FACULDADE DE ENGENHARIA DA UNIVERSIDADE DO PORTO

Calibração adaptativa e extração de características de um sistema catadióptrico

João Pedro Ribeiro Morais

Mestrado Integrado em Engenharia Eletrotécnica e de Computadores

Orientador: Paulo José Cerqueira Gomes da Costa

Coorientador: António Paulo Gomes Mendes Moreira

17 de Julho de 2017

Resumo

Um sistema de visão catadióptrico é constituído por um conjunto câmara-espelho em que, por exemplo, o espelho tem a forma de um parabolóide. Isso permite dotar o sistema de uma área de visão elevada, pagando o preço de um nível de distorção considerável. Nos casos em que a imagem vai ser diretamente processada sem intervenção humana a distorção resultante não é crítica desde que seja adequadamente compensada. Esta distorção é altamente não linear e deve levar em conta os efeitos do desalinhamento dos eixos câmara-espelho. Numa situação em que essa geometria pode variar, é necessário um mecanismo adaptativo para lidar com essa dinâmica. Outro desafio, neste tipo de sistemas, está relacionado com os algoritmos de extração de características que devem ser modificados para lidar com a distorção presente: quer fazendo modificações nesses algoritmos que lidem com o facto de que figuras como retas e círculos perdem muitas das suas características típicas, quer estabelecendo mecanismo de retificação da imagem que permitam usar os algoritmos nas suas formas clássicas.
Nesta dissertação serão expostos os modelos matemáticos dos componentes destes sistemas bem como o modelo matemático de visão. Este modelo terá em conta a reflexão dos raios no espelho.
Por fim será mostrado de que forma um sistema deste tipo pode ser calibrado através de uma rede neuronal.

Abstract

A catadioptric system is composed of a camera-mirror pair, where the mirror has a parabolic shape. This endows the system with a high area of vision, although with a considerable level of distortion. In the cases where the image will be directly processed without human intervention this distortion isn't critical unless it's compensated. This is a non-linear distortion and one must take into account the effects of the misalignment of the camera mirror axis. In a situation where that geometry can vary it's necessary to have a dynamic model to deal with those dynamics. Another challenge on this type of systems is that the algorithms used to extract characteristics must be modified to deal with the distortion: either by making changes to those algorithms that deal with the fact that shapes like straight lines and circles lose many of their typical characteristics, or by establishing an image rectification mechanism that allows the use of the algorithms in their traditional forms.

In this dissertation the mathematical models of the components of those systems will be exposed as well as the mathematical model of vision. This model is based on the reflection of the rays in the mirror.

Finally it will be shown how a system of this type can be calibrated through a neural network.

Agradecimentos

"Aqueles que passam por nós, não vão sós, não nos deixam sós. Deixam um pouco de si, levam um pouco de nós."

No sucesso e no insucesso existem pessoas que se distinguem na nossa vida pelas mais diversas razões.

À Faculdade de Engenharia, por ter sido uma casa que me acolheu nos últimos anos, educando-me, e pela oportunidade de convívio que me possibilitou com colegas e professores excelentes.

Ao professor Paulo Gomes da Costa e ao professor António Paulo Moreira, pela orientação e apoio dados durante este projeto e pelos conhecimentos que me transmitiram.

À Tuna de Engenharia da Universidade do Porto, por todos os momentos de grande amizade, acompanhados sempre de música e animação. Fiz grandes amigos e vivi momentos que sem dúvida levarei para a vida.

Ao agrupamento de escuteiros 10 de Cedofeita, e a todos os elementos que o compõem, pela escola de vida que foi, e continua a ser. Ensinou-me a ser uma pessoa e cidadão melhor a cada dia, tendo sido um contributo fundamental no meu desenvolvimento.

A todos os meus amigos, em especial ao Ricardo Figueiredo, João Brandão, Sérgio Pinto, Inês Faria, Raquel Ribeiro, Raquel Kritinas, Sofia Malheiro, Sara Lemos, João Lemos, Clarisse Alemão, Francisco Ferreira, João Reis, Pedro Casanova, Pedro Relvas, Pedro Afonso, Sérgio Matos e Inês Barbosa por todo o apoio e motivação para a concretização desta dissertação e por serem as pessoas excelentes que são.

Aos meus pais, Madalena e José Alberto, por todos os valores e educação que me deram e por me apoiarem incondicionalmente em todos os momentos.

Ao meu avô Morais, que continua presente na minha vida sempre que lhe é possível.

À minha avó Lóló, avó Maria e avô Hilário, que de certo teriam muito gosto em ler esta dissertação.

A Deus, pelo dom da sapiência.

João Pedro Ribeiro Morais

"Eis o que eu considero triunfo: ser feliz.
Mas a felicidade não é apenas passiva; quer dizer,
não se alcança sentando-se a gente à espera dela;
isso seria coisa insignificante – o prazer"

Robert Baden-Powell

Conteúdo

Lista de Figuras

Lista de Tabelas

Abreviaturas e Símbolos

2-D	Duas dimensões
3-D	Três dimensões
4-D	Quatro dimensões
FANN	*Fast Artificial Neural Network Library*
FEUP	Faculdade de Engenharia da Universidade do Porto
OpenCV	*Open Source Computer Vision Library*
RGB	*Red-Green-Blue*
SVP	*Single View-Point*

u	Primeira coordenada do plano da imagem
v	Segunda coordenada do plano da imagem
f	Distância focal da câmara
λ	Parâmetro da câmara que descreve a correspondência de um ponto da imagem e pontos infinitos no referencial da câmara
x_c	Coordenada em x no referencial da câmara
y_c	Coordenada em y no referencial da câmara
z_c	Coordenada em z no referencial da câmara
u_0	Primeira coordenada do *principal point*
v_0	Segunda coordenada do *principal point*
u_c	Primeira coordenada do centro da imagem
v_c	Segunda coordenada do centro da imagem
u_r	Primeira coordenada com a origem das coordenadas no centro da imagem
v_r	Segunda coordenada com a origem das coordenadas no centro da imagem
α	Distorção dos pixels
λ_u	Raio sem considerar a distorção
λ_x	Coordenada em x do raio, sem considerar distorção, no referencial da câmara
λ_y	Coordenada em y do raio, sem considerar distorção, no referencial da câmara
λ_c	Centro de distorção
λ_{cx}	Coordenada em x do centro de distorção
λ_{cy}	Coordenada em x do centro de distorção
λ_d	Raio depois de considerar a distorção
x_m	Coordenada em x no referencial do espelho
y_m	Coordenada em y no referencial do espelho
z_m	Coordenada em z no referencial do espelho
a	Parâmetro do espelho
b	Parâmetro do espelho

R_m^c	Transformação rotacional do espelho para a câmara
θ_m	Ângulo de rotação do espelho em torno do eixo y
ψ_m	Ângulo de rotação do espelho em torno do eixo x
T_m	Distância entre a câmara e o referencial do espelho, expresso no sistema de coordenadas da câmara
H_m^c	Transformação homogénea do espelho para a câmara
R_c^m	Tranformação rotacional da câmara para o espelho
H_c^m	Transformação homogénea da câmara para o espelho
R_c^w	Transformação rotacional da câmara para o mundo
θ_c	Ângulo de rotação da câmara em torno do eixo y
ψ_c	Ângulo de rotação da câmara em torno do eixo x
ϕ_c	Ângulo de rotação da câmara em torno do eixo z
T_c	Distância entre a câmara e o referencial do mundo, expresso no sistema de coordenadas do mundo
λ_{cm}	Ponto do raio no referencial da câmara
λ_m	Ponto do raio no referencial do espelho
$\vec{\lambda}_m'$	Orientação do raio no referencial do espelho
λ_m''	Centro otico da câmara no referencial do espelho
$\vec{N}$	Vetor normal de superfície no referencial do espelho
$r_i\vec{m}$	Orientação do raio incidente no referencial do espelho
$\vec{r_{rm}}$	Orientação do raio refletido no referencial do espelho
γ	Ângulo genérico entre dois vetores
x_x	Coordenada real, em x, do painel de xadrez no sistema de coordenadas do mundo
y_x	Coordenada real, em y, do painel de xadrez no sistema de coordenadas do mundo
z_x	Coordenada real, em z, do painel de xadrez no sistema de coordenadas do mundo
h_e	Altura desde o chão à base do espelho
u_N	Primeira coordenada do plano da imagem normalizada
v_N	Segunda coordenada do plano da imagem normalizada
$r_e\vec{x}$	Vetor de direção do espelho para o painel de xadrez
$r_e\vec{x}N$	Vetor de direção do espelho para o painel de xadrez normalizado

Capítulo 1

Introdução

Nesta secção, será apresentado o que é um sistema catadióptrico bem como várias características do mesmo, como por exemplo, o tipo de espelho utilizado. Será também apresentada a revisão bibliográfica que serviu de base para esta dissertação tal como as motivações para a elaboração da mesma. Por fim, será explicado de que forma esta dissertação se irá estruturar.

1.1 Sistemas de visão catadióptrico

Um sistema catadióptrico é um sistema composto por um espelho côncavo e por uma câmara. Este tipo de sistema possui uma elevada área de visão, o que permite ter à disposição uma quantidade de informação do ambiente envolvente bastante superior, comparando-o com um sistema que não possua o espelho. Por outras palavras, é possível, com estes sistemas, obter-se informação visual a 360º, como se pode ver na figura 1.1.

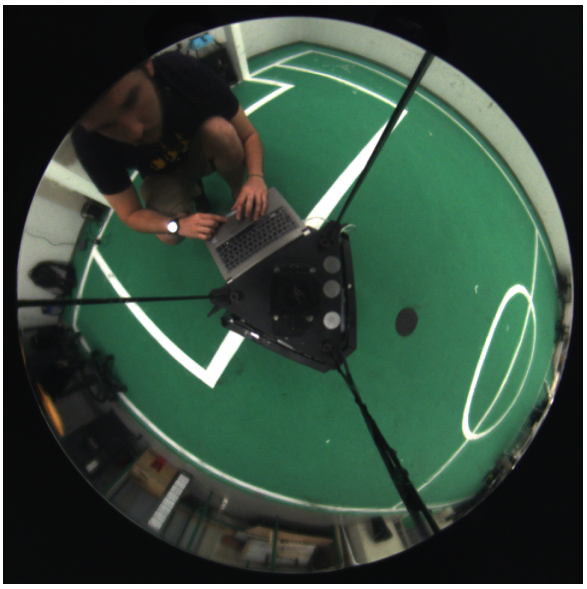
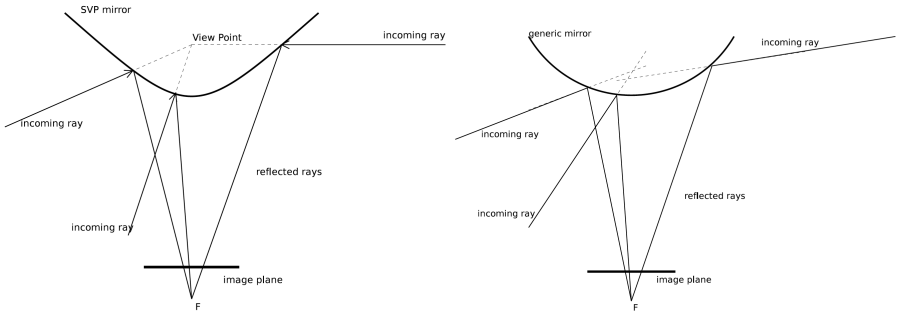


Figura 1.1: Imagem de visão a 360º adquirida pelo robot. Imagem adquirida no laboratório.

Como podemos comprovar em [1], nestes sistemas existem duas categorias: os SVP e os *non*-SVP.

Nos SVP, as retas dos raios de chegada ao espelho, que são refletidos para o plano da imagem, intersectam-se todas num único ponto chamado *Effective Viewpoint*. Os únicos espelhos em que se consegue ter esta características são os espelhos hiperbólicos (acompanhados de uma câmara em perspectiva) ou espelhos parabolóides (acompanhados por uma câmara ortográfica). O facto de se ter um único ponto de intersecção permite um mapeamento de todos os pontos do ambiente no plano da imagem. Neste projeto o sistema é de categoria SVP e cujo o espelho é hiperbólico. Isto deve-se ao facto de ser o conjunto presente nos robots de futebol robótico da equipa 5DPO da FEUP.



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

Figura 1.2: Comparação entre um sistema SVP (a) e um sistema *non*-SVP (b). Note-se que em (a) todos os raios se interceptam ao contrário de (b). Imagem retirada de [1].

No entanto, para além das vantagens já descritas, os sistemas SVP têm um grande problema. Devido à secção vertical curva dos espelhos e da resolução espacial mal distribuída, o chão e a câmara ocupam grande parte da imagem, o que faz com que haja pouco espaço na imagem para o ambiente envolvente.

Este tipo de sistemas é usado em diversas aplicações como por exemplo: controlo de trajetórias, como se vê em [2], ou no futebol robótico. O grande problema destes sistemas é, que são altamente não lineares, o que implica que, para cada aplicação, seja necessária uma calibração para se obter a melhor aproximação possível dos pontos do plano da imagem para os pontos reais no mundo.

1.2 Revisão Bibliográfica

1.2.1 Calibração de sistemas catadióptricos

Nesta secção serão apresentados alguns métodos utilizados por outros autores, que também serviram de base para esta dissertação. Podemos observar em [2] que foi utilizado um sistema, cujo espelho tem a forma esférica, para explicar de que forma foi feita a calibração.

Um sistema deste género converte o mapeamento de um ponto 3-D numa esfera unitária através de um centro C, seguindo-se uma projeção para o plano da imagem através de um centro de projeção O. Normalmente assume-se que este centro C assume o centro do sistema de coordenadas.

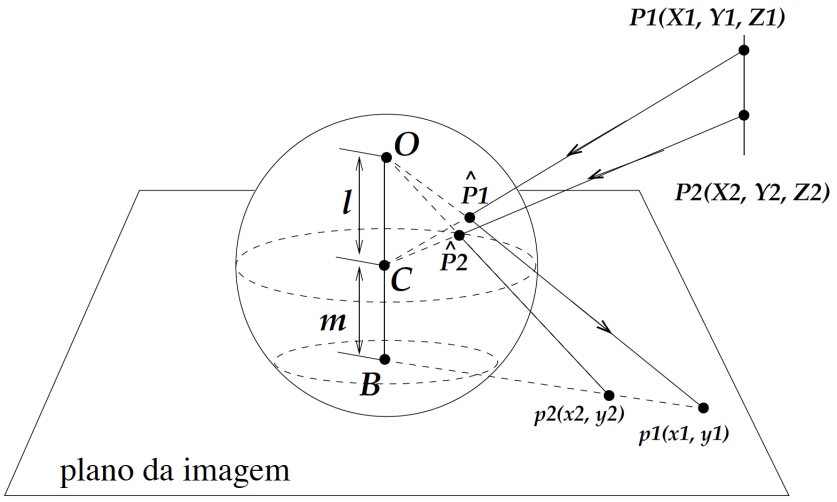


Figura 1.3: Modelo geral de projeção com espelho do tipo esférico

Tendo em conta o modelo apresentado, a projeção de um ponto $P = (X,Y,Z)$ num ponto $p = (x,y)$ é obtida por uma equação que relaciona ambas as coordenadas.

$$\begin{bmatrix} x \\ y \end{bmatrix} = \frac{l+m}{lR-Z} \begin{bmatrix} X \\ Y \end{bmatrix} \tag{1.1}$$

Na equação 1.2 estão presentes as variáveis l e m que representam, respetivamente, a distância do centro da esfera ao centro de projeção O e a distância do centro da esfera ao plano da imagem B e em que R é a distancia do ponto P ao centro da esfera. O parâmetro l varia entre os valores zero e um em que se $l = 1$, tem-se um sistema com espelho parabólico e se $l = 0$ o espelho é plano. Se o valor de l for um número entre zero e um, exclusivé, o espelho do sistema será hiperbólico ou elíptico. No entanto, os parâmetros intrínsecos da câmara e a distância focal não são considerados da equação.

No que diz respeito à calibração do sistema, esta foi feita utilizando uma estrutura 3-D com quadrados pretos e brancos, à semelhança de um tabuleiro de xadrez. Esta teve duas fases: definir o centro da imagem e estimar os valores dos parâmetros l e m do modelo apresentado. No entanto, para ser feita a calibração foi também preciso conhecer os parâmetros intrínsecos da câmara.

O centro da imagem foi encontrado usando-se o alinhamento do eixo de visão da câmara com o plano da imagem. De seguida, as linhas verticais 3-D foram projetadas em 2-D. Desta forma o centro omnidirecional foi encontrado pela interceção das linhas radiais. As equações das retas radiais foram definidas a partir de dois pontos pertencentes a cada uma dessas retas. As equações

foram depois agrupadas sobre a forma de matriz formando um sistema de equações. O centro (x_0, y_0) pode ser então estimado resolvendo o sistema através de mínimos quadrados.

$$\begin{bmatrix} x_1 \\ y_1 \\ \vdots \\ x_n \\ y_n \end{bmatrix} = \begin{bmatrix} 1 & 0 & \Delta_{x_1} & \dots & 0 \\ 0 & 1 & \Delta_{y_1} & \dots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & 0 & 0 & 0 & \Delta_{x_N} \\ 0 & 1 & 0 & 0 & \Delta_{y_N} \end{bmatrix} \begin{bmatrix} x_0 \\ y_0 \\ k_1 \\ \vdots \\ k_N \end{bmatrix} \tag{1.2}$$

A estimação dos parâmetros l e m foi feita usando-se um conjunto de pontos em que se conheciam as suas coordenadas no espaço 3-D e a sua projeção em 2-D. Para a estimação foi utilizada uma função de custo F em que foram comparadas as coordenadas dos pontos 3-D (X,Y,Z) e a sua projeção (u,v). PG é considerada a função que representa a equação que faz as projeções do modelo e que tem em conta os parâmetros intrínsecos da câmara. Desta forma obtemos

$$F = \sum_{i}^{N} \|(u_i, v_i) - PG(X_i, Y_i, Z_i)\|^2 \tag{1.3}$$

A partir do momento em que se tem a função de custo, os valores de l e m são calculados com o objetivo de minimizar a função custo. Desta forma temos

$$(l,m) = \min_{l,m} \sum_{i}^{N} \|(u_i, v_i) - PG(X_i, Y_i, Z_i)\|^2 \tag{1.4}$$

Estes parâmetros foram estimados através de um algoritmo não linear que se baseia em mínimos quadrados. Este método não implica conhecer a equação do espelho, visto que, é obtido através de um modelo geral de projeção.

Através de [3], pode ver-se que foi feita uma abordagem de calibração tendo em conta o modelo matemático de visão através de reflexão no espelho. Este modelo foi também desenvolvido e utilizado em [4]. Para esta abordagem era previamente conhecida a distância focal da câmara, assim como a equação do espelho. Tendo-se acesso a estas informações é possível calcular o raio que sai da câmara em direção ao espelho. Tendo este raio calculado é possível prever o ponto em que este incide no espelho. Conhecendo a equação do espelho, calcula-se o vetor normal a um plano tangente ao espelho no ponto exato em que o raio se encontra. Conhecendo este vetor, é calculado o ângulo entre este e o raio e é então projetado o ponto no mundo sabendo que essa projeção irá ser feita por um vetor que faz o mesmo ângulo com o vetor normal. Esta projeção é depois feita para o plano do chão. É possível comprovar observando a imagem 1.4.

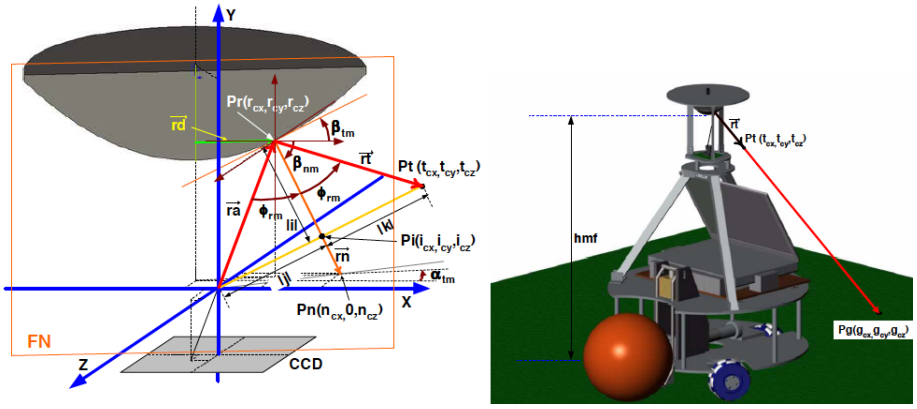


Figura 1.4: (a) Cálculo da normal do plano tangente ao espelho no ponto *Pr* e a equação do raio refletido. (b) Projeção do ponto refletido, no plano do chão. Imagem retirada de [3]

.

Foram também tidos em conta desalinhamentos entre a câmara e o espelho que podem causar erros de cálculo nas projeções calculadas. Para isso, foi construída uma matriz de transformação homogénea composta por três matrizes de rotação e um vetor de translação. Neste caso, d_y considera-se zero visto que o eixo y corresponde à altura e, neste modelo, esta é invariável.

$$R_x(\rho)\cdot R_y(\gamma)\cdot R_z(\theta)\cdot T = \begin{bmatrix} t1_{\rho\gamma\theta} & t2_{\rho\gamma\theta} & t3_{\rho\gamma} & d_x \\ t1_{\rho\theta} & t2_{\rho\theta} & t3_{\rho} & 0 \\ t1_{\rho\gamma\theta} & t2_{\rho\gamma\theta} & t3_{\rho\gamma} & d_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{1.5}$$

Em [5] a abordagem baseou-se no mesmo princípio. Foi proposto um método para estimar a forma do espelho e a sua posição, bem como os parâmetros extrínsecos de um sistema catadióptrico em que os referênciais não estão centrados. Este método necessita de imagens de um padrão planar com linhas e pontos cuja posição seja conhecida, neste caso em xadrez, e que a câmara esteja calibrada internamente.

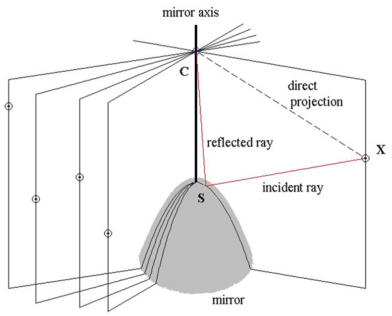


Figura 1.5: Projeção do ponto óptico *C* num ponto do mundo *X* em que *S* é o ponto de reflexão. Imagem retirada de [5]

.

Como se pode ver em [6], este descreve como aumentar a eficiência de um sistema catadióptrico e neste caso é descrito como foi obtida a forma do espelho. Foram feitas várias simulações 3-D até que se chegou a um modelo de espelho ideal. Este foi modelado em *SolidWorks* e a sua curvatura dependia da região do espelho. Desta forma, sabendo todos os parâmetros do espelho, dependendo da sua região, é possível calibrar o sistema sem ter de estimar parâmetros para além de possíveis desalinhamentos.

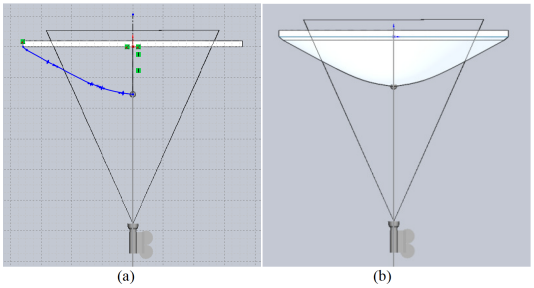


Figura 1.6: (a) Planeamento do espelho, antes de ser obtido por revolução, com as linhas correspondentes às diferentes curvaturas. (b) Espelho final obtido por revolução. Imagem retirada de [6]

Em [7] é utlizada uma rede neuronal para calibrar uma câmara. Esta rede tem em consideração uma relação não linear entre coordenadas em 2-D e em 3-D. Para isto, foi necessário conhecer pontos no mundo. Utilizando-se padrões de xadrez foram extraídos pontos da imagem obtida para se poder estabelecer uma correspondência. Para a deteção dos pontos foi utilizado um algoritmo de deteção de cantos de *Harris*. Depois de obtida uma correspondência entre 2-D e 3-D, que serviu como informação essencial para o treino, foi projetada uma rede neuronal.

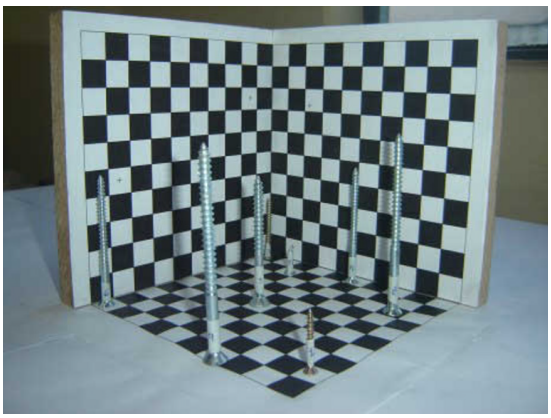


Figura 1.7: Estrutura 3-D utilizada para a calibração. Imagem retirada de [7]

1.2.2 Bibliotecas *OpenCV*

O *OpenCV* é um conjunto de bibliotecas *opensource* de processamento de imagem. Desta forma, estas possibilitam que qualquer pessoa possa utilizar e modificar o código de forma a obter

a solução desejada para um determinado problema. Estas bibliotecas, partilhadas ou estáticas, dividem-se em:

- *core* - Define estruturas de dados básicas, incluindo matrizes multidimensionais e funções básicas usadas noutras bibliotecas.
- *imgproc* - Tem como foco o processamento de imagem que inclui filtragem linear e não linear, transformações geométricas de imagem (redimensionamento, distorção, remapeamento genérico baseado em tabela), conversão de espaço de cores, histogramas.
- *video* - Diz respeito à análise de vídeo, estimação de movimento, remoção de *background*, entre outros.
- *calib3d* - Composta por algoritmos geométricos de *multiple-view*, calibração de câmara, estimação de posição de objectos, e algoritmos de reconstrução 3-D.
- *features2d* - Fundamentalmente usada para detetar características salientes.
- *objdetect* - Usada na deteção de objectos com classes pré-definidas (por exemplo, caras, olhos, carros).
- *highgui* - Uma interface de captura de imagem e de vídeo, com compatibilidade de *codecs*.
- *gpu* - Biblioteca aceleradora, por GPU, de algoritmos das restantes bibliotecas.

O *OpenCV* é composto por mais de 2500 algoritmos optimizados e tem interfaces *C/C++*, *Python*, *Java* e *Matlab*. É também possível desenvolver aplicações em *Windows*, *Android*, *Linux* e *Mac OS*. Informações retiradas de [8] e [9].

Figura 1.8: Logótipo das bibliotecas *OpenCV*. Imagem retirada de [8]

1.2.3 Redes Neuronais

O cérebro humano é composto por uma quantidade enorme de neurónios. Estes, ligados entre si em rede, tomam decisões do dia a dia baseando-se muitas vezes em fenómenos anteriores. Desta forma, à semelhança da função biológica, surgiram as redes neuronais artificiais. Estas redes são compostas por neurónios que se encontram ligados entre si por conexões. [10] Estas redes são organizadas em camadas sendo elas: *input*, *output* e *hidden layers*.

Na *input layer* encontram-se os neurónios que recebem a informação, no fundo, é a ligação que a rede tem com o exterior: depois de receber os dados, estes passam para a *hidden layer*. Nesta

camada, que pode ter várias *layers*, processa-se o tratamento dos dados, baseado num processo de aprendizagem, que por sua vez, envia a previsão de resultados para a *output layer*. Esta última camada é composta pelos neurónios que apresentam o resultado da previsão da rede. [11]

A fim de se conseguir ter uma previsão de resultados, é necessário treinar a rede neuronal através de um processo de aprendizagem. Este engloba fornecer à rede neuronal dados reais, em que são conhecidos os *inputs* e *outputs*, para que a rede se consiga ajustar a esses valores. No entanto, deve ter-se em atenção o número de neurónios da *hidden layer* para que não aconteça um fenómeno chamado *over-fitting*, que é, a rede ficar de tal forma ajustada aos valores de treino que a impossibilita de prever valores de uma forma eficiente.

Através deste treino, são atribuídos pesos às ligações entre neurónios e ajustados de forma a que os valores de *output* sejam próximos do pretendido. [12]

1.2.4 *Feedforward*

Uma rede *feedforward* é um tipo de rede neuronal. Em redes deste género a informação propaga-se dos *inputs* para os *outputs* sem a informação ter possibilidade de voltar para trás. É possível ver-se os neurónios de entrada e de saída enquanto que, os neurónios da *hidden layer* estão ocultos. Desta forma, os neurónios de *input* recebem a informação, os neurónios intermédios, tendo em conta o treino feito à priori, calculam as saídas correspondentes.

As redes *feedforward* em camadas são muito usadas. Um dos motivos para isto acontecer é que, a partir de um conjunto escasso de *inputs* e *outputs* utilizado no treino, conseguem prever de forma aproximadamente correta as saídas de valores desconhecidos. Por outro lado, existe um algoritmo de treino chamado *backpropagation* que consegue, normalmente, encontrar valores aceitáveis para os pesos das ligações entre neurónios provocando assim uma maior fiabilidade na previsão de valores. Informações retiradas de [13].

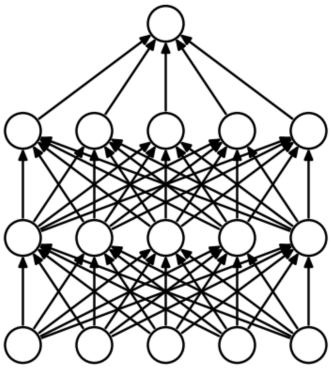


Figura 1.9: Exemplo de uma rede neuronal de *feedforward* em que os *inputs* encontram-se na base e os *outputs* encontram-se no topo. Imagem retirada de [14]

1.2.5 *Backpropagation*

Backpropagation é um método que pode ser usado para treinar uma rede neuronal. Este baseia-se em calcular o gradiente da função de perda em relação aos pesos das ligações entre neurónios.

Através de um algoritmo de descida de gradiente, o método é usado para optimizar o desempenho da rede. Este algoritmo é dividido em duas etapas, propagação e atualização de pesos.

Na primeira etapa, os dados de entrada vão-se propagando, camada a camada, até à saída da rede. De seguida, o resultado é comparado com as saídas desejadas e, através de uma função de perda, é calculado o erro em cada um dos neurónios de saída. Os valores do erro são então propagados para trás, começando nas saídas até cada neurónio ter um erro associado que representa a sua contribuição para o resultado final. Estes valores de erro são utilizados para calcular o gradiente da função de perda.

Na segunda etapa, o gradiente é sujeito a um método de otimização usado para atualizar os pesos de todas as ligações de modo a minimizar o erro. Informação retirada de [15].

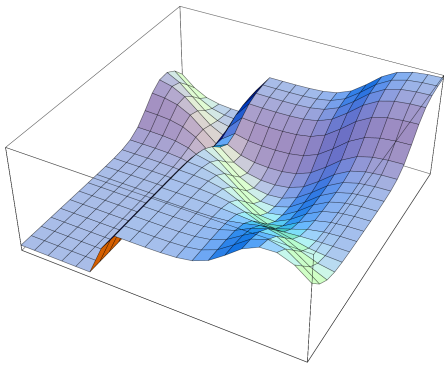


Figura 1.10: Exemplo de um mínimo local numa função de erro. Imagem retirada de [15]

1.2.6 Bibliotecas *FANN*

A *Fast Artificial Neural Network Library* é uma biblioteca *opensource* de redes neuronais. Implementada em C, é utilizada para obter redes totalmente conectadas e com poucas conexões. É relativamente fácil de utilizar, no que diz respeito ao tratamento de dados para o treino, é versátil, bem documentada e rápida de usar. Esta é compatível com várias interfaces gráficas, como por exemplo a *Agiel Neural Network*. Informações retiradas de [16]

Figura 1.11: Logótipo das bibliotecas *FANN*. Imagem retirada de [16]

1.3 Motivações e metodologia

Os sistemas catadióptricos são bastante influenciados por desalinhamentos que possam existir. Este problema verifica-se na equipa de futebol robótico 5DPO, da FEUP, e deve-se a erros de montagem ou a embates que possam existir em ambiente real de jogo. Tendo em conta que, na calibração de visão feita aos robots, atualmente, não são tidos em conta esses desalinhamentos,

surgiu a necessidade de encontrar uma forma de calibrar o sistema mesmo que estes estejam presentes. Para isso pretende-se fazer uma calibração que permita, utilizando pontos conhecidos no mundo e pontos conhecidos na imagem, estimar um mapeamento imagem-mundo mesmo que se verifiquem desalinhamentos nos seus elementos óticos.

Nesta dissertação, teve-se como base os modelos da câmara, espelho, transformação de coordenadas e visão descritos em [4]. Para a extração de características do mundo foram utilizadas as bibliotecas *OpenCV* compiladas em C++. As informações adquiridas foram todas tratadas e organizadas numa folha de cálculo. Por fim, a rede neuronal foi concebida utilizando as bibliotecas FANN através da interface gráfica *Agiel Neural Network*.

1.4 Estrutura da Dissertação

Para além da introdução, esta dissertação contém mais quatro capítulos. No capítulo 2, são descritos os modelos da câmara, espelho bem como o modelo de visão que será utilizado nesta dissertação. No capítulo 3 será apresentada e explicada a forma como foi feita a extração de características de uma estrutura de xadrez concebida para a calibração do sistema. No capítulo 4 abordar-se-á de que forma foi calibrado o sistema a partir de uma rede nauronal. Por último, no capítulo 5 serão apresentadas as conclusões do trabalho desenvolvido bem como os trabalhos futuros que se possam ainda levar a cabo.

Capítulo 2

Componentes e modelos do sistema de Visão

Como já referido anteriormente os sistemas catadióptricos de visão são constituídos por um espelho e por uma câmara. Cada um destes componentes tem características próprias que condicionam a eficiência do sistema.

Um grande problema associado a estes sistemas é o desvio que os seus componentes podem ter entre si, que provoca erros de medição se estas condicionantes não forem tidas em conta a quando da calibração do sistema.

Deste modo, neste capítulo, serão apresentados os modelos da câmara e do espelho utilizado, bem como as transformações de coordenadas necessárias para que se possa ter em conta as possíveis peculiaridades inerentes à montagem dos componentes do sistema no robot.

É também apresentado um modelo matemático que pode ser utilizado para calibrar o sistema. Este baseia-se na refleção dos raios incidentes no espelho, provenientes da câmara, e na estimação dos parâmetros do espelho. Esta estimação, através de mínimos quadráticos, utiliza o erro entre pontos do mundo conhecidos e os pontos correspondentes aos raios refletidos.

2.1 Modelo da Câmara

Para se obter o modelo da câmara, foi utilizado o modelo descrito em [4], em que este é obtido através de um modelo *pinhole* simples, e que será descrito de seguida. Este modelo tem como objetivo descrever uma relação matemática entre as coordenadas 3-D de um ponto e a sua projeção no plano da imagem.
Este modelo descreve uma projeção de um ponto 3-D, de coordenadas (x_c, y_c, z_c), num ponto 2-D no plano da imagem, de coordenadas (u, v).

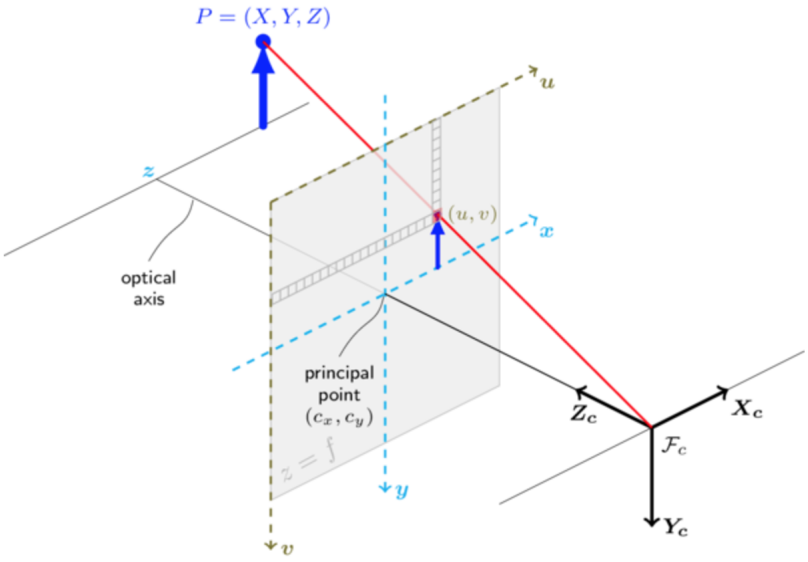


Figura 2.1: Representação do modelo *pinhole* da câmara. Imagem retirada de [17]

Definindo-se f (distância focal), como sendo a distância entre o centro ótico e o plano da imagem, obtemos

$$\lambda \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \begin{bmatrix} f & 0 & 0 \\ 0 & f & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_c \\ y_c \\ z_c \end{bmatrix} \tag{2.1}$$

O modelo inicial não inclui os efeitos provocados pela lente e utiliza a origem das coordenadas como ponto principal. No entanto, a maior parte dos sistemas consideram a origem das coordenadas no canto superior esquerdo. Desta forma é necessário dar-lhe um *offset*, obtendo-se

$$\lambda \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \begin{bmatrix} f_u & 0 & u_0 \\ 0 & f_v & v_0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_c \\ y_c \\ z_c \end{bmatrix} \tag{2.2}$$

É assumido também que os pixels são quadrados não distorcidos mas, como esta permissa nem sempre é válida, é possível colocar parâmetros na equação para que esta tenha estes fatores em conta.

$$\lambda \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \begin{bmatrix} f_u & \alpha & u_0 \\ 0 & f_v & v_0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_c \\ y_c \\ z_c \end{bmatrix} \tag{2.3}$$

Neste caso, assume-se que os pixels são quadrados ($f_u = f_v$) e que não têm distorção ($\alpha = 0$). No caso estudado, pretendeu-se fazer o inverso, ou seja, calcular a projeção de um ponto na imagem para 3-D. Invertendo o modelo da câmara, e definindo λ_x como x_c/z_c e λ_y como y_c/z_c, temos

$$\begin{bmatrix}\lambda_x\\ \lambda_y\end{bmatrix} = \begin{bmatrix}\frac{1}{f_u} & -\frac{\alpha}{f_u f_v} & \frac{-u_0}{f_u}+\frac{\alpha v_o}{f_u f_v}\\ 0 & \frac{1}{f_v} & -\frac{v_0}{f_v}\end{bmatrix}\begin{bmatrix}u\\ v\\ 1\end{bmatrix} \tag{2.4}$$

Para aumentar a validade deste modelo, e visto que este não tinha inicialmente em consideração a distorção provocada pela lente, foram adicionadas as equações de distorção de *Brown* que têm em conta distorções geométricas.

Começando por representar os raios sem distorção obteve-se

$$\lambda_u = \begin{bmatrix}\lambda_x\\ \lambda_y\end{bmatrix} \tag{2.5}$$

e representou-se também o centro de distorção como

$$\lambda_c = \begin{bmatrix}\lambda_{cx}\\ \lambda_{cy}\end{bmatrix} \tag{2.6}$$

Tendo considerado $r^2 = (\lambda_x - \lambda_{cx})^2 + (\lambda_y - \lambda_{cy})^2$, cada raio distorcido foi dado por

$$\begin{aligned}\lambda_d = \lambda_u + (\lambda_u - \lambda_c)\sum_{i=1}^{N_r}(R_i \cdot r^{2i}) + \\ + \begin{bmatrix}T_1(r^2+2(\lambda_x-\lambda_{cx})^2) & 2T_2(\lambda_x-\lambda_{cx})(\lambda_y-\lambda_{cy})\\ T_2(r^2+2(\lambda_y-\lambda_{cy})^2) & 2T_1(\lambda_x-\lambda_{cx})(\lambda_y-\lambda_{cy})\end{bmatrix}(1+\sum_{i=3}^{N_t}(T_i \cdot r^{2(i-2)}))\end{aligned} \tag{2.7}$$

onde T_i representa i-ésimo coeficiente tangencial de distorção e R_i o i-ésimo coeficiente radial de distorção. N_t e N_r representam, respectivamente, as ordens da distorção radial e tangencial. O efeito causado pela distorção é uma mudança de direção do raio incidente na lente. Desta forma é então possível calcular a correspondência entre o pixel e os raios distorcidos através de

$$\lambda_d = \begin{bmatrix}\frac{1}{f_u} & -\frac{\alpha}{f_u f_v} & \frac{-u_0}{f_u}+\frac{\alpha v_o}{f_u f_v}\\ 0 & \frac{1}{f_v} & -\frac{v_0}{f_v}\end{bmatrix}\begin{bmatrix}u\\ v\\ 1\end{bmatrix} \tag{2.8}$$

2.2 Modelo do Espelho

O modelo do espelho foi previamente obtido em [4], como será apresentado de seguida. Para calcular o modelo do espelho, foi utilizada a equação de uma hiperbolóide de duas folhas em que o espelho corresponde a uma delas. Embora o espelho seja limitado, a equação 2.9 descreve-o de forma infinita. O espelho tem um determinado raio e encontra-se num robot de futebol robótico. Este está fixo num suporte cujo fundo é preto.

O princípio de funcionamento de visão através do espelho baseia-se na reflexão de raios que serão projetados tendo em conta a curvatura do espelho. Desta forma, todos os casos em que os

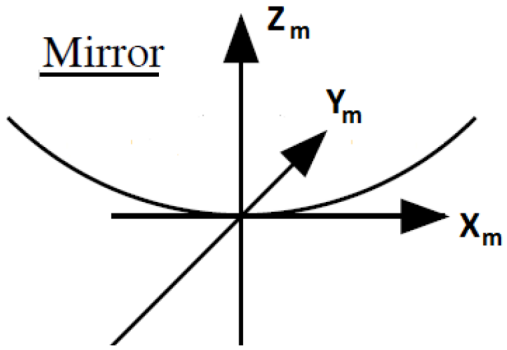


Figura 2.2: Representação do modelo do espelho. Imagem retirada de [4]

pixels correspondam a raios de visão que não interceptem o espelho, irão corresponder a um ponto de cor preta.

$$\frac{(z_m + a)^2}{a^2} - \frac{x_m^2 + y_m^2}{b^2} = 1 \tag{2.9}$$

O parâmetro a deve ser positivo para que a hiperboloide tenha, efetivamente, duas folhas.

2.3 Transformação de coordenadas

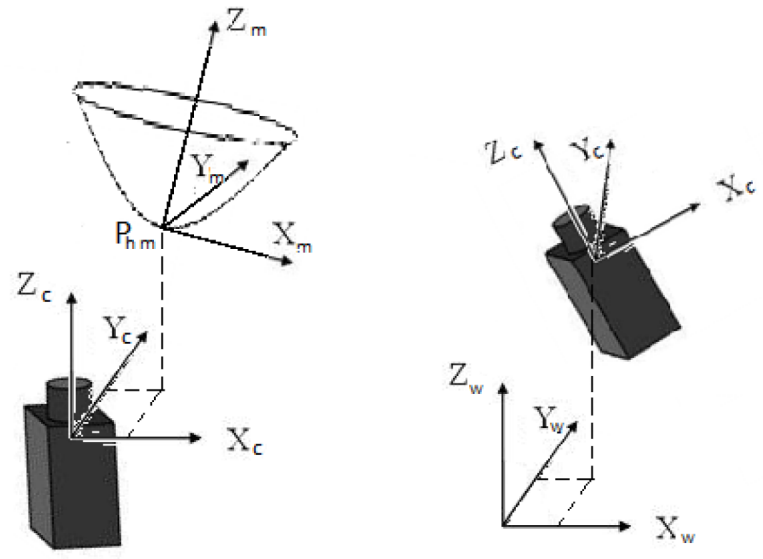
Nos sistemas catadióptricos ideais os seus componentes (câmara, espelho) estão perfeitamente alinhados, simplificando muito a calibração dos mesmos. No entanto, em ambiente real, estas características geralmente, não se verificam. Os possíveis desalinhamentos foram descritos em [4].

Para além do referencial inerente ao espelho e do referencial inerente à câmara, existe também o referencial do mundo. Estes três referencais podem estar desalinhados entre si como se pode ver na figura 2.3.

Se for tida em consideração a figura 2.3(a), observam-se vários casos possíveis de desalinhamento entre a câmara e o espelho. A nível de translações, o espelho pode estar deslocado, em relação ao eixo da câmara, segundo o eixo do x ou segundo o eixo do y. No que diz respeito à coordenada z, esta representa a distancia entre a câmara e o espelho. Em relação às rotações, o espelho pode estar rodado em torno dos eixos x e/ou y da câmara. Em z não foi necessário considerar pois, como o espelho é obtido através de revolução, esta rotação não é notada. Num caso ideal o ponto P_{hm} estaria perfeitamente alinhado com a origem do referencial da câmara e o eixo Z_m teria a mesma direção de Z_c.

Passando para o desalinhamento ilustrado na figura 2.3(b), a nível de translação, este pode ocorrer segundo os eixos x e y. No que diz respeito à coordenada z, esta representa a distância entre o plano do chão e a câmara. Como desalinhamentos por rotação, para além dos referidos anteriormente, referentes ao conjunto câmara-espelho, acrescenta-se a rotação segundo o eixo de

z. Visto que a câmara se encontra fixa num robot, e que este sofre rotações, haverá situações em que a câmara estará rodada em relação a Z_w.



(a) Misalignment between camera and mirror

(b) Misalignment of camera in relation to world reference

Figura 2.3: Possíveis desalinhamentos dos referenciais. (a) Desalinhamento entre câmara e espelho. (b) Desalinhamento entre a câmara e o mundo. Imagem retirada de [4]

.

Para modelar a transformação de coordenadas, da câmara para o espelho, foram usadas transformações homogéneas compostas pela matriz de rotação e pelo vetor de translação. Para estas transformações consideraram-se apenas os ângulos de *yaw* e de *pitch*, visto que o *roll* não tem um efeito grave.

$$R_m^c = R_{y,\theta_m} R_{x,\psi_m} = \begin{bmatrix} \cos\theta_m & \sin\theta_m \cdot \sin\psi_m & \sin\theta_m \cdot \cos\psi_m \\ 0 & \cos\psi_m & -\sin\psi_m \\ -\sin\theta_m & \cos\theta_m \cdot \sin\psi_m & \cos\theta_m \cdot \cos\psi_m \end{bmatrix} \tag{2.10}$$

em que θ_m é o ângulo de *pitch* e ψ_m o ângulo de *yaw*.

O vetor de translação, em coordenadas da câmara, definiu-se como

$$T_m = \begin{bmatrix} t_{mx} \\ t_{my} \\ t_{mz} \end{bmatrix} \tag{2.11}$$

Através das deduções anteriores obteve-se a matriz homogénea, de transformação de coordenadas do espelho para a câmara,

$$H_m^c = \begin{bmatrix} R_m^c & T_m \\ 0_{1,3} & 1 \end{bmatrix} = \begin{bmatrix} \cos\theta_m & \sin\theta_m \cdot \sin\psi_m & \sin\theta_m \cdot \cos\psi_m & t_{mx} \\ 0 & \cos\psi_m & -\sin\psi_m & t_{my} \\ -\sin\theta_m & \cos\theta_m \cdot \sin\psi_m & \cos\theta_m \cdot \cos\psi_m & t_{mz} \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{2.12}$$

A transformação homogénea inversa é

$$H_c^m = \begin{bmatrix} R_c^m & -R_c^m T_m \\ 0_{1,3} & 1 \end{bmatrix} = \begin{bmatrix} R_m^{c\,T} & -R_m^{c\,T} T_m \\ 0_{1,3} & 1 \end{bmatrix} \tag{2.13}$$

Para obter a transformação R_c^m apenas é necessário transpor a matriz R_m^c, visto que esta é uma matriz ortogonal.

Para modelizar o modelo de transformação de coordenadas da câmara para o mundo, foram também utilizadas transformações homogéneas. Neste caso foi necessário considerar também o ângulo de *roll*, devido ao facto dessa rotação já interferir diretamente com a transformação.

$$R_c^w = R_{z,\phi_c} R_{y,\theta_c} R_{x,\psi_c} =$$

$$= \begin{bmatrix} \cos\phi_c \cdot \cos\theta_c & -\sin\phi_c \cdot \cos\psi_c + \cos\phi_c \cdot \sin\theta_c \cdot \sin\psi_c & \sin\phi_c \cdot \sin\psi_c + \cos\phi_c \cdot \sin\theta_c \cdot \cos\psi_c \\ \sin\phi_c \cdot \cos\theta_c & \cos\phi_c \cdot \cos\psi_m + \sin\phi_c \cdot \sin\theta_c \cdot \sin\psi_c & -\cos\phi_c \cdot \sin\psi_c + \sin\phi_c \cdot \sin\theta_c \cdot \cos\psi_c \\ -\sin\theta_m & \cos\theta_m \cdot \sin\psi_m & \cos\theta_m \cdot \cos\psi_m \end{bmatrix} \tag{2.14}$$

Para se obter a orientação do vetor em coordenadas do mundo, multiplica-se esta matriz pelo vetor na referência da câmara.
O vetor de translação é dado por

$$T_c = \begin{bmatrix} t_{cx} \\ t_{cy} \\ t_{cz} \end{bmatrix} \tag{2.15}$$

Desta forma, foi obtida a matriz homogénea de transformação de coordenadas da câmara para o mundo como sendo

$$H_c^w = \begin{bmatrix} R_c^w & -R_c^w T_c \\ 0_{1,3} & 1 \end{bmatrix} \tag{2.16}$$

2.4 Modelo Matemático de Visão por Reflexão

Uma possível abordagem a este problema, que foi descrita em [4] e será apresentadas de seguida, é utilizar um modelo de *Back-Propagation Ray Tracing* que tem em conta o raio incidente no espelho e calcula o raio refletido para o mundo.

Considerando $\lambda_{cm} = s(\lambda_x, \lambda_y, 1)$ como o raio incidente em cada pixel, em coordenadas da câmara, obteve-se a equação do raio em coordenadas do espelho, através de uma transformação homogénea.

$$\lambda_m = H_c^m x(s\lambda_x, s\lambda_y, s, 1) = R_c^m xs(\lambda_x, \lambda_y, 1) - R_c^m xT_m = s\vec{\lambda}'_m - \lambda''_m \tag{2.17}$$

em que

$$\vec{\lambda}'_m = \begin{bmatrix} \lambda'_{mx} \\ \lambda'_{my} \\ \lambda'_{mz} \end{bmatrix} \tag{2.18}$$

$$\lambda''_m = \begin{bmatrix} \lambda''_{mx} \\ \lambda''_{my} \\ \lambda''_{mz} \end{bmatrix} \tag{2.19}$$

Depois de obtida a relação 2.17, procedeu-se à substituição da equação do raio na hiperboloide

$$\frac{(s\lambda'_{mz} - \lambda_{mz} + a)^2}{a^2} - \frac{(s\lambda'_{mx} - \lambda_{mx})^2 + (s\lambda'_{my} - \lambda_{my})^2}{b^2} = 1 \tag{2.20}$$

$$\begin{aligned} \left(\frac{{\lambda'_{mz}}^2}{a^2} - \frac{{\lambda'_{my}}^2 + {\lambda'_{mx}}^2}{b^2} \right) s^2 + \left(\frac{2\lambda'_{mz}(a - \lambda''_{mz})}{a^2} - \frac{2\lambda'_{my}\lambda''_{my} + 2\lambda'_{mx}\lambda''_{mx}}{b^2} \right) s \\ + \left(\frac{(a - \lambda''_{mz})^2}{a^2} - \frac{{\lambda''_{mx}}^2 + {\lambda''_{my}}^2}{b^2} - 1 \right) = 0 \end{aligned} \tag{2.21}$$

Para simplificar, a equação foi divida da seguinte forma

$$s_a = \frac{{\lambda'_{mz}}^2}{a^2} - \frac{{\lambda'_{my}}^2 + {\lambda'_{mx}}^2}{b^2} \tag{2.22}$$

$$s_b = \frac{2\lambda'_{mz}(a - \lambda''_{mz})}{a^2} - \frac{2\lambda'_{my}\lambda''_{my} + 2\lambda'_{mx}\lambda''_{mx}}{b^2} \tag{2.23}$$

$$s_c = \frac{(a - \lambda''_{mz})^2}{a^2} - \frac{{\lambda''_{mx}}^2 + {\lambda''_{my}}^2}{b^2} - 1 \tag{2.24}$$

$$s_m = \frac{-s_b \pm \sqrt{s_b^2 - 4s_a s_c}}{2s_a} \tag{2.25}$$

Em todos os casos em que $s_a = 0$ ou que $(s_b^2 - 4s_a s_c) < 0$, os raios não intercetam o espelho. Tendo-se a expressão de s_m calcula-se então a coordenada z do espelho. Obtendo-se os resultados é necessário verificar quais deles estão corretos. Visto que se considerou o espelho como a folha positiva da hiperbolóide, apenas os resultados positivos, para z, são válidos. Caso haja mais do que um valor positivo terá de se optar pelo menor entre eles. Isto deve-se ao facto de o valor menor corresponder à primeira intersecção do raio com o espelho que representa o ponto em que o raio é refletido.

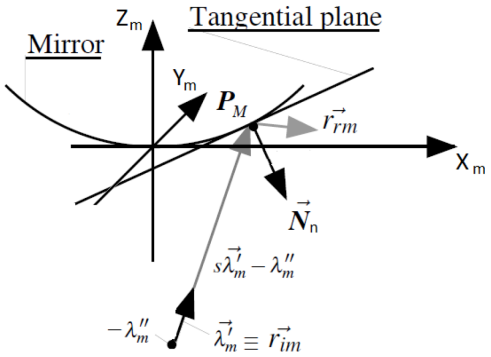


Figura 2.4: Representação da reflexão do raio associada ao espelho. Imagem obtida de [4].

Por observação da imagem 2.4, é possível perceber como se processa a reflexão de um raio proveniente da câmara. Este, com uma orientado segundo o vetor $\vec{\lambda}'_m$, incide no espelho no ponto P_M (ponto de interceção com hiperbolóide, em coordenadas do espelho) e é refletido segundo um vetor de orientação $\vec{r_{rm}}$.

$$P_M = s_m \vec{\lambda}'_m - \lambda''_m \tag{2.26}$$

A reflexão é calculada através de $\vec{N}_n$ que é o vetor normal a um plano tangente que passa na hiperbolóide no ponto em que o raio incidiu. Com $\vec{N}_n$, e sabendo $\vec{\lambda}'_m$, é possível calcular o ângulo entre eles que será exatamente igual ao ângulo que $\vec{N}_n$ fará com $\vec{r_{rm}}$. A fim de calcular $\vec{N}_n$, que corresponde ao gradiente $\nabla F(x_m, y_m, z_m)$ no ponto (x_m, y_m, z_m) da hiperbolóide, foi necessário, previamente definir

$$F(x_m, y_m, z_m) = -\frac{(z_m + a)^2}{a^2} + \frac{x_m^2 + y_{m^2}}{b^2} + 1 \tag{2.27}$$

Em que $F(x_m, y_m, z_m)$ representa um plano tangente à hiperboloide no ponto (x_m, y_m, z_m). De seguida calculou-se o gradiente desse plano, que tem direção para fora do espelho,

$$\vec{N}(x_m, y_m, z_m) = \nabla F(x_m, y_m, z_m) = \begin{bmatrix} \frac{2x_m}{b^2} \\ \frac{2y_m}{b^2} \\ -\frac{2(z_m+a)}{a^2} \end{bmatrix} \tag{2.28}$$

O ângulo entre o gradiente e o raio incidente no espelho, pode ser calculado através de uma equação genérica que relaciona as coordenadas de dois vetores com o seu angulo

$$\cos\gamma = \frac{x_1x_2 + y_1y_2 + z_1z_2}{\sqrt{x_1^2 + y_1^2 + z_1^2}\sqrt{x_2^2 + y_2^2 + z_2^2}} \tag{2.29}$$

Conhecendo coordenadas reais de pontos, seria possível escrever uma função $PC(u,v)$, que consideraria como entradas as coordenadas dos pixels da imagem, e cujo resultado seria a projeção desses mesmos pontos no mundo (em 3-D). Comparando as saídas de $PC(u,v)$ com os valores reais, seria possível escrever a função custo G como

$$G = \sum_i^N \|(x_{xi}, y_{xi}, z_{xi}) - PC(u_i, v_i)\|^2 \tag{2.30}$$

Por fim, tendo a função custo, seria possível estimar os valores de a e b, presentes no vetor $\vec{N}_n$, que minimizam o erro, através de mínimos quadráticos.

$$(a,b) = \min_{a,b} \sum_i^N \|(x_{xi}, y_{xi}, z_{xi}) - PC(u_i, v_i)\|^2 \tag{2.31}$$

Capítulo 3

Extracção de Características do Sistema

Neste capítulo irá ser apresentada uma estrutura que foi construída com placas de xadrez assim como as características da mesma. Esta foi utilizada para retirar pontos do espaço e é apresentada a forma de como isso foi feito.

De seguida foi necessário pré-processar a imagem para que se conseguissem retirar as informações pretendidas.

Por fim, neste capítulo serão apresentadas as conclusões relativamente à abordagem a ter para a calibração do sistema.

3.1 Estrutura de Painéis de Xadrez

Figura 3.1: Robot dentro da estrutura de calibração. Imagem adquirida no laboratório.

Para se ter acesso a pontos conhecidos no mundo, foi construída uma estrutura com quatro paredes em que cada uma delas era composta por um painel de xadrez. Cada painel era composto por sessenta e quatro quadrados, em que trinta e dois eram pretos e os restantes eram brancos. Cada quadrado tinha oito centímetros de lado. Esta estrutura foi colocada em volta do robot.

Foi também utilizada uma estrutura deste género em [2]. A imagem adquirida pode ser vista na imagem 3.2.

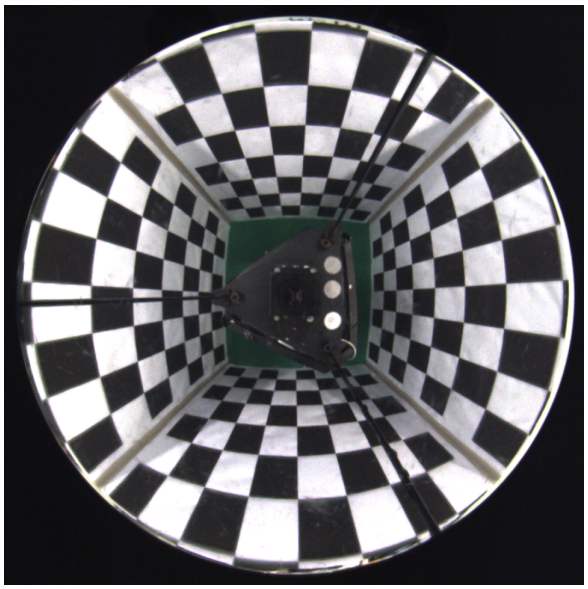


Figura 3.2: Imagem obtida pela câmara com a estrutura de xadrez.

Através da geometria do painel de xadrez é possível saber, à priori, uma quantidade elevada de pontos do mundo, que correspondem aos cantos dos quadrados. Desta forma, sabendo as coordenadas desses pontos no mundo e sabendo as coordenadas em pixels na imagem, seria possível estimar os parâmetros da equação do espelho de modo a que, o erro inerente à equação que fará a conversão das coordenadas da imagem para o mundo, fosse o mínimo possível.

Esta estrutura foi escolhida por diversos fatores e o facto de ser facilmente transportada, visto ser desmontável, foi um fator preponderante. O facto de ser um xadrez branco e preto, faz com que este seja mais facilmente detetável na imagem. Sendo um quadrado um quadrilátero regular de quatro lados com mesmo comprimento e quatro ângulos retos, torna-se uma figura geométrica bastante fiável para a calibração do sistema. Estas características do quadrado fazem com que se saiba, à priori, que todos os pontos adquiridos na estrutura formam linhas retas e não linhas curvas, como se observa na imagem 3.2. O aspeto curvo é causado pelo espelho e pela distorção da lente da câmara.

3.2 Tratamento de imagem

A imagem 3.2 foi obtida por um robot de futebol robótico da equipa 5DPO da FEUP. Neste robot, a estrutura que suporta a câmara e o espelho é composta por três hastes de cor preta que provocam alguma perda do campo de visão nas regiões da imagem onde estes se encontram. Desta forma foi necessário pré processar a imagem e para isso desenvolveu-se um algoritmo em C++ utilizando as bibliotecas OpenCV.

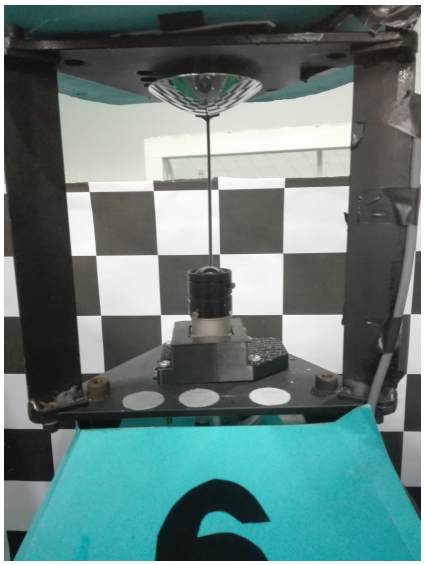


Figura 3.3: Suporte da câmara e do espelho no robot de futebol robótico. Imagem adquirida no laboratório.

Inicialmente a imagem RGB foi convertida para padrões de cinza. Neste caso, como o objetivo foi detetar o padrão de xadrez da imagem, esta não precisou de estar a cores. A conversão, tornou também o processamento mais leve.

Como referido anteriormente, o suporte do espelho é composto por três hastes pretas. Estas hastes impedem que, em determinadas zonas da imagem, seja possível haver uma continuidade nos quadrados do padrão utilizado para a calibração.

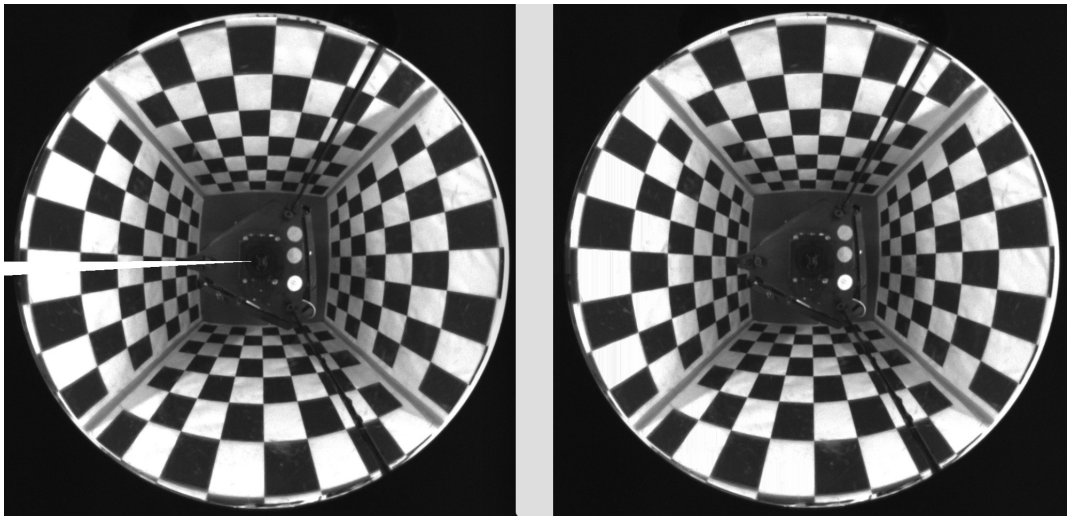


Figura 3.4: Do lado esquerdo é possível ver a imagem com uma fatia branca, que se refere à área a corrigir. Do lado direito é possível ver a imagem corrigida sem a haste.

A fim de se obter planos de xadrez completos, ou pelo menos o mais aproximado disso possível, foi feita uma remoção das hastes da imagem. Esta não afetou o resultado de forma negativa, pelo facto destas estarem localizadas em regiões do xadrez com quadrículas e não com cantos. Para fazer a remoção foram definidas regiões, através da interseção de retas, que taparam a região a tratar. O resultado pode ser visto na imagem 3.4.

De seguida foi implementada uma função que, correndo coluna a coluna a matriz da imagem, dá um novo valor aos pixels dessa região, tendo em conta o valor que estes efetivamente deveriam ter. O resultado pode também ser visto na imagem 3.4.

Depois de corrigidas as regiões da imagem foram definidas quatro regiões de interesse na imagem, cada uma delas para uma matriz de xadrez diferente. Esta divisão foi feita para que a deteção dos painéis fosse feita apenas no painel pretendido.

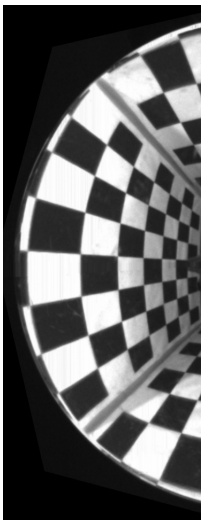


Figura 3.5: Imagem representativa de uma das regiões definidas para se proceder à detecção do painel de xadrez.

A deteção do xadrez foi feita com a função *findChessboardCorners()* de OpenCV. Esta função necessita que lhe seja indicada a imagem, da qual se pretende extrair os pontos, o tamanho do padrão de xadrez a detetar, a variável de destino do resultado e a variável que irá receber as informações que a função retornará. A função *findChessboardCorners()* tem como base a função *cornerSubPix()*, que observa os vetores de uma determinada vizinhança e compara as direções como se pode ver em [18].

A deteção teve de ser feita para cada um dos planos de xadrez. De seguida os resultados foram desenhados na imagem, através da função *drawChessboardCorners()*.

```
1  //Tamanho dos pontos interiores do xadrez
2  Size patternsize1(7,6);
3  Size patternsize2(7,6);
4  Size patternsize3(7,5);
```

```
5   Size patternsize4(7,6);
6
7   //Cantos do xadrez
8   vector<Point2f> centers1;
9   vector<Point2f> centers2;
10  vector<Point2f> centers3;
11  vector<Point2f> centers4;
12
13  //Flags
14  bool patternfound1;
15  bool patternfound2;
16  bool patternfound3;
17  bool patternfound4;
18
19  void deteta_xadrez()
20  {
21          patternfound1 = findChessboardCorners(gray1, patternsize1, centers1);
22          patternfound2 = findChessboardCorners(gray2, patternsize2, centers2);
23          patternfound3 = findChessboardCorners(gray3, patternsize3, centers3);
24          patternfound4 = findChessboardCorners(gray4, patternsize4, centers4);
25    for(int i=0; i<centers1.size(); i++)
26          {
27                  centers1[i].x += 550;
28          }
29          for(int i=0; i<centers4.size(); i++)
30          {
31                  centers4[i].y += 594;
32          }
33    cout<< "Resultado detacao xadrez 1 -> " << patternfound1 << endl;
34          cout<< "Resultado detacao xadrez 2 -> " << patternfound2 << endl;
35          cout<< "Resultado detacao xadrez 3 -> " << patternfound3 << endl;
36          cout<< "Resultado detacao xadrez 4 -> " << patternfound4 << endl;
37  }
38
39  void desenha_xadrez()
40  {
41          drawChessboardCorners(image, patternsize1, Mat(centers1), patternfound1);
42          drawChessboardCorners(image, patternsize2, Mat(centers2), patternfound2);
43          drawChessboardCorners(image, patternsize3, Mat(centers3), patternfound3);
44          drawChessboardCorners(image, patternsize4, Mat(centers4), patternfound4);
45  }
```

Obtidos os pontos pretendidos nas regiões de interesse definidas, foi necessário reposicioná-los na imagem original, para se obter as verdadeiras coordenadas de cada um deles. O resultado pode ser visto na figura 3.6

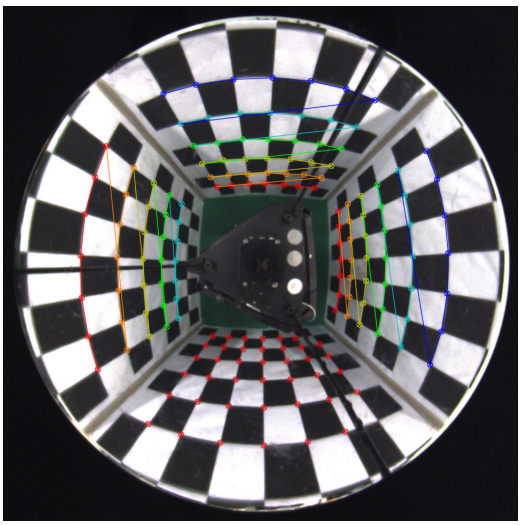


Figura 3.6: Imagem com a representação dos pontos de xadrez detectados.

Obtidas todas as informações necessárias dos pontos, estas foram então escritas num ficheiro e foram introduzidas numa folha de cálculo, onde se escontravam as coordenadas no mundo dos cantos dos quatros paineis de xadrez. Estas coordenadas foram obtidas com a origem no centro do robot.

Tabela 3.1: Exemplo de pontos no mundo e a sua correspondência nos pixels da imagem.

Mundo			**Imagem**	
x_x	y_x	z_x	u_r	v_r
-0,30	-0,45	0,128	-90,0	135,5
-0,20	-0,45	0,128	-60,0	137,5
-0,10	-0,45	0,128	-30,0	140,5
0,00	-0,45	0,128	4,5	140,5
0,10	-0,45	0,128	36,5	139
0,20	-0,45	0,128	65,0	139

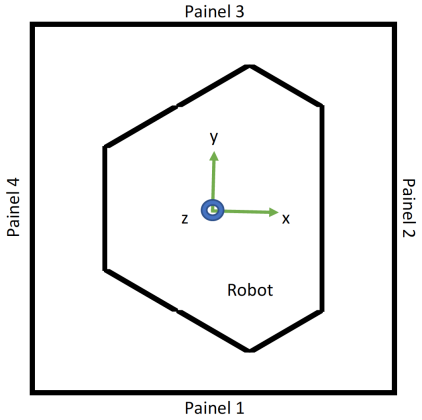


Figura 3.7: Imagem com a representação da estrutura de xadrez à volta do robot com o referêncial no mundo.

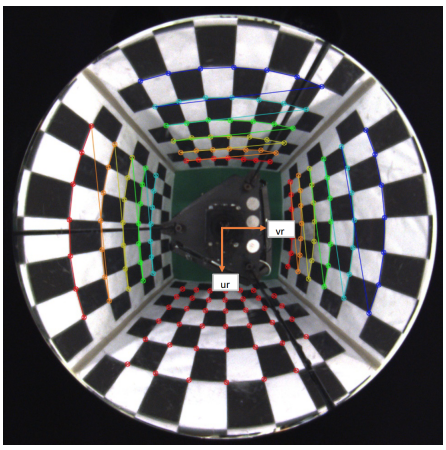


Figura 3.8: Representação do referêncial na imagem adquirida.

3.3 Conclusões

Tendo em conta o modelo apresentado em 2.4 o objetivo seria calibrar o sistema, estimando não só os parâmetros a e b da equação do espelho, mas também as translações e rotações dos referênciais da câmara e do espelho. Ao estimar esses valores, seria possível estimar o raio refletido, proveniente da câmara para o mundo. Essa estimação seria feita através do método de mínimos quadráticos. Para calcular o erro, usar-se-ia as coordenadas reais, conhecidas, dos pontos do xadrez e as coordenadas estimadas, desses mesmos pontos, tendo em conta o modelo de reflexão do raio proveniente da câmara.

O raio refletido corresponde, não apenas a um ponto, mas sim a uma recta de direção igual ao raio refletido. Ou seja, cada pixel da imagem corresponde, no mundo, a uma infinidade de pontos sendo assim complicado ter noção de profundidade.

Desta forma, e pelo facto deste modelo ter associado um conjunto elevado de parâmetros a estimar, devido principalmente a fatores extrínsecos, foram utilizadas redes neuronais, para calibrar o sistema, como será descrito no capítulo 4. Ao serem utilizadas redes neuronais, é possível estimar o mapeamento imagem-mundo mesmo que existam outros fatores que possam interferir no sistema catadióptrico, para além dos desvios referidos no capítulo 2.

Capítulo 4

Calibração com Rede Neuronal

Neste capítulo será apresentada a rede neuronal utilizada para calibrar o sistema catadióptrico anteriormente descrito. Serão descritos os dados utilizados para a rede, bem como o tratamento necessário para que possam ser utilizados. Irá falar-se das características da rede neuronal utilizada, bem como o treino da rede. Para obter uma melhor eficiência na previsão dos valores, decidiu-se fazer quatro redes neuronais. Visto ter-se utilizado as bibliotecas *FANN* de redes neuronais, irá ser mostrada a interface utilizada. Por último serão apresentados os resultados obtidos.

4.1 Dados para a Rede Neuronal

Como se viu no capítulo 3, foram extraídas da imagem as coordenadas, em pixels, de pontos da estrutura de xadrez construída. Esta informação fez com que fosse possível estabelecer uma correspondência direta entre as coordenadas da imagem e as coordenadas no mundo.

$$(u, v) \rightarrow (x_x, y_x, z_x)$$

Esta correspondência foi organizada numa folha de cálculo e, utilizando essa informação, decidiu-se usar uma rede neuronal para calibrar o sistema.

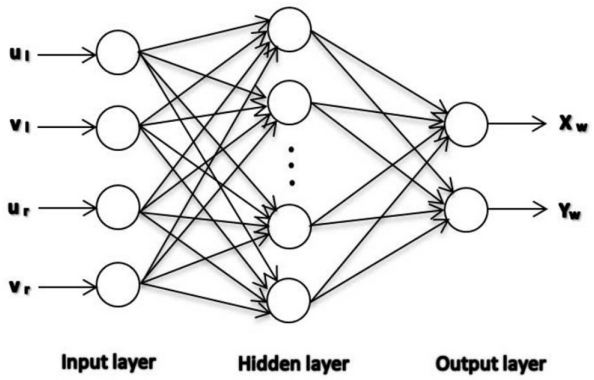


Figura 4.1: Exemplo de uma rede neuronal. Imagem obtida de [7].

Desta forma, para ser possível usar os valores, estes tiveram de sofrer uma normalização. Este processo faz com que todos os dados estejam limitados, neste caso no intervalo $[-1,1]$, fazendo com que todos os valores que sejam utilizados na rede estejam na mesma ordem de grandeza. Assim é evitado que os dados influenciem, de forma negativa, os neurónios da rede neuronal.

No que diz respeito às coordenadas (u,v) da imagem, foi necessário dividir ambas pelo valor máximo que a coordenada do pixel pode tomar, tendo em conta que a origem das coordenadas se encontra no centro da imagem.

$$(u_N, v_N) = \left(\frac{u-u_c}{\frac{u_{mx}}{2}}, \frac{v-v_c}{\frac{v_{mx}}{2}}\right) = \left(\frac{u_r}{482}, \frac{v_r}{482}\right) \tag{4.1}$$

Tendo-se considerado a origem das coordenadas no centro do robot, possuindo as coordenadas reais dos pontos do xadrez, e sabendo a altura aproximada a que o espelho se encontra do chão, foi possível definir um vetor $\vec{r}_{ex}$ cuja direção seria do espelho para o painel de xadrez. De seguida foi também feita uma normalização dos vectores calculados dividindo-os pela sua norma.

$$\vec{r_{ex}} = (x_x, y_x, z_x) - (0,0,h_e) \tag{4.2}$$

$$\vec{r_{exN}} = \frac{\vec{r_{ex}}}{\|\vec{r_{ex}}\|} \tag{4.3}$$

Como será explicado mais à frente em 4.3, tanto para treinar uma rede neuronal, como para a testar, é necessário ter os *inputs* e os *outputs* correspondentes. Assim, depois de normalizados, os dados foram divididos em dois ficheiros, em que um deles seria para treinar a rede e o outro para testar o resultado desse treino.

Tabela 4.1: Exemplo de alguns valores utilizados do ficheiro de treino.

u_N	v_N	$\hat{i}$	$\hat{j}$	$\hat{k}$
-0,19	0,28	-0,36	-0,55	-0,75
-0,06	0,29	-0,13	-0,58	-0,80
0,08	0,29	0,13	-0,58	-0,80
-0,21	0,32	-0,40	-0,60	-0,69
-0,07	0,33	-0,14	-0,65	-0,75
0,09	0,33	0,14	-0,65	-0,75
0,23	0,31	0,40	-0,60	-0,69
-0,17	0,37	-0,31	-0,69	-0,65
0,01	0,38	0,00	-0,73	-0,68
0,18	0,37	0,31	-0,69	-0,65

Os ficheiros devem conter valores diferentes, para que efetivamente seja possível verificar se a rede neuronal consegue calcular valores corretos, tendo em conta o treino feito. No entanto, ambos os conjunto de dados devem cobrir todas as zonas de interesse.

Tabela 4.2: Exemplo de alguns valores utilizados do ficheiro de teste.

u_N	v_N	$\hat{i}$	$\hat{j}$	$\hat{k}$
-0,12	0,29	-0,25	-0,57	-0,78
0,01	0,29	0,00	-0,59	-0,81
0,13	0,29	0,25	-0,57	-0,78
-0,14	0,32	-0,28	-0,63	-0,73
0,01	0,33	0,00	-0,65	-0,76
0,16	0,32	0,28	-0,63	-0,73
-0,25	0,36	-0,44	-0,66	-0,62
-0,08	0,38	-0,16	-0,72	-0,68
0,10	0,38	0,16	-0,72	-0,68
0,26	0,36	0,44	-0,66	-0,62

4.2 Estrutura da Rede

Existem muitos tipos de redes neuronais, sendo utilizada uma de topologia *feedforward*. Estas têm como objetivo produzir um *output*, cujo o erro seja o menor possível, em relação aos verdadeiros valores a prever, e são compostas por *layers*. Estas dividem-se em *input layers*, *hidden layers* e *output layers* e cada uma delas é constituída por neurónios. [19]

Neste caso foi utilizada uma rede neuronal com dois neurónios na *input layers*, três neurónios na *output layers* e dezasseis neurónios na *hidden layers*, como se poderá ver mais à frente em 4.2.1 e 4.2.2. A rede foi treinada em *backpropagation*.

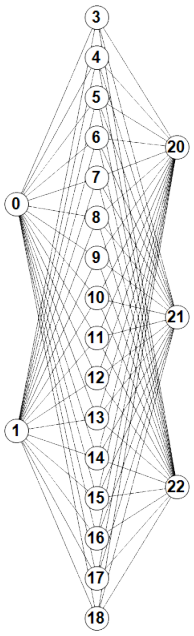


Figura 4.2: Diagrama da rede neuronal utilizada no problema.

A fim de obter uma melhor eficiência na previsão dos valores da saída da rede, foi decidido fazer-se quatro redes neuronais, cada uma delas correspondente a um dos painéis de xadrez descritos no capítulo. 3

4.2.1 *Input* e *Output Layer*

Numa rede neuronal, na *input layer*, encontram-se os neurónios que têm contacto direto com o exterior da rede. No fundo são os neurónios que recebem as informações do exterior. Neste caso, tendo em conta a aplicação da rede, tem-se como entradas os valores de u_N e v_N o que faz com que seja necessária a existência de dois neurónios de entrada.

Os *outputs* da rede são as coordenadas do vetor unitário $\vec{r_{ex}}$. Assim é necessário que a *output layer* seja composta por três neurónios, em que cada um deles corresponde a cada uma das coordenadas.

4.2.2 *Hidden Layer*

A *Hidden Layer* é a camada que faz a ligação entre os neurónios da *input layer* e da *output layer*, ou seja, recebe os valores de *input* e prevê os *outputs*. Esta camada tem um número de neurónios variável, que interfere diretamente no treino feito à rede. Esse número está diretamente ligado ao facto da rede conseguir aproximar corretamente os valores de saída ou obter essa aproximação com um grande erro.

Esta camada é a camada que prevê os valores de saída. Deste modo, quanto maior for o número de neurónios, melhor será essa aproximação. Ainda assim existem algumas condicionantes no que diz respeito à função matemática que calcula os *outputs*, como será explicado em 4.3.

Desta forma, depois de feito o treino da rede, chegou-se à conclusão que dezasseis neurónios na *hidden layer* seriam suficientes. Chegou-se a esta conclusão, calculando os erros à medida que se variou o número de neurónios. Como se pode comprovar na tabela 4.3, à medida que o número de neurónios, aumentou o erro diminiu. No entanto, pode comprovar-se na tabela 4.4 que o menor erro de previsão está associado a uma *hidden layer* com dezasseis neurónios.

Tabela 4.3: Cálculo do erro de treino variando o número de neurónios na rede neuronal 1.

Nº de Neurónios	erro em $\hat{i}$	erro em $\hat{j}$	erro em $\hat{k}$	erro médio total
14	0,018471389	0,006508931	0,01423548	0,013071933
16	0,012311366	0,005684771	0,012939248	0,010311795
18	0,008580867	0,00259412	0,004203001	0,005125996

Tabela 4.4: Cálculo do erro de teste variando o número de neurónios na rede neuronal 1.

Nº de neurónios	erro em $\hat{i}$	erro em $\hat{j}$	erro em $\hat{k}$	erro médio total
14	0,018985724	0,010745837	0,018671161	0,016134241
16	0,013349896	0,009987064	0,016109614	0,013148858
18	0,019939926	0,012232229	0,016337157	0,016169771

4.3 Treino da Rede Neuronal

Quando se treina uma rede neuronal, com um determinado conjunto de *inputs* e *outputs*, pretende-se que esta seja capaz de ajustar os seus parâmetros internos, de forma a obter os *outputs* utilizados para esse mesmo treino. Mesmo assim, não se pretende que a rede seja demasiado específica para os valores utilizados no treino, possibilitando que esta consiga prever valores que não sejam ainda conhecidos à priori. Nos casos em que a rede não consegue calcular valores desconhecidos, diz-se que esta está *over-fitted*. O treino de uma rede neuronal é semelhante a um problema de optimização, pretendendo que o erro mínimo quadrático seja o menor possível. Este problema pode ser resolvido de muitas formas. Neste caso, esta abordagem foi feita por *backpropagation*. Este é um treino supervisionado em que, depois de calculado o erro, são corrigidos os pesos de todas as conexões entre neurónios desde as saídas até às entradas. É um algoritmo de descida de gradiente [20] [21].

4.3.1 *Agiel Neural Network*

Para proceder ao treino da rede neuronal foi utilizada uma interface gráfica. Existem várias interfaces compatíveis e baseadas nas bibliotecas *FANN* mas, neste caso, foi utilizada a *Agiel Neural Network*. Esta é simples de usar e permite visualizar, graficamente, a diferença entre os *outputs* reais e os *outputs* produzidos pela rede neuronal.

Para se utilizar esta interface, foi necessário em primeiro lugar, importar o ficheiro de treino e o ficheiro de teste no separador *Data Source Preparation*. De seguida, no separador *Network Setup* foi necessário indicar qual seria o ficheiro de treino, assim como quais as colunas correspondestes aos *inputs* e aos *outputs*. Indicou-se também o número de *hidden layers* pelas quais a rede seria composta e quantos neurónios teria cada uma delas.

Com a rede definida passou-se então ao treino da mesma. Este treino foi feito no separador *Training Run*, ajustando-se alguns parâmetros, dos quais o *Maximum Epochs* e o *Maximum Error* para $1 \cdot 10^5$ e para $1 \cdot 10^{-10}$, respetivamente. Este último parâmetro define-se tendo em conta o erro máximo desejado. O *Maximum Epochs*, diz respeito à quantidade de iterações que a rede fará durante o treino para que o erro possa convergir para o erro máximo desejado.

4.4 Resultados Finais

Depois de realizar todos os procedimentos explicados anteriormente, foi possível obter um gráfico de resultados, tanto para o treino como para o teste feito à rede com valores, à priori, desconhecidos. Foi também calculado o erro associado a cada um destes processos.

Tabela 4.5: Erros associados ao treino de cada uma das redes neuronais.

Nº da Rede Neuronal	erro em $\hat{i}$	erro em $\hat{j}$	erro em $\hat{k}$	erro médio total
1	0,012311366	0,005684771	0,012939248	0,010311795
2	0,001440037	0,003260086	0,002569552	0,002423225
3	0,001949329	0,001479147	0,002850707	0,002093061
4	0,002501677	0,000719928	8,5262E-05	0,001102289

Tabela 4.6: Erros associados ao teste de cada uma das redes neuronais.

Nº da Rede Neuronal	erro em $\hat{i}$	erro em $\hat{j}$	erro em $\hat{k}$	erro médio total
1	0,013349896	0,009987064	0,016109614	0,013148858
2	0,015727446	0,023031322	0,024268364	0,021009044
3	0,019586223	0,020180988	0,02517337	0,02164686
4	0,039541434	0,022659537	0,026770257	0,029657076

Como se pode ver, por exemplo na imagem 4.7, a rede neuronal conseguiu aproximar bastante os *outputs* com os valores utilizados no treino. Isto indica, que a rede conseguiu encontrar uma forma de se ajustar a todos os valores usados. Quando foram testados valores, que à priori eram desconhecidos, a rede também conseguiu fazer uma boa aproximação aos valores reais, embora com algum erro associado. Pode ser visto o resultado na imagem 4.8.

Numa rede neuronal, quanta mais informação for utilizada para treiná-la, melhor será a eficiência da mesma na previsão de valores, nunca esquecendo o possível *over-fitting*. Para os testes feitos, foram utilizados cerca de quarenta pares *input-output*. Destes, apenas vinte foram usados para treinar a rede e os outros vinte para testá-la.

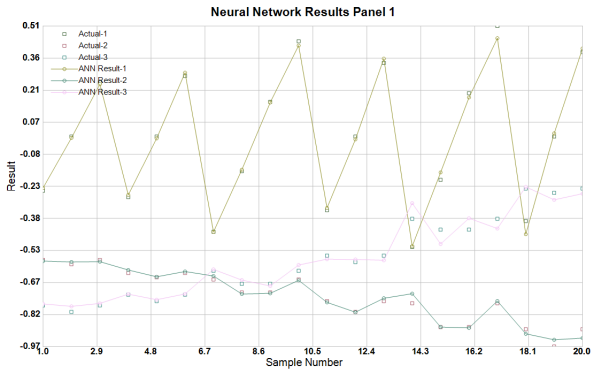


Figura 4.3: Gráfico de resultados referente ao treino feito na rede neuronal número um.

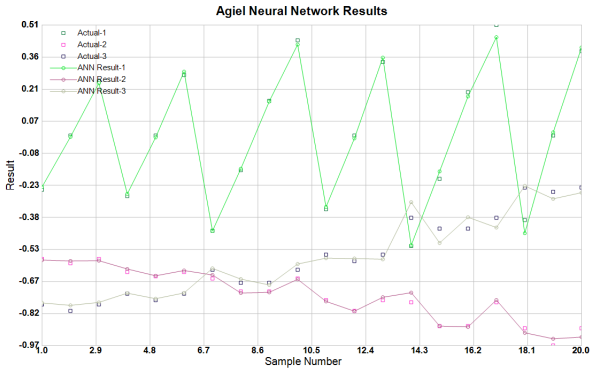


Figura 4.4: Gráfico de resultados referente ao teste feito na rede neuronal número um.

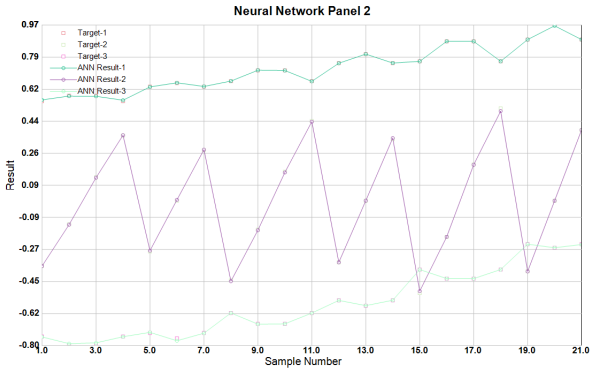


Figura 4.5: Gráfico de resultados referente ao treino feito na rede neuronal número dois.

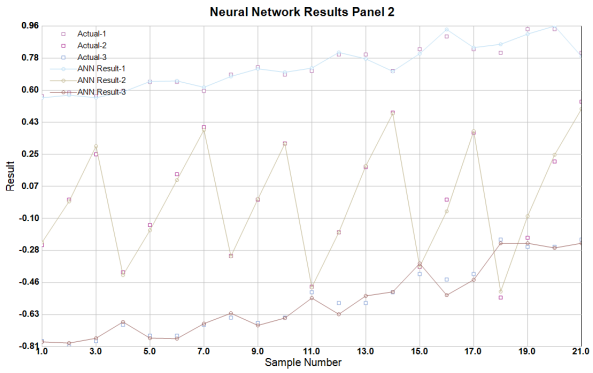


Figura 4.6: Gráfico de resultados referente ao teste feito na rede neuronal número dois.

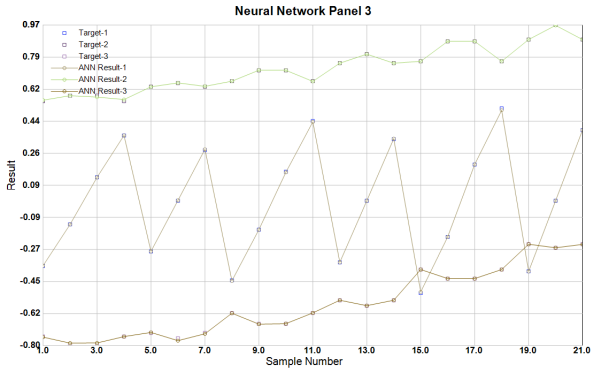


Figura 4.7: Gráfico de resultados referente ao treino feito na rede neuronal número três.

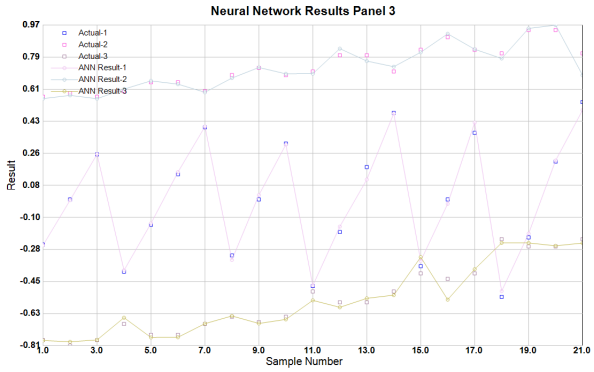


Figura 4.8: Gráfico de resultados referente ao teste feito na rede neuronal número três.

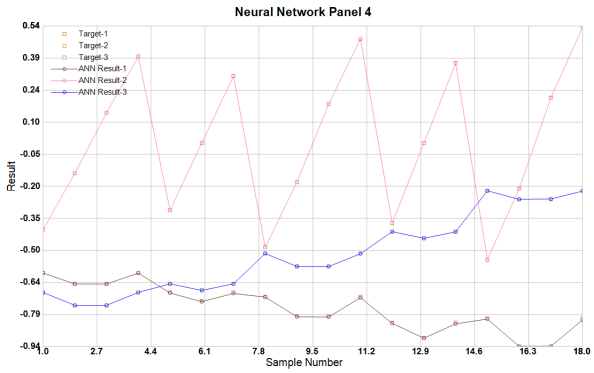


Figura 4.9: Gráfico de resultados referente ao treino feito na rede neuronal número quatro.

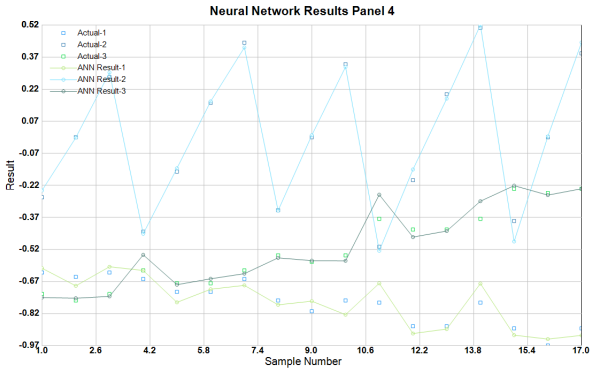


Figura 4.10: Gráfico de resultados referente ao teste feito na rede neuronal número quatro.

Capítulo 5

Conclusões e Trabalho Futuro

Neste capítulo serão expostas as conclusões do trabalho feito, bem como as dificuldades que surgiram durante o seu desenvolvimento. Por último, serão sugeridas possíveis melhorias e trabalhos futuros que se possam levar a cabo.

5.1 Satisfação dos Objectivos

Inicialmente foi pensado fazer-se uma calibração do sistema catadióptrico através de um modelo matemático de visão por reflexão. Para essa calibração foi construída uma estrutura com padrões de xadrez para que fosse possível obter pontos conhecidos no mundo. Este modelo possibilitaría estimar os parâmetros da equação do espelho, os *offsets* da câmara e do espelho bem como os ângulos de rotação de cada um deles. No entanto, este seria bastante complexo de implementar matemáticamente, tendo em conta a quantidade de variáveis a estimar. Desta forma, utilizando-se na mesma a estrutura de xadrez, decidiu-se utilizar uma rede neuronal para efetuar a calibração. A rede, costruída facilmente em C, poderá tornar-se mais útil para os robots de futebol robótico. Isto, tendo em conta que possibilita a previsão de posições e direções sem que seja necessário estimar diretamente todos os parâmetros para que o modelo referido anteriormente funcione. Outra das vantagens da utilização da rede neuronal é o ser possível controlar o erro pretendido e treinar a rede para que esse erro possa convergir para o pretendido.

Surgiram várias dificuldades durante o desenvolvimento do projeto. O primeiro problema prendeu-se com a existência de partes da imagem que estavam tapadas pelas hastes do suporte do espelho. Este problema foi resolvido pré-processando a imagem antes da extração de características. Existe também a questão da parte de extração de características da imagem e a parte da calibração estarem separadas. Este fator, associado ao facto dos ficheiros de treino da rede neuronal não estarem a ser obtidos automaticamente mas sim manualmente numa folha de cálculo, provoca que o tempo dispensado para o tratamento de informação e para a calibração em si seja elevado.

Ainda assim, apesar das dificuldades encontradas, o projeto foi desenvolvido da melhor forma possível, fazendo com que a calibração deste sistema possa ser implementada nos robots de futebol robótico da equipa 5DPO da FEUP.

5.2 Trabalho Futuro

Como trabalho futuro propõe-se que o algoritmo seja otimizado de forma a ser executado automaticamente. Ou seja, as informações serem tratadas a quando da extração de características, para que não seja necessário que os ficheiros para o treino da rede neuronal sejam construídos manualmente.

Em relação à rede e ao seu treino, a fim de se ter mais informações e aumentar a fiabilidade da mesma, propõe-se que a matriz de xadrez seja composta por mais quadrados para assim haver mais pontos utilizáveis a cobrir a área da imagem a calibrar. Poder-se-ia integrar o algoritmo nos robots de futebol robótico, para que a calibração de visão dos mesmos seja um processo mais rápido e fácil de executar. Assim, será possível corrigir a calibração no caso de um choque acidental da bola com o suporte do espelho em ambiente real de jogo, por exemplo.

Por último, seria interessante experimentar outras abordagens para a recolha dos dados, como por exemplo colocar uma tira no chão e rodar o robô sobre si próprio.

Anexo A

Ficheiro de treino e de teste das redes neuronais

As duas primeiras colunas dos ficheiros correspondem aos *inputs* e as três útimas colunas correspondem ao *output*.

A.0.1 Treino Rede 1

1	−0,19	0,28	−0,36	−0,55	−0,75
2	−0,06	0,29	−0,13	−0,58	−0,80
3	0,08	0,29	0,13	−0,58	−0,80
4	−0,21	0,32	−0,40	−0,60	−0,69
5	−0,07	0,33	−0,14	−0,65	−0,75
6	0,09	0,33	0,14	−0,65	−0,75
7	0,23	0,31	0,40	−0,60	−0,69
8	−0,17	0,37	−0,31	−0,69	−0,65
9	0,01	0,38	0,00	−0,73	−0,68
10	0,18	0,37	0,31	−0,69	−0,65
11	−0,29	0,43	−0,48	−0,71	−0,51
12	−0,10	0,45	−0,18	−0,80	−0,57
13	0,12	0,45	0,18	−0,80	−0,57
14	0,31	0,42	0,48	−0,71	−0,51
15	−0,25	0,54	−0,37	−0,83	−0,41
16	0,01	0,56	0,00	−0,90	−0,44
17	0,27	0,53	0,37	−0,83	−0,41
18	−0,43	0,62	−0,54	−0,81	−0,22
19	−0,16	0,70	−0,21	−0,94	−0,26
20	0,18	0,69	0,21	−0,94	−0,26

A.0.2 Treino Rede 2

1	0,30	−0,20	0,55	−0,36	−0,75
2	0,31	−0,08	0,58	−0,13	−0,80
3	0,31	0,05	0,58	0,13	−0,80
4	0,29	0,18	0,55	0,36	−0,75
5	0,35	−0,16	0,63	−0,28	−0,73
6	0,35	−0,01	0,65	0,00	−0,76
7	0,34	0,13	0,63	0,28	−0,73
8	0,39	−0,26	0,66	−0,44	−0,62
9	0,41	−0,11	0,72	−0,16	−0,68
10	0,41	0,07	0,72	0,16	−0,68
11	0,38	0,24	0,66	0,44	−0,62
12	0,47	−0,22	0,76	−0,34	−0,55
13	0,48	−0,02	0,81	0,00	−0,58
14	0,47	0,18	0,76	0,34	−0,55
15	0,53	−0,36	0,77	−0,51	−0,38
16	0,58	−0,15	0,88	−0,20	−0,43
17	0,58	0,09	0,88	0,20	−0,43
18	0,54	0,32	0,77	0,51	−0,38

19	0,69	−0,33	0,89	−0,39	−0,24
20	0,73	−0,04	0,97	0,00	−0,26
21	0,70	0,26	0,89	0,39	−0,24

A.0.3 Treino Rede 3

1	−0,17	−0,30	−0,36	0,55	−0,75
2	−0,05	−0,31	−0,13	0,58	−0,80
3	0,09	−0,31	0,13	0,58	−0,80
4	0,21	−0,30	0,36	0,55	−0,75
5	−0,13	−0,34	−0,28	0,63	−0,73
6	0,02	−0,35	0,00	0,65	−0,76
7	0,17	−0,35	0,28	0,63	−0,73
8	−0,23	−0,39	−0,44	0,66	−0,62
9	−0,06	−0,41	−0,16	0,72	−0,68
10	0,12	−0,41	0,16	0,72	−0,68
11	0,27	−0,38	0,44	0,66	−0,62
12	−0,17	−0,47	−0,34	0,76	−0,55
13	0,03	−0,48	0,00	0,81	−0,58
14	0,23	−0,47	0,34	0,76	−0,55
15	−0,31	−0,54	−0,51	0,77	−0,38
16	−0,08	−0,58	−0,20	0,88	−0,43
17	0,16	−0,58	0,20	0,88	−0,43
18	0,37	−0,53	0,51	0,77	−0,38
19	−0,25	−0,70	−0,39	0,89	−0,24
20	0,05	−0,72	0,00	0,97	−0,26
21	0,33	−0,68	0,39	0,89	−0,24

A.0.4 Treino Rede 4

1	0,40	−0,46	−0,60	−0,40	−0,69
2	0,32	−0,19	−0,65	−0,14	−0,75
3	0,30	0,13	−0,65	0,14	−0,75
4	0,36	0,41	−0,60	0,40	−0,69
5	0,47	−0,27	−0,69	−0,31	−0,65
6	0,44	−0,03	−0,73	0,00	−0,68
7	0,46	0,23	−0,69	0,31	−0,65
8	0,58	−0,31	−0,71	−0,48	−0,51
9	0,55	−0,12	−0,80	−0,18	−0,57
10	0,54	0,09	−0,80	0,18	−0,57
11	0,57	0,28	−0,71	0,48	−0,51
12	0,63	−0,19	−0,83	−0,37	−0,41
13	0,61	−0,01	−0,90	0,00	−0,44
14	0,62	0,16	−0,83	0,37	−0,41
15	0,69	−0,23	−0,81	−0,54	−0,22
16	0,67	−0,09	−0,94	−0,21	−0,26
17	0,67	0,07	−0,94	0,21	−0,26
18	0,68	0,21	−0,81	0,54	−0,22

A.0.5 Teste Rede 1

1	−0,12	0,29	−0,25	−0,57	−0,78
2	0,01	0,29	0,00	−0,59	−0,81
3	0,13	0,29	0,25	−0,57	−0,78
4	−0,14	0,32	−0,28	−0,63	−0,73
5	0,01	0,33	0,00	−0,65	−0,76
6	0,16	0,32	0,28	−0,63	−0,73
7	−0,25	0,36	−0,44	−0,66	−0,62
8	−0,08	0,38	−0,16	−0,72	−0,68
9	0,10	0,38	0,16	−0,72	−0,68
10	0,26	0,36	0,44	−0,66	−0,62
11	−0,20	0,44	−0,34	−0,76	−0,55
12	0,01	0,45	0,00	−0,81	−0,58
13	0,22	0,43	0,34	−0,76	−0,55
14	−0,35	0,51	−0,51	−0,77	−0,38
15	−0,12	0,55	−0,20	−0,88	−0,43
16	0,14	0,55	0,20	−0,88	−0,43
17	0,37	0,50	0,51	−0,77	−0,38
18	−0,31	0,66	−0,39	−0,89	−0,24
19	0,01	0,71	0,00	−0,97	−0,26
20	0,33	0,65	0,39	−0,89	−0,24

A.0.6 Teste Rede 2

1	0,30	−0,14	0,57	−0,25	−0,78
2	0,31	−0,02	0,59	0,00	−0,81
3	0,30	0,12	0,57	0,25	−0,78
4	0,34	−0,23	0,60	−0,40	−0,69
5	0,35	−0,10	0,65	−0,14	−0,75
6	0,35	0,05	0,65	0,14	−0,75
7	0,34	0,20	0,60	0,40	−0,69
8	0,40	−0,19	0,69	−0,31	−0,65
9	0,41	−0,02	0,73	0,00	−0,68
10	0,40	0,15	0,69	0,31	−0,65
11	0,45	−0,31	0,71	−0,48	−0,51
12	0,48	−0,13	0,80	−0,18	−0,57
13	0,48	0,08	0,80	0,18	−0,57
14	0,45	0,27	0,71	0,48	−0,51
15	0,56	−0,26	0,83	−0,37	−0,41
16	0,59	−0,03	0,90	0,00	−0,44
17	0,57	0,22	0,83	0,37	−0,41
18	0,64	−0,44	0,81	−0,54	−0,22
19	0,71	−0,19	0,94	−0,21	−0,26
20	0,72	0,12	0,94	0,21	−0,26
21	0,66	0,39	0,81	0,54	−0,22

A.0.7 Teste Rede 3

1	−0,11	−0,30	−0,25	0,57	−0,78
2	0,02	−0,31	0,00	0,59	−0,81
3	0,15	−0,30	0,25	0,57	−0,78
4	−0,20	−0,34	−0,40	0,60	−0,69
5	−0,05	−0,35	−0,14	0,65	−0,75
6	0,10	−0,35	0,14	0,65	−0,75
7	0,24	−0,33	0,40	0,60	−0,69
8	−0,15	−0,40	−0,31	0,69	−0,65
9	0,03	−0,41	0,00	0,73	−0,68
10	0,20	−0,40	0,31	0,69	−0,65
11	−0,26	−0,46	−0,48	0,71	−0,51
12	−0,07	−0,48	−0,18	0,80	−0,57
13	0,13	−0,48	0,18	0,80	−0,57
14	0,31	−0,45	0,48	0,71	−0,51
15	−0,20	−0,57	−0,37	0,83	−0,41
16	0,04	−0,59	0,00	0,90	−0,44
17	0,27	−0,56	0,37	0,83	−0,41
18	−0,38	−0,66	−0,54	0,81	−0,22
19	−0,11	−0,72	−0,21	0,94	−0,26
20	0,20	−0,71	0,21	0,94	−0,26
21	0,44	−0,64	0,54	0,81	−0,22

A.0.8 Teste Rede 4

1	0,36	−0,34	−0,63	−0,28	−0,73
2	0,30	−0,03	−0,65	0,00	−0,76
3	0,33	0,28	−0,63	0,28	−0,73
4	0,50	−0,37	−0,66	−0,44	−0,62
5	0,45	−0,15	−0,72	−0,16	−0,68
6	0,45	0,10	−0,72	0,16	−0,68
7	0,48	0,34	−0,66	0,44	−0,62
8	0,56	−0,22	−0,76	−0,34	−0,55
9	0,54	−0,02	−0,81	0,00	−0,58
10	0,55	0,19	−0,76	0,34	−0,55
11	0,64	−0,27	−0,77	−0,51	−0,38
12	0,62	−0,10	−0,88	−0,20	−0,43
13	0,61	0,08	−0,88	0,20	−0,43
14	0,63	0,24	−0,77	0,51	−0,38
15	0,68	−0,16	−0,89	−0,39	−0,24
16	0,67	−0,01	−0,97	0,00	−0,26
17	0,67	0,14	−0,89	0,39	−0,24

Referências

[1] Alberto Colombo, Matteo Matteucci, e DG Sorrenti. On the calibration of non single viewpoint catadioptric sensors. *... 2006: Robot Soccer World Cup X*, páginas 194–205, 2007. URL: http://link.springer.com/chapter/10.1007/978-3-540-74024-7{_}17.

[2] R. Vassallo M. Filho F. Pereira, C. Gava. Calibração de sistemas catadióptricos e detecção da pose de robôs móveis por segmentação de imagens omnidirecionais. 9 2005.

[3] Nuno Lau Luís Almeida Bernardo Cunha, José Azevedo. Obtaining the inverse distance map from a non-SVP hyperbolic catadioptric robotic vision system. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 5001 LNAI:417–424, 2008.

[4] Tiago Pereira. Calibration of catadioptric vision systems. Tese de mestrado, Faculdade de Engenharia da Universidade do Porto, 7 2012.

[5] Luis Perdigoto e Helder Araujo. Estimation of mirror shape and extrinsic parameters in axial non-central catadioptric systems. *Image and Vision Computing*, 54:45–59, 2016.

[6] Nino Pereira Gil Lopes, Fernando Ribeiro. Catadioptric system optimisation for omnidirectional RoboCup MSL Robots. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 7416 LNCS:318–328, 2012.

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

[8] OpenCV Team. About. Consultado em 20 de Junho de 2017. URL: http://opencv.org/about.html.

[9] OpenCV Team. Introduction. Consultado em 20 de Junho de 2017. URL: http://docs.opencv.org/2.4/modules/core/doc/intro.html#introduction.

[10] S.Samundeswari Dr. G.P.Rameshkumar. Neural Network, Artificial Neural Network (ANN) and Biological Neural Network (BNN) in Soft Computing. *International Journal of Engineering Sciences Research Technology*, 3:1159–1161, 3 2014.

[11] Priyanka Wankar Sonali. B. Maind. Research Paper on Basic of Artificial Neural Network. *International Journal on Recent and Innovation Trends in Computing and Communication*, 2:96–100, 2014.

[12] Xila Liu Fei Tong. Samples selection for artificial neural network training in preliminary structural design. *Tsinghua Science and Technology*, 10:233–239, 2005.

[13] Lawrence Davis David J. Montana. Training feedforward neural networks using genetic algorithms. *Proceedings of the 11th International Joint Conference on Artificial intelligence - Volume 1*, 1:762–767, 1989.

[14] Alex Krizhevsky Ilya Sutskever Ruslan Salakhutdinov Nitish Srivastava, Geoffrey Hinton. Dropout: A simple way to prevent neural networks from overfitting. *Journal of Machine Learning Research*, 15:1929–1958, 6 2014.

[15] R. Rojas. *Neural Networks*. Springer-Verlag, 1996.

[16] FANN Team. Fann. Consultado em 20 de Junho de 2017. URL: http://leenissen.dk/fann/wp/.

[17] opencv dev team. Camera calibration and 3d reconstruction, 6 2017. Consultado em 18 de Junho de 2017. URL: http://docs.opencv.org/2.4/modules/calib3d/doc/camera_calibration_and_3d_reconstruction.html.

[18] opencv dev team. cornersubpix., 6 2017. Consultado em 18 de Junho de 2017. URL: http://docs.opencv.org/2.4/modules/imgproc/doc/feature_detection.html#void.

[19] C. Mendes P. Horta. Aplicação das redes neuronais artificiais à detecção dos mercados euronext mais rentáveis. 5 2007.

[20] Training an ann. Consultado em 18 de Junho de 2017. URL: http://leenissen.dk/fann/wp/help/neural-network-theory/.

[21] André Siqueira Ruela. Redes neurais feedforward e backpropagation. Universidade Federal de Ouro Preto.