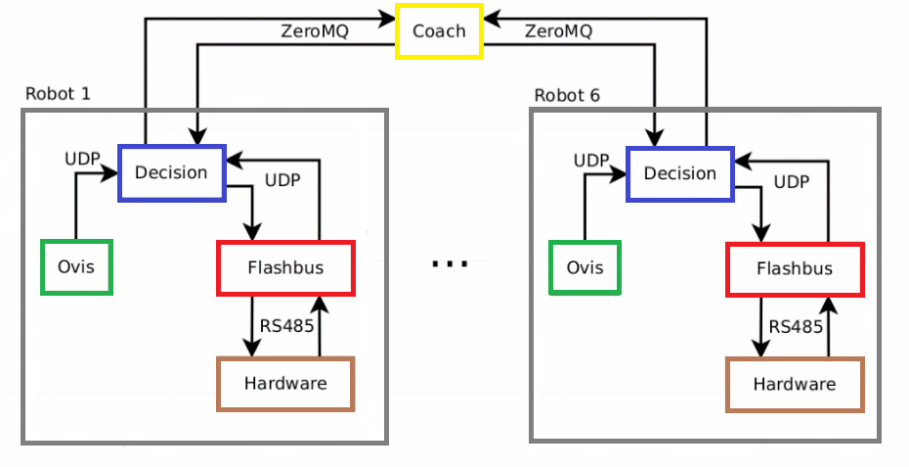


Figura 2.4: Arquitetura do *software* nos robôs da equipa 5DPO. Imagem adaptada de [2].

Como se observa acima, cada robô tem três aplicações que são executadas separadamente, mas que trocam dados entre si utilizando o protocolo UDP. O *Big Omni Vision* (Ovis) é o programa responsável pela aquisição e processamento de imagem. O principal objetivo deste *software* passa pela deteção de linhas de campo e objetos de interesse, como a bola e obstáculos, que podem ser os robôs adversários ou elementos externos ao jogo, como pessoas. Para se identificar as linhas, procede-se à deteção de transições verde-branco em linhas com direção radial, a partir do centro da imagem, tal como se verifica na figura abaixo [2].

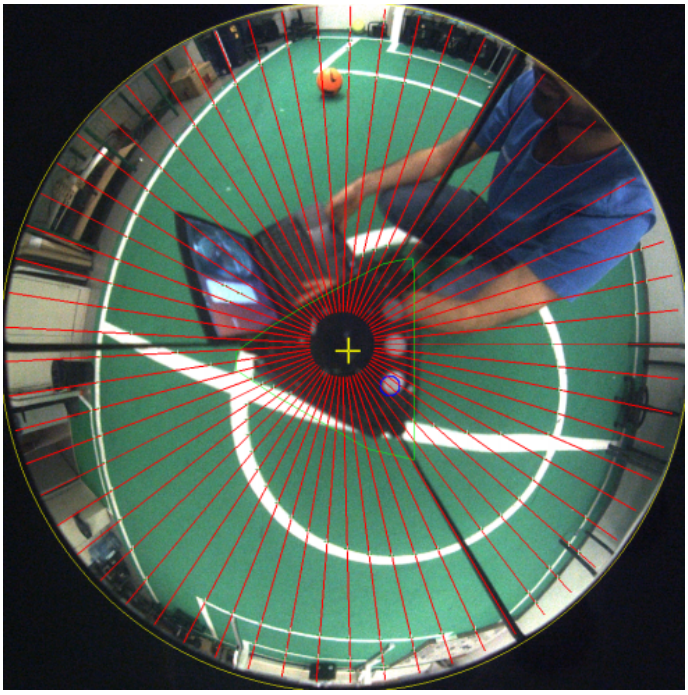


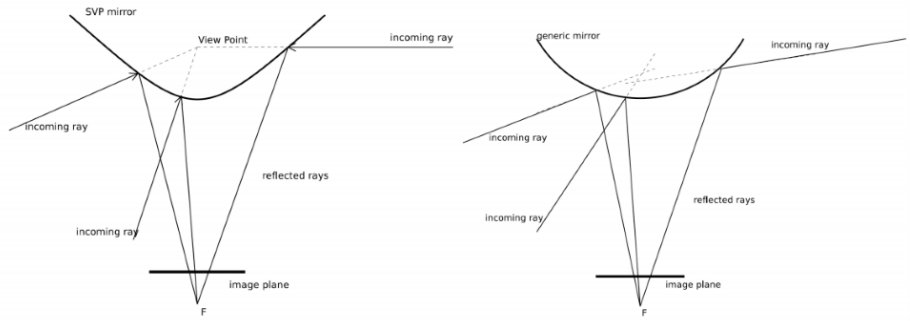
Figura 2.5: Imagem obtida através do programa *OVIS* nos robôs da equipa 5DPO [2].

2.2 Classificação de Sistemas de Visão Catadióptricos

Tal como referido anteriormente, os sistemas de visão catadióptricos são formados por um conjunto câmara-espelho, em que a câmara está apontada para um espelho côncavo e este está orientado para o plano a partir do qual se pretende obter a imagem.

O espelho pode apresentar diferentes formas como parabólica, hiperbólica, elíptica, cónica, planar, entre outras. Em [3] é adotado um método de classificação de sistemas de visão em função da tipologia do espelho utilizado no sistema. Assim, os sistemas catadióptricos podem ser classificados como *single view-point* (SVP) ou como *non-SVP*.

Nos sistemas SVP, as retas que projetam os raios incidentes no espelho, que sofrem reflexão para o plano da imagem, intersetam-se num único ponto no interior do espelho. Os únicos espelhos que apresentam estas características e podem ser classificados como SVP são os espelhos hiperbólicos, desde que acompanhados por uma câmara em perspetiva, ou espelhos paraboloides, em conjunto com uma câmara ortográfica. De seguida, apresenta-se uma imagem comparativa das duas categorias de sistemas.



(a) central sensor (b) non-central sensor

Figura 2.6: Sistema SVP (a) com raios a intersetarem-se no mesmo ponto, ao contrário do que acontece com o sistema non-SVP. Imagem obtida de [3].

Em termos de vantagens da utilização de cada sistema, com SVP é possível obter um mapeamento de cada ponto do mundo no plano da imagem, desde que a câmara e o espelho estejam perfeitamente alinhados, entre si e em relação ao referencial global. Por outro lado, os sistemas non-SVP não têm a restrição dos alinhamentos, não havendo limitações quanto à forma do espelho e da sua posição. Contudo, com esses sistemas não há um mapeamento através de uma função que transforme uma imagem omnidirecional numa em perspetiva.

No que concerne aos robôs da equipa 5DPO, à luz da classificação adotada em [3], o tipo de configuração é SVP, sendo que os espelhos têm forma hiperbólica.

Na prática, mesmo assumindo um sistema SVP, o alinhamento do conjunto câmara-espelho é difícil de concretizar mecanicamente, pelo que é necessária uma correta calibração que permita mapear a transformação não linear entre coordenadas na imagem e coordenadas no mundo.

2.3 Calibração de Sistemas de Visão

Fazendo equivaler projeções esféricas a sistemas catadióptricos, em [11] é apresentado um modelo matemático com o objetivo de determinar os parâmetros intrínsecos da câmara, como a posição relativa entre câmara e espelho. Este modelo de calibração está associado a vários tipos de sistemas SVP, tal como uma câmara ortográfica em conjunto com um espelho parabólico, ou então uma câmara de perspetiva associada a espelhos hiperbólicos, elípticos ou planos.

Por sua vez, em [12], é apresentada uma calibração utilizada pela equipa CAMBADA em que foi feita uma calibração tendo em conta o modelo matemático de visão através de reflexão no espelho. A partir da distância focal da câmara e da equação do espelho, é possível calcular o raio que é refletido do espelho para a câmara. Assim, sabendo o ponto em que o raio incide no espelho, calcula-se o vetor normal a um plano tangente ao espelho nesse mesmo ponto. O ângulo entre este vetor e o raio que é refletido no espelho, é utilizado para fazer uma projeção no mundo, que permite determinar o ponto correspondente ao pixel da imagem. Neste método tem-se em conta desalinhamentos entre a câmara e o espelho que podem causar erros de cálculo nas projeções calculadas. Como tal, é utilizada uma matriz de transformação composta por três matrizes de rotação e um vetor de translação.

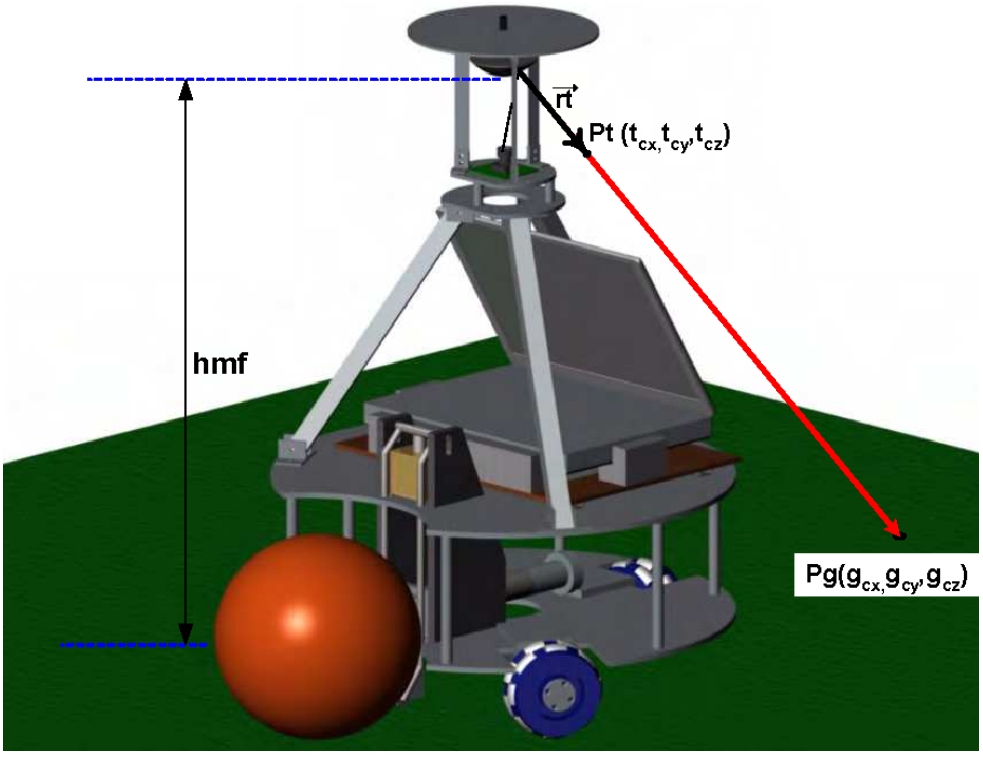


Figura 2.7: Estudo do modelo da propagação do raio de reflexão no espelho. Imagem obtida de [4].

Seguindo o mesmo princípio, mas em ambiente de simulação, [13] mostra como foi modelado o sistema de visão e como foi feita a simulação para a calibração, através dos dados extraídos das

linhas do campo.

Em [14] é utilizada uma solução não paramétrica que implementa um algoritmo de *Levenberg-Marquardt* Dividido, que permite a otimização dos parâmetros de uma função de mapeamento entre as coordenadas na imagem e as coordenadas no mundo. Em comparação com outras soluções para calibração de câmaras que implementam o algoritmo de *Levenberg-Marquardt*, neste caso a solução apresentada permite reduzir a complexidade de N^3 para N caso a matriz Jacobiana utilizada seja esparsa.

Por outro lado, em [15] é apresentada uma solução que faz uso de redes neuronais no processo de calibração. Através da recolha de algumas amostras da posição de pontos no mundo, com o auxílio de padrões de xadrez, os correspondentes valores dos pixéis na imagem são determinados através de um algoritmo de deteção de cantos de *Harris*. Depois de recolhidos alguns pares de pontos 3D e a sua correspondente projeção 2D, no plano da imagem, esses dados são utilizados no treino da rede neuronal.

Ainda no tópico das redes neuronais, [5] apresenta um método semelhante ao anterior, mas adaptado para os robôs de futebol robótico da equipa 5DPO, a mesma plataforma de testes a utilizar nesta dissertação.

Figura 2.8: Padrões de xadrez usados para recolher dados para treino de redes neuronais. Imagem obtida de [5].

Por fim, em [16], é utilizada uma abordagem diferente, centrada no projeto do espelho. Recorrendo a ferramentas de simulação 3D, são projetados vários espelhos com diferentes curvaturas e,

para cada um, é simulada a imagem projetada pelo mesmo. Posteriormente, é feita uma comparação das diferentes imagens obtidas e, para cada região, é determinada o espelho que produz a melhor imagem. No final é obtida uma solução que resulta da integração das melhores soluções de curvaturas para cada região. Para além disso, também foi estudada a forma como a cabeça do robô seria montada, de forma a reduzir as oclusões no espelho devido aos mecanismos responsáveis pela sua fixação.

2.4 Caraterização do Problema

Na sequência da revisão bibliográfica efetuada nas secções anteriores segue-se, inicialmente, uma definição da problemática que orienta esta dissertação, seguida da proposta de solução que a visa solucionar.

2.4.1 Definição do problema

Tendo em conta as motivações e objetivos que orientam a dissertação, somos agora capazes de definir o problema e apresentar uma proposta de solução para o mesmo.

Assim, dada a importância dos sistemas de visão omnidirecional em aplicações com robôs móveis, em particular o caso do futebol robótico, é pertinente que haja uma correta calibração do conjunto câmara-espelho, para que a informação extraída possa ser utilizada de forma fiável. Tal como se verificou, existem várias formas de solucionar este problema, porém, nem todas se adequam a este contexto em particular, por várias razões. Entre elas está, por exemplo, o facto de o conjunto câmara-espelho ter desalinhamentos e assimetrias, resultantes não só de uma possível má instalação mecânica dos equipamentos, mas também decorrentes dos choques entre robôs durante as partidas ou do embate da bola.

Deste modo, o problema passa por estudar e implementar um processo de calibração do sistema de visão catadióptrico presente nos robôs da equipa de futebol MSL 5DPO.

2.4.2 Solução proposta

Na sequência do que foi constatado anteriormente, verifica-se que uma calibração não paramétrica se afigura como a solução mais adequada nesta aplicação. Assim, para conseguir o mapeamento correto entre os pixéis na imagem 2D e os correspondentes pontos no mundo 3D, deverá ser usada uma função matemática que faça essa transformação, recebendo como entradas a posição dos pixéis detetados na imagem e que na saída indique a posição correspondente na realidade.

Portanto, a solução proposta terá de passar por diferentes fases, desde a determinação da função matemática que mais se adequa para fazer o mapeamento imagem-mundo, até à utilização de um método de otimização que permita encontrar a solução ótima para os parâmetros dessa função, passando ainda pelo processo de medição de alguns pontos a distâncias fixas do robô e correspondentes pixéis na imagem para auxiliar no ajuste dos parâmetros da função.

Capítulo 3

Função de Mapeamento 2D-3D

Neste capítulo começa-se por especificar o contexto em que a solução proposta se insere e são detalhados alguns conceitos que permitem a sua execução. De seguida é mostrado como foi realizado um ensaio e todas os preparativos necessários para recolher amostras do sistema. Por fim, são retiradas algumas conclusões dos resultados obtidos.

3.1 Enquadramento do Método Proposto

Tendo em conta a solução proposta para este tema de dissertação, o plano de trabalhos passa, numa fase inicial, por analisar os pontos detetados na imagem, e verificar se existe alguma relação matemática evidente que permita fazer a correspondência desses pontos a distâncias no mundo. Para conseguir fazer essa análise, era determinante a amostragem de algumas distâncias, em relação ao robô, conhecidas a priori.

Para proceder à amostragem, foi necessário escolher e promover uma forma de marcar os pontos no mundo. Como tal, a solução passaria por marcar transições de cores no ambiente de testes, neste caso o chão do campo de futebol robótico, uma vez que o *software* implementado nos robôs, em particular o programa *Ovis*, já permitia a deteção de vários tipos de transições de cores: verde-branco, branco-verde e verde-branco-verde. Entre estas opções, a escolha recaiu sobre produzir uma forma de originar a deteção de transições verde-branco.

Assim, de seguida é explicado mais detalhadamente como a imagem é obtida nesse mesmo *software* e como cada pixel é identificado.

No *Ovis* a imagem adquirida tem um forma retangular com uma resolução de 962×962 pixéis. Por convenção, estabeleceu-se que a imagem é representada como uma matriz de pixéis e definida por dois eixos perpendiculares segundo direções denominadas x e y. Desta forma, cada pixel é caracterizado por uma abcissa e uma ordenada, sendo que o pixel no canto superior esquerdo corresponde à origem e tem como coordenadas (0, 0). De seguida segue uma figura que ilustra as características explicadas.

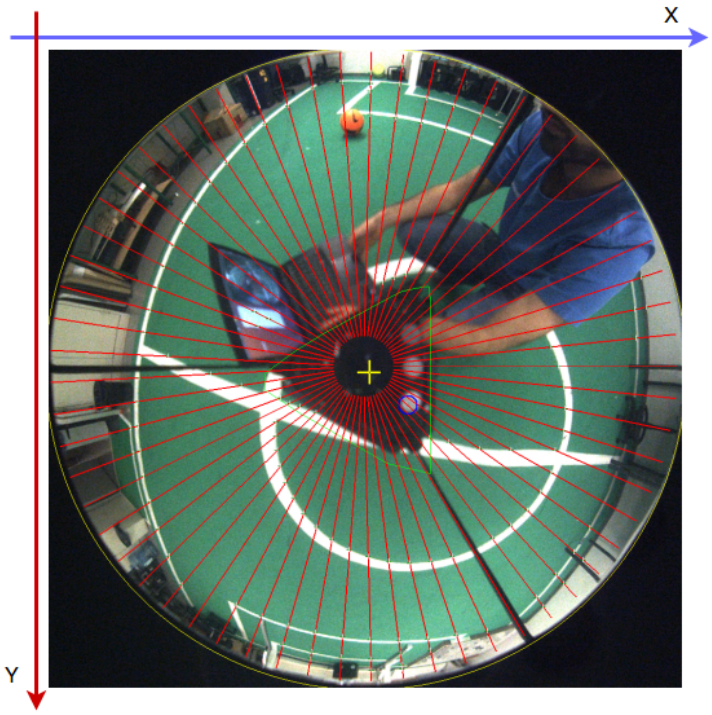


Figura 3.1: Imagem capturada no *Ovis* identificada pelos eixos. Imagem adaptada de [2].

Por outro lado, é de destacar que, devido à forma geométrica circular do espelho, ao qual a câmara está apontada, a imagem apresenta zonas exteriores mais escuras que correspondem à parte superior do robô, onde o espelho está fixado. O topo do robô é suportado por três hastes refletidas no espelho e que são visíveis na figura 3.1. Na região central da imagem é também refletido o próprio robô e a câmara, tal como se verifica. As regiões mencionadas não são relevantes e por norma são ignoradas, apesar de ocuparem uma parte significativa da imagem, o que constitui uma das desvantagens dos sistemas de visão catadióptricos. Para além disso, a imagem é ainda caraterizada pelo seu centro, marcado pela cruz assinalada a cor amarela.

Como foi mencionado, o *software Ovis*, através de análise e processamento de imagem, permite determinar, ao longo de raios, desenhados a cor vermelha na figura, os pixéis onde é encontrada uma transição de cores. Assim, idealizou-se marcar tiras de veludo com transições, que seriam colocadas no chão em torno do robô, com a vantagem de que poderiam ser reutilizadas em diferentes robôs e noutras ocasiões, sem que fosse necessário proceder à marcação do campo de futebol robótico de cada vez que fossem necessárias.

Assim sendo, uma vez obtidas as tiras, optou-se pela marcação de transições de cores com um intervalo de cinco graus entre cada transição, em relação ao ângulo que é feito com a vertical do robô, de ora em diante definido por ρ.

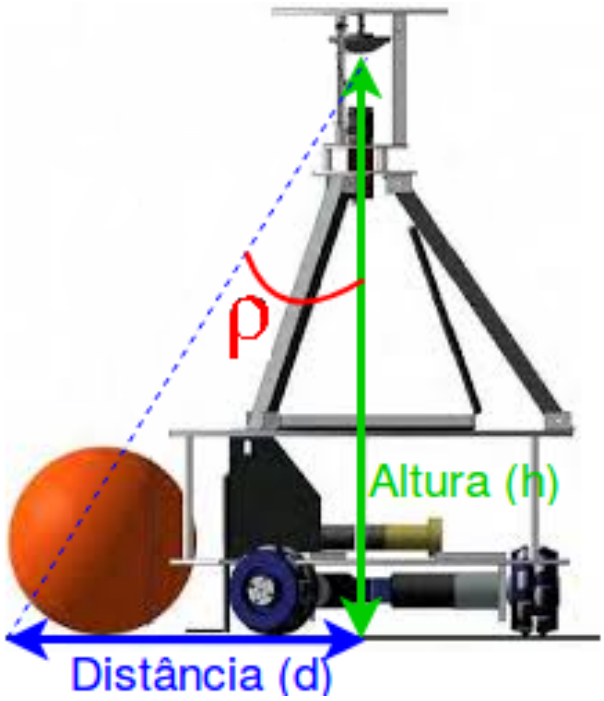


Figura 3.2: Ângulo ρ em relação à vertical do robô, considerado para a marcação das transições nas tiras. Imagem adaptada de [4].

Da imagem disponibilizada acima resultam algumas convenções adotadas para a marcação das tiras, desde logo o ângulo ρ, já mencionado, a altura, medida na vertical entre o ponto inferior do espelho e o solo, e ainda, a distância de qualquer ponto no meio envolvente do robô, calculada a partir da projeção no solo do ponto inferior do espelho. Neste contexto, é possível extrapolar a relação trigonométrica:

$$\tan(\rho) = \frac{d}{h} \tag{3.1}$$

3.2 Conceção do Método de Mapeamento

A partir da expressão matemática anterior, e uma vez feita a medição da altura do robô, que neste caso corresponde a $76,5$ centímetros, somos capazes de calcular a distância em relação ao centro do robô a que cada marcação na tira deverá ser feita. Antes disso, foram tomadas em conta algumas considerações, como por exemplo os limites de distância possíveis, máximo e mínimo. Assim, estipulou-se que, devido ao comprimento do veludo e do espaço disponível no campo de futebol robótico da faculdade, a transição mais longínqua não deveria ser marcada a uma distância superior a três metros. Para além disso, verificou-se que as transições mais próximas do robô ficavam ocultadas da imagem, devido à reflexão do próprio robô no espelho, criando assim uma zona de ângulo morto. Tendo em conta estas condicionantes, e uma vez que para ρ igual a 20^{o}

resulta uma distância de cerca de 27 centímetros, que não é visível na imagem, enquanto que um ângulo ρ de 80^o corresponde a uma distância que ultrapassa os 3 metros, definiu-se que o intervalo de transições a marcar seria entre 25^o e 75^o. Na tabela 3.1 estão discriminadas as transições a marcar.

ρ (em graus)	Distância (metros)
25	0.357
30	0.442
35	0.536
40	0.642
45	0.765
50	0.912
55	1.093
60	1.325
65	1.641
70	2.102
75	2.855

Tabela 3.1: Ângulo e distância correspondente de cada transição a marcar.

Seguindo todas as especificidades mencionadas, o resultado final da construção das tiras está presente na figura seguinte.

Figura 3.3: Resultado final da tira com as marcações feitas.

3.3 Ensaio com a Metodologia Proposta

Nesta fase, tendo sido elaborada a tira de calibração, o próximo passo passaria pela sua colocação no robô para correr o programa que permitisse obter as transições, para futura análise. Porém, nesse processo constatou-se existir um desalinhamento acentuado do conjunto câmara-espelho com a consequência de introduzir distorções na imagem obtida.

3.3.1 Trabalho prévio

Assim, uma vez que o objetivo nesta fase é a análise da relação matemática entre a imagem e a realidade, seria conveniente fazer essa análise com o mínimo possível de ruído a afetar os resultados, ou seja, um desalinhamento pronunciado do centro do espelho com a câmara altera a reflexão do espelho ao ponto de poder invalidar essas conclusões.

Como tal, antes de prosseguir, tentou reduzir-se ao máximo os desalinhamentos mecânicos. Para o efeito, o espelho de cada robô foi removido e, com o auxílio de um torno mecânico, foi demarcado o centro, que corresponde ao ponto inferior, quando aplicado no robô.

Figura 3.4: Espelho após a marcação do centro.

De seguida, procedeu-se à montagem do espelho novamente no robô. Através da observação da imagem obtida no programa, as posições foram ajustadas até que a marcação no espelho estivesse alinhada com a reflexão da câmara.

