Departamento de Física
Faculdade de Ciências e Tecnologia da Universidade de Coimbra
Ano Lectivo de 2007/2008

Sistema para calibrar um endoscópio rígido dentro da sala de cirurgia

Cadeira de Projecto

Hugo Joel de Jesus Simões
Aluno nº 2003125180

Departamento de Física

Faculdade de Ciências e Tecnologia da Universidade de Coimbra

Ano Lectivo de 2007/2008

Sistema para calibrar um endoscópio rígido dentro da sala de cirurgia

Cadeira de Projecto

Hugo Joel de Jesus Simões

Mestrado Integrado em Engenharia Biomédica

Orientadores:

João Pedro Barreto

Paulo Menezes

Setembro de 2008

Agradecimentos

Depois deste ano de trabalho, era imprescindível agradecer a algumas pessoas por todo o seu apoio e ajuda prestada durante a realização deste projecto. Ao professor João Pedro Barreto, agradeço por toda a ajuda que me prestou, quer na formulação matemática quer na implementação dos algoritmos. Estendo os meus agradecimentos ao professor Paulo Menezes também por todo o apoio e ajuda que me disponibilizou. E também ao Michel Antunes e ao engenheiro João Santos, por toda a ajuda prestada, sobretudo na fase de implementação de algoritmos.

Agradeço ainda a todos os meus colegas de laboratório por terem convivido comigo e de certa forma terem colaborado para a realização deste projecto.

Por último, quero agradecer a todos os meus amigos pela força que me deram, bem como à minha namorada e à minha família, particularmente aos meus pais, irmãos e cunhada.

A todos, um muito obrigado!

Resumo

A cirurgia artroscópica tem assumido um papel preponderante na área da cirurgia médica. O facto de apenas serem necessárias duas pequenas incisões tem a vantagem de facilitar a recuperação do paciente já que os tecidos não são tão danificados. Contudo, devido à baixa qualidade do vídeo artroscópico, é necessário que o médico seja experiente, o que implica um grande treino para poder realizar artroscopia. Pretende-se construir um sistema de navegação cirúrgica de modo facilitar a aprendizagem do médico e até mesmo aumentar a taxa de sucesso deste tipo de cirurgia. No entanto, para que tal seja possível, deve-se efectuar processamento das imagens artroscópicas de modo a se poder extrair o máximo de informação. Este relatório aborda um método de segmentação da área útil da imagem artroscópica baseado num valor de *threshold* adaptativo, em conjunto com um método de estimação robusta, o RANSAC, e um filtro de Kalman. Conseguiu-se obter um algoritmo robusto e estável e, além disso, capaz de efectuar o processamento da imagem em tempo real, como pretendido. Desenvolveu-se também um novo método de extracção de cantos automático para extrair informação necessária para proceder à calibração do artroscópio, baseado na entropia dos ângulos do gradiente. Apesar da elevada distorção radial, verificou-se que o método desenvolvido consegue marcar os cantos da grelha com boa precisão. É ainda descrito um modelo teórico para proceder à contagem do número de quadrículas. Embora sejam necessários efectuar alguns ajustes aos métodos de extracção e contagem de cantos, conclui-se que os primeiros resultados práticos pressupõem que este método seja robusto e tenha grande precisão na detecção de cantos.

Abstract

Arthroscopic surgery has gained a front role in the scope of medical surgery. The fact that only two small incisions are necessary for this procedure makes the recovery of the patient quicker, as not so much tissue is damaged. Nevertheless, because of the low quality of the arthroscopic video, the doctor needs to be very experienced, which means that large training in this type of surgery is required. The aim is to build a system of surgical navigation so as to make the learning process easier on the part of the doctor and even to increase the success rate of this type of surgery. However, for this to be possible, we need to process the arthroscopic images so that we can extract the most information possible. This report puts forth an approach based on a method of segmentation of the useful area on the arthroscopic image, based on an adaptative threshold value, together with a method of robust estimation, RANSAC, and a Kalman filter. We were able to obtain an algorithm which is both robust and stable and, besides, able to process the image in real time, as intended. We have also developed a new automatic method of extraction of corners in order to extract the necessary information to calibrate the arthroscope, based on the entropy of the gradient angles. In spite of the high radial distortion, this method has proved to mark the corners of the grid with good precision. We also describe a theoretical model to count the number of squares. Although some adjustments are still needed as regards the methods of extraction and corner counting, we conclude that the first practical results presuppose this method to be robust and to have great precision in corner detection.

Índice

Agradecimentos **3**
Resumo **4**
Abstract **5**
Índice **6**
Índice de figuras **8**

Capítulo 1: Introdução **11**

Capítulo 2: Introdução às cónicas **14**
2.1 Cónica 14
2.2 Parâmetros de caracterização de uma cónica 15
2.2.1 Centro da cónica 15
2.2.2 Ângulo de rotação 15
2.2.3 Cálculo dos semi-eixos maior e menor 17
2.2.4 Cálculo da box envolvente 18
2.3 Calculo da equação geral da cónica a partir dos seus parâmetros 19
2.4 Determinação da cónica a partir de pontos da imagem 20
2.5 Método específico de *fit* de elipses 21
2.6 Detecção de pontos da fronteira na imagem 22
2.6.1 Método baseado em máximos de gradientes 23
2.6.2 *Threshold* adaptativo baseado no histograma 24
2.7 Detecção da marca da lente e cálculo da sua rotação 26

Capítulo 3: Implementação do Algoritmo de Segmentação **28**
3.1 O artroscópio 28
3.2 Análise ao comportamento dos histogramas 29
3.2.1 O efeito do white-set 29
3.2.2 Cálculo do valor de *threshold* 35
3.3 Estimação robusta: RANSAC 37
3.4 Filtro de Kalman 40
3.5 Cálculo da percentagem de pixeis de fundo 43
3.6 Testes ao algoritmo de segmentação implementado 46
3.7 Conclusões 50

Capítulo 4: Extracção de Cantos da Grelha **51**
4.1 Calibração de câmaras 51
4.2 Alguns algoritmos de extracção de cantos existentes 53
4.2.1 Toolbox de calibração de câmaras de Bouguet 53
4.2.2 Função *FindChessBoardCorners* do OpenCV 55
4.3 Aplicação dos algoritmos de detecção de cantos descritos às imagens artroscópicas 56