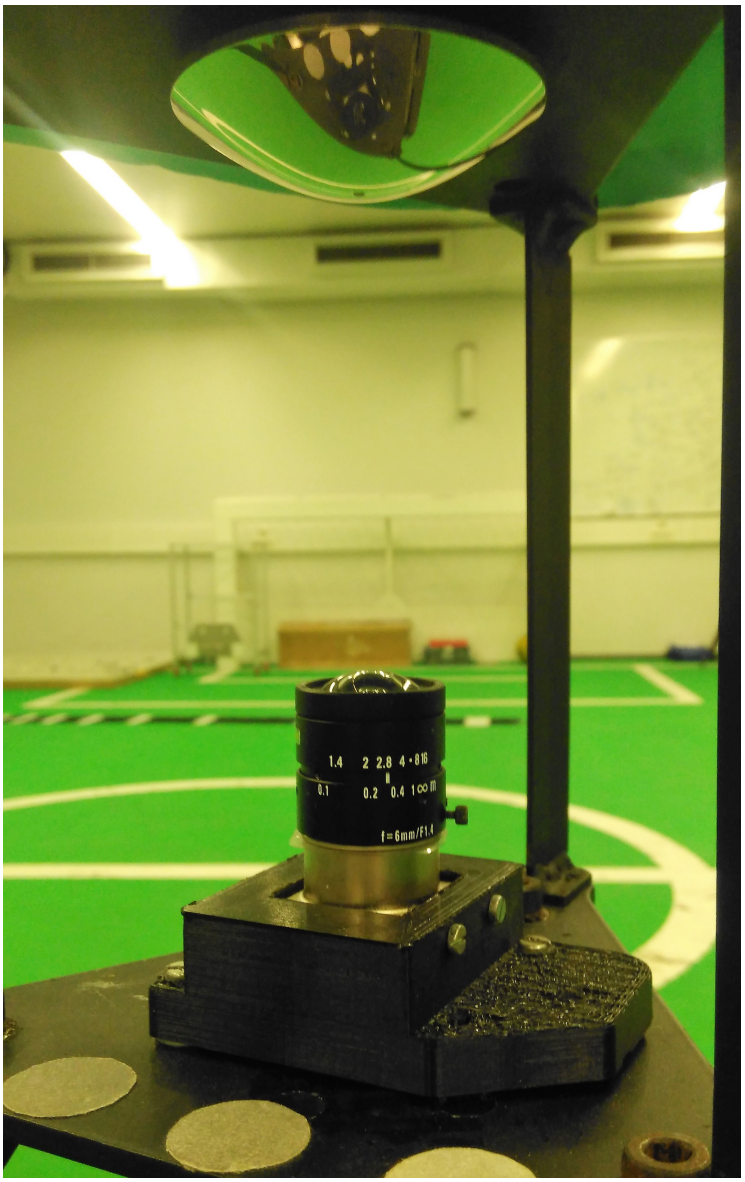


Figura 3.5: Conjunto câmara-espelho após o ajuste do alinhamento.

Uma vez minimizados os efeitos das imprecisões mecânicas, embora não de forma definitiva, dado que os robôs continuam sujeitos a colisões e choques com a bola em situações de jogo que podem desajustar o alinhamento do sistema de visão, era possível partir para a recolha de amostras, feita com os robôs imobilizados pelo que o sistema se manteria afinado, tal como se pretendia.

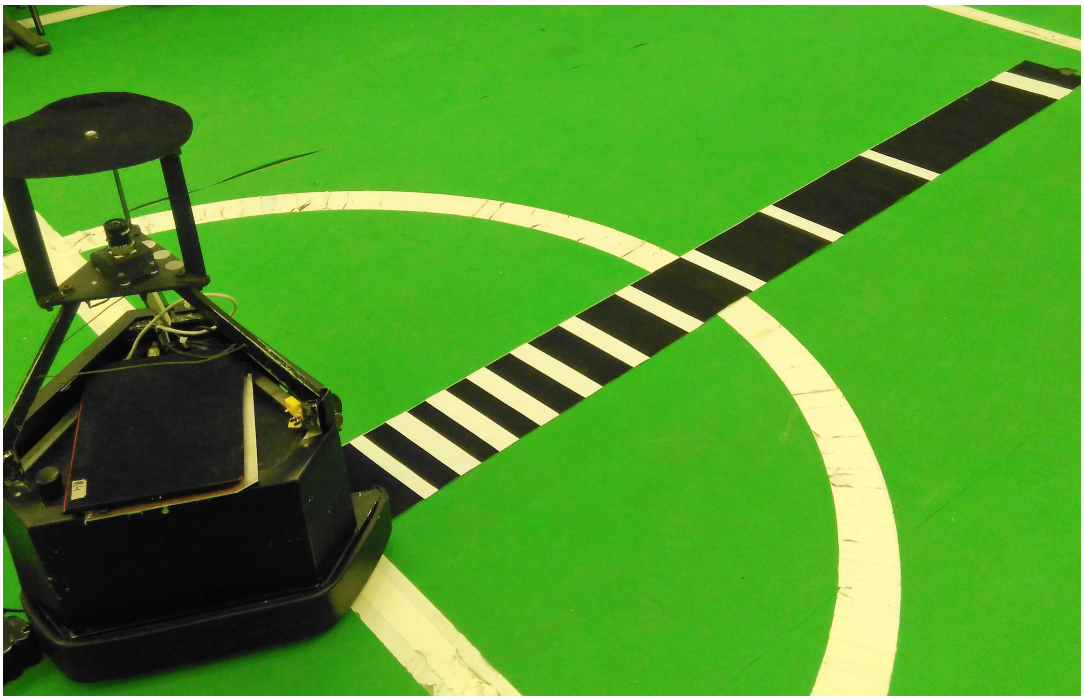


Figura 3.6: Tira posicionada no robô.

Deste modo, procedeu-se à disposição da tira construída na zona frontal do robô, tal como é mostrado em 3.6, e executou-se o *software Ovis* associado à visão. Como tal, foi necessário configurar algumas definições. Desde logo, os raios traçados na imagem e ao longo dos quais são encontradas as transições. Assim, utilizou-se apenas um raio, na direção da tira, e definiu-se o comprimento do raio para apenas ficar sobreposto sobre a tira. Com este processo evitava-se a deteção de transições do ambiente circundante, irrelevantes nesta fase.

Por outro lado, tal como foi dito anteriormente, a funcionalidade implementada no programa de deteção de transições apenas permite detetar transições de cores entre verde e branco. Por força da tira disponível alternar entre preto e branco, alterou-se no *Ovis* a representação HSV da cor verde para a representação correspondente à cor preta. Assim sendo, o programa passou a interpretar o preto como verde e foi possível detetar as transições.

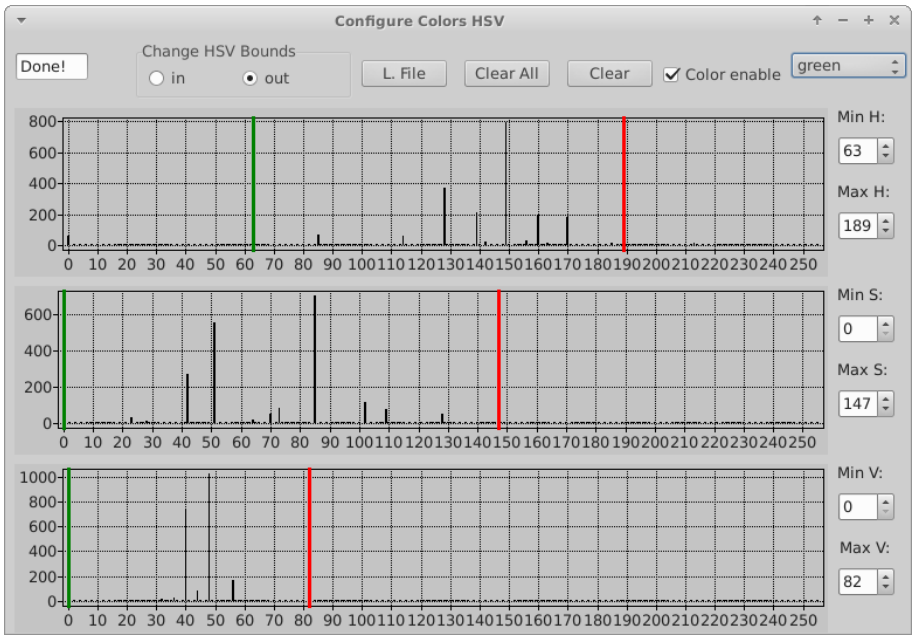


Figura 3.7: Representação HSV da cor preta interpretada como cor verde.

Na figura seguinte, referente a uma imagem pós-processamento, observa-se que a tira é interpretada como sendo verde e branca.

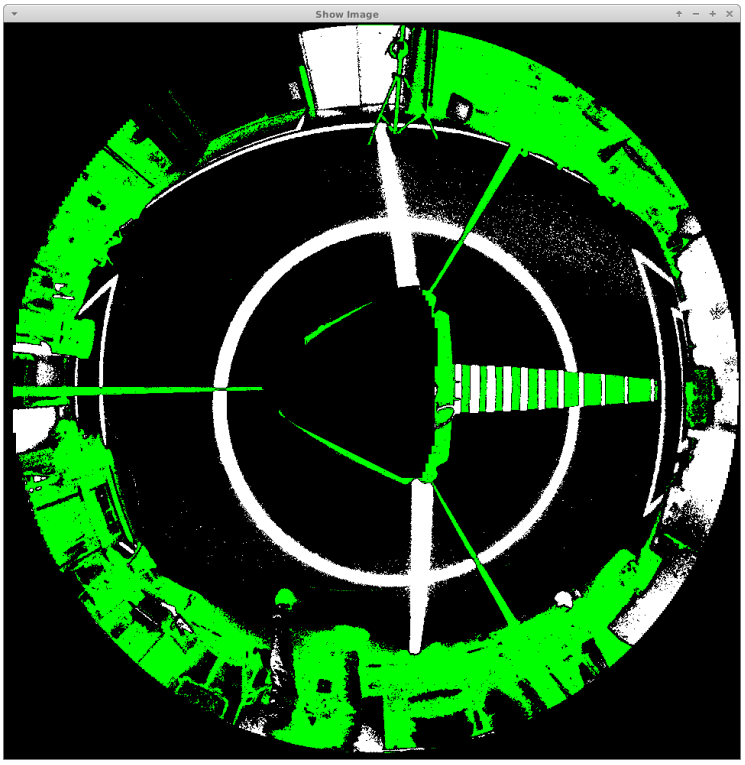


Figura 3.8: Imagem processada após alterar a representação HSV.

Após os procedimentos referidos, o programa é, por fim, capaz de detetar as transições na tira, como é evidenciado na seguinte figura.

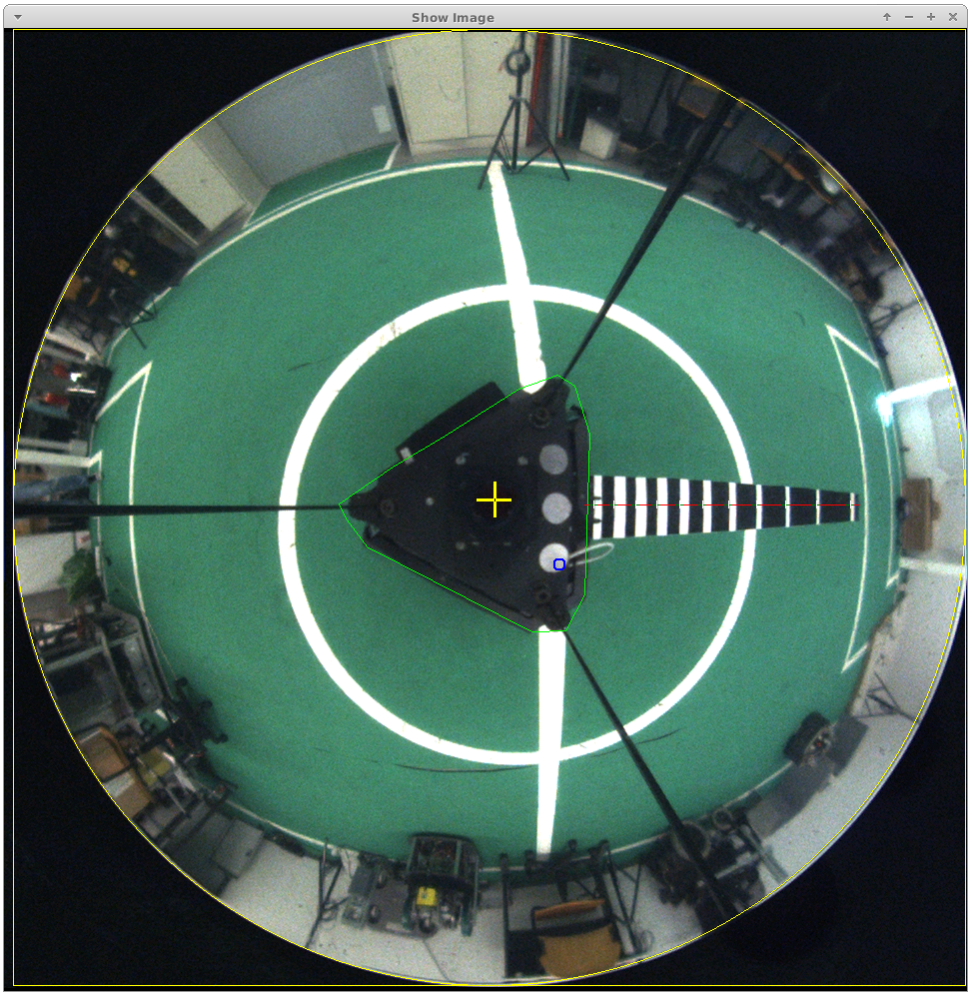


Figura 3.9: Raio sobreposto sobre a tira com transições detetadas.

3.3.2 Recolha de amostras

Finalmente, no atual estádio de trabalho, fez-se no programa *Ovis* a recolha manual das características das transições que se pretendia detetar, nomeadamente as coordenadas dos pixéis na imagem. Na tabela seguinte seguem os dados recolhidos para cada transição.

ρ (em graus)	Distância (metros)	X (pixéis)	Y (pixéis)
25	0.357	582.2	477
30	0.442	603.1	477
35	0.536	624.7	477
40	0.642	646.5	477
45	0.765	669.1	477
50	0.912	692.6	477
55	1.093	719.2	477
60	1.325	746.6	477
65	1.641	776.5	477
70	2.102	806.9	477
75	2.855	841.3	477

Tabela 3.2: Coordenadas na imagem de cada transição detetada.

Da mesma forma que se definiu que, no mundo 3D, a distância era medida em relação a um referencial centrado na projeção do ponto inferior do espelho no solo, também neste caso se convencionou que a distância em termos de pixéis seria definida em relação ao ponto da imagem 2D correspondente ao centro do espelho. Daqui resulta que a distância de qualquer ponto da imagem ao centro é definida por uma relação geométrica simples, expressa na equação matemática:

$$d_{px} = \sqrt{(x_{px} - x_c)^2 + (y_{px} - y_c)^2} \tag{3.2}$$

em que d_{px} é a distância ao centro da imagem em pixéis, x_{px} e y_{px} definem as coordenadas na imagem de cada transição e, finalmente, x_c e y_c estabelecem as coordenadas do centro da imagem, como é aclarado na seguinte imagem.

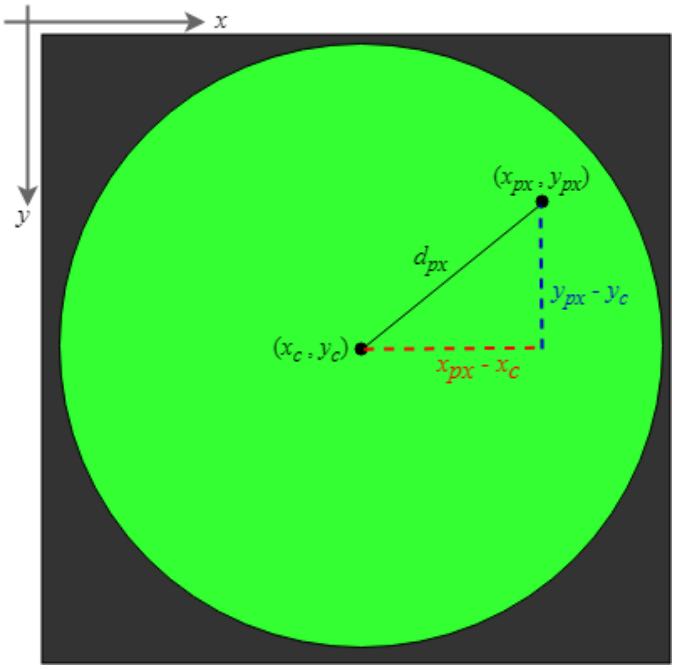


Figura 3.10: Distância, em pixéis, ao centro da imagem, de qualquer ponto (x_{px}, y_{px}).

Sabendo que, neste ensaio, o centro tem como coordenadas (477, 477) e aplicando na equação 3.2 os valores das coordenadas de cada transição, consegue-se extrapolar os valores das distâncias de cada marca na imagem.

ρ (em graus)	Distância (metros)	Distância (pixéis)
25	0.357	105.2
30	0.442	126.1
35	0.536	147.7
40	0.642	169.5
45	0.765	192.1
50	0.912	215.6
55	1.093	242.2
60	1.325	269.6
65	1.641	299.5
70	2.102	329.9
75	2.855	364.3

Tabela 3.3: Distância em pixéis ao centro da imagem para cada transição.

Partindo da tabela 3.3 estão reunidas as condições para a análise da função de mapeamento que relaciona a imagem obtida bidimensional com a realidade tridimensional. Com efeito, inserindo os valores numa folha de cálculo podem ser traçados gráficos de dispersão que permitem avaliar qual das relações é mais apropriada. Nesse sentido, surgem duas possibilidades, ângulo ρ em

função da distância na imagem ou, em alternativa, distância em metros em função da distância na imagem. Os traçados para cada um desses casos serão analisados na secção seguinte.

3.4 Análise de Funções de Mapeamento

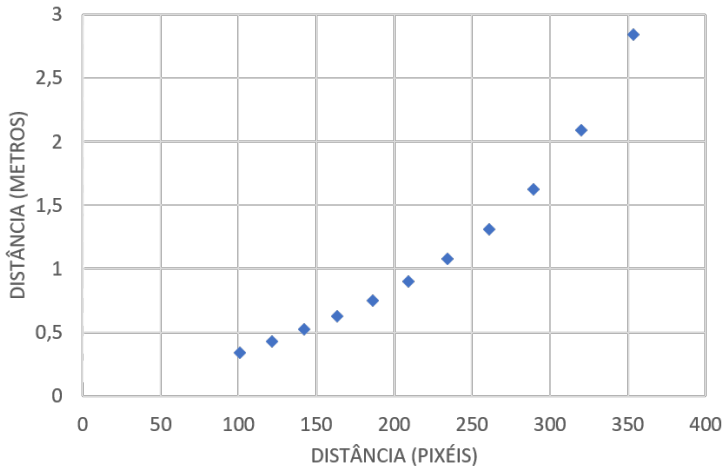


Figura 3.11: Distância na realidade em metros em função da distância em pixéis ao centro da imagem.

Como é possível constatar, se o mapeamento da distância em pixéis for feito para a distância em metros, o comportamento da curva que se ajusta aos pontos carateriza-se por um aumento progressivo do declive conforme aumenta a distância do robô, o que evidencia o comportamento de uma função não linear.

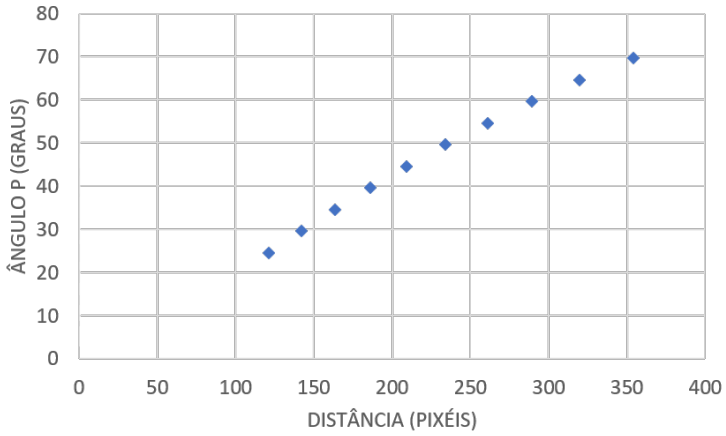


Figura 3.12: Ângulo ρ em graus em função da distância em pixéis ao centro da imagem.

Por outro lado, ao analisar o traçado da relação do ângulo com a distância na imagem, observa-se que, conforme a distância ao robô cresce, a tendência dos pontos é caraterizada também por um aumento, mas com declive aproximadamente constante, o que implica que entre as duas grandezas em causa há também uma relação não linear, mas mais próxima de uma reta.

Logo, com base nos resultados e nas ilações tiradas, estabeleceu-se que o mapeamento 2D-3D deverá ser feito entre a distância na imagem em pixéis e o ângulo ρ, sendo que uma função polinomial seria uma boa candidata para definir a função matemática dessa transformação, devido à não linearidade do sistema.

Capítulo 4

Modelo e Sistema de Calibração

Ao longo do presente capítulo é mostrado com é feita a calibração e em que direções. Segue-se a explicação do algoritmo de otimização escolhido. De seguida, procede-se à explicação da parte de *software* desenvolvida e a respetiva integração nos programas já implementados nos robôs. Por fim, são sintetizados os procedimentos para realizar a calibração.

4.1 Direções de Calibração

Em relação à deteção de características pelo sistema de visão, até este momento apenas foi abordada a distância em relação ao robô, que resulta da conversão do ângulo ρ, a partir de 3.1, resultando em

$$d = \tan(\rho) \times h \tag{4.1}$$

Porém, conhecendo apenas a distância não é possível saber a localização concreta dos elementos detetados pelo robô, como a posição da bola, pois para um determinado valor de distância d a posição é redundante, podendo ser qualquer uma na circunferência de raio d e centrada no robô. Como tal, é necessário definir outro grau de liberdade, neste caso o ângulo em relação ao robô visto do topo. Este ângulo é definido como θ e varia entre -180º e 180º, sendo que a direção de 0º corresponde à linha radial do centro para a parte frontal do robô. Deste modo, a posição dos elementos, como a bola, passa a ser dada por

$$\begin{aligned} x_r &= \cos(\theta) \times d \\ y_r &= \sin(\theta) \times d \end{aligned} \tag{4.2}$$

em que x_r e y_r representam as coordenadas em relação ao referencial centrado no robô.

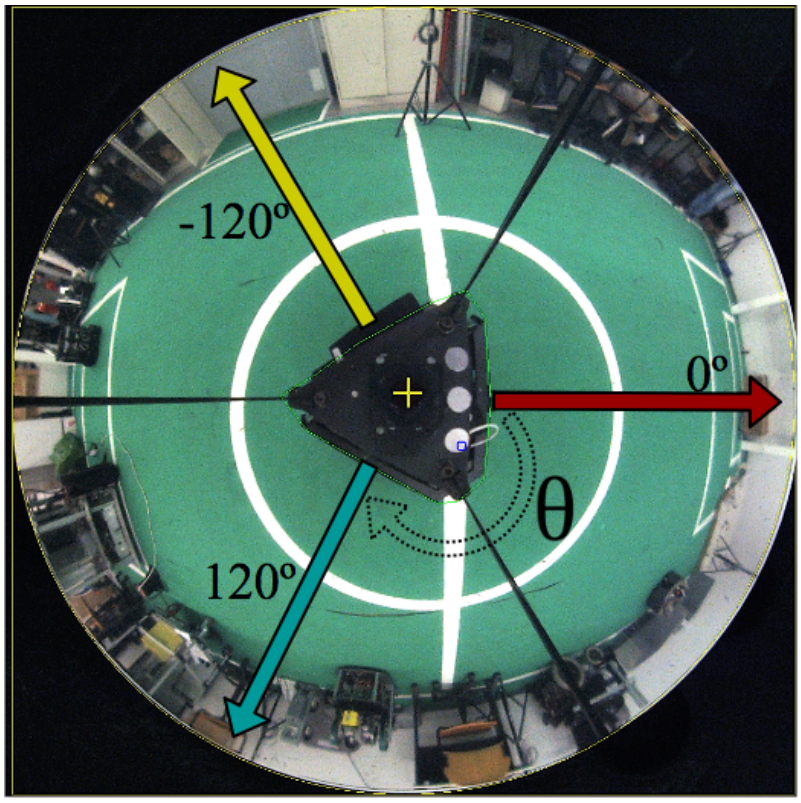


Figura 4.1: Ângulo θ, em graus, em torno do robô e direções onde serão colocadas as tiras.

Tendo em conta esta condição, e uma vez que, sempre que no futuro seja necessário fazer uma calibração usando o método proposto nesta dissertação, o alinhamento mecânico entre o espelho e a câmara pode não ser garantido, definiu-se fazer o mapeamento 2D-3D em regiões distintas em torno do robô.

Assim, a recolha de pontos de transições será feita ao longo de três direções, igualmente espaçadas entre si como na figura 4.1, e o método de otimização permitirá encontrar três funções de regressão com parâmetros distintos, sendo que a função a utilizar para transformar qualquer ponto da imagem na sua posição na realidade resultará da fusão destas três calibrações, tal como será referido adiante.

Nesse sentido, reproduziram-se mais duas tiras, semelhantes à feita anteriormente 3.3, com o mesmo número de transições marcadas às mesmas distâncias, tal como é apresentado na figura seguinte.

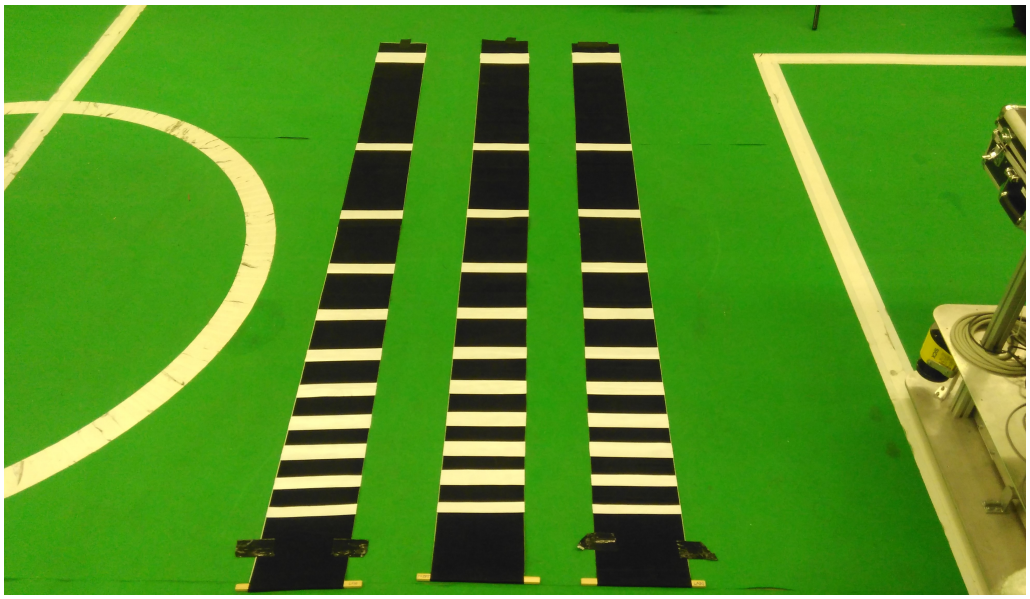


Figura 4.2: Conjunto de tiras a usar nas diferentes direções, com as marcações às mesmas distâncias.

4.2 Algoritmo de Otimização

Na sequência das ilações retiradas no capítulo anterior, a opção pelo método de ajuste de uma curva a um conjunto de pontos pressupõe a utilização de um método de análise de regressão. Uma vez que a função é não linear, a soma dos mínimos quadrados deverá ser minimizada por uma técnica iterativa.

Nesse sentido, elegeu-se o algoritmo de Levenberg-Marquardt como o processo iterativo que permitiria estimar os parâmetros de uma função polinomial que fizesse a correspondência entre distâncias na imagem e ângulo na realidade.

4.2.1 Levenberg-Marquardt

Tal como foi mencionado, este algoritmo é utilizado na resolução de problemas não lineares. Neste secção vai ser explicado o método e como são obtidas as soluções, com base em [17–28].

Assim, dado um conjunto de m pontos (x_i, y_i), aos quais se pretende ajustar uma curva, o objetivo é estimar os parâmetros β da função de regressão $f(x, \beta)$, tal que a soma dos quadrados dos desvios $S(\beta)$ seja minimizada, tal como é expresso na seguinte equação.

$$\hat{\beta} = \quad \text{argmin}_{\beta} S(\beta) \equiv \quad \text{argmin}_{\beta} \sum_{i=1}^{m} [y_i - f(x_i, \beta)]^2 \tag{4.3}$$

Na equação 4.3 está formulado o problema. A respetiva solução é obtida através de um algoritmo, tal como é descrito a seguir.

Inicialmente, é definida uma estimativa inicial dos parâmetros $\hat{\beta}$. Em cada passo do algoritmo é somado um valor δ à estimativa anterior, passando $\beta+\delta$ a ser a estimativa atual. Assim que os critérios de paragem sejam correspondidos, o processo iterativo termina.

O valor de δ é obtido seguindo um conjunto de princípios que caracterizam este algoritmo. Assim, aproximando a função $f(x_i,\beta+\delta)$ pela sua linearização tem-se que:

$$f(x_i,\beta+\delta) \approx f(x_i,\beta)+\mathrm{J}_i\delta \tag{4.4}$$

em que J_i corresponde ao gradiente de f em ordem a β, formulado em 4.5.

$$\mathrm{J}_i = \frac{\partial f(x_i,\beta)}{\partial \beta} \tag{4.5}$$

Partindo da linearização feita em 4.4, a soma do quadrado dos desvios $S(\beta+\delta)$ é dada por:

$$S(\beta+\delta) \approx \sum_{i=1}^{m}[y_i - f(x_i,\beta)-\mathrm{J}_i\delta]^2 \tag{4.6}$$

o que em notação matricial corresponde a

$$[y-f(\beta)]^T[y-f(\beta)]-2[y-f(\beta)]^T\mathrm{J}\delta+\delta^T\mathrm{J}^T\mathrm{J}\delta \tag{4.7}$$

sendo J a matriz Jacobiana. Sabendo que, a soma do quadrado dos desvios $S(\beta)$ é mínima quando o gradiente é nulo em relação a β, então, igualando a zero a derivada de 4.7 em ordem a δ e reordenando a equação tem-se:

$$(\mathrm{J}^T\mathrm{J})\delta \quad = \quad \mathrm{J}^T[y-f(\beta)] \tag{4.8}$$

onde a linha i da matriz Jacobiana corresponde ao gradiente J_i, enquanto o elemento i dos vetores y e f corresponde a y_i e $f(x_i,\beta)$, respetivamente. Este é um conjunto de equações lineares a partir de onde é possível extrapolar δ.

Ora, a partir de 4.8, Levenberg incluiu um fator de amortecimento λ não negativo, resultando na seguinte equação

$$(\mathrm{J}^T\mathrm{J}+\lambda\mathrm{I})\delta \quad = \quad \mathrm{J}^T[y-f(\beta)] \tag{4.9}$$

onde I representa a matriz identidade. Este fator λ, ajustado a cada iteração, influencia o comportamento do algoritmo. Se S decresce rapidamente, o fator de amortecimento pode ter um valor mais baixo, aproximando o algoritmo ao método de Gauss-Newton. Por outro lado, se a

redução de S for insuficiente, um valor mais alto de λ leva a um passo na direção do gradiente descendente. O algoritmo para na condição do valor de δ ser inferior a um limite pré-estabelecido ou se a redução de S for inferior à desejada. Neste caso, o último valor de β é adotado como a solução final.

Não obstante, a desvantagem de 4.9 surge quando o fator de amortecimento toma valores muito elevados, o que faz com que não seja possível inverter $\mathrm{J}^T\mathrm{J} + \lambda\mathrm{I}$. Então, percebendo que seria possível escalar cada componente do gradiente de acordo com a curvatura, para levar a um maior movimento nas direções onde o gradiente é menor, Marquardt substituiu a matriz identidade em 4.9 pela matriz diagonal de $\mathrm{J}^T\mathrm{J}$, obtendo

$$[\mathrm{J}^T\mathrm{J} + \lambda\,\mathrm{diag}(\mathrm{J}^T\mathrm{J})]\delta = \mathrm{J}^T[y - f(\beta)] \tag{4.10}$$

dando origem, deste modo, ao algoritmo de Levenberg-Marquardt, que pode ser sintetizado pelo seguinte processo iterativo, em que i representa a iteração atual:

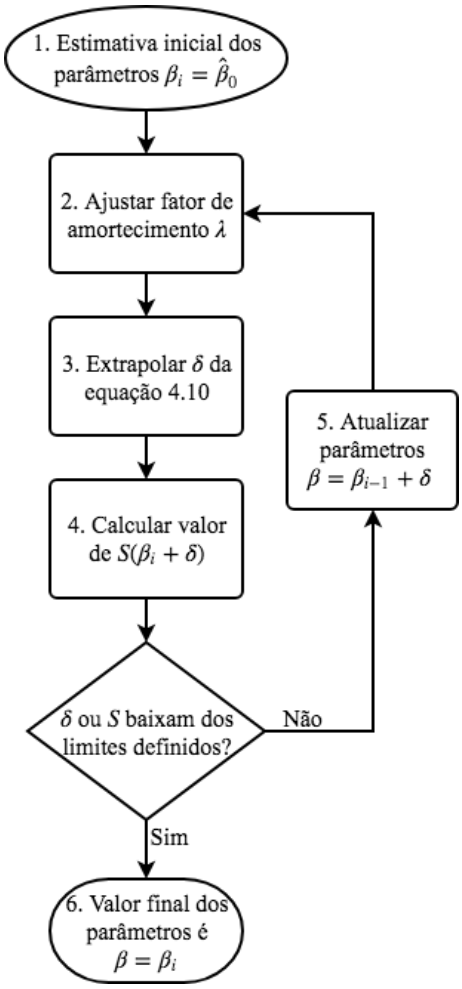


Figura 4.3: Algoritmo de Levenberg-Marquardt representado num fluxograma.

É importante salientar que na escolha do valor inicial dos parâmetros β_0, poderá ser utilizado qualquer valor, desde que haja apenas um mínimo. Porém, numa situação de múltiplos mínimos, o algoritmo só converge para o mínimo global se a estimativa inicial já for um valor próximo do final.

Por outro lado, o ajuste de λ no ponto 2 também influencia o desempenho do algoritmo. Existem algumas técnicas heurísticas que visam auxiliar na escolha do fator de amortecimento, que garantem a convergência local do algoritmo. Porém, a convergência global do algoritmo tem desvantagens associadas, como a morosidade do processo perto da solução ótima.

Nesse sentido, Marquardt forneceu algumas recomendações para a atualização de λ. Deverá começar-se pela escolha de um valor λ_o e outro fator $v > 1$. Após uma iteração, calcula-se o valor da soma do quadrado dos desvios para o vetor de parâmetros atualizado com $\lambda = \lambda_0/v$ e com $\lambda = \lambda_0$. Assim, se o primeiro caso resultar numa redução de S, então λ/v é considerado o novo valor de λ. Por outro lado, se λ/v não resultar numa melhoria, mas λ sim, então o fator de amortecimento mantém-se e o algoritmo continua. Por fim, caso nenhuma das duas alternativas produza um resultado melhor, o amortecimento é aumentado, multiplicando-o por v até obter uma melhoria em S, resultando num fator de amortecimento $\lambda_0 v^k$, para qualquer valor de K.

Neste momento, definido o algoritmo de otimização adotado, é conveniente salientar como se enquadra no problema desta tese. Assim, neste caso existe um conjunto de 11 pontos, que correspondem ao conjunto das transições, num gráfico de ângulo ρ em função da distância em pixéis. A função de regressão é uma função polinomial de ordem n, sendo que o estudo da ordem será feito mais tarde, e pode ser expressa por

$$f(x,\beta) = a_0 + a_1 x + a_2 x^2 + \cdots + a_{n-1} x^{n-1} + a_n x^n = \sum_{i=0}^{n} a_i x^i \tag{4.11}$$

o que em notação matricial equivale a

$$F = \begin{bmatrix} 1 & x & \cdots & x^{n-1} & x^n \end{bmatrix} \begin{bmatrix} a_0 \\ a_1 \\ \vdots \\ a_{n-1} \\ a_n \end{bmatrix} = X\beta \tag{4.12}$$

onde o vetor de parâmetros β corresponde, neste caso, aos coeficientes da função polinomial, ou seja, $\beta = \begin{bmatrix} a_0 & a_1 & a_2 & \cdots & a_{n-1} & a_n \end{bmatrix}^T$. Em suma, assim se demonstra como a utilização do método de LM neste contexto permite determinar os coeficientes que melhor aproximam a curva polinomial que, a partir de distâncias na imagem, determina o ângulo ρ correspondente na realidade, tal como se pretende.

4.3 *Software*

Nesta secção é explicado o código desenvolvido, que permite a aplicação do algoritmo escolhido, assim como a interface gráfica para contacto com o utilizador e a consequente integração com o *software* dos robôs.

4.3.1 Otimização e Extração de Parâmetros

Em ambientes de cálculo numérico como o *Matlab* e o *Octave*, entre outros, a implementação do algoritmo de Levenberg-Marquardt está embutida em *toolboxes* de otimização. Para além disso, existem implementações sob a forma de bibliotecas de código que podem ser adaptadas para diferentes aplicações.

Nesse sentido, não se encetaram esforços no desenvolvimento do algoritmo de LM, optando-se pela inclusão de uma biblioteca [29] no programa, implementada originalmente na linguagem de programação *C* e adaptada para *Object Pascal*, uma linguagem derivada de *Pascal*. Como tal, a ferramenta utilizada para programar nesta linguagem foi o IDE *Lazarus*. Este programa tem ainda a vantagem de possuir ferramentas de criação de GUI para interface com o utilizador.

Por conseguinte, na biblioteca escolhida é disponibilizada a função *lmmin()* que determina um vetor de parâmetros *par*, que minimiza a soma dos quadrados dos elementos de *fvec*. O vetor *fvec* é computado por uma função *evaluate()* providenciada pelo utilizador. Em caso de sucesso, o vetor *par* representa o mínimo local, não necessariamente o global, dependendo sempre da estimativa inicial.

```
void lmmin( const int n_par, double *par, const int m_dat, const
    void *data, void *evaluate(), const
lm_control_struct *control, lm_status_struct *status );
```

O excerto de código anterior, onde é representada a função de otimização, é caraterizada por alguns argumentos, que são explicados de seguida:

- ***n_par***

  Número de variáveis livres, que define o tamanho do vetor de parâmetros. Neste caso será sempre igual à ordem da função polinomial mais uma unidade.

- ***par***

  Vetor de parâmetros. Como *input* da função deverá conter uma estimativa dos parâmetros. No final da chamada à função, o *output* constitui a solução encontrada para minimizar $||fvec||$.

- ***m_dat***

  Tamanho do vetor *fvec*, que corresponde ao número de pontos do gráfico ao qual se pretende ajustar a curva de regressão. Deve satisfazer a condição *n_par* <= *m_dat*.

- ***data***

  Este apontador é ignorado pelo algoritmo de otimização, sendo que serve apenas para ser reencaminhado como argumento para a rotina *evaluate* providenciada pelo utilizador. Neste caso, utilizou-se uma estrutura de dados para cada direção, *data0*, *data120* e *data240*, contendo diversas informações como a posição das transições detetadas, os erros associados e a ordem do polinómio de mapeamento.

- ***evaluate***

  Apontador para a função fornecida pelo utilizador que, a cada iteração, computa os *m_dat* elementos do vetor de parâmetros *fvec*, através da subtração do valor dado pela estimativa atual dos parâmetros *par* a *y*, isto é, o desvio da estimativa do ângulo obtido com a função polinomial em relação ao valor real. Se a soma dos quadrados do valor de retorno de *fvec* cumprir as condições de paragem, a rotina *lmmin()* termina a otimização.

- ***control***

  Conjunto de parâmetros para afinar o processo de otimização, como as tolerâncias máximas admitidas para o erro e o passo usado para calcular a matriz Jacobiana. Na maioria dos casos, o valor padrão $\&lm\_control\_double$ é adequado.

- ***status***

  Uma estrutura de dados do tipo *record*, usada para retornar informação acerca do processo de minimização, como a norma de $fvec$ e o número de iterações.

4.3.2 Interface Gráfica

De seguida, depois de ter o código implementado e as invocações às funções da biblioteca feitas, estava concretizado o programa que, a partir de um conjunto de pontos, determinava os parâmetros que melhor definem a regressão polinomial correspondente. Posteriormente, partiu-se para a realização de uma interface gráfica, para facilitar a visualização do processo de calibração, e ainda, tornar mais intuitiva a interação do utilizador com o programa.

Então, foi desenhada a interface, que se carateriza por uma janela que alterna entre quatro abas, sendo a primeira destinada às opções, onde se podem configurar alguns parâmetros. As restantes, dizem respeito, para cada direção em que é feita a calibração, às funcionalidades de otimização e exibição dos resultados obtidos.

Na primeira aba é possível configurar alguns parâmetros, como o número de transições medidas em cada direção, a altura em metros do ponto inferior do espelho, a ordem da função polinomial para cada direção, e ainda, a localização do centro da imagem, em pixéis.

Os valores mostrados na imagem para cada parâmetro são predefinidos e, a menos que sejam alterados e validados, são estes os valores usados nos cálculos de otimização.

Caso novos valores sejam inseridos, os mesmos apenas são válidos se pertencerem ao intervalo entre certos limites estipulados. Assim, o número de medidas deverá ser entre 11 e 15, dando alguma margem para calibrações futuras, caso se pretenda aumentar o número de transições. A altura deverá estar compreendida entre 65 e 85 centímetros, não se prevendo que haja alterações estruturais nos robôs que coloquem o espelho a uma altura que não pertença a este intervalo. O polinómio da função de mapeamento poderá ser de ordem 2, no mínimo, e de ordem 6, no máximo. Por fim, estipulou-se que o centro da imagem não deverá tomar valores negativos nem para além dos limites da imagem.

Se o valor for validado, então a caixa com o valor editável apresenta a cor verde, senão essa caixa terá um fundo vermelho.

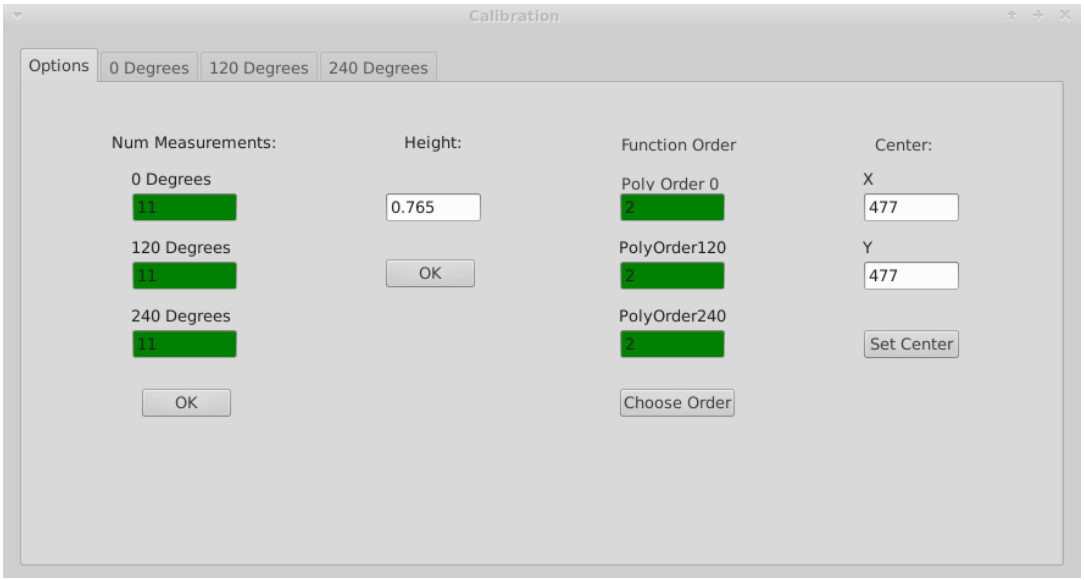


Figura 4.4: Aba de opções da interface gráfica.

As restantes abas têm uma interface semelhante, mas cada uma é correspondente a cada direção em que será feita a calibração. Neste caso, a janela visualizada é composta por uma tabela com várias colunas referentes a diferentes grandezas, mostradores com valores e alguns botões com funcionalidades que serão descritas de seguida.

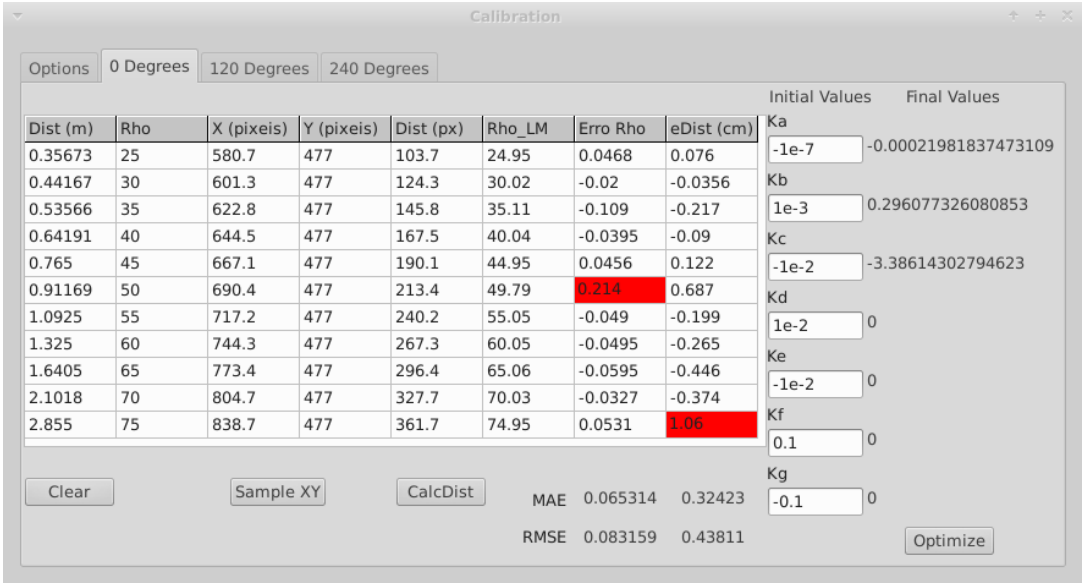


Figura 4.5: Aba do programa com valores na tabela para θ na direção de 0^o.

Em relação à tabela, existem alguns valores que estão sempre presentes à partida e têm valores fixos, que indicam a localização real das transições detetadas nas tiras. Esses valores correspondem à distância em metros *Dist(m)* de cada transição à projeção do centro do robô no solo, e o seu ângulo *Rho* em relação ao ponto inferior do espelho, tal como ilustrado a seguir.

O botão *Clear* permite, tal como o nome sugere, limpar os valores da tabela, permanecendo os valores fixos das colunas *Dist(m)* e *Rho*.

Por outro lado, o botão *Sample XY* permite obter e guardar na tabela o valor das transições em pixéis segundo as direções X e Y e a direção θ do raio em que foram detetadas. Essas informações são provenientes do *Ovis* e posteriormente transferidas para o *Decision* e armazenadas. Porém, os valores também podem ser inseridos manualmente na tabela.

Após a obtenção das coordenadas em pixéis de cada transição, o botão *CalcDist* calcula e mostra na tabela a distância total em pixéis *Dist(px)*, de cada transição ao centro da imagem, aplicando a fórmula definida em 3.2.

Finalmente, clicando no botão *Optimize* é corrido o algoritmo que, partindo dos valores iniciais atribuídos aos coeficientes K do polinómio, mostrados à direita da tabela, implementa o método de otimização Levenberg-Marquardt para determinar os valores finais de K que minimizam o erro da função de regressão. Quando o processo iterativo termina, os valores finais de cada parâmetro são mostrados debaixo de *Final Values*. Uma nota para a nomenclatura adotada quanto aos parâmetros K, em que K_a corresponde ao coeficiente líder do polinómio, isto é, à variável de ordem superior. A título de exemplo, se a ordem escolhida para o polinómio for 3, então K_a é o coeficiente de x^3, enquanto K_d é o termo independente.

Uma vez obtidos os valores finais de K, esses parâmetros são usados na função de ajuste para computar o valor do ângulo *Rho_LM*. De seguida, é calculado o erro absoluto e_i para a distância *eDist(cm)* e para o ângulo *ErroRho*, que resulta da diferença entre o valor real e o valor obtido com

a função de aproximação, $e_i = y_i - f(x,k)$. Depois, o erro máximo absoluto, ora em distância, ora em ângulo, é destacado a vermelho. Para além disso, são também calculados e representados o erro médio absoluto *MAE* e o erro quadrático médio *RMSE*, seguindo as fórmulas 4.13 e 4.14, respetivamente.

$$\text{MAE} = \frac{\sum_{i=1}^{n} |e_i|}{n} \tag{4.13}$$

$$\text{RMSE} = \sqrt{\frac{\sum_{i=1}^{n} |e_i|^2}{n}} \tag{4.14}$$

Para além da figura 4.5, em anexo pode ver-se figuras com tabelas preenchidas com valores para as restantes direções, sendo que as calibrações foram feitas com ordem polinomial 2, altura de 0.765 metros e centro com coordenadas $(477,477)$.

4.3.3 Obtenção das Coordenadas das Transições

Tal como foi afirmado, o *Ovis* processa a imagem de forma contínua, para detetar transições verde-branco ao longo dos raios definidos para o efeito. As transições, à medida que são detetadas, são transferidas para o *Decision*. Nessa camada da aplicação, assim que são recebidas, essas informações são guardadas em estruturas de dados do tipo *record* onde são guardados vários detalhes, entre os quais as coordenadas da posição dos pixéis das transições e o ângulo θ do raio onde foram detetadas.

Deste modo, como a funcionalidade associada ao botão *Sample XY* está dependente da capacidade do programa aceder às informações transferidas para o *Decision*, procedeu-se à integração do *software* desenvolvido nesse programa. Assim sendo, para correr a aplicação de calibração passou a ser necessário clicar no botão *Calibration* colocado na interface do *Decision*.

Por conseguinte, sempre que o botão *Sample XY*, que permite a amostragem das transições para a tabela, fosse clicado, tomaram-se alguns cuidados no processamento dos dados. Ora, havia rejeição dos dados se o número de amostras não fosse igual ao número de transições, fazendo com que se alguma das transições não fosse detetada, não houvesse uma associação errada de distâncias. Por exemplo, se a 5ª transição da tira não fosse detetada, a 6ª seria interpretada no seu lugar e todas as seguintes seriam erradamente interpretadas.

Para além disso, no momento em que o botão *Sample XY* é premido, caso os dados não sejam rejeitados, são recolhidas 10 amostras para cada transição e no final é calculada a média aritmética, para filtragem do ruído de medição, uma vez que existem algumas oscilações na funcionalidade de deteção de transições do *Ovis*.

Deste modo, concluiu-se a integração do *software* desenvolvido na camada de aplicação *Decision* presente nos robôs.

4.3.4 Integração no *Ovis*

4.3.4.1 Estudo da Ordem Polinomial

Neste momento, obtidas as tiras como padrão de calibração, feito o programa de otimização e encerrada a integração no *Decision*, estavam reunidas as condições para aplicar o método de calibração e retirar algumas ilações dos comportamentos observados.

Assim, colocando as tiras de volta dos robôs, fez-se correr o programa e fizeram-se várias calibrações, alterando sucessivamente a ordem polinomial. Com isto pretendia-se perceber qual o grau mais adequado a utilizar.

Deste modo, tendo obtido os parâmetros, traçaram-se gráficos para perceber o comportamento das curvas e computaram-se os erros MAE e RMSE para o ângulo e distância. Na figura seguinte está presente um exemplo dos polinómios de diferentes ordens para uma das direções. Os gráficos e tabelas das outras direções estão disponíveis em anexo.

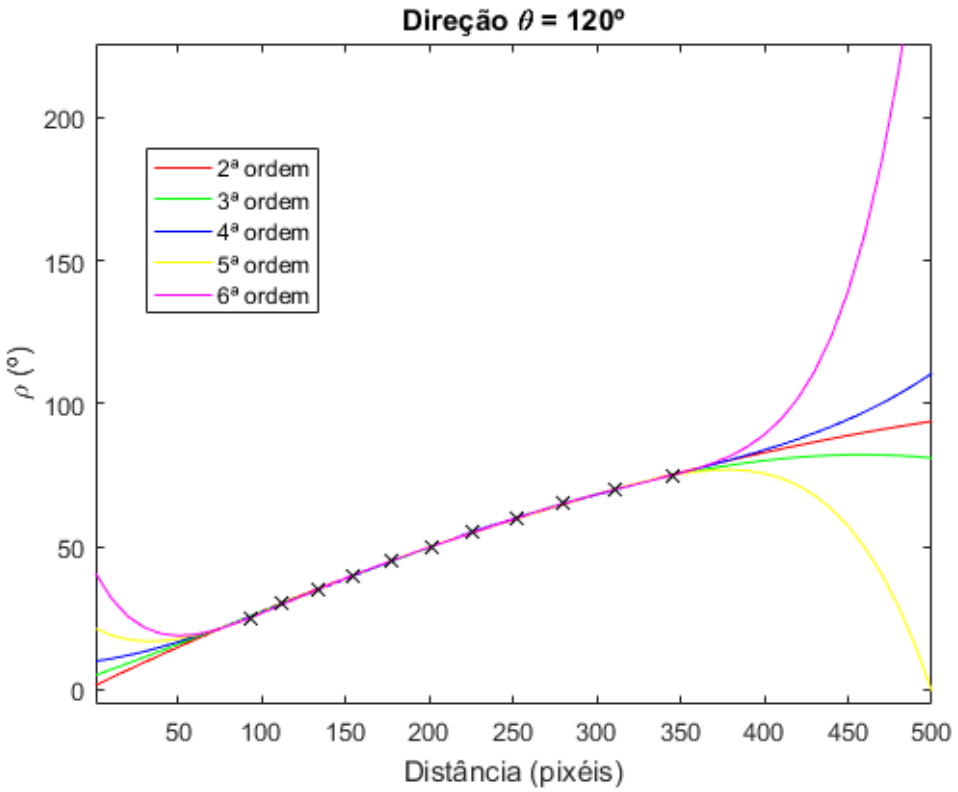


Figura 4.6: Função $\rho(d_{px})$ para diferentes ordens polinomiais, na direção de 120º.

Ordem Polinomial	MAE	RMSE
2	0.0881	0.1127
3	0.0848	0.1088
4	0.0671	0.1012
5	0.0681	0.1011
6	0.0663	0.1003

Tabela 4.1: Direção 120º - MAE e RMSE para o erro do ângulo ρ (graus).

Ordem Polinomial	MAE	RMSE
2	0.4407	0.6412
3	0.3313	0.3989
4	0.1867	0.2386
5	0.1989	0.2474
6	0.1690	0.2247

Tabela 4.2: Direção 120º - MAE e RMSE para o erro em distância (centímetros).

Como se pode observar, para as distâncias entre 90 e 380 pixéis, aproximadamente, o comportamento das curvas é semelhante seja qual for a ordem da função polinomial, pois esta região corresponde ao intervalo de distâncias na imagem onde estão os padrões de calibração. Porém, focando a atenção nos extremos do gráfico, que correspondem, quer a zonas mais próximas do robô do que a primeira transição ($35cm$), quer a zonas para lá do alcance das tiras ($2.85m$), o que se infere é que conforme a ordem polinomial aumenta, maior é a alteração do declive das curvas.

Por outro lado, as tabelas 4.1 e 4.2 permitem inferir que, no geral, o erro é menor conforme o grau polinomial aumenta.

Na verdade, para ordens polinomiais elevadas, o que se verifica é o fenómeno de *overfitting* [30], em que a função se ajusta muito bem aos dados amostrados do sistema, neste caso as posições das transições nas tiras, mas é ineficaz na previsão de resultados. Daqui se depreende que, devido ao comportamento aproximadamente linear do sistema, o uso de funções de grau inferior é mais vantajoso, pois, por exemplo, usando a função polinomial de ordem 6, para uma bola afastada do robô que seja detetada na imagem a uma distância de cerca de 450 pixéis será assumido um valor de ρ superior a 150º, o que não é viável.

Para além disso, outra das vantagens da utilização de uma ordem polinomial inferior tem a ver com o esforço computacional, pois em situações de jogo o mapeamento é feito em tempo real. E ainda, por vezes é necessário fazer o cálculo inverso, para descobrir a que distância na imagem corresponde um determinado ângulo ρ ou distância real, o que implica inverter a função de mapeamento. Ora, a inversão de polinómios de ordem elevada obriga à utilização de métodos numéricos, com maior esforço computacional do que, por exemplo, a resolução de uma simples equação quadrática para um polinómio de segunda ordem.

Levando em consideração estas conclusões, definiu-se que o polinómio a utilizar seria de segunda ordem, sendo necessário obter apenas três parâmetros na otimização.

No *software* dos robôs, as funções de tratamento e mapeamento da imagem estão implementadas no *Ovis*, pelo que foi necessário alterar algumas destas funcionalidades para os robôs passarem a utilizar a calibração obtida. Outra questão abordada foi a forma como as três calibrações seriam unificadas em uma só.

4.3.4.2 Fusão das Três Calibrações

Assim, para qualquer ponto na imagem caraterizado por (x_x, y_x, θ_x), foram atribuídos diferentes pesos (w_a, w_b, w_c) às funções de calibração de cada direção, em função da proximidade do ponto da imagem a essa direção.

Para o efeito, a atribuição de pesos foi feita de forma inversamente proporcional à distância do ponto a mapear a essa direção. Assim, quanto mais próximo de uma direção em que foi feita a calibração se está, mais peso é atribuído à respetiva função.

Deste modo, implementou-se um algoritmo para o cálculo dos pesos baseado no seguinte fluxograma:

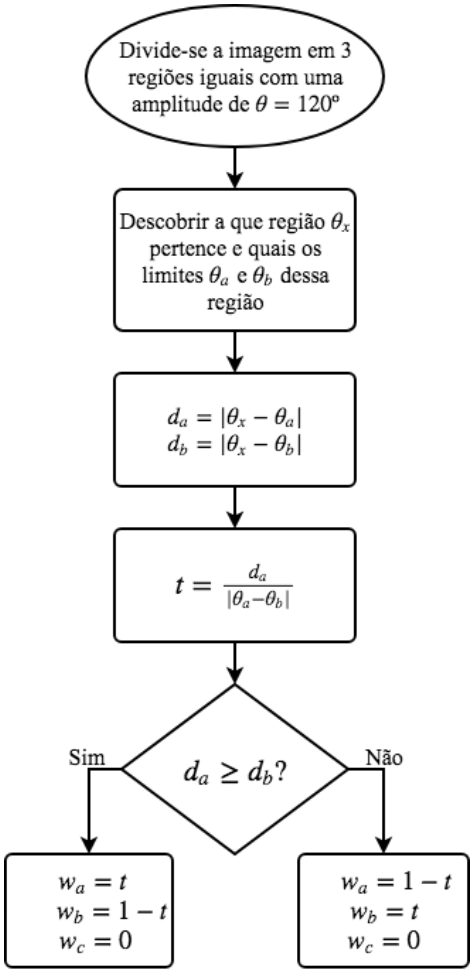


Figura 4.7: Algoritmo da fusão das 3 funções de calibração representado num fluxograma.

Em anexo segue um excerto do código do programa *Ovis*, contendo uma função que transforma pixéis em ângulos ρ e θ, onde está implementado este algoritmo que integra as três calibrações numa única.

4.4 Síntese dos Procedimentos para a Calibração

Por fim, são listados todos os procedimentos a seguir que permitem executar o método de calibração proposto nesta dissertação.

1. Colocar as tiras de volta do robô como em 4.8;

2. Com o robô operacional, correr o *software* necessário para o seu funcionamento, entre os quais o *Decision* e o *Ovis*;

3. No *Ovis*:

   (a) Escolher como tipo de transições a detetar o verde-branco;

   (b) Ajustar a representação HSV da cor verde para a correspondente à cor preta;

   (c) Definir apenas 3 raios sobre a imagem, com comprimento suficiente para intersetar as tiras em toda a sua extensão;

   (d) No final o resultado deverá ser semelhante ao observável em 4.9.

4. No *Decision*:

   (a) Abrir a aplicação de calibração clicando no botão *Calibration*;

   (b) Configurar os parâmetros na aba *Options*;

   (c) Na aba seguinte, fazer *Sample XY* seguido de *CalcDist*;

   (d) Por fim, clicar em *Optimize*;

   (e) Os parâmetros obtidos são guardados automaticamente num ficheiro de texto. Passar esses parâmetros para o *Ovis*.

5. O processo de calibração está concluído.

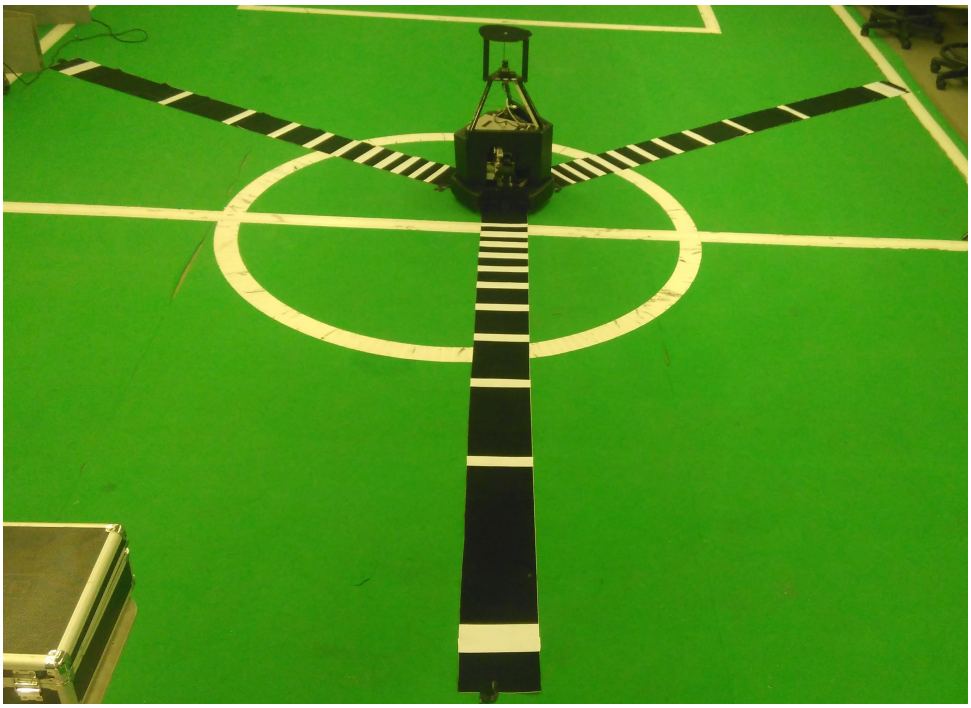
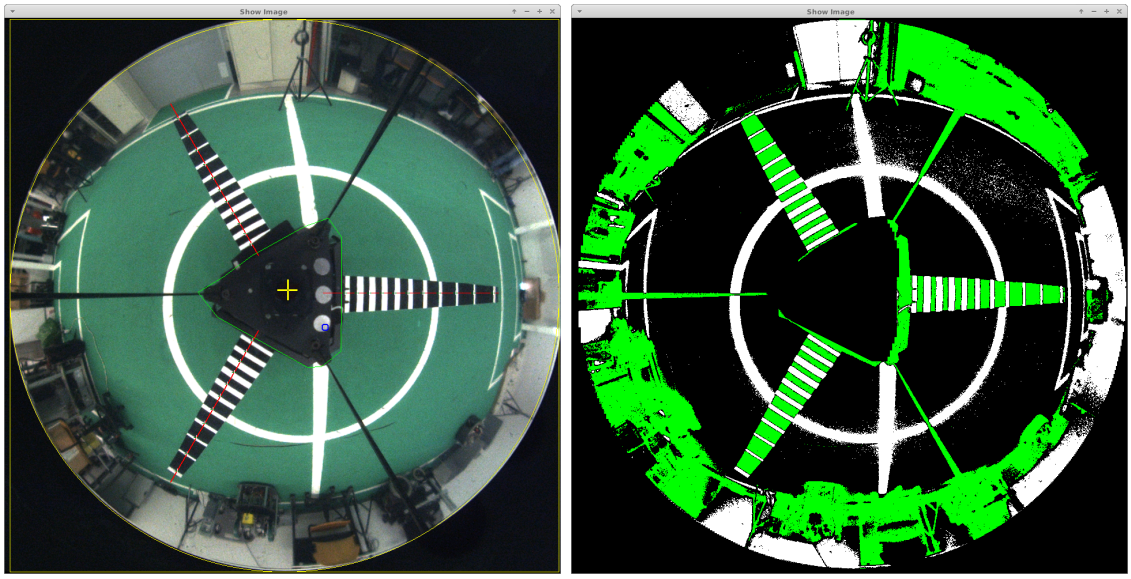


Figura 4.8: Tiras dispostas em torno do robô.



(a) Imagem não Processada (b) Imagem Processada

Figura 4.9: *Ovis* com três Tiras espaçadas por 120º.

Capítulo 5

Testes e Resultados

Ao longo do presente capítulo é mostrado que testes foram feitos para averiguar a viabilidade da calibração obtida com este método.

No primeiro caso, é feito um teste com a funcionalidade de estimação de posição e velocidade da bola, comparando o método de calibração atual com o anterior.

Seguidamente, é feito um teste distinto, em que o robô é colocado em rotação com velocidade angular de 1 rad/s, com a bola parada a diferentes distâncias, amostrando as posições da bola observadas pelo sistema de visão, usando os parâmetros obtidos com o novo método de calibração.

Finalmente, são retiradas algumas ilações dos testes efetuados.

5.1 Estimação de Posição e Velocidade da Bola

Tal como foi mencionado antes, os robôs da equipa 5DPO já possuíam múltiplos programas e funcionalidades implementadas. Entre essas está também a capacidade de estimação da posição e velocidade da bola, quer em relação a um referencial local, ao robô neste caso, quer no referencial global. Este procedimento, demonstrado em [2], implementa filtros de Kalman, com o vetor de estado referenciado no robô e no referencial global. Para além disso, está intrinsecamente dependente de algumas variáveis, e da sua fusão, como a odometria e o sistema de visão.

Como tal, definiu-se que esta funcionalidade seria útil para testar o sistema de visão e aferir as melhorias obtidas com o método proposto nesta dissertação. Em alguns casos foram comparados com os resultados do método de calibração usado no ano anterior.

5.1.1 Ensaios com Diferentes Calibrações

Os ensaios realizados dizem respeito à trajetória percorrida pelo robô em torno da bola colocada no centro do meio-campo, descrevendo um quadrado como na figura 5.1, com 1 metro

de largura e com velocidade linear de 1 m/s. Enquanto percorre esta trajetória, o robô faz uma estimativa da posição e velocidade da bola, em relação a diferentes referenciais.

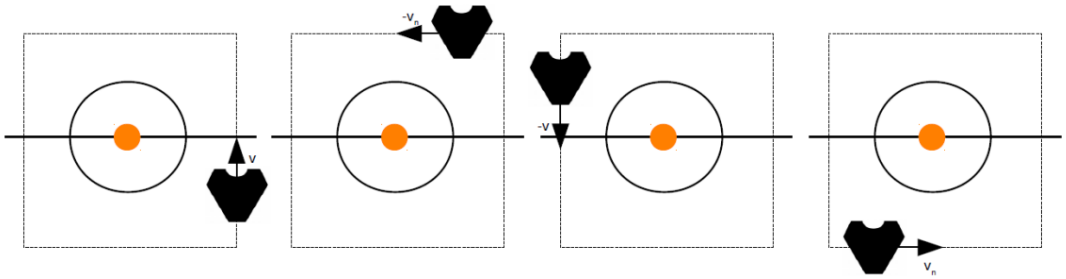


Figura 5.1: Sequência da trajetória definida pelo robô, com a bola imóvel no centro do campo.

Deste modo, seguem-se alguns resultados obtidos com a calibração implementada anteriormente, seguidos dos resultados para os mesmos testes, mas reproduzidos com a calibração feita com o novo método.

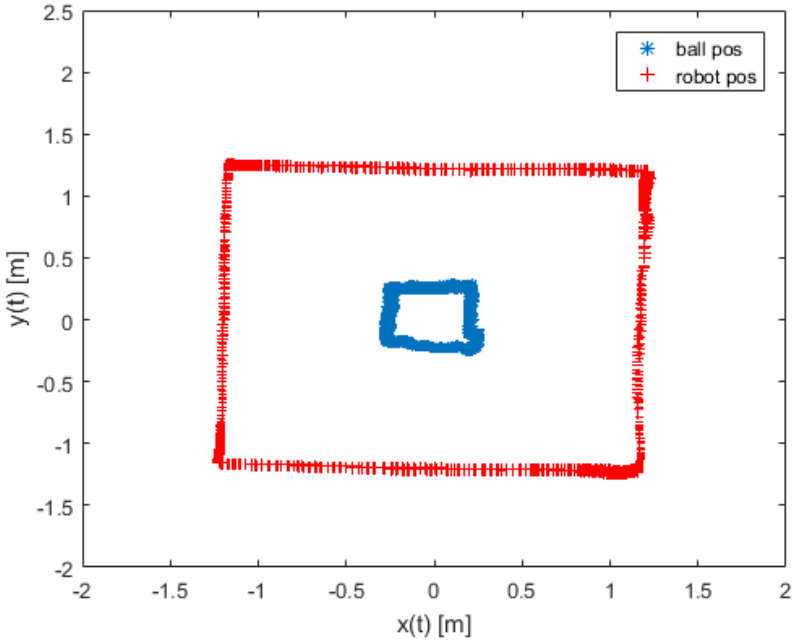


Figura 5.2: Posição estimada do robô e da bola no referencial global - ensaio com calibração antiga.

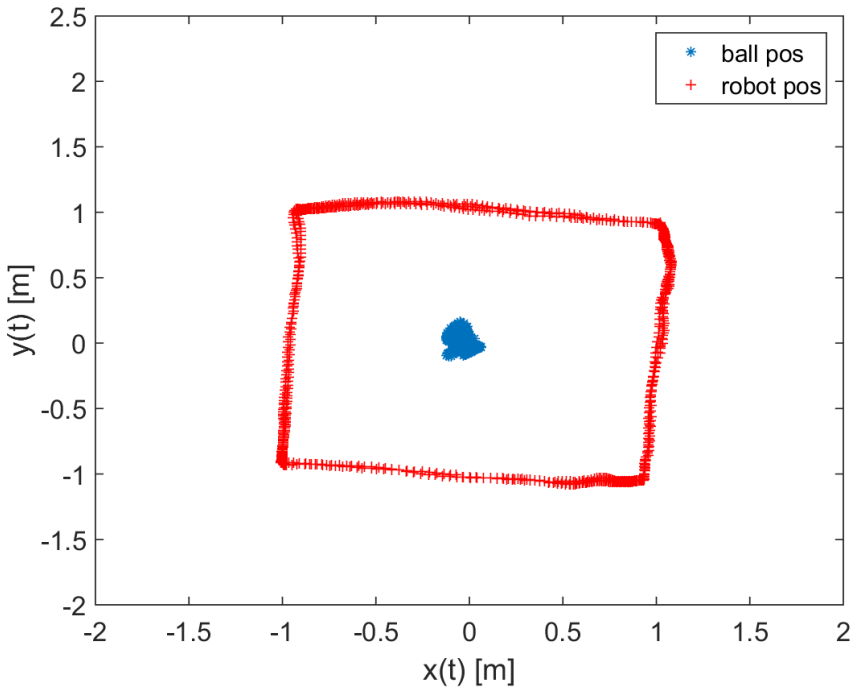


Figura 5.3: Posição estimada do robô e da bola no referencial global - ensaio com calibração proposta.

No primeiro caso, a figura 5.2 permite verificar que, apesar de a bola permanecer imóvel durante o ensaio, o robô estima que a sua posição se altera com o tempo, transmitindo a ideia que também a bola descreveu a trajetória de um quadrado, com cerca de meio metro de largura.

Na segunda figura 5.3, verifica-se que a posição estimada da bola no referencial global é consistente. Embora o gráfico não defina um ponto perfeito, a posição no gráfico permite inferir que a bola está no centro do terreno e que não acompanha a trajetória quadrada do robô.

Assim, em comparação com o ano passado, a bola, que permanece imóvel no centro do quadrado, tem a sua posição estimada com maior precisão com a nova calibração do que com a calibração precedente, o que atesta que este método de calibração produz resultados que resultam numa melhoria do sistema de visão dos robôs.

Para além disso, outra observação que pode ser feita prende-se com o facto do uso de padrões de calibração em direções distintas e a correspondente adaptação de um modelo matemático para cada uma dessas direções, resulta claramente numa vantagem. Isto com base em que, com a anterior calibração, conforme o robô se deslocava e a bola passava a ser refletida em novas posições do espelho o mapeamento passava a ser diferente, resultando no gráfico quadrado visível em 5.2. O mesmo não se verifica em 5.3, onde a posição da bola está confinada ao centro do referencial do robô.

5.2 Bola a distâncias fixas com robô em rotação

Nesta secção, serão apresentados os resultados obtidos para os ensaios efetuados com o robô em rotação sobre si próprio, com a bola parada a diferentes distâncias, usando apenas o método de calibração proposto.

Numa primeira fase são mostrados os resultados com a bola parada à distância de 1 metro do robô. Porém, num dos ensaios são utilizados os parâmetros obtidos de uma calibração feita com cerca de quatro semanas, considerada desatualizada, e no outro ensaio são usados os parâmetros de uma calibração feita na hora dos testes. Com estes resultados pretende-se entender o que acontece à calibração com o passar do tempo e perceber como aumenta o erro.

Seguidamente, usando apenas os parâmetros obtidos no momento do ensaio, são mostrados mais testes com a bola colocada a diferentes distâncias, para perceber o que acontece ao erro de observação do sistema de visão conforme a distância ao robô aumenta.

Para cada ensaio realizado será apresentado um conjunto de 6 gráficos, referentes à posição da bola em relação ao referencial local do robô, às respetivas coordenadas x e y nesse mesmo referencial, à distância em metros e o correspondente erro, e finalmente, ao ângulo ρ observado e o desvio em relação ao valor real.

5.2.1 Bola a 1 metro com calibração desatualizada

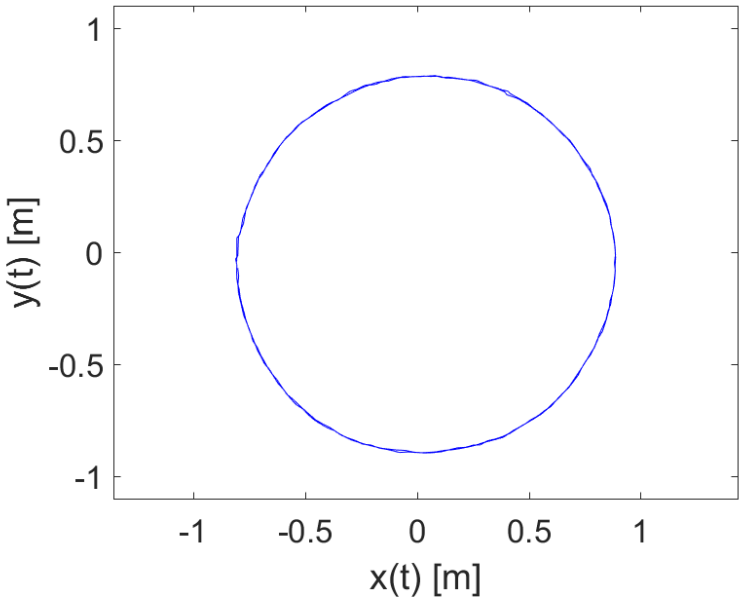


Figura 5.4: Posição observada da bola a 1 metro do robô em rotação - calibração desatualizada.

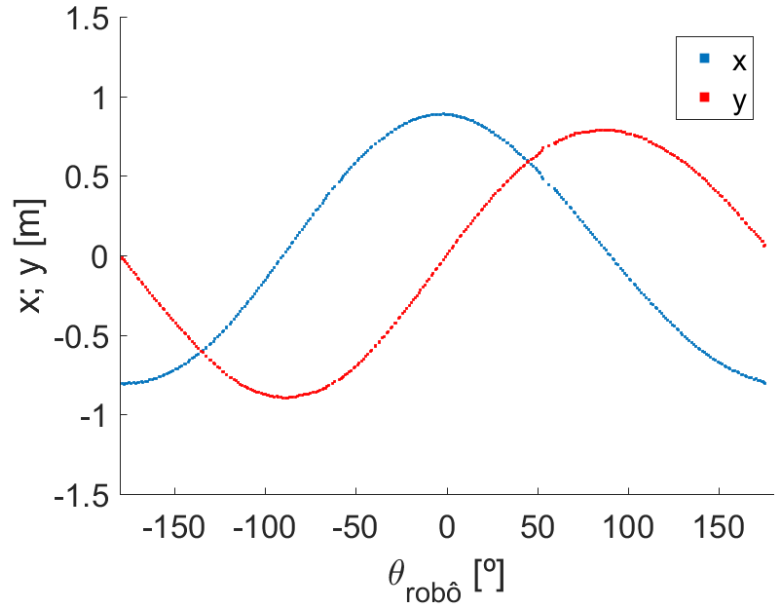


Figura 5.5: Coordenadas observadas da bola a 1 metro do robô em rotação - calibração desatualizada.

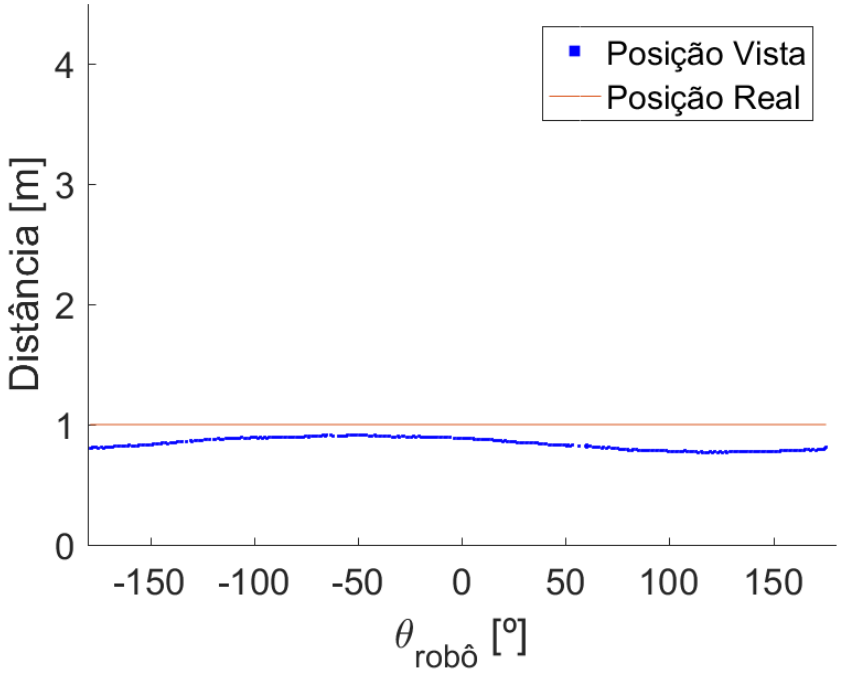


Figura 5.6: Distância observada da bola a 1 metro do robô em rotação - calibração desatualizada.

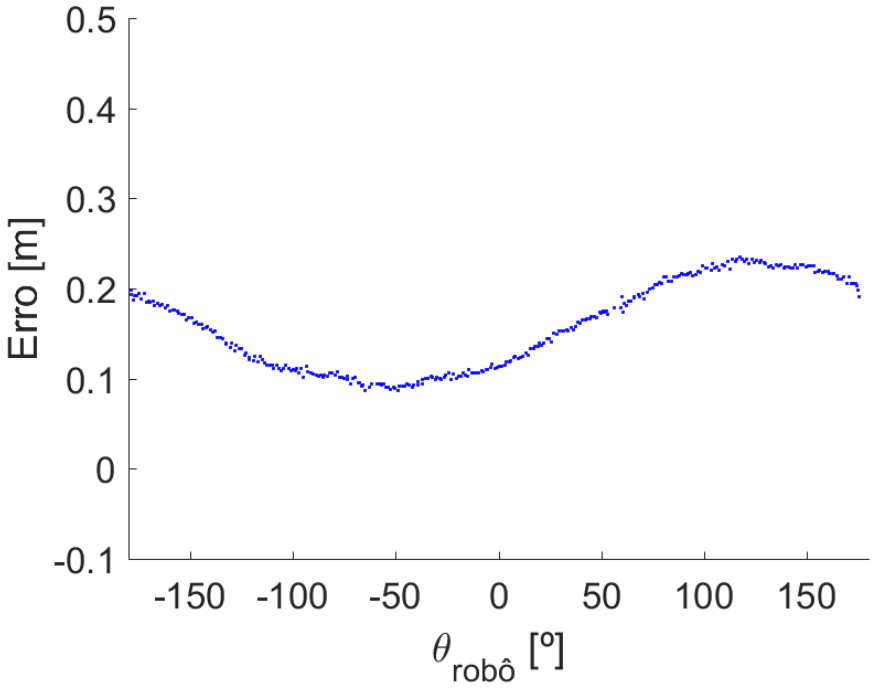


Figura 5.7: Erro da distância observada - calibração desatualizada.

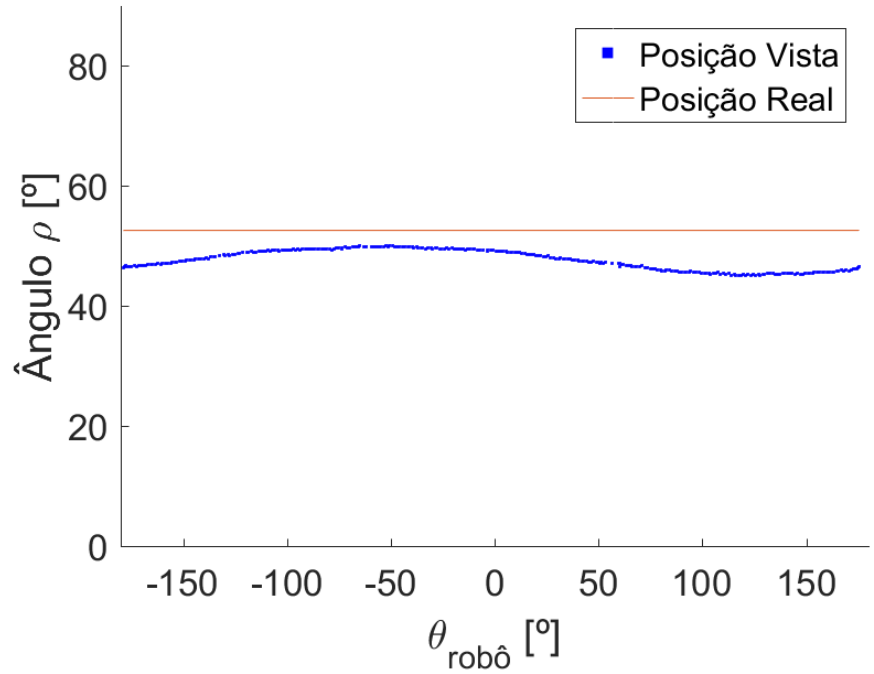


Figura 5.8: Ângulo ρ observado da bola a 1 metro do robô em rotação - calibração desatualizada.

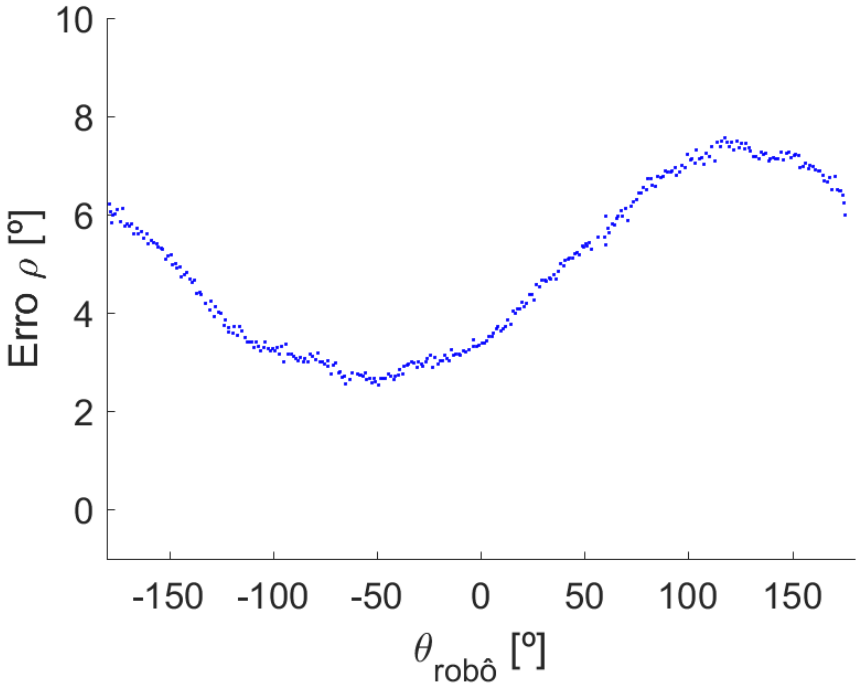


Figura 5.9: Erro do ângulo ρ observado - calibração desatualizada.

5.2.2 Bola a 1 metro com calibração recente

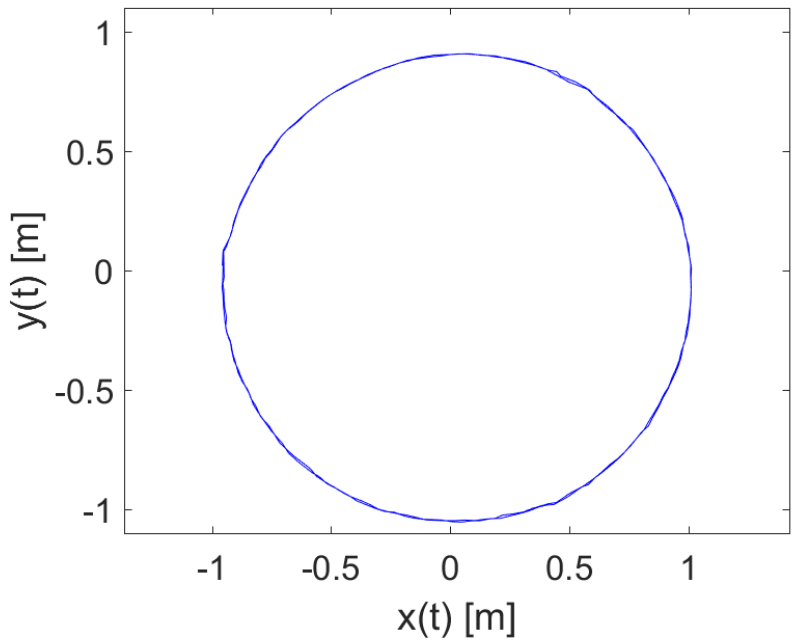


Figura 5.10: Posição observada da bola a 1 metro do robô em rotação - calibração recente.

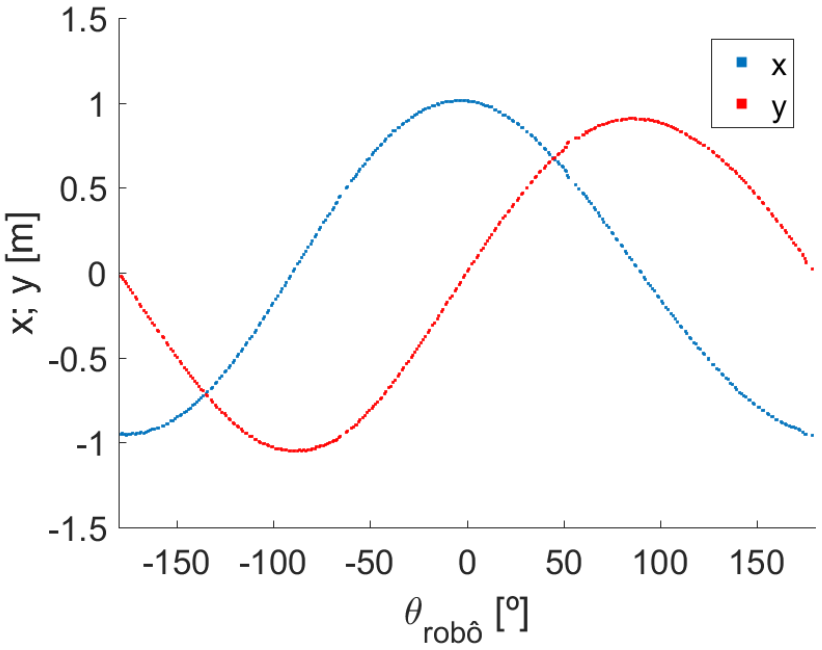


Figura 5.11: Coordenadas observadas da bola a 1 metro do robô em rotação - calibração recente.

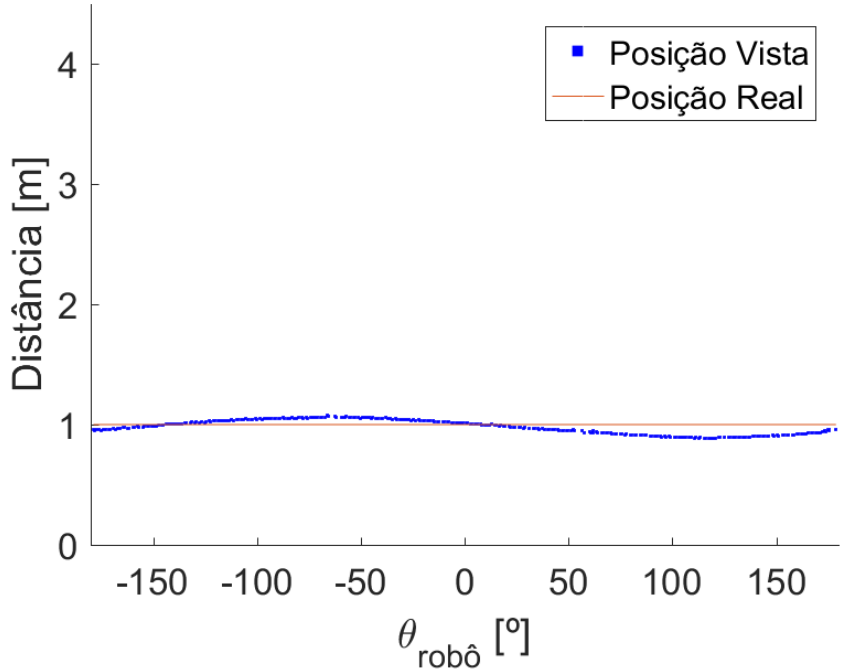


Figura 5.12: Distância observada da bola a 1 metro do robô em rotação - calibração recente.

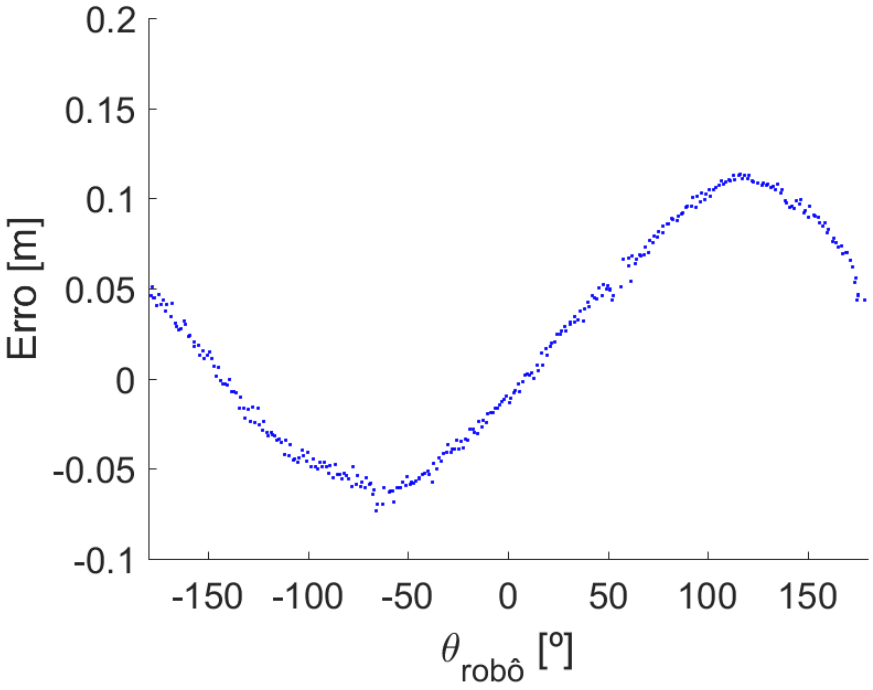


Figura 5.13: Erro da distância observada - calibração recente.

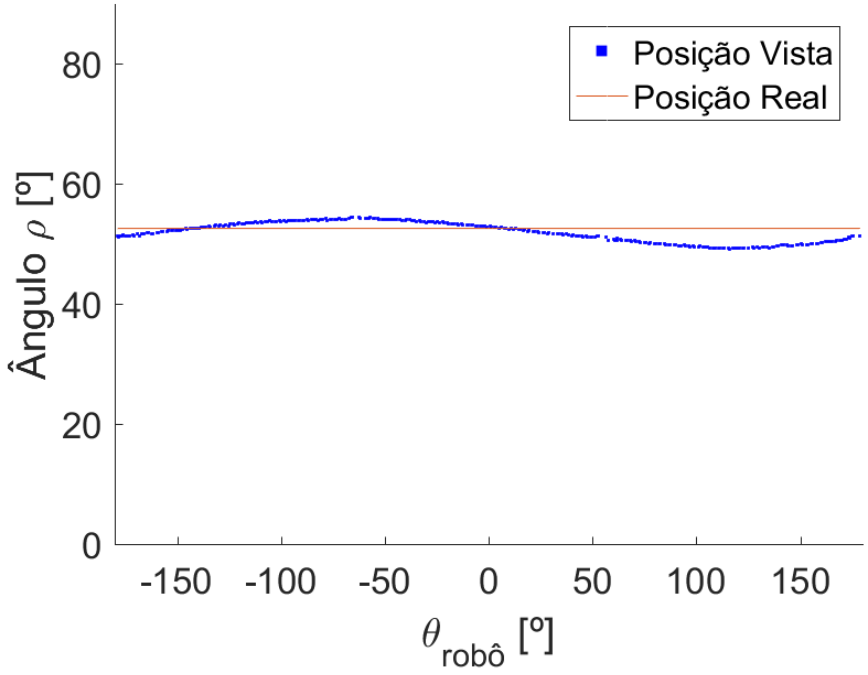


Figura 5.14: Ângulo ρ observado da bola a 1 metro do robô em rotação - calibração recente.

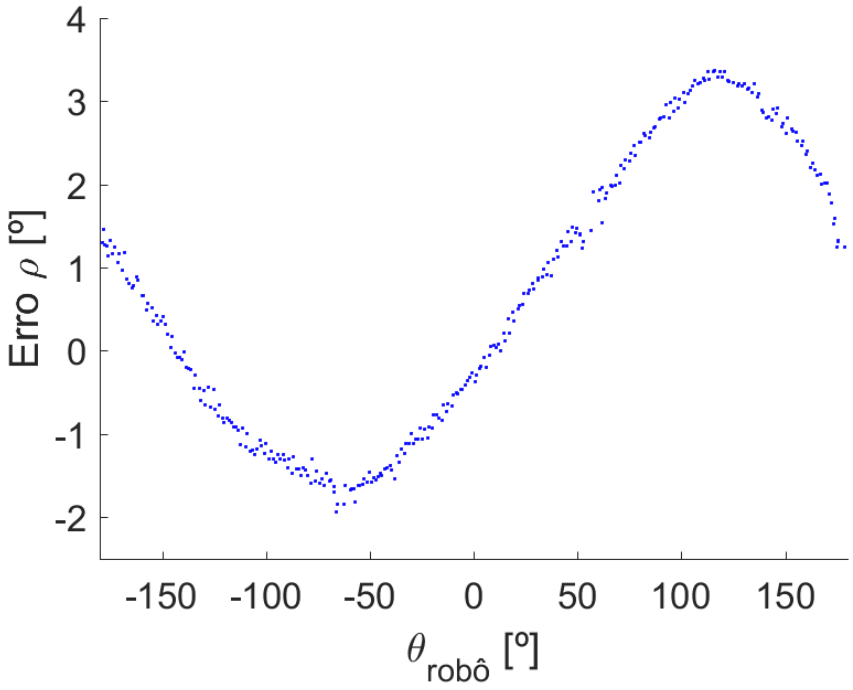


Figura 5.15: Erro do ângulo ρ observado - calibração recente.

Distância (m)	MAE (m)	MAE (º)
(a) 1	0.1601	4.9659
(b) 1	0.0556	1.5883

Tabela 5.1: MAE para o erro de observação em distância (metros) e em ângulo ρ (º), para a bola a 1 metro de distância, com a calibração desatualizada (a) e com a recente (b).

Comparando as figuras 5.4 e 5.10, em que os ensaios foram feitos com a bola colocada à mesma distância em relação ao robô em rotação, usando no primeiro caso uma calibração desatualizada, feita há um período de tempo de cerca de quatro semanas, e no segundo uma calibração feita no momento do ensaio, verifica-se que, de facto, com o passar do tempo, o uso geral do robô e as colisões que este sofre levam ao desalinhamento entre a câmara e o espelho, resultando em piores resultados no sistema de visão. O aumento do erro é observável nos gráficos 5.7 e 5.9, em comparação com 5.13 e 5.15, respetivamente, e também corroborado na tabela 5.1, onde, para a distância de 1 metro, o erro médio absoluto é superior para a calibração desatualizada.

Deste modo, conclui-se ser de maior relevância calibrar o sistema de visão pontualmente, de preferência antes de cada jogo, uma vez que ao longo do tempo os parâmetros do sistema de visão tendem a perder fiabilidade.

5.2.3 Bola a 1.5 metros com calibração recente

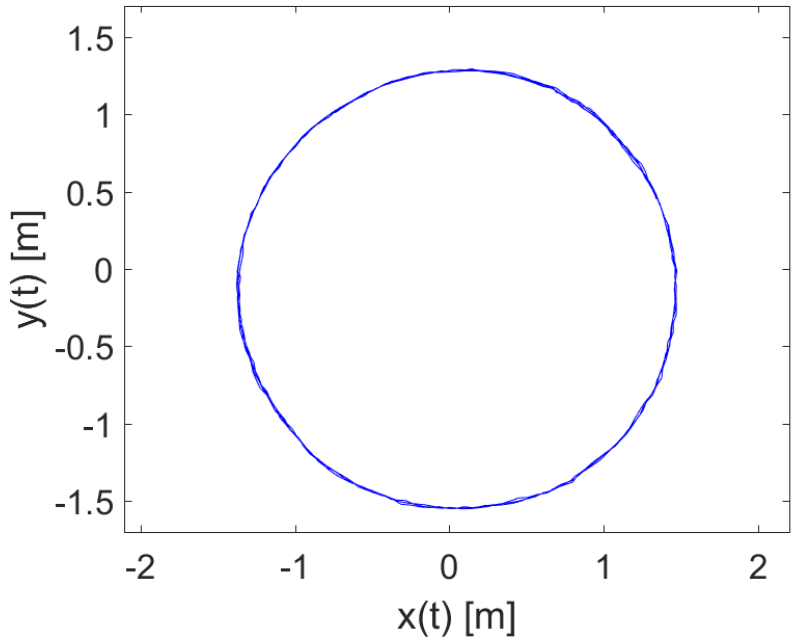


Figura 5.16: Posição observada da bola a 1.5 metros do robô em rotação.

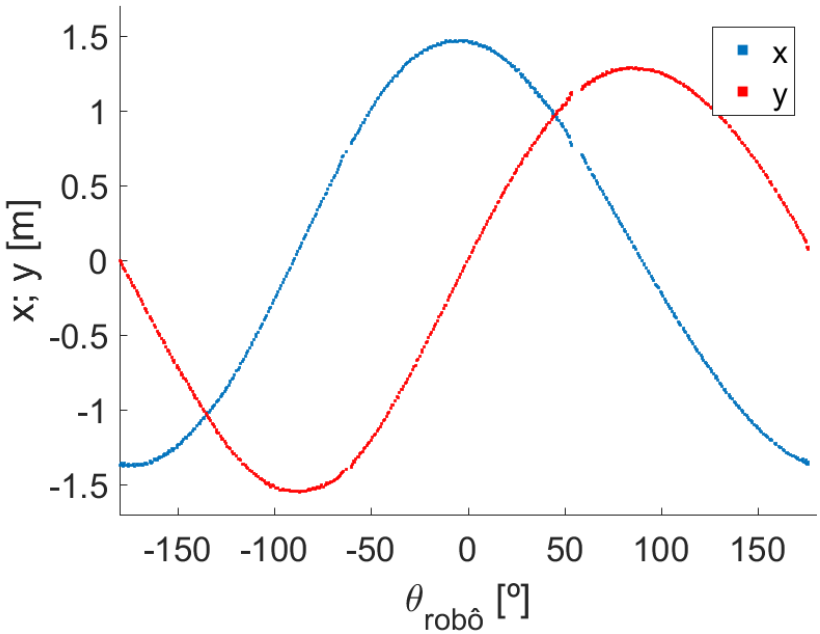


Figura 5.17: Coordenadas observadas da bola a 1.5 metros do robô em rotação.

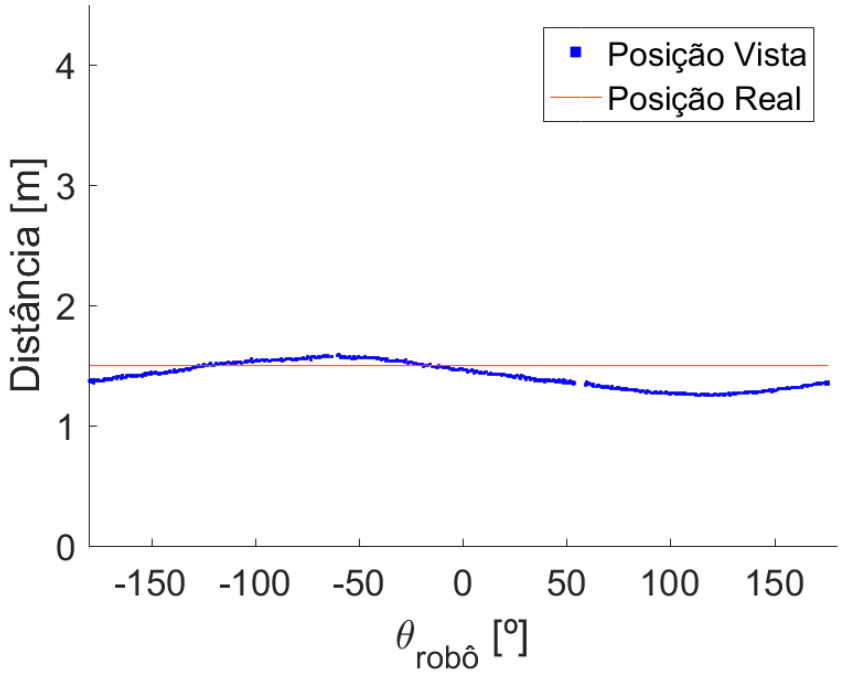


Figura 5.18: Distância observada da bola a 1.5 metros do robô em rotação.

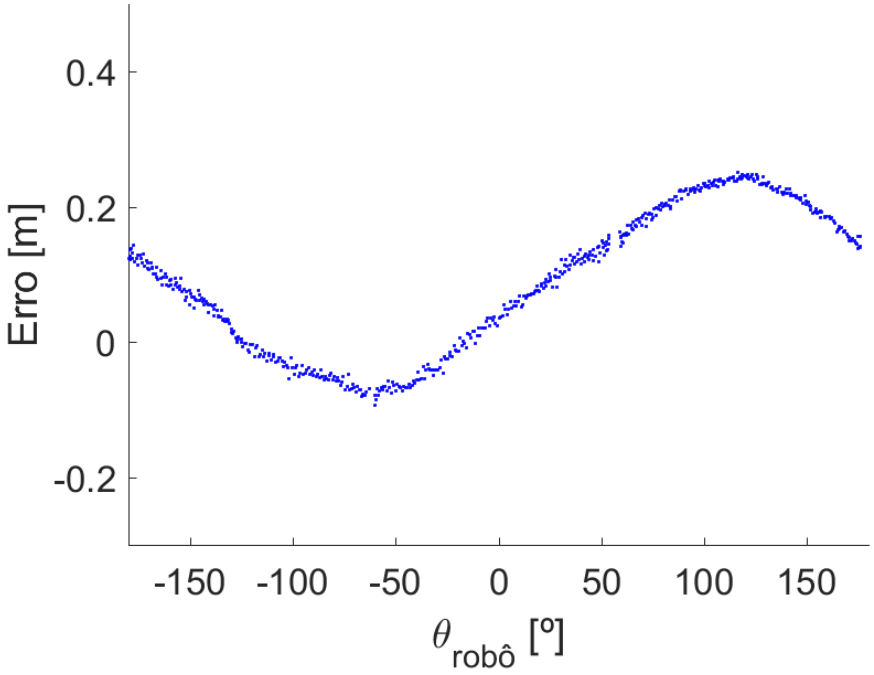


Figura 5.19: Erro da distância observada da bola a 1.5 metros.

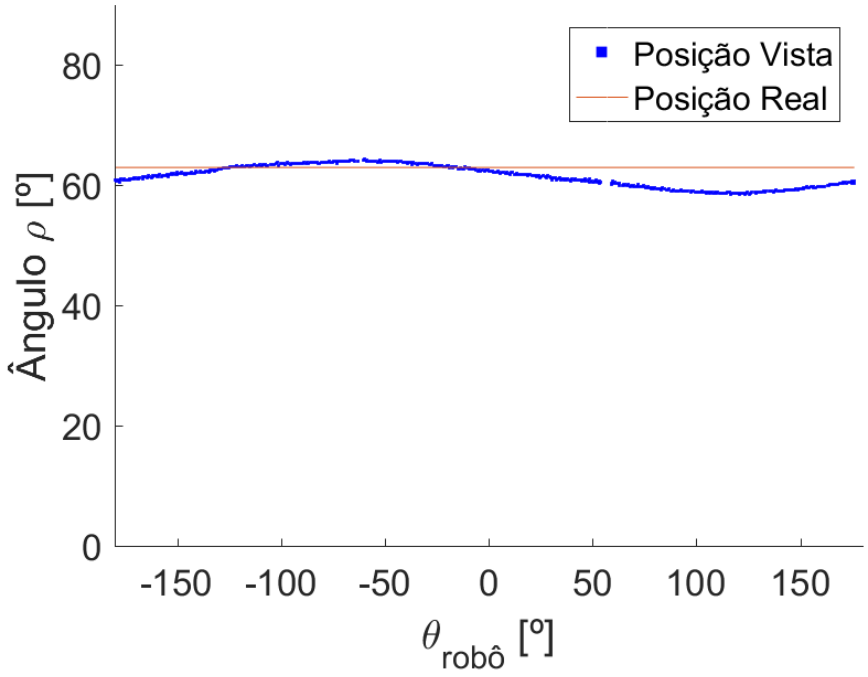


Figura 5.20: Ângulo ρ observado da bola a 1.5 metros do robô em rotação.

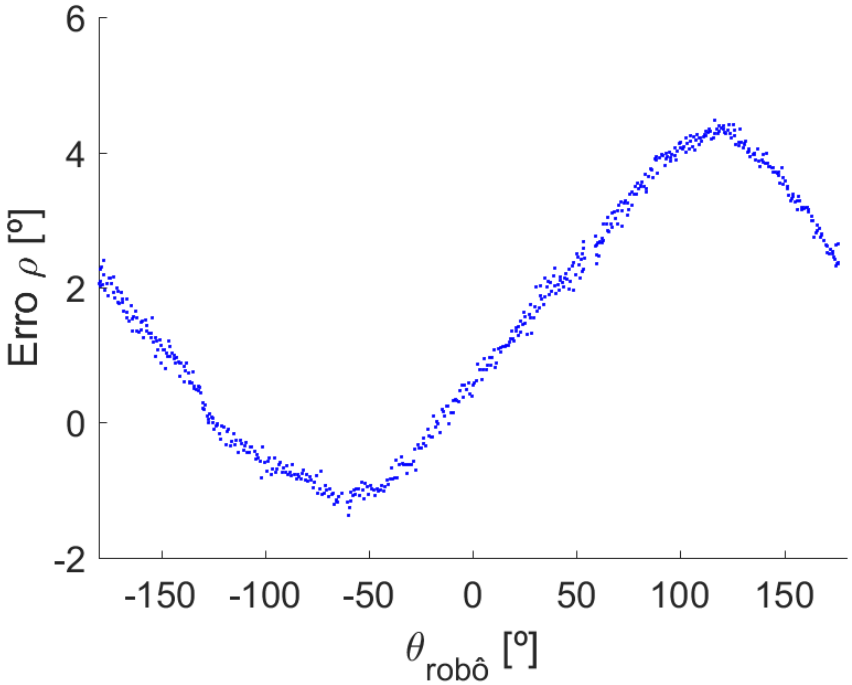


Figura 5.21: Erro do ângulo ρ observado da bola a 1.5 metros.

O erro absoluto médio de observação, quer do ângulo, quer da distância, está expresso na seguinte tabela, onde é comparado ao ensaio anterior, realizado com os mesmos parâmetros de calibração, obtidos no momento do ensaio.

Distância (m)	MAE (m)	MAE (º)
1	0.0556	1.5883
1.5	0.1183	1.9960

Tabela 5.2: MAE para o erro de observação em distância (metros) e em ângulo ρ (º), para a bola a 1.5 metros do robô, em comparação com o ensaio anterior.

Nesta situação, o que se verifica é que, ao colocar a bola a uma distância maior, o erro médio da distância observada passa de cerca de 5.6 centímetros para 11.8 centímetros, enquanto para o ângulo se regista um aumento do erro médio de cerca de 1.59º para aproximadamente 2º.

5.2.4 Bola a 2 metros com calibração recente

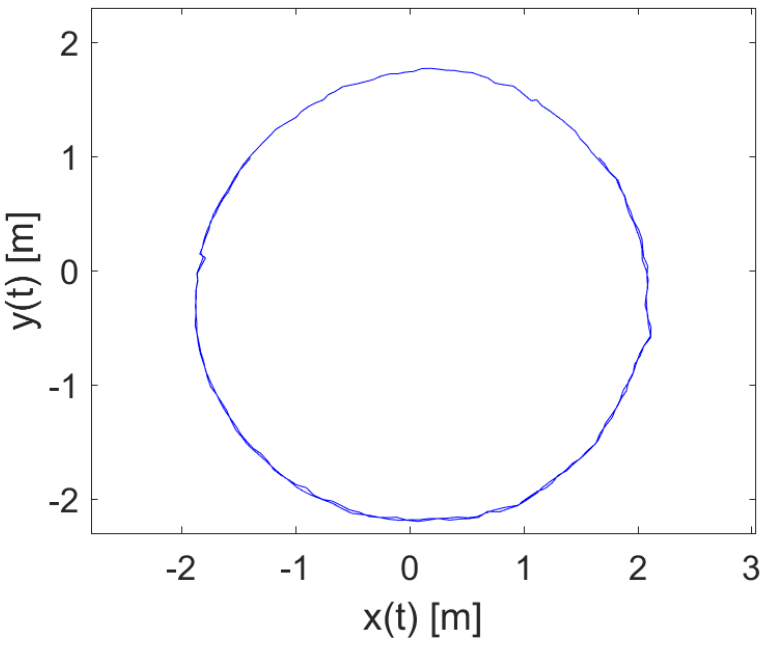


Figura 5.22: Posição observada da bola a 2 metros do robô em rotação.

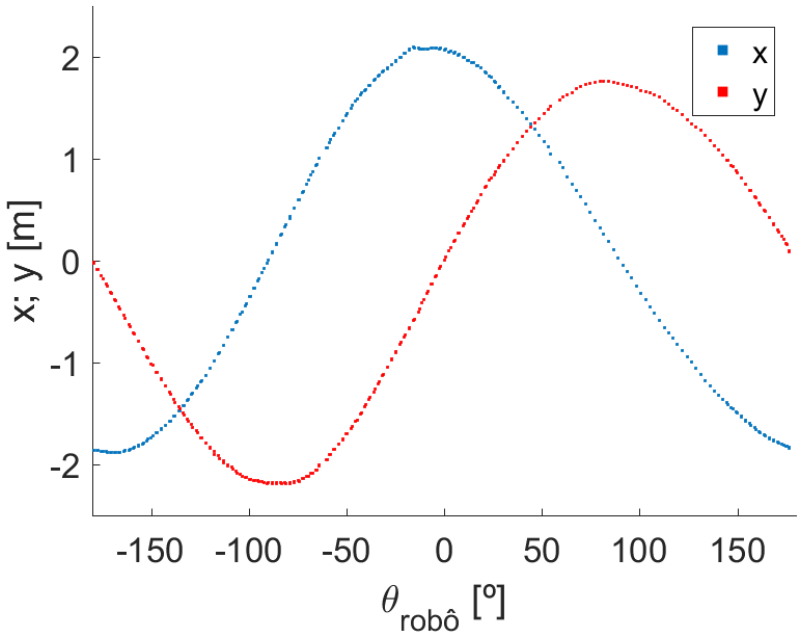


Figura 5.23: Coordenadas observadas da bola a 2 metros do robô em rotação.

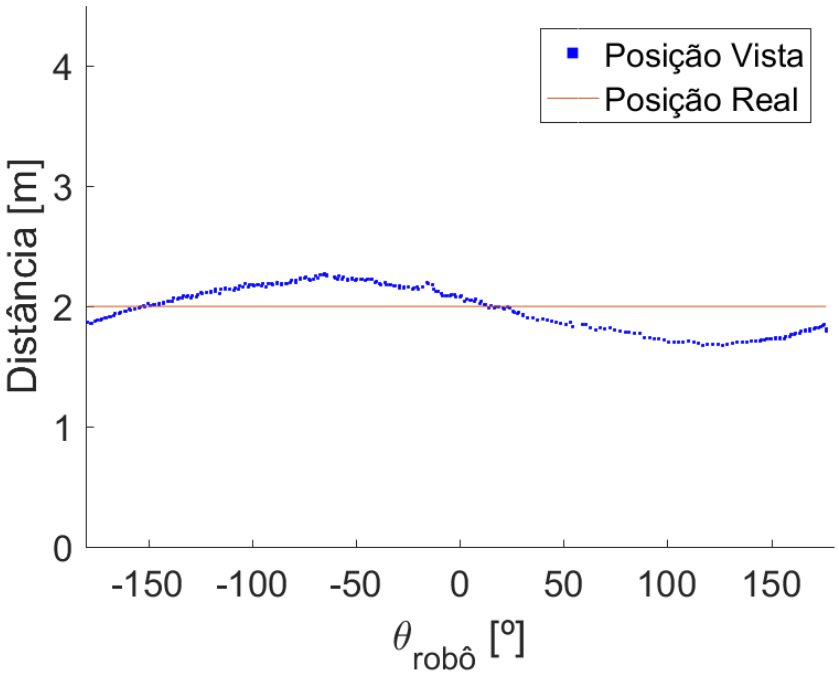


Figura 5.24: Distância observada da bola a 2 metros do robô em rotação.

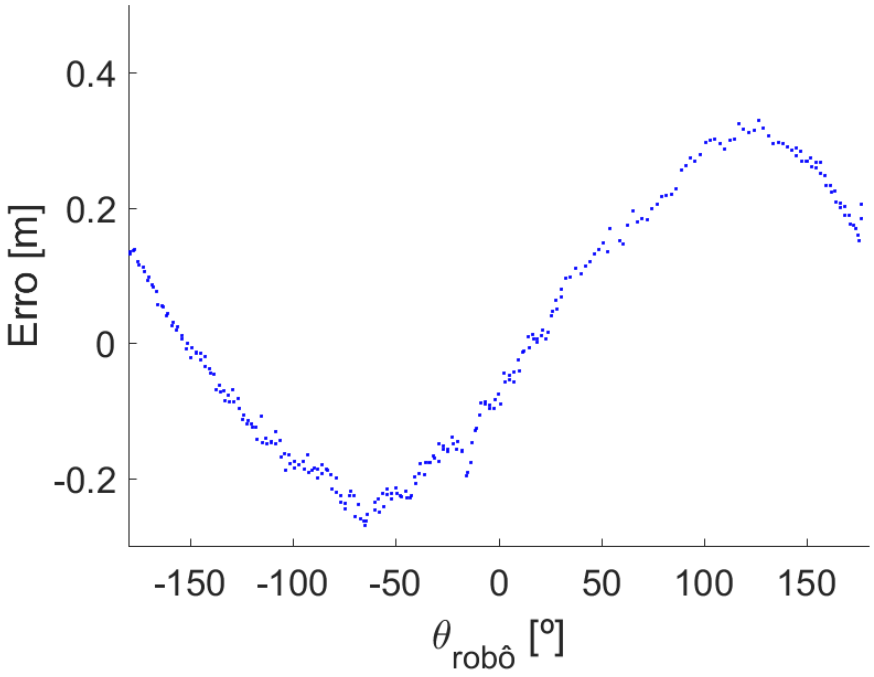


Figura 5.25: Erro da distância observada da bola a 2 metros.

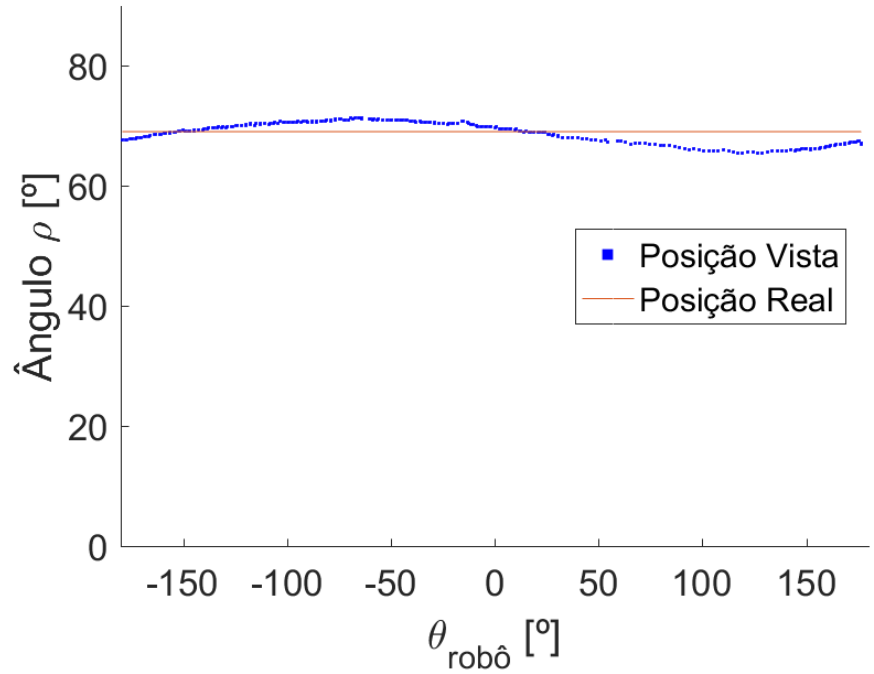


Figura 5.26: Ângulo ρ observado da bola a 2 metros do robô em rotação.

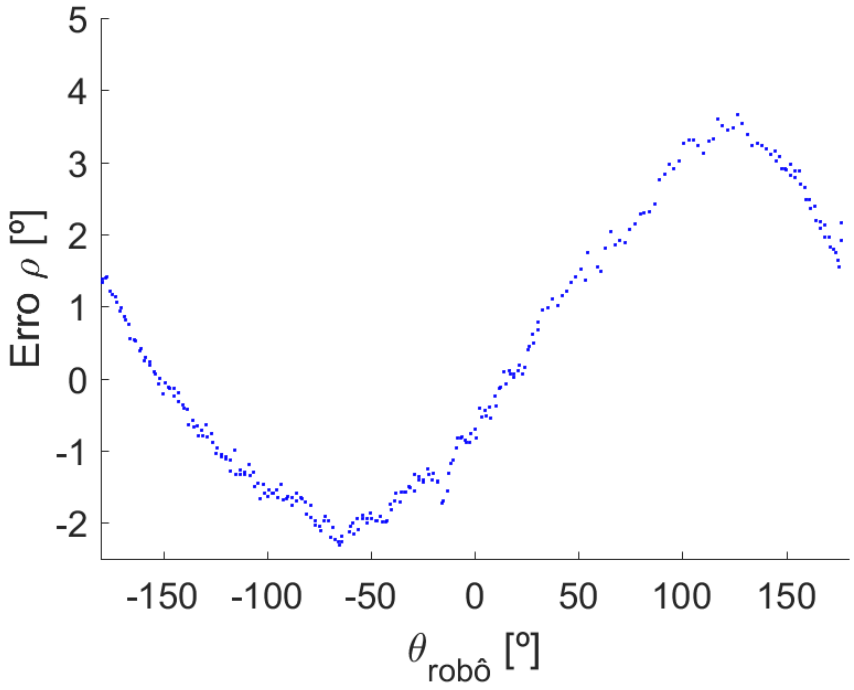


Figura 5.27: Erro do ângulo ρ observado da bola a 2 metros.

Também para este caso, foi calculado o erro absoluto médio de observação da bola. Esses valores, tal como os obtidos nos ensaios anteriores, foram compilados na tabela seguinte, para efeitos de comparação.

Distância (m)	MAE (m)	MAE (º)
1	0.0556	1.5883
1.5	0.1183	1.9960
2	0.1590	1.5551

Tabela 5.3: MAE para o erro de observação em distância (metros) e em ângulo ρ (º), para a bola a 2 metros do robô, em comparação com os ensaios anteriores.

Como se depreende da tabela 5.3, com a bola a 2 metros de distância do robô, o erro absoluto médio da distância observada passa para cerca de 16 centímetros, enquanto para o ângulo se verifica uma diminuição do erro médio para cerca de 1.5º, em relação aos dois ensaios anteriores.

5.2.5 Bola a 2.5 metros com calibração recente

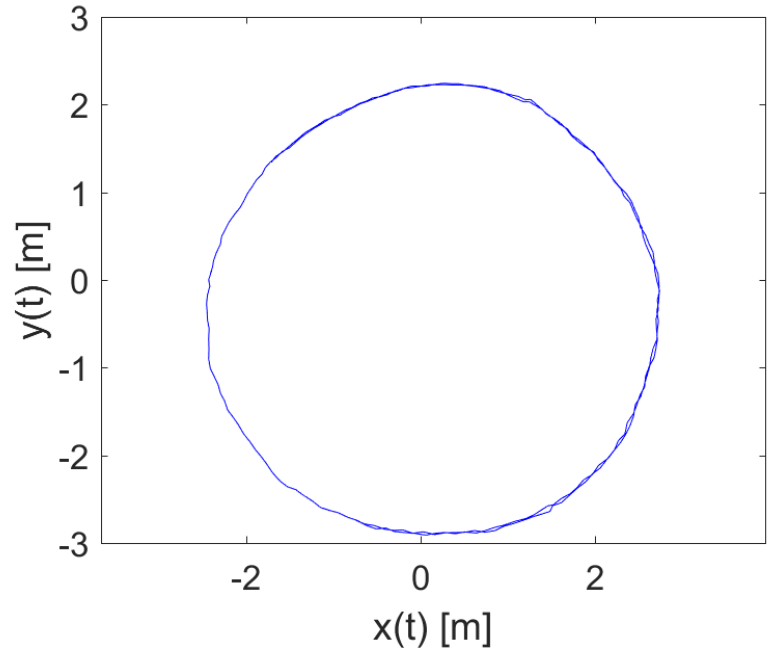


Figura 5.28: Posição observada da bola a 2.5 metros do robô em rotação.

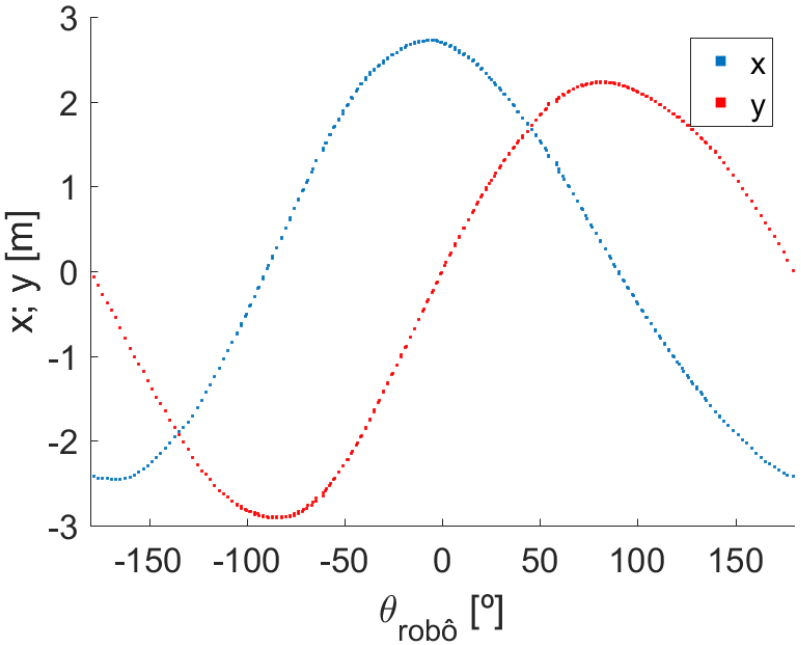


Figura 5.29: Coordenadas observadas da bola a 2.5 metros do robô em rotação.

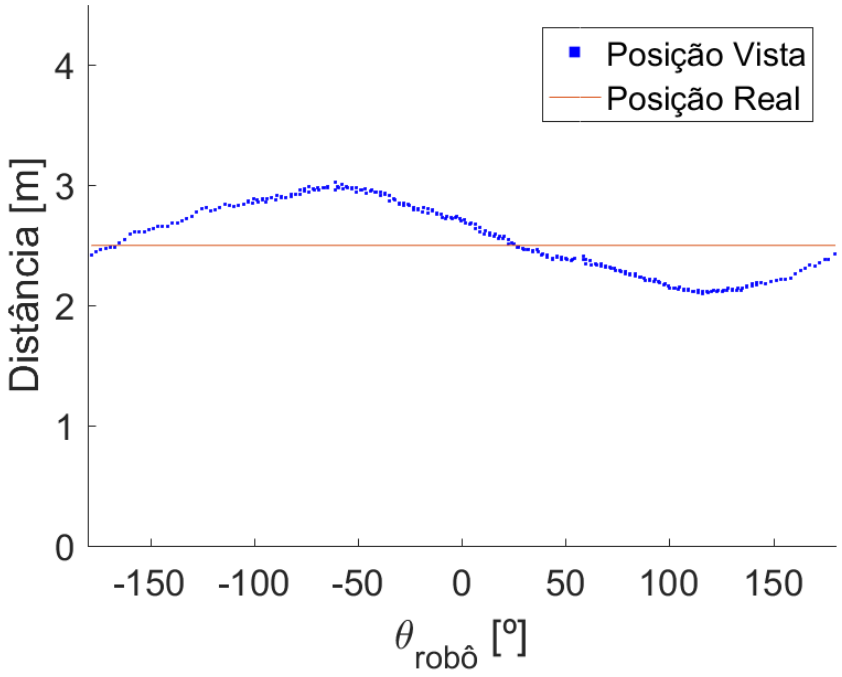


Figura 5.30: Distância observada da bola a 2.5 metros do robô em rotação.

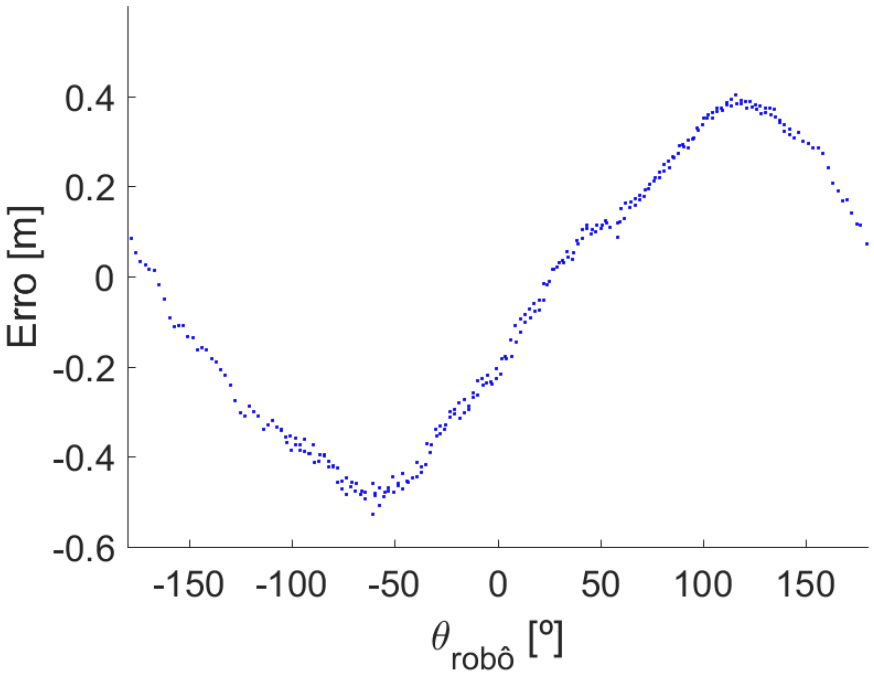


Figura 5.31: Erro da distância observada da bola a 2.5 metros.

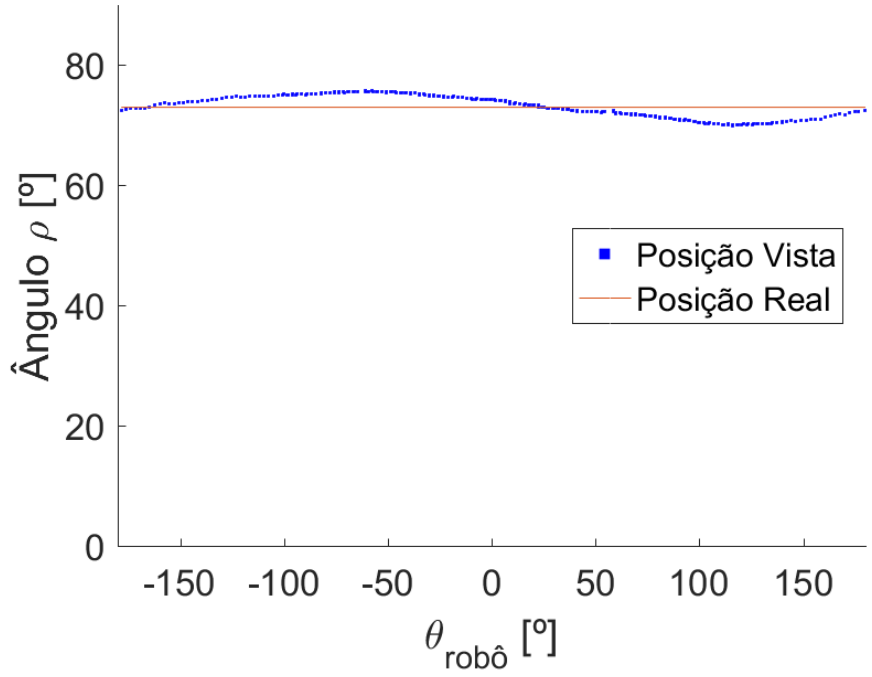


Figura 5.32: Ângulo ρ observado da bola a 2.5 metros do robô em rotação.

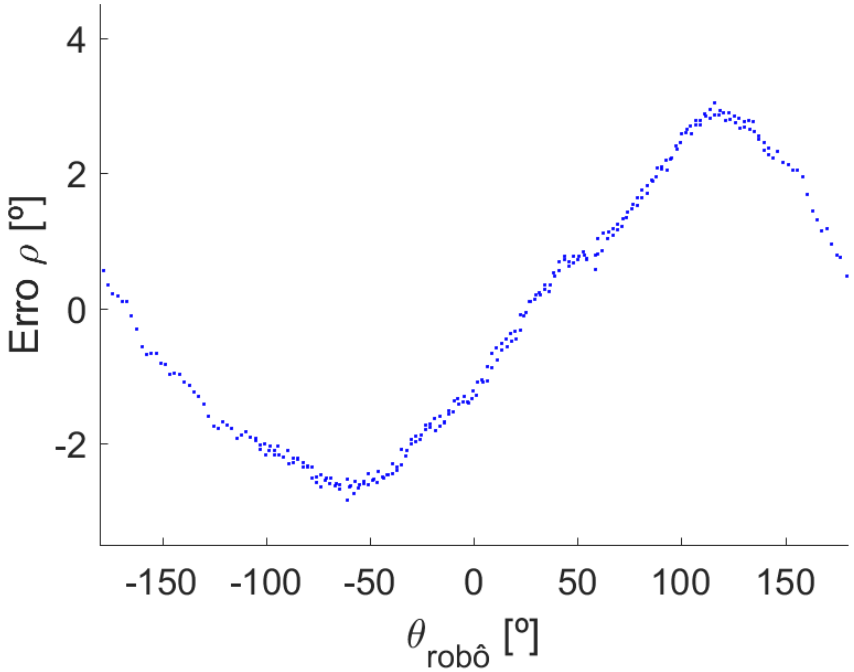


Figura 5.33: Erro do ângulo ρ observado da bola a 2.5 metros.

Distância (m)	MAE (m)	MAE (º)
1	0.0556	1.5883
1.5	0.1183	1.9960
2	0.1590	1.5551
2.5	0.2629	1.6482

Tabela 5.4: MAE para o erro de observação em distância (metros) e em ângulo ρ (º), para a bola a 2.5 metros do robô, em comparação com os ensaios anteriores.

Por outro lado, para a bola a 2.5 metros de distância do robô, que corresponde a um valor próximo da última marcação do padrão de calibração usado, confirma-se a tendência para o aumento do erro absoluto médio da distância observada com o afastamento da bola, passando para cerca de 26 centímetros. Já em relação ao ângulo, verifica-se um erro absoluto médio de cerca de 1.6º.

5.2.6 Bola a 3.5 metros com calibração recente

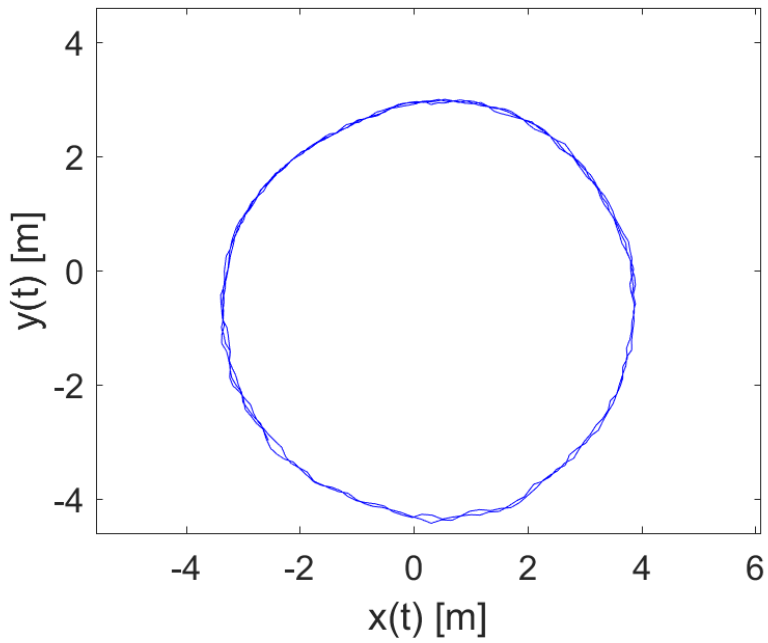


Figura 5.34: Posição observada da bola a 3.5 metros do robô em rotação.

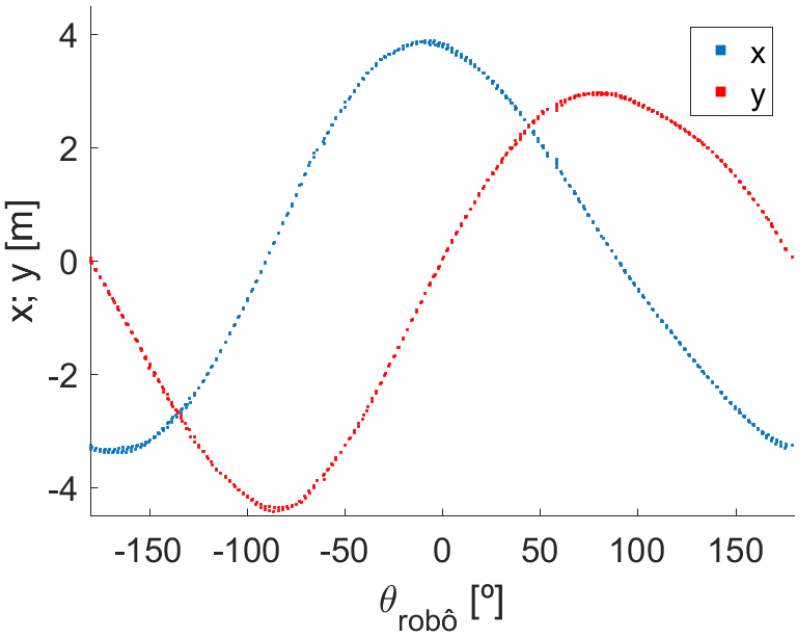


Figura 5.35: Coordenadas observadas da bola a 3.5 metros do robô em rotação.

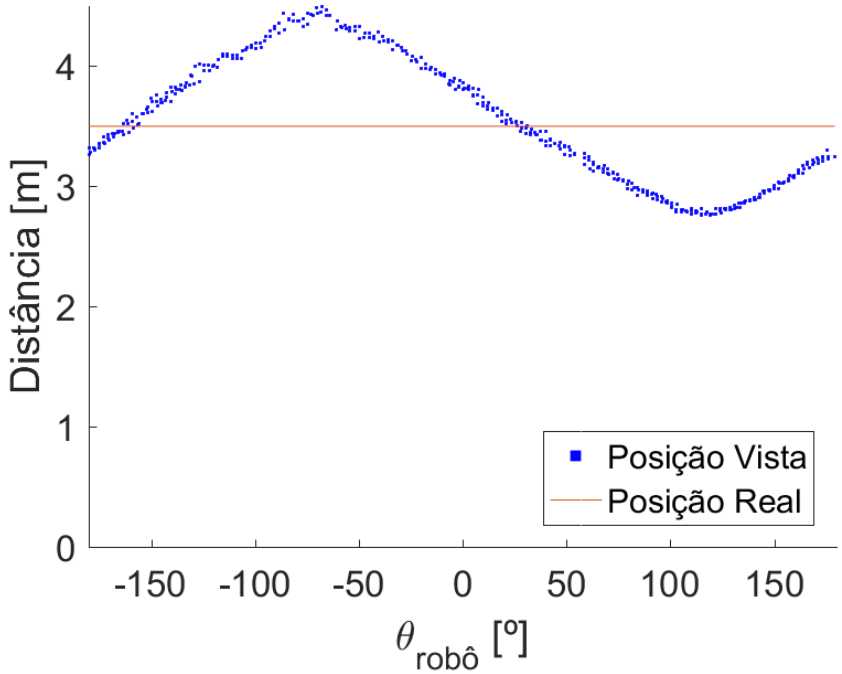


Figura 5.36: Distância observada da bola a 3.5 metros do robô em rotação.

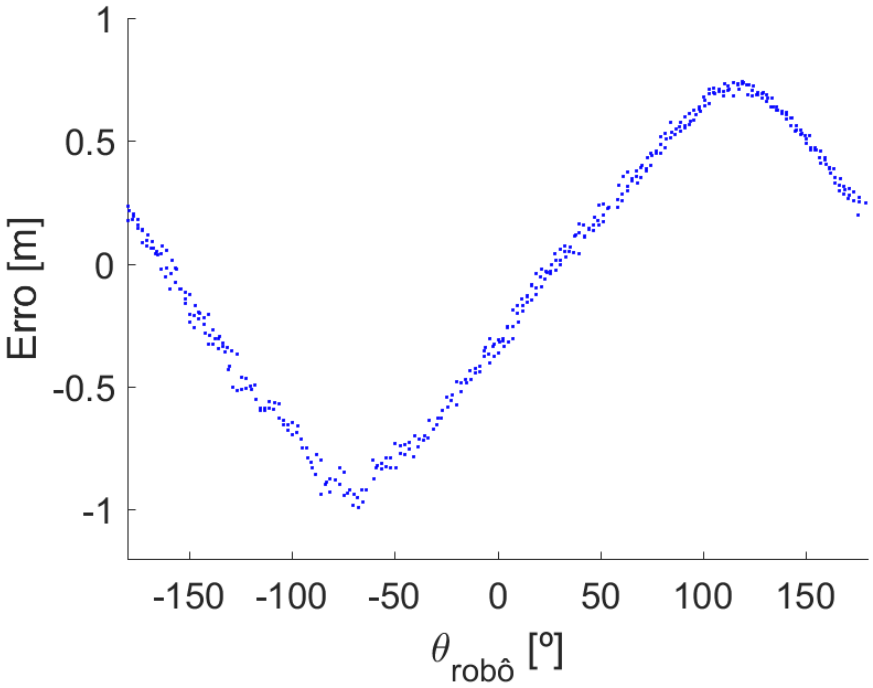


Figura 5.37: Erro da distância observada da bola a 3.5 metros.

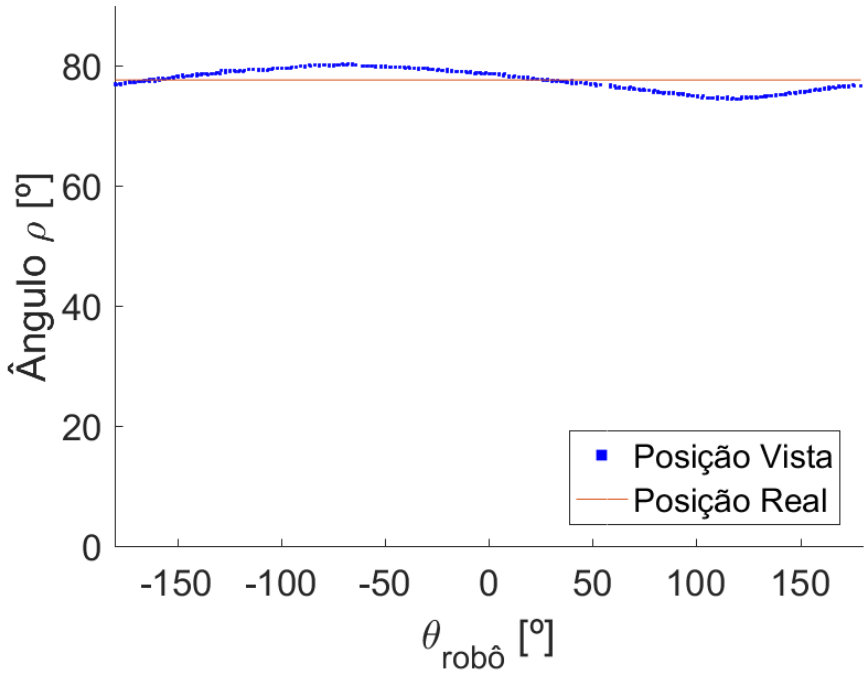


Figura 5.38: Ângulo ρ observado da bola a 3.5 metros do robô em rotação.

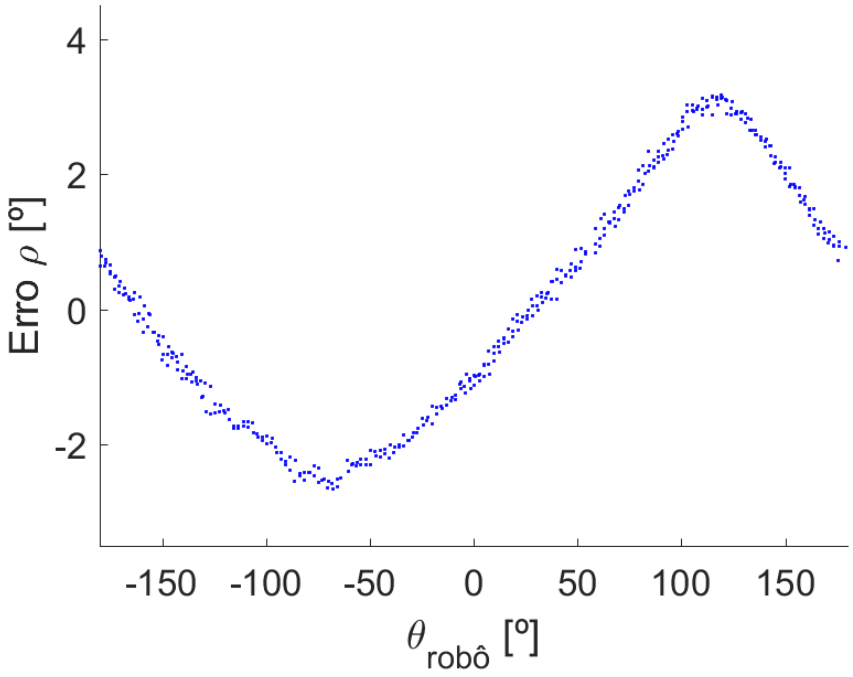


Figura 5.39: Erro do ângulo ρ observado da bola a 3.5 metros.

Distância (m)	MAE (m)	MAE (º)
1	0.0556	1.5883
1.5	0.1183	1.9960
2	0.1590	1.5551
2.5	0.2629	1.6482
3.5	0.4384	1.5104

Tabela 5.5: MAE para o erro de observação em distância (metros) e em ângulo ρ (º), para a bola a 3.5 metros do robô, em comparação com os ensaios anteriores.

Finalmente, foi colocada a bola a 3.5 metros de distância do robô, um valor bem distante para além da última transição do padrão de calibração usado, para aferir como o sistema se comporta fora dos limites do padrão de calibração utilizado.

Também neste caso, o erro absoluto médio da distância observada continua a aumentar, passando para cerca de 44 centímetros. Com estes resultados infere-se que o erro da distância observada aumenta com o afastamento em relação ao robô.

No que diz respeito ao ângulo, verifica-se um erro absoluto médio de cerca de 1.51º, o valor mais baixo de todos os ensaios realizados. Estes resultados atestam que não há uma tendência no comportamento do erro de observação do ângulo com o afastamento da bola.

5.3 Análise de Resultados

Neste capítulo, começou-se na secção 5.1 por mostrar os resultados de um teste em que, com a bola parada no centro do terreno de jogo, e colocando o robô a descrever uma trajetória de um quadrado em torno do robô, foram recolhidos dados do sistema de visão. Foram feitos ensaios utilizando o método de calibração anterior e o proposto, chegando-se à conclusão que a calibração proposta constitui uma melhoria em relação à anterior.

De seguida, na secção 5.2 foram abordados os ensaios realizados em que o robô, numa posição fixa do campo de futebol, foi colocado em rotação sobre si próprio, e, com a bola parada a distâncias conhecidas *a priori*, foram recolhidos pontos periodicamente a cada 40 milissegundos, da posição observada da bola. Para cada distância a que a bola foi colocada, foram traçados gráficos para a posição da bola em relação ao referencial local do robô, para as respetivas coordenadas x e y nesse mesmo referencial, para a distância em metros e o correspondente erro, e finalmente, para o ângulo ρ e o desvio em relação à realidade.

Para além disso, na tabela seguinte estão sintetizados os resultados obtidos para os testes efetuados.

Distância (m)	MAE (m)	MAE (^{o})
(a) 1	0.1601	4.9659
(b) 1	0.0556	1.5883
(b) 1.5	0.1183	1.9960
(b) 2	0.1590	1.5551
(b) 2.5	0.2629	1.6482
(b) 3.5	0.4384	1.5104

Tabela 5.6: MAE para o erro de observação em distância (metros) e em ângulo ρ (^{o}), para a bola a diferentes distâncias, com a calibração desatualizada (a) e com a recente (b).

Numa primeira fase, os ensaios foram feitos com a bola colocada à mesma distância em relação ao robô em rotação, usando no primeiro caso uma calibração feita há cerca de quatro semanas e considerada desatualizada, em comparação com uma calibração feita no momento do ensaio. Assim, verifica-se que, com o passar do tempo, os parâmetros vão perdendo validade e produzem piores resultados no sistema de visão. Com efeito, observando a tabela 5.6, para a distância de 1 metro com a calibração desatualizada, o erro médio absoluto é superior a todos os ensaios realizados com os parâmetros recentes. Como tal, os resultados demonstram que se deve calibrar o sistema de visão pontualmente, de preferência antes de cada jogo.

Por outro lado, usando somente a calibração feita no momento dos testes, também foram realizados outros ensaios, colocando a bola a diferentes distâncias, nomeadamente, a 1.5, 2, 2.5 e 3.5 metros, para aferir a qualidade dos resultados obtidos conforme a distância ao robô aumenta. Tal como se verifica em 5.6, o erro médio absoluto (MAE) da distância a que a bola é observada aumenta com a distância real a que a mesma se encontra. Pelo contrário, no que diz respeito ao ângulo ρ da bola, o MAE não aumenta necessariamente com o seu afastamento. Deste modo, embora o desvio em relação ao valor real de ρ da bola não aumente com o seu afastamento, verifica-se um

aumento do erro na distância observada devido ao comportamento da função tangente, presente na expressão 3.1, que permite converter ângulos em distâncias.

Ora, tendo em conta que a última transição do padrão de calibração utilizado está a cerca de 2.85 metros, estes resultados sugerem que, no futuro, possam ser utilizadas transições marcadas nas tiras a um ângulo ρ superior a 75^{o}, com vista a minimizar o erro obtido a distâncias maiores.

Para além disso, é de destacar o facto de existirem algumas descontinuidades observadas nos gráficos. Este comportamento deve-se aos momentos em que há falhas na observação da bola, pois as hastes do suporte obstruem a reflexão da bola no espelho.

Finalmente, a forma circular dos gráficos obtidos da posição da bola no referencial local, permite ainda inferir que o algoritmo utilizado para a fusão das três calibrações é um método apropriado, que produz transições suaves entre as diferentes funções.

Capítulo 6

Conclusões e Trabalho Futuro

Por fim, neste capítulo é feito um balanço do trabalho realizado, destacando as dificuldades encontradas e os objetivos alcançados. No final, sugerem-se melhorias e aspetos a ter em conta em trabalhos futuros.

6.1 Satisfação dos Objetivos

O objetivo inicial passava por estudar e implementar um método de calibração de um sistema de visão na plataforma de testes escolhida para o efeito, os robôs móveis com visão omnidirecional da equipa de futebol robótico 5DPO.

Inicialmente, definiram-se alguns conceitos e explicaram-se algumas conceções assumidas, como o ângulo em relação à vertical do robô, ρ, e o ângulo em torno do robô visto do topo, θ. De seguida, para fazer um estudo da relação matemática do sistema imagem-mundo, percecionado pelo sistema de visão, escolheu-se criar um padrão de calibração usando transições marcadas numa tira, espaçadas por cinco graus em relação ao ângulo ρ. Assim que foi terminado, tomaram-se algumas precauções, uma vez que se estava a fazer um estudo matemático do qual se pretendia retirar conclusões importantes, seria relevante reduzir os ruídos dos ensaios, como os desalinhamentos mecânicos entre a câmara e o espelho. Deu-se então início a alguns ensaios, obtendo as coordenadas na imagem das transições colocadas diante do robô na tira. Traçaram-se gráficos e avaliaram-se relações, chegando-se à conclusão que, embora as grandezas fossem não lineares, um bom candidato para função de mapeamento seria um polinómio, que fizesse corresponder distâncias na imagem ao ângulo ρ no mundo.

Tendo a escolha recaído numa função polinomial, rapidamente se percebeu que seria necessário um método de otimização, que permitisse descobrir os parâmetros da função que tornassem mais correto o ajuste da sua curva, minimizando a soma do quadrado dos desvios. Nesse sentido, estudou-se e optou-se pelo método de Levenberg-Marquardt. Foi então implementado esse algoritmo num programa que, a partir de um conjunto de pontos, estimava os parâmetros que minimizavam o erro na regressão polinomial.

Para melhorar a interação com o utilizador, o programa desenvolvido foi associado a uma interface gráfica e integrado no software dos robôs, permitindo obter dados do processamento das transições do padrão de calibração diretamente para o programa de otimização.

Aproveitando o trabalho realizado, fizeram-se testes, utilizando três tiras igualmente espaçadas em torno do robô, para perceber qual a melhor opção para o grau polinomial a usar. Chegou-se à conclusão que, devido ao baixo esforço computacional na inversão da função e melhores resultados obtidos no centro da imagem, mais perto do robô, e na parte exterior da imagem, na zona mais longínqua captada pelo sistema de visão, o polinómio a usar seria de segundo grau.

Por fim, como com o método utilizado se obtinham três funções de mapeamento, projetou-se uma forma de unificar as três calibrações numa só, e implementou-se o resultado nos robôs.

De seguida, passou-se à fase de testes e obtenção de resultados. Para aferir as possíveis melhorias em relação a anteriores métodos de calibração, fizeram-se alguns ensaios, usando as funcionalidades de estimação de posição da bola dos robôs, que é altamente dependente sistema de visão. Para além disso, para o método de calibração implementado, foram realizados testes de observação da bola, com o robô em rotação, em diferentes contextos, com parâmetros desatualizados, obtidos há um maior período de tempo, e com parâmetros obtidos no momento, para verificar a validade da calibração com o passar do tempo. Por último, a bola foi colocada a diferentes distâncias do robô, para avaliar a eficácia da calibração com o afastamento em relação ao robô.

As dificuldades encontradas ao longo da execução do trabalho prenderam-se, essencialmente, com os desalinhamentos entre os elementos do sistema de visão, nomeadamente, entre a câmara e o espelho.

Como conclusão satisfatória, pode-se afirmar ter conseguido implementar um procedimento de calibração que permite obter melhores resultados e que pode ser adotado como o novo método de calibração dos robôs da equipa de futebol robótico 5DPO.

Por outro lado, os aspetos negativos estão relacionados principalmente com o facto de este processo ser moroso e, caso haja uma colisão durante um jogo que inviabilize os parâmetros obtidos, não há forma de o robô poder ser calibrado a tempo de recuperar o sistema de visão durante esse jogo.

6.2 Trabalho Futuro

No futuro, sugere-se uma reforma da estrutura dos robôs, como por exemplo, colocar um tubo de acrílico envolvendo o espelho e a câmara, à semelhança do que é feito noutras equipas de futebol robótico, para garantir que, mesmo que haja colisões durante os jogos, e consequente deformação mecânica de alguma parte do robô, o alinhamento entre a câmara e o espelho se mantém.

Para além disso, seria positivo aumentar a distância das transições detetadas, sugerindo-se em alternativa, que se coloque apenas uma tira de maior comprimento e, com o robô em rotação, se obtenham mais funções para diferentes direções, para além das três já utilizadas. Por exemplo, com a tira disposta à sua frente, o robô poderia rodar em intervalos de θ igual a 30^{o}, originando assim doze funções de mapeamento, tornando, à partida, os resultados mais fidedignos.

Outra sugestão surge na sequência de evitar a morosidade do processo de calibração. Em alternativa, seria interessante poder fazer um ajuste dos parâmetros pontualmente, em situações em que a calibração não cumpra determinados critérios. Assim, quando o sistema de visão do robô estivesse a obter dados incorretos, este poderia deslocar-se a uma zona do campo e pelas linhas que delimitam o terreno de jogo, proceder à calibração.

Anexo A

Dados da Aplicação

A.1 Aplicação com Resultados nas Tabelas

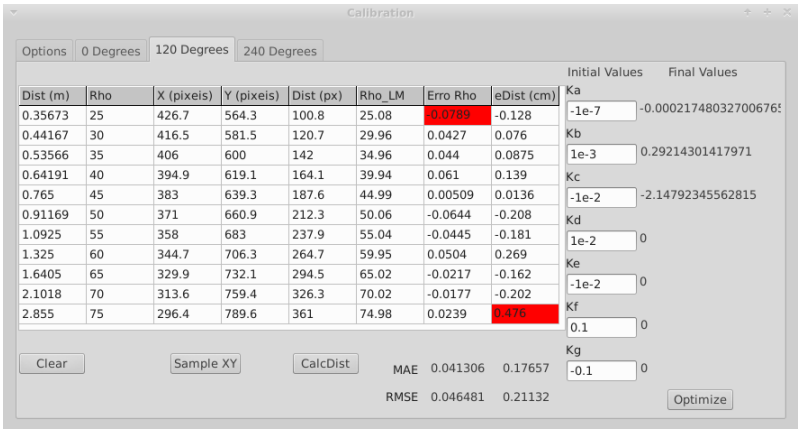


Figura A.1: Aba do programa com valores na tabela para θ na direção de 120º.

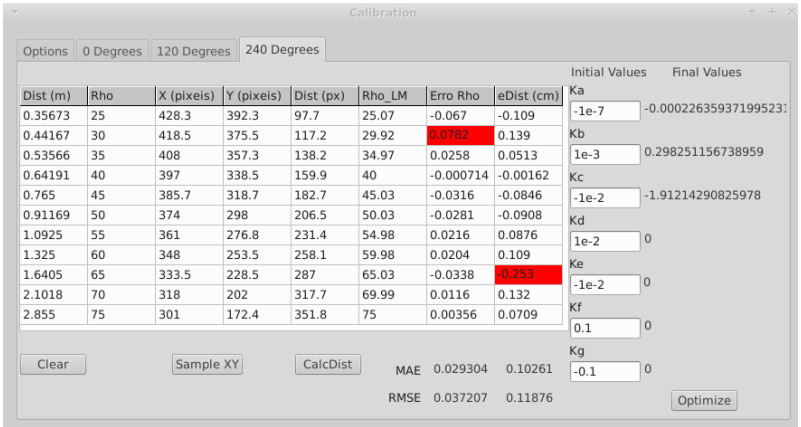


Figura A.2: Aba do programa com valores na tabela para θ na direção de -120º.

A.2 Polinómios de Mapeamento de Diferentes Ordens

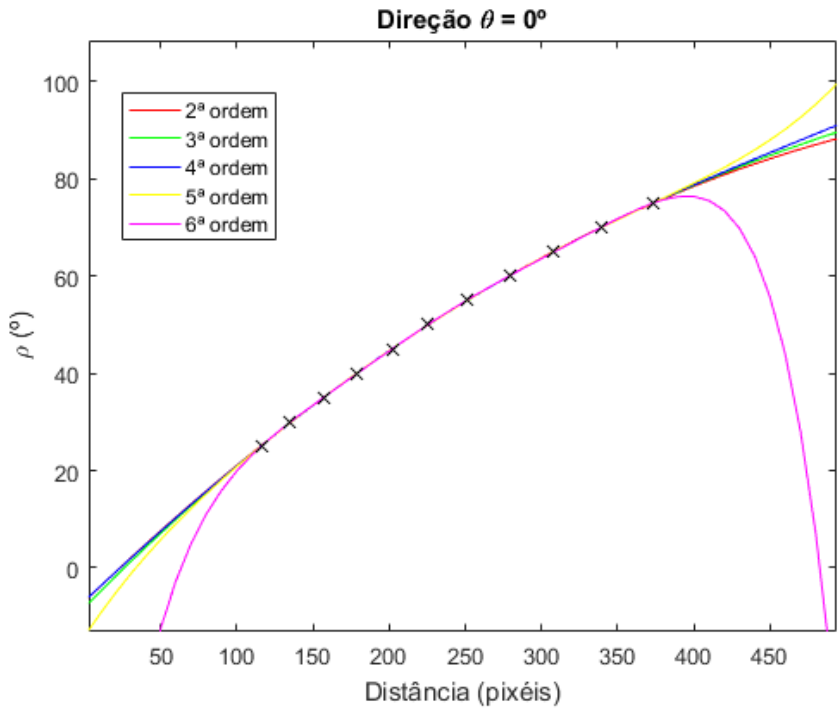


Figura A.3: Função na direção de 0º.

Ordem Polinomial	MAE	RMSE
2	0.0899	0.1042
3	0.0798	0.0951
4	0.0787	0.0941
5	0.0807	0.0919
6	0.0429	0.0490

Tabela A.1: Direção 0º - MAE e RMSE para o erro do ângulo ρ (graus).

Ordem Polinomial	MAE	RMSE
2	0.4226	0.5679
3	0.2572	0.2804
4	0.2585	0.2819
5	0.3243	0.3764
6	0.1941	0.2499

Tabela A.2: Direção 0º - MAE e RMSE para o erro em distância (centímetros).

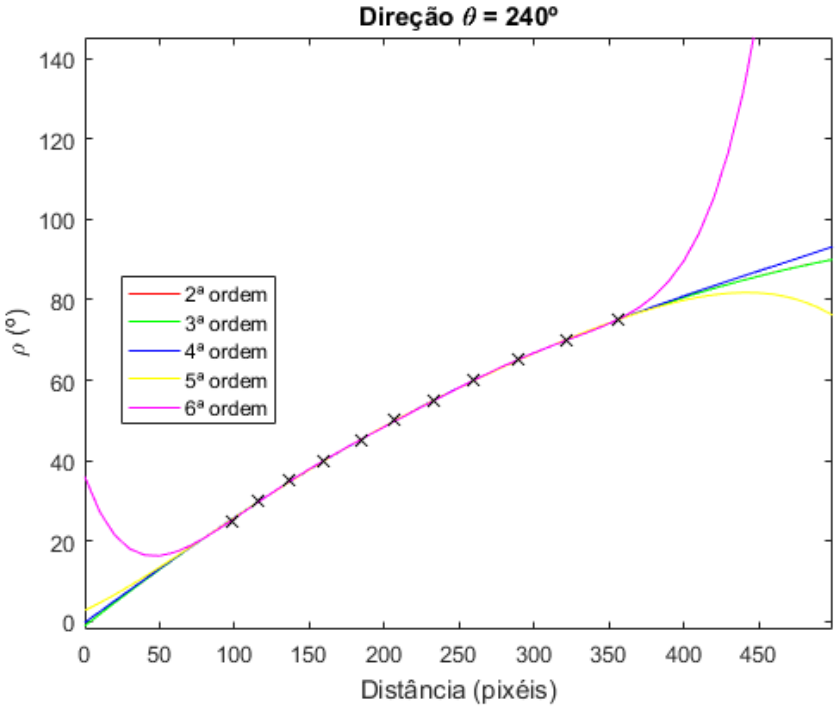


Figura A.4: Função na direção de -120º.

Ordem Polinomial	MAE	RMSE
2	0.1745	0.2305
3	0.1510	0.2159
4	0.1568	0.1927
5	0.0986	0.1306
6	0.0716	0.1050

Tabela A.3: Direção -120º - MAE e RMSE para o erro do ângulo ρ (graus).

Ordem Polinomial	MAE	RMSE
2	0.5830	0.7239
3	0.4613	0.5879
4	0.6549	0.7751
5	0.4877	0.6530
6	0.2222	0.2975

Tabela A.4: Direção -120º - MAE e RMSE para o erro em distância (centímetros).

A.3 Código de Função de Transformação 2D-3D

```
procedure CamUVToRoTeta(u,v: double; out ro,teta: double);
var
  dt, du, dv, da: double;
  theta0, theta120, theta240: double;
  tx, t: double;
begin
  theta0:= 0;
  theta120:= 120 *pi()/180;
  theta240:= -120 *pi()/180;

  du:= u-Camera.cu;
  dv:= Camera.cv-v;
  dt:= sqrt(du*du + dv*dv);

  if dt<1e-3 then begin
    ro:=0;
    teta:=0;
  end else
  begin
   teta:= -atan2(dv, du);
   if (teta >= theta0) and (teta <= theta120) then begin
     tx:= theta120 - theta0;
     t:= (teta - theta0)/tx;
     ro:= min(87.7, (1-t)*((K.Ka0*dt + K.Kb0)*dt + K.Kc0) + t*((
        K.Ka120*dt + K.Kb120)*dt + K.Kc120));
   end else
   if (teta >= theta240) and (teta < theta0) then begin
     tx:= theta0 - theta240;
     t:= (teta - theta240)/tx;
     ro:= min(87.7, (1-t)*((K.Ka240*dt + K.Kb240)*dt + K.Kc240)
        + t*((K.Ka0*dt + K.Kb0)*dt + K.Kc0));
   end else
   if (teta < theta240) or (teta > theta120) then begin
     tx:= 2*pi() + theta240 - theta120;
     if teta < 0 then begin
       t:= (theta240 - teta)/tx;
       ro:= min(87.7, (1-t)*((K.Ka240*dt + K.Kb240)*dt + K.Kc240
          ) + t*((K.Ka120*dt + K.Kb120)*dt + K.Kc120));
```

```
      end else
      if teta > 0 then begin
        t:= (teta - theta120)/tx;
        ro:= min(87.7, (1-t)*((K.Ka120*dt + K.Kb120)*dt + K.Kc120
            ) + t*((K.Ka240*dt + K.Kb240)*dt + K.Kc240));
      end;
    end;
  end;
  ro:= ro*pi()/180;
end;
```

Referências

[1] HumanoidSoccer [online]. URL: https://cosmos-magazine.imgix.net/file/spina/photo/11365/1708012{_}robocup2017{_}2-824445706.jpg?fit=clip{&}w=835 [último acesso em 2018-02-04].

[2] Pedro Miguel da Silva Rocha Relvas. Fusão sensorial e cooperação em equipas de robôs móveis. Tese de mestrado, Faculdade de Engenharia da Universidade do Porto, jul 2017. URL: https://repositorio-aberto.up.pt/handle/10216/105366.

[3] Hiroaki Kitano, Minoru Asada, Yasuo Kuniyoshi, Itsuki Noda, e Eiichi Osawa. RoboCup. Em *Proceedings of the first international conference on Autonomous agents - AGENTS '97*, páginas 340–347, New York, New York, USA, 1997. ACM Press. URL: http://portal.acm.org/citation.cfm?doid=267658.267738, doi:10.1145/267658.267738.

[4] Bernardo Cunha, Jose Azevedo, e Nuno Lau. Calibration of Non-SVP Hyperbolic Catadioptric Robotic Vision Systems. Em *Robot Vision*. InTech, mar 2010. URL: http://www.intechopen.com/books/robot-vision/calibration-of-non-svp-hyperbolic-catadioptric-robotic-vision-systems, doi:10.5772/9303.

[5] João Pedro Ribeiro Morais. Calibração adaptativa e extração de características de um sistema catadióptrico. Tese de mestrado, Faculdade de Engenharia da Universidade do Porto, Julho 2017.

[6] Objective. URL: http://www.robocup.org/objective [último acesso em 2018-02-03].

[7] RoboCupSoccer. URL: http://www.robocup.org/domains/1 [último acesso em 2018-02-03].

[8] start - 5dpo Robotics (FEUP, Portugal). URL: https://web.fe.up.pt/{~}robosoc/en/doku.php [último acesso em 2018-02-03].

[9] RulesBook. URL: http://wiki.robocup.org/images/4/41/Rulebook{_}MSL-19.1.pdf [último acesso em 2018-02-04].

[10] Middle Size League - RoboCup Wiki. URL: http://wiki.robocup.org/Middle{_}Size{_}League{#}Rules [último acesso em 2018-02-03].

[11] João P. Barreto e Helder Araujo. Geometric properties of central catadioptric line images and their application in calibration. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 27(8):1327–1333, aug 2005. URL: http://ieeexplore.ieee.org/document/1453519/, doi:10.1109/TPAMI.2005.163.

[12] Bernardo Cunha, José Azevedo, Nuno Lau, e Luis Almeida. Obtaining the Inverse Distance Map from a Non-SVP Hyperbolic Catadioptric Robotic Vision System. Em *RoboCup 2007: Robot Soccer World Cup XI*, páginas 417–424. Springer Berlin Heidelberg, Berlin, Heidelberg, 2007.

[13] Tiago Pereira. Calibration of Catadioptric Vision Systems. masters, Faculdade de Engenharia da Universidade do Porto, Julho 2012.

[14] José Alexandre de França, Maria Bernadete de Morais França, Marcela Hitomi Koyama, e Tiago Polizer da Silva. Uma implementação do algoritmo Levenberg-Marquardt dividido para aplicações em visão computacional. *Semina: Ciências Exatas e Tecnológicas*, 30(1):51, jul 2009.

[15] André Vitor Bosisio Moura Pedra, Marcio Mendonça, Marco Antonio Ferreira Finocchio, Lúcia Valéria Ramos de Arruda, e José Eduardo Cogo Castanho. Camera Calibration Using Detection and Neural Networks. *IFAC Proceedings Volumes*, 46(7):245–250, may 2013.

[16] Gil Lopes, Fernando Ribeiro, e Nino Pereira. Catadioptric System Optimisation for Omnidirectional RoboCup MSL Robots. Em *Lecture Notes in Computer Science (LNCS)*, volume 7416 LNCS, páginas 318–328. Springer Verlag, 2012. URL: http://link.springer.com/10.1007/978-3-642-32060-6{_}27, doi:10.1007/978-3-642-32060-6_27.

[17] Kenneth Levenberg. A Method for the Solution of Certain Non-Linear Problems in Least Squares. *Quarterly of Applied Mathematics*, 2:164–168, 1944.

[18] Donald W. Marquardt. An Algorithm for Least-Squares Estimation of Nonlinear Parameters. *Journal of the Society for Industrial and Applied Mathematics*, 11(2):431–441, jun 1963. URL: http://epubs.siam.org/doi/10.1137/0111030, doi:10.1137/0111030.

[19] André Girard. Excerpt from Revue d'optique théorique et instrumentale. *Rev. Opt*, 37:225–241, 397–424, 1958.

[20] C G Wynne. Lens Designing by Electronic Digital Computer: I. *Proceedings of the Physical Society*, 73(5):777–787, may 1959. URL: http://stacks.iop.org/0370-1328/73/i=5/a=310?key=crossref.a54c46778a21b2f084326f1f2bd146c9, doi:10.1088/0370-1328/73/5/310.

[21] David D. Morrison. Methods for nonlinear least squares problems and convergence proofs. *Proceedings of the Jet Propulsion Laboratory Seminar on Tracking Programs and Orbit Determination*, páginas 1–9, 1960.

[22] Christian Kanzow, Nobuo Yamashita, e Masao Fukushima. Levenberg–Marquardt methods with strong local convergence properties for solving nonlinear equations with convex constraints. *Journal of Computational and Applied Mathematics*, 172(2):375–397, dec 2004. URL: http://linkinghub.elsevier.com/retrieve/pii/S0377042704001256, doi:10.1016/j.cam.2004.02.013.

[23] Jose Pujol. The solution of nonlinear inverse problems and the Levenberg-Marquardt method. *GEOPHYSICS*, 72(4):W1–W16, jul 2007. URL: http://library.seg.org/doi/10.1190/1.2732552, doi:10.1190/1.2732552.

[24] Philip E. Gill e Walter Murray. Algorithms for the Solution of the Nonlinear Least-Squares Problem. *SIAM Journal on Numerical Analysis*, 15(5):977–992, oct 1978. URL: http://epubs.siam.org/doi/10.1137/0715063, doi:10.1137/0715063.

[25] C. T. Kelley. *Iterative methods for optimization*. SIAM, 1999.

[26] Jorge. Nocedal e Stephen J. Wright. *Numerical optimization*. Springer, 2nd edição, 2006.

[27] Tilo Strutz. *Data fitting and uncertainty : a practical introduction to weighted least squares and beyond*. Springer Vieweg, 2016.

[28] Jorge J. Moré e Daniel C. Sorensen. Computing a Trust-Region Step. *SIAM J. Sci. Stat. Comput.*, páginas 553–572, 1983.

[29] Joachim Wuttke. lmfit – a C library for Levenberg-Marquardt least-squares minimization and curve fitting, 2004-2013. URL: http://apps.jcns.fz-juelich.de/lmfit.

[30] Miguel Cárdenas-Montes. Sobreajuste - Overfitting. URL: http://wwwae.ciemat.es/{~}cardenas/docs/lessons/sobreajuste.pdf.