4.4 Implementação de novos métodos de detecção de cantos de grelhas 57
4.4.1 Método baseado no detector de Harris 58
4.4.2 Método baseado na entropia dos ângulos do gradiente 60
4.5 Contagem das quadrículas 62
4.6 Conclusões 64

Capítulo 5: Considerações Finais 65

Anexos 66
A. Esquema de funcionamento do algoritmo de segmentação implementado em C 66
B. Configuração do algoritmo de segmentação: *configuracao.h* 67
C. Documentação das funções criadas em C 69
D. Documentação das funções criadas em MATLAB 74

Bibliografia 78

Índice de figuras

Figura 1.1: Imagem artroscópica típica 12

Figura 2.1: Esquema do sinal dos ângulos da cónica 17

Figura 2.2: As tangentes à cónica nos pontos S^- e S^+ intersectam-se num ponto S 19

Figura 2.3: Imagem de uma folha branca adquirida pelo artroscópio e com a fonte de luz ligada 23

Figura 2.4: Pontos detectados (assinalados a vermelho) pelo método proposto **(a)** para uma superfície branca e **(b)** para a imagem de uma grelha 24

Figura 2.5: (a) Imagem obtida por binarização; **(b)** imagem obtida após filtragem da imagem (a); **(c)** imagem obtida após aplicação de erosão na imagem (b); **(d)** imagem obtida após subtracção da imagem (c) à imagem (b) 25

Figura 2.6: Pontos detectados (assinalados a vermelho) pelo método proposto 25

Figura 2.7: Imagem adquirida pelo artroscópio. A verde está assinalada a marca da lente 26

Figura 3.1: Componentes do sistema de artroscopia utilizado: 1 – artroscópio; 2 – câmara de 3 CCD; 3 – unidade de controlo; 4 – fonte de luz; 5 – fibra óptica (guia de luz) 28

Figura 3.2: Imagem de uma superfície branca obtida **(a)** antes de efectuar o white-set e com fonte de luz, **(b)** depois de efectuar o white-set com e com fonte de luz, e **(c)** depois de efectuar o white-set e sem fonte de luz 30

Figura 3.3: Histogramas correspondentes às imagens da figura 3.2 **(a)** histograma da componente R, **(b)** histograma da componente G, **(c)** histograma da componente B, e **(d)** histograma da imagem em escala de cinzentos 31

Figura 3.4: Imagem de uma grelha obtida **(a)** antes de efectuar o white-set e com fonte de luz, **(b)** depois de efectuar o white-set com e com fonte de luz, e **(c)** depois de efectuar o white-set e sem fonte de luz 31

Figura 3.5: Histogramas correspondentes às imagens da figura 3.4 **(a)** histograma da componente R, **(b)** histograma da componente G, **(c)** histograma da componente B, e **(d)** histograma da imagem em escala de cinzentos 32

Figura 3.6: Simulação de uma imagem real obtida com fonte de luz **(a)** antes e **(b)** depois de efectuar o white-set. **(c)** Imagem de uma cena do mundo obtida após realizar o white-set 33

Figura 3.7: Histogramas correspondentes às imagens (a) e (b) da figura 3.6 **(a)** histograma da componente R, **(b)** histograma da componente G, **(c)** histograma da componente B, e **(d)** histograma da imagem em escala de cinzentos 33
Figura 3.8: Histogramas correspondentes à imagem da figura 3.6(c) 34
Figura 3.9: Gráficos de ganho impostos pelo white-set numa imagem **(a)** de uma superfície branca e **(b)** de uma grelha de xadrez 34
Figura 3.10: Histograma típico de uma imagem de uma grelha 35
Figura 3.11: Resultados da detecção de pontos obtidos para **(a)** imagem de superfície branca, **(b)** imagem de uma grelha, **(c)** imagem de uma cena do mundo, **(d)** simulação de imagem real, **(e)** imagem de grelha obtida sem fonte de luz, onde o valor de corte foi calculado com base no histograma da imagem em escala de cinzentos, e **(f)** imagem de grelha obtida sem fonte de luz, onde o valor de corte foi calculado com base no histograma da componente verde da imagem RGB 36
Figura 3.12: (a) Conjunto de pontos com a presença de alguns *outliers*. **(b)** Ajuste do conjunto de pontos a uma recta ignorando os pontos *outliers* (a vermelho) 38
Figura 3.13: (a) Pontos detectados (assinalados a vermelho) pelo método proposto em 2.6.2. **(b)** Pontos detectados (assinalados a vermelho) após ter sido aplicado o RANSAC aos pontos estimados pelo método descrito em 2.6.2 39
Figura 3.14: Esquema das equações calculados durante o algoritmo do filtro de Kalman 42

Figura 3.15: (a) Exemplo de histograma com artefacto, o qual pode levar a erros de segmentação. **(b)** Resultado da binarização da imagem obtido 44
Figura 3.16: Imagem típica de superfície branca (folha de papel) utilizada para calcular percentagem de pixeis de fundo 44
Figura 3.17: Histograma de uma imagem de uma superfície branca 45
Figura 3.18: Resultado da binarização da imagem da figura 3.16, utilizando um valor de corte de 40 45
Figura 3.19: Formas possíveis de mostrar os resultados **(a)** apenas se mostra a cónica e o ângulo da marca da lente, e **(b)** mostra todos os parâmetros da cónica, o ângulo da marca da lente, a cónica (a vermelho), os pontos *inliers* (a amarelo) e os *outliers* (a azul) estimados no RANSAC e a box da cónica (a verde) 46
Figura 3.20: (a) Norma (em pixeis) e **(b)** ângulo (em radianos) do vector definido pelo centro da cónica e o centro da imagem no caso da câmara estar fixa 47
Figura 3.21: (a) Norma (em pixeis) e **(b)** ângulo (em radianos) do vector definido pelo centro da cónica e o centro da imagem no caso da câmara rodar em relação à lente 47

Figura 3.22: (a) Norma (em pixeis) e **(b)** ângulo (em radianos) do vector definido pelo centro da cónica e o centro da imagem no caso de ser aplicada tensão à lente 47
Figura 3.23: Comprimento dos semi-eixos **(a)** maior e **(b)** menor (em pixeis) em cada um dos casos estudados 48
Figura 3.24: Ângulo de rotação da cónica, em graus, no caso **(a)** da câmara estar fixa, **(b)** de ser aplicada tensão na lente, e **(c)** da câmara rodar em relação à lente 49
Figura 3.25: Ângulo de rotação da marca da lente, em graus, no caso **(a)** da câmara estar fixa, **(b)** de ser aplicada tensão na lente, e **(c)** da câmara rodar em relação à lente 49

Figura 4.1: Relação entre o plano imagem e o plano da câmara e o plano da câmara e o mundo 3D.....51
Figura 4.2: Relação entre o plano da imagem e o mundo 3D.....52
Figura 4.3: Resumo do procedimento de calibração de uma câmara.....53
Figura 4.4: Exemplo de como se marcam os quatro cantos externos da grelha de calibração. O primeiro canto que se marca é a origem.....53
Figura 4.5: Resultado da detecção de cantos de uma grelha obtido pela toolbox de calibração de câmaras de Bouguet.....54
Figura 4.6: Parâmetros de entrada e de saída da função *FindChessBoardCorners*.....55
Figura 4.7: Resultado da detecção de cantos de uma grelha obtido pela função *FindChessBoardCorners*.....56
Figura 4.8: (a) Resultado da detecção de cantos da grelha numa imagem artroscópica obtido com a toolbox de calibração de câmaras de Bouguet. **(b)** Resultado da detecção de cantos da grelha numa imagem artroscópica obtido com a função *FindChessBoardCorners* 57
Figura 4.9: Ideia base do detector de Harris. (a) zona constante: não existe variações ao longo das todas as direcções; (b) aresta: não existem variações ao longo da direcção da aresta; e (c) canto: variações significantes em todas as direcções 58
Figura 4.10: Resultado obtido pelo método de Harris.....59
Figura 4.11: Esquematização dos quatro conjuntos principais de ângulos dos gradientes da imagem de uma grelha.....60
Figura 4.12: Resultado obtido pelo método de entropia dos ângulos do gradiente.....62
Figura 4.13: Primeira iteração do processo de contagem de quadrículas 63
Figura 4.14: Segunda iteração do processo de contagem de quadrículas 64

Capítulo 1: Introdução

A artroscopia é um procedimento cirúrgico minimamente invasivo utilizado para examinar e tratar lesões no interior das articulações e é feita com o recurso a um artroscópio, um tipo de endoscópio rígido que pode ser inserido dentro da articulação com recurso a uma pequena incisão. A grande vantagem deste tipo de cirurgia relativamente às restantes é que as articulações não precisam de ser totalmente abertas. São necessárias apenas duas pequenas incisões: uma para o artroscópio e outra para os instrumentos cirúrgicos. Assim, consegue-se facilitar a recuperação do paciente e até mesmo aumentar a taxa de sucesso da cirurgia devido ao menor trauma a que os tecidos conectivos são sujeitos. [1]

Esta técnica é bastante utilizada na reconstrução do Ligamento Cruzado Anterior (LCA) no joelho. O LCA é um estabilizador primário do joelho que une o fémur à tíbia a nível intra-articular e tem como função estabilizar o movimento anterior da tíbia relativamente ao fémur quando o joelho se movimenta da flexão para a extensão. A sua lesão ocorre frequentemente e pode surgir quando existe rotação do joelho, situação que acontece sempre que existe uma variação brusca da direcção. Uma vez que o tecido ligamentar não regenera é necessário proceder a uma intervenção cirúrgica para o tratamento deste tipo de lesões, onde o ligamento roto é substituído por um enxerto. Este procedimento tem os seguintes passos: (1) colheita de tecido para substituir o ligamento; (2) remoção do ligamento roto; (3) abertura de um túnel ósseo que atravessa o joelho nos pontos de inserção do ligamento original; (4) introdução do enxerto no joelho através do túnel aberto; (5) fixação do enxerto usando parafusos bio-absorviveis. [2]

Este procedimento deve ser realizado por cirurgiões especializados, com alguns anos de treino e que conheçam bem a anatomia do joelho, já que a navegação no interior da cavidade é realizado com recurso a um artroscópio. A abertura do túnel para introduzir o enxerto é o passo mais crítico deste procedimento. Este deve estar correctamente posicionado. Basta um pequeno desvio para que sejam geradas tensões anormais durante o movimento, o que pode levar à rotura do enxerto introduzido. Tal situação causa dor e instabilidade no paciente e gera novas lesões, o que pode levar à necessidade de uma segunda intervenção cirúrgica. Estudos estatísticos mostram que a taxa de sucesso da técnica de reconstrução do LCA é inferior a 85% e em caso desta falhar, em 50% dos casos é devida ao mau posicionamento do enxerto. Podem ser utilizados sistemas computacionais de modo a auxiliar a navegação no interior do joelho e assim, tentar melhorar a taxa de sucesso desta técnica, bem como diminuir os requisitos de treino e a curva de aprendizagem do cirurgião. [2]

Assim, pretende-se desenvolver um navegador cirúrgico de auxílio à reconstrução do LCA. Um navegador cirúrgico consiste num sistema de visualização que fornece informação sobre a posição dos

instrumentos cirúrgicos relativamente aos ossos, em tempo real. A posição e orientação dos objectos exteriores ao corpo humano são facilmente determinadas usando um sistema de seguimento óptico: uma cabeça estéreo com câmaras infra-vermelhas segue um conjunto de marcadores ópticos acoplados a cada um dos objectos, permitindo o cálculo da posição dos mesmos relativamente a um sistema de referência no mundo. No entanto, para as estruturas anatómicas, neste caso, a tíbia e o fémur, é mais difícil de obter a sua posição e orientação, já que estas não se encontram visíveis do exterior. Assim, a solução pode passar pela reconstrução da cavidade do joelho a partir de uma sequência de vídeo artroscópico. Os resultados de reconstrução poderão ser registados com uma imagem volumétrica pré-operatória de modo a determinar a posição do fémur/tíbia em relação à câmara. Uma vez que se conhece a posição da câmara no mundo, consegue-se determinar a posição dos ossos relativamente ao sistema de referência do mundo. [2]

Para que se consiga retirar o máximo de informação do vídeo artroscópico, é necessário proceder à calibração da câmara. A calibração consiste na determinação da matriz de projecção da câmara, a qual representa o mapeamento entre as coordenadas de um ponto no mundo e as coordenadas de um ponto na imagem. Este é um tema bastante estudado na literatura. No entanto, a maioria dos métodos existentes destinam-se a ser aplicados em cenas do quotidiano adquiridas por uma câmara em perspectiva convencional. [2]

O primeiro objectivo deste trabalho consiste em efectuar a segmentação da área útil da imagem artroscópica, em tempo real. A seguinte figura mostra uma imagem artroscópica típica.

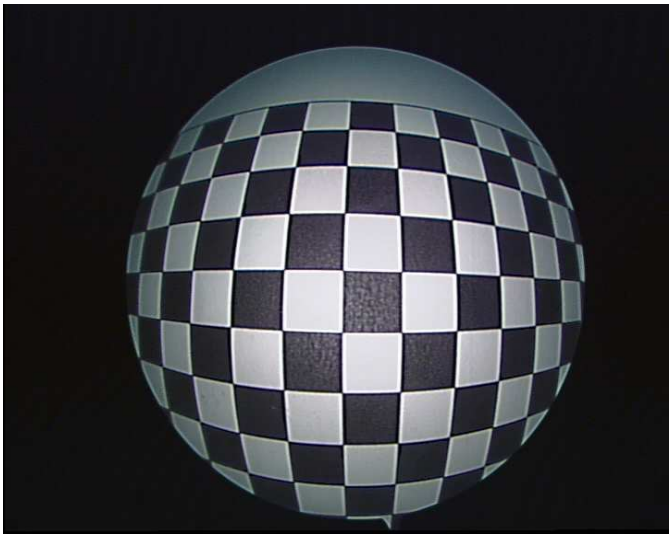

Figura 1.1: Imagem artroscópica típica.

Neste caso, a fronteira entre a zona útil e o fundo da imagem vai ser aproximada por uma cónica, mais especificamente, uma elipse. Após a equação da elipse fronteira ser determinada, é possível extrair informação paramétrica desta e inferir acerca do movimento da lente artroscópica entre *frames* (por exemplo, no caso de haver tensão ou torção da lente, ou no caso de haver rotação da lente relativamente à câmara). A informação obtida pode então ser utilizada para fazer correcções na posição da câmara relativamente ao sistema de referência do mundo. O grande desafio desta tarefa é conseguir implementar um bom método de segmentação, robusto e estável, e que seja rápido, uma vez que se pretende que seja implementado em tempo real.

O segundo objectivo consiste na construção de um algoritmo de detecção de cantos de uma grelha de xadrez, de modo a obter-se informação útil que possa ser utilizada para a calibração da câmara. Como se verifica na figura 1.1, a imagem artroscópica apresenta bastante distorção radial, artefacto pelo qual a maioria dos algoritmos de detecção de cantos falha. Assim, pretende-se obter um método rápido, eficiente, preciso, de fácil aplicação e completamente autónomo de detecção de cantos da grelha que possa ser efectuado pelo cirurgião dentro da sala de operações. E com a informação extraída, proceder-se à calibração da câmara. A distorção radial que as imagens artroscópicas apresentam e a necessidade do método ser preciso são o grande desafio desta tarefa.

Este trabalho faz parte de um projecto designado ArtrhoNav, financiado pela Fundação para a Ciência e Tecnologia e que tem por objectivo desenvolver um sistema de navegação para a Cirurgia de Reconstrução do Ligamento Cruzado Anterior por Via Minimamente Invasiva e envolve o Instituto de Sistemas e Robótica (onde este trabalho foi realizado), o grupo de Cirurgia do Joelho dos Hospitais da Universidade de Coimbra, o Centro de Informática e Sistemas da Universidade de Coimbra e conta ainda com a colaboração da Critical Software.

Este relatório está dividido em 5 capítulos: no primeiro capitulo é feita uma introdução ao tema e são apresentados os objectivos deste trabalho. No segundo capítulo é descrita alguma teoria sobre as cónicas e é formulada uma solução para a detecção de pontos da fronteira. No terceiro capitulo o algoritmo de segmentação é implementado. São feitos estudos aos histogramas de imagens e são feitos alguns testes ao algoritmo de segmentação implementado. No quarto capítulo é apresentado um método de detecção automática de cantos de uma grelha e é formulada uma solução para proceder À contagem do número de quadrículas da grelha. Por último, no quinto capítulo são feitas as considerações finais acerca deste projecto.

Capítulo 2: Introdução às cónicas

Como já foi dito, o primeiro objectivo deste trabalho foi efectuar a segmentação da zona útil da imagem artroscópica em tempo real. A fronteira foi aproximada por uma cónica, mais especificamente uma elipse e após a determinação da sua equação, extraindo parâmetros da elipse, tais como o comprimento dos semi-eixos, a posição do centro e o ângulo de rotação, quer da cónica, quer da marca da lente, é possível retirar alguma informação acerca do movimento da lente entre *frames*. Além disso, no caso de se pretender armazenar informação para efectuar posterior processamento, uma vez que se tem informação acerca da zona útil da imagem, permite poupar tempo e espaço no armazenamento de informação, já que a imagem segmentada é menor.

Neste capítulo vai ser feita uma introdução teórica sobre cónicas: como calcular a sua equação, como extrair determinados parâmetros a partir da informação obtida. Na parte final do capítulo irão ser expostos as principais aproximações que foram feitas para a extracção de pontos da fronteira.

2.1 Cónica

Uma cónica é uma curva descrita por uma equação de segunda ordem no plano e pode ser parameterizada por uma matriz simétrica, tal que

$$\Omega = \begin{bmatrix} a & b & d \\ b & c & e \\ d & e & f \end{bmatrix}. \qquad (2.1)$$

Um ponto $X = (x, y, w)$ pertence à curva Ω se, e só se, $X^T \Omega X = 0$ (2.2). Expandindo esta condição,

$$\begin{bmatrix} x & y & \omega \end{bmatrix} \begin{bmatrix} a & b & d \\ b & c & e \\ d & e & f \end{bmatrix} \begin{bmatrix} x \\ y \\ \omega \end{bmatrix} =$$

$$ax^2 + 2bxy + cy^2 + 2dx\omega + 2ey\omega + f\omega^2 = 0$$

e fazendo $\omega = 1$, a expressão torna-se na equação geral de uma curva planar quadrática

$$ax^2 + 2bxy + cy^2 + 2dx + 2ey + f = 0\,. \quad (2.3)$$

As cónicas não degeneradas classificam-se em três grandes grupos: elipses, parábolas e hipérboles. Uma cónica diz-se não degenerada se a matriz Ω for *full rank*. Se, ao invés, a matriz for deficiente de *rank*, a cónica gerada classifica-se como sendo degenerada. Existem dois tipos principais

de cónicas degeneradas: uma par de linhas distintas (no caso de a matriz ser de *rank* 2) e uma linha repetida (no caso de a matriz ser de *rank* 1). [3]

2.2 Parâmetros de caracterização de uma cónica

Na secção anterior definiu-se uma cónica e a sua equação geral. A partir desta, é possível determinar parâmetros de caracterização de uma curva cónica, tais como, o centro, o ângulo de rotação, o comprimento dos semi-eixos e determinar a box envolvente da cónica.

2.2.1 Centro da cónica

Um ponto x e uma cónica Ω definem uma linha $l = \Omega.x$. Então, a linha l é chamada a polar de x em relação a Ω e o ponto x é o pólo de l em relação a Ω.

O centro da cónica é o pólo da linha no infinito relativamente à cónica. O centro C da cónica Ω é dado por

$$C = \Omega^* \pi_\infty$$

onde Ω^* é o envelope cónico (adjunta) de Ω e π_∞ é a linha do infinito na posição canónica ($\pi_\infty = (0 \quad 0 \quad 1)^t$). No caso da matriz ser simétrica e não-singular, $\Omega^* = \Omega^{-1}$. Ou seja, o centro é dado por

$$C = \Omega^{-1}\pi_\infty. \qquad (2.4)$$

2.2.2 Ângulo de rotação

Na secção 2.1, definiu-se

$$ax^2 + 2bxy + cy^2 + 2dx + 2ey + f = 0 \quad (2.5)$$

como sendo a equação geral de uma curva planar quadrática. Olhando para a equação, facilmente se conclui que o termo $2bxy$ apenas existe no caso de a cónica estar rodada (ou seja, os seus eixos maior e menor não estarem alinhados com os eixos do referencial). Através deste termo, é possível determinar qual o ângulo de rotação da cónica. [4] Para um ângulo θ arbitrário, a rotação entre um ponto x e um ponto x' é definida por

$$\begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} \cos\theta & \sin\theta \\ -\sin\theta & \cos\theta \end{bmatrix} \begin{bmatrix} x' \\ y' \end{bmatrix}.$$

Calculando as expressões para x, y, xy, x^2 e y^2, tem-se que

$$
\begin{aligned}
x &= x'\cos\theta + y'\sin\theta \\
y &= -x'\sin\theta + y'\cos\theta \\
xy &= -x'^2\cos\theta\sin\theta + x'y'\left(\cos^2\theta - \sin^2\theta\right) + y'^2\cos\theta\sin\theta \\
x^2 &= x'^2\cos^2\theta + 2x'y'\cos\theta\sin\theta + y'^2\sin^2\theta \\
y^2 &= x'^2\sin^2\theta - 2x'y'\sin\theta\cos\theta + y'^2\cos^2\theta.
\end{aligned}
$$

Fazendo a substituição na (2.5) e agrupando os termos, tem-se que

$$
\begin{aligned}
& x'^2(a\cos^2\theta + c\sin^2\theta - 2b\cos\theta\sin\theta) \\
& + x'y'\left[2a\cos\theta\sin\theta - 2c\sin\theta\cos\theta + 2b(\cos^2\theta - \sin^2\theta)\right] \\
& + y'^2(a\sin^2\theta + c\cos^2\theta + 2b\cos\theta\sin\theta) \\
& + x'(2d\cos\theta - 2e\sin\theta) + y'(2d\sin\theta + 2e\sin\theta) \\
& + f = 0.
\end{aligned}
$$

Rescrevendo a equação na forma da equação geral de uma curva planar quadrática, tem-se

$$a'x'^2 + 2b'x'y' + c'y'^2 + 2d'x' + 2e'y' + f' = 0 \quad (2.6)$$

onde os coeficientes são

$$
\begin{aligned}
a' &= a\cos^2\theta + c\sin^2\theta - 2b\cos\theta\sin\theta \\
b' &= (a-c)\cos\theta\sin\theta + b(\cos^2\theta - \sin^2\theta) \\
c' &= a\sin^2\theta + c\cos^2\theta + 2b\cos\theta\sin\theta) \\
d' &= d\cos\theta - e\sin\theta \\
e' &= d\sin\theta + e\sin\theta \\
f' &= f.
\end{aligned}
$$

Como já foi dito anteriormente, para que a cónica tenha a os seus eixos alinhados com os eixos do referencial, o termo $2b'x'y'$ terá de ser nulo. Através desta condição, é possível determinar o ângulo entre o eixo da cónica e a horizontal.

$$
\begin{aligned}
b' &= (a-c)\cos\theta\sin\theta + b(\cos^2\theta - \sin^2\theta) \\
&= b\cos(2\theta) - \frac{1}{2}(c-a)\sin(2\theta) = 0
\end{aligned}
$$

Resolvendo em ordem a θ, obtém-se

$$\theta = -\frac{1}{2}\cot^{-1}\left(\frac{c-a}{2b}\right)$$

ou, para ser mais fácil a sua aplicação

$$\theta = -\frac{1}{2}\tan^{-1}\left(\frac{2b}{c-a}\right).$$

É importante referir que se convencionou que o ângulo θ é positivo no sentido horário e negativo no sentido anti-horário.

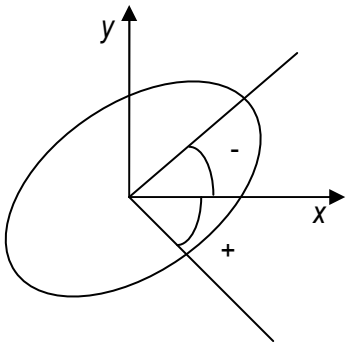

Figura 2.1: Esquema do sinal dos ângulos da cónica.

2.2.3 Cálculo dos semi-eixos maior e menor

Voltando à equação (2.6), e aplicando uma rotação θ, obtemos a seguinte equação

$$a'x'^2 + c'y'^2 + 2d'x' + 2e'y' + f' = 0$$

onde os coeficientes a', c', d', e' e f' têm os valores já referidos na secção anterior. A existência de termos em x' e/ou y' são um indicador para o facto da cónica não estar centrada na origem do referencial. No entanto, é possível efectuar uma translação de modo a torná-la centrada na origem. [4] Assim, da equação anterior,

$$a'(x'^2 + \frac{2d'}{a'}x') + c'(y'^2 + \frac{2e'}{c'}y') + f' = 0$$

Somando $\frac{d'^2}{a'}$ e $\frac{e'^2}{c'}$ em ambos o membros da equação anterior tem-se

$$a'(x' + \frac{d'}{a'})^2 + c'(y' + \frac{e'}{c'})^2 = -f' + \frac{d'^2}{a'} + \frac{e'^2}{c'}. \quad (2.7)$$

Definindo $x'' \equiv x' + d'/a'$, $y'' \equiv x' + e'/c'$ e $f'' \equiv -f + d'^2/a' + f'^2/c'$ e substituindo em (2.7)

$$a'x''^2 + c'y''^2 = f''$$
$$\Leftrightarrow \frac{a'}{f''}x''^2 + \frac{c'}{f''}y''^2 = 1.$$

A equação geral de uma elipse centrada na origem e com os seus eixos alinhados com os eixos do sistema de referência é

$$\frac{x^2}{M^2} + \frac{y^2}{m^2} = 1,$$

onde M e m representam, respectivamente, os semi-eixos maior e menor da cónica.

Comparando as duas equações, conclui-se que os comprimentos dos semi-eixos são

$$M = \sqrt{\frac{f''}{a'}} \text{ e } m = \sqrt{\frac{f''}{c'}},$$

onde a', c' e f'' têm os valores já definidos anteriormente.

2.2.4 Cálculo da box envolvente

Em [3] foi demonstrado que uma recta r intersecta uma cónica Ω em dois pontos, P^- e P^+, cujas as coordenadas são dadas por

$$P^{\pm} = \left(\pm\sqrt{-r^t\Omega^* r}I + \tilde{r}\Omega\right)I_s r, \quad (2.8)$$

onde

$$\Omega = \begin{bmatrix} a & b & d \\ b & c & e \\ d & e & f \end{bmatrix}, \ \Omega^* = \begin{bmatrix} fc-e^2 & ed-bf & be-cd \\ ed-bf & af-d^2 & bd-ae \\ be-cd & bd-ae & ac-b^2 \end{bmatrix}, \ I = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix},$$

$$I_s = \begin{bmatrix} 0 & 0 & -1 \\ 0 & 0 & -1 \\ 1 & 1 & 0 \end{bmatrix}, \ r = (r_1, r_2, r_3)^t \text{ e } \tilde{r} = \begin{bmatrix} 0 & -r_3 & r_2 \\ r_3 & 0 & -r_1 \\ -r_2 & r_1 & 0 \end{bmatrix}.$$

Note-se que a recta r pode ser definida por $r = \Omega.S$, onde S é o ponto de intersecção das rectas tangentes à cónica nos pontos S^- e S^+. A figura 2.2 ilustra esta situação.

Para determinar a box da cónica, é necessário calcular, por um lado, as coordenadas dos dois pontos cujas as rectas tangentes à cónica nesses pontos são verticais e, por outro lado, os dois pontos cujas as rectas tangentes à cónica nesses pontos são horizontais. Duas rectas paralelas entre si

intersectam-se no infinito. Portanto, para calcular os pares de pontos pretendidos, basta forçar a que o ponto S seja um ponto definido no infinito. Em geometria projectiva, uma recta $t = (1,0,0)^t$ é uma recta horizontal no infinito. Definindo $S = (1,0,0)^t$, calculando $r = \Omega.S$ e substituindo em (2.8), calculam-se os pontos que delimitam a cónica verticalmente (em cima e em baixo). Por outro lado, uma recta $t = (0,1,0)^t$ é uma recta vertical no infinito. Definindo $S = (0,1,0)^t$, calculando $r = \Omega.S$ e substituindo em (2.8), calculam-se os pontos que delimitam a cónica horizontalmente (à esquerda e à direita).

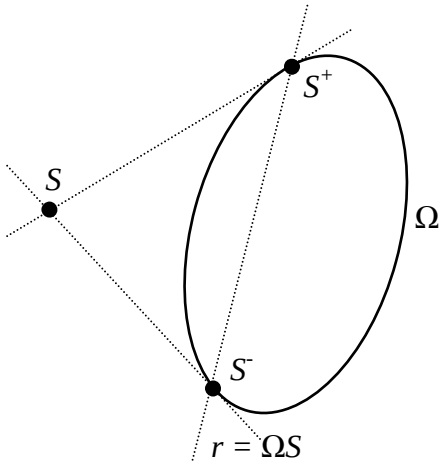

Figura 2.2: As tangentes à cónica nos pontos S^- e S^+ intersectam-se num ponto S.

2.3 Calculo da equação geral da cónica a partir dos seus parâmetros

Nas secções anteriores explicou-se como chegar aos parâmetros de caracterização de uma cónica. Nesta secção, pretende-se efectuar o processo inverso, ou seja, determinar a equação geral da cónica a partir do comprimento dos semi-eixos, do seu ângulo de rotação e da posição do centro. Para tal, definem-se A e B como sendo o comprimento dos semi-eixos maior e menor, respectivamente, $C = (C_x, C_y)$ o centro da cónica e ϕ o seu ângulo de rotação.

Partindo da equação geral da elipse, tem-se

$$\frac{x^2}{A^2} + \frac{y^2}{B^2} = 1 \Leftrightarrow \frac{B^2 x^2}{A^2 B^2} + \frac{A^2 y^2}{A^2 B^2} = 1 \Leftrightarrow B^2 x^2 + A^2 y^2 - A^2 B^2 = 0.$$

Passando para a forma matricial, obtém-se

$$\Omega' = \begin{bmatrix} B^2 & 0 & 0 \\ 0 & A^2 & 0 \\ 0 & 0 & -A^2 B^2 \end{bmatrix}.$$

Ω' é uma cónica (mais especificamente, uma elipse) centrada na origem e com os seus eixos alinhados com os eixos do referencial. Para obter a cónica Ω desejada, é necessário efectuar uma mudança de referencial, através de uma rotação R (induzida pelo facto da cónica estar rodada de um certo ângulo ϕ) e uma translação t (devida ao facto da cónica não estar centrada na origem do referencial). Para um ponto, tem-se que

$$x = Tx' \quad (2.9)$$

onde

$$T = \begin{bmatrix} R & t \\ 0^T & 1 \end{bmatrix} = \begin{bmatrix} \cos\phi & -\sin\phi & C_x \\ \sin\phi & \cos\phi & C_y \\ 0 & 0 & 1 \end{bmatrix}.$$

Da equação (2.9) vem que $x' = T^{-1}x$. Substituindo na equação (2.2), tem-se que

$$\Omega = (T^{-1})^T . \Omega' . T^{-1}$$

onde Ω é matriz de parametrização da cónica definida pelos parâmetros descritos anteriormente.

2.4 Determinação da cónica a partir de pontos da imagem

Como já foi dito anteriormente, uma cónica pode ser parametrizada por uma matriz simétrica, tal que

$$\Omega = \begin{bmatrix} a & b & d \\ b & c & e \\ d & e & f \end{bmatrix}.$$

Outra forma de parametrizar uma curva cónica é através do vector $\omega = (a, b, c, d, e, f)^t$. [3]

O algoritmo de *fitting* de cónicas determina a curva cónica que melhor se ajusta a um dado conjunto dos pontos. Para que se possa estimar uma cónica inequivocamente, são necessários, pelo menos, cinco pontos. Existem várias aproximações que podem ser feitas de modo a obter o *fit* da cónica. Neste caso, foi usado o método dos mínimos quadrados baseado em distâncias algébricas. [3]

Considere-se que se têm um conjunto de pontos distintos $p_i = (x_i, y_i)^t$ com $i = 1,...,M$ situados num determinado plano. A distância algébrica α_i entre um ponto p_i e uma cónica Ω é dada por $\alpha_i = \alpha(p_i) = ax_i^2 + 2bx_i y_i + cy_i^2 + 2dx_i + 2ey_i + f, i = 1,...,M$. [3]

Note-se que, no caso do ponto estar situado sobre a curva cónica, a distância algébrica será nula. A matriz A (definida em (2.10)) é chamada de matriz de design. [3]

$$A = \begin{bmatrix} x_1^2 & 2x_1y_1 & y_1^2 & 2x_1 & 2y_1 & 1 \\ x_2^2 & 2x_2y_2 & y_2^2 & 2x_2 & 2y_2 & 1 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ x_M^2 & 2x_My_M & y_M^2 & 2x_M & 2y_M & 1 \end{bmatrix} \qquad (2.10)$$

O vector de distâncias algébricas entre a cónica e todos os pontos é $(\alpha_1, \alpha_2, ..., \alpha_M) = A.\omega$. Se os pontos se situam sobre a curva cónica então, a matriz A tem *rank* 5 e ω está no espaço nulo da matriz. [3]

Suponha-se agora que o conjunto de pontos a partir do qual se pretende estimar a cónica tem mais do que 5 elementos. Geralmente, os pontos estão afectados por algum ruído e a matriz A é de *rank* 6. Se a matriz A for *full rank*, não existe espaço nulo e a única solução de $A.\omega = 0$ é $\omega = 0$. [3] Normalmente, os pontos são ajustados pela cónica que minimiza a soma dos quadrados das distâncias algébricas. [3] Ou seja, pretende-se encontrar o mínimo da função ϕ definida como sendo

$$\phi(\omega) = \sum_{i=1}^{M} \alpha_i^2 = \omega^t A^t A \omega.$$

A solução $\omega = 0$ é um mínimo global de ϕ. Para evitá-lo, deve-se restringir ω. Existem vários métodos propostos. O método utilizado foi o método mínimos quadrados normal. [3] Este método estima a cónica ω que minimiza ϕ sobre a restrição $\hat{\omega}^t \hat{\omega} = 1$. [3] A função objectivo está descrita na seguinte equação, onde a restrição é introduzida usando um multiplicador de Lagrange λ

$$\phi(\omega, \lambda) = \omega^t A^t A \omega + \lambda(\omega^t \omega - 1).$$

A cónica ω que minimiza ϕ sob estas condições é obtida resolvendo o sistema de valores próprios $A^t A \hat{\omega} = \lambda \hat{\omega}$ e é dada pelo vector próprio correspondente ao menor valor próprio da matriz $A^t A$. [3]

2.5 Método específico de *fit* de elipses

Na secção anterior foi feita a descrição do método para obter a cónica que melhor ajusta os pontos. Também já foi referido anteriormente que uma curva cónica não degenerada pode ser uma elipse, uma hipérbole ou uma parábola. No caso aqui descrito, o ajuste dos pontos mais adequado é através de uma elipse.

Seja $\omega = (a, b, c, d, e, f)^T$ a matriz de parametrização de uma curva cónica. É possível determinar qual a classe a que a cónica pertence (no caso de não ser degenerada). Assim, se:

$b^2 - ac < 0$, a cónica é uma elipse ou um círculo (note-se que um círculo é uma elipse com a particularidade dos eixos maior e menor terem a mesma dimensão);

$b^2 - ac = 0$, a cónica é uma parábola;

$b^2 - ac > 0$, a cónica é uma hipérbole.

Fitzgibbon e Fisher propuseram um método específico para o cálculo de elipses. [5] Considere-se a seguinte matriz

$$C = \begin{bmatrix} 0 & 0 & 2 & 0 & 0 & 0 \\ 0 & -1 & 0 & 0 & 0 & 0 \\ 2 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}.$$

Se uma cónica ω verificar $\omega^t C \omega = 1$, então ω deverá ser uma elipse ou um círculo. [5] O método de ajuste proposto estima a curva, minimizando a distância algébrica dos pontos sob a restrição $\omega^t C \omega = 1$. [5] A função objectivo resultante está descrita na seguinte equação, onde a restrição é introduzida usando um multiplicador de Lagrange λ

$$\phi(\omega, \lambda) = \omega^t A^t A \omega + \lambda(\omega^t C \omega - 1).$$

Pode provar-se que a elipse/círculo ω que minimiza ϕ é o vector próprio correspondente ao único valor próprio positivo do sistema de valores próprios generalizado $A^t A \omega = \lambda C \omega$. Note-se que, no caso de C ser substituído por $-C$, o método passa a ser específico para o cálculo de hipérboles. [5]

2.6 Detecção de pontos da fronteira na imagem

Na secção foi feita uma descrição de como fazer o ajuste dos pontos a uma curva cónica ou mais especificamente, a uma elipse (secção 2.5) a partir de um conjunto de pontos iniciais. Nesta secção irão ser descritos os principais métodos implementados, bem como as aproximações feitas.

2.6.1 Método baseado em máximos de gradientes

A seguinte figura representa uma imagem adquirida a partir do artroscópio.

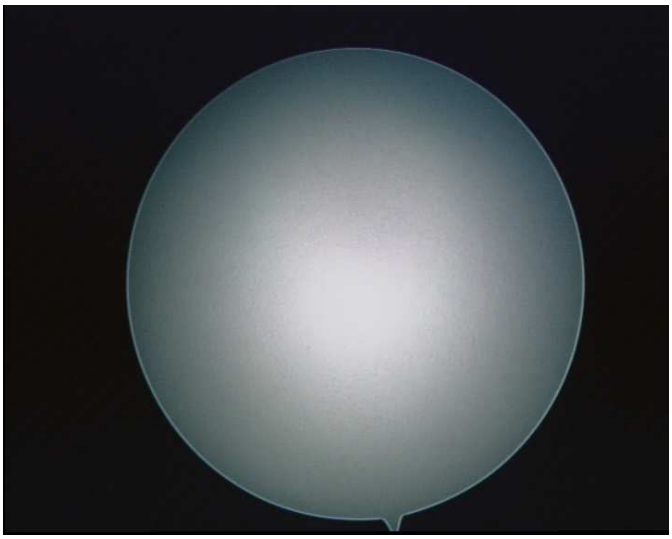

Figura 2.3: Imagem de uma folha branca adquirida pelo artroscópio e com a fonte de luz ligada.

A primeira aproximação feita para determinar os pontos da fronteira, foi percorrer toda a imagem e encontrar máximos de gradiente. Como se verifica na figura 2.3, estes máximos estão localizados sobre a região da fronteira entre o fundo e a zona útil da imagem (pois é onde existem maiores variações de cor). Assim, o procedimento é:

- Converter a imagem do formato RGB para uma imagem de níveis de cinzento;
- Para a primeira coluna da imagem, calcular as diferenças entre uma linha da imagem e a linha anterior até ser encontrada uma diferença entre duas linhas de pelo menos 9 níveis de cinzento;
- Encontrada esta diferença, o ponto correspondente é guardado e o procedimento é repetido até se percorrem todas as colunas da imagem;
- No caso de não ser encontrada uma diferença entre duas linhas de pelo menos 9 níveis de cinzento, não é armazenado nenhum ponto referente a essa coluna;
- Este procedimento é efectuado no sentido crescente do número de linhas e no sentido descendente de modo a apanhar os pontos de cada um dos lados.

Importa referir que o valor da diferença de 9 níveis foi obtido experimentalmente, através da análise de resultados para diferentes valores. É também importante perceber que apesar de parecer bastante uniforme, os pixeis do fundo da imagem podem ter variações elevadas do nível de cor de um pixel para o outro na vizinhança.

A seguinte figura mostra o resultado obtido pela aplicação deste método.

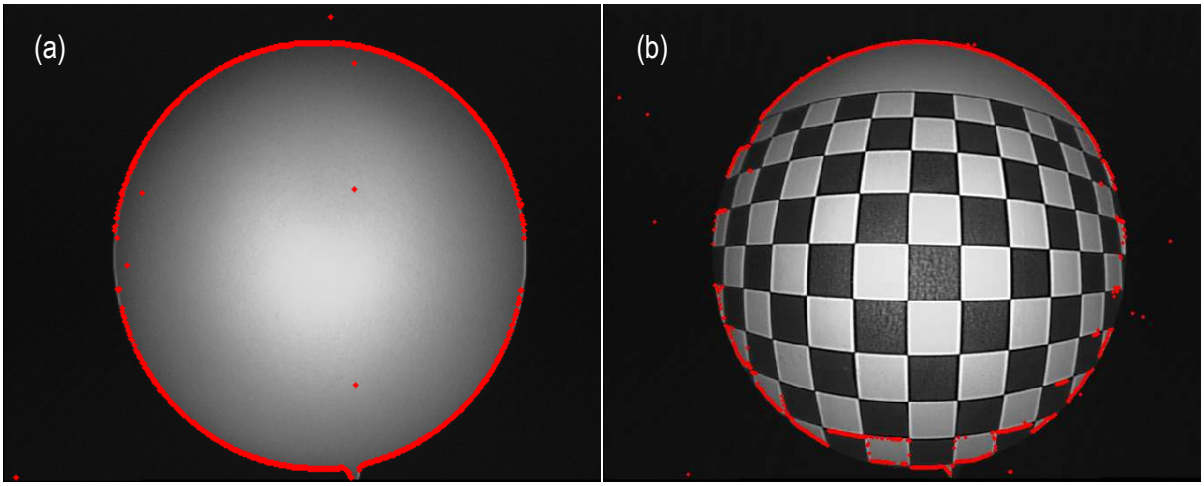

Figura 2.4: Pontos detectados (assinalados a vermelho) pelo método proposto **(a)** para uma superfície branca e **(b)** para a imagem de uma grelha.

Analisando a figura, conclui-se que o resultado está próximo do pretendido. Com excepção para uma pequena percentagem dos pontos, todos os pontos estão localizados sobre a zona da fronteira que se pretende detectar. Além disso, esses pontos *outliers* podem ser eliminados por um método de estimação robusta, por exemplo, aplicando-lhe um RANSAC (como irá ser explicado mais adiante). Para uma imagem típica de calibração, uma grelha de xadrez, o resultado obtido também foi satisfatório, como se verifica na figura 2.4(b). Também neste caso, se verificam alguns pontos situados fora da zona de fronteira que, no entanto, podem ser facilmente eliminados, utilizando um método de estimação robusta.

Apesar dos bons resultados obtidos, este método foi abandonado por ser um método lento, que não satisfaz a necessidade de se ter um processamento em tempo real (como se pretende).

2.6.2 *Threshold* adaptativo baseado no histograma

Um outro método utilizado baseou-se no cálculo de um *threshold* adaptativo baseado no histograma das imagens. Após o cálculo do valor do *threshold*, a imagem é convertida para o formato binário (preto e branco). É então aplicado um filtro mediana com uma máscara de 7x7 pixeis na imagem, de modo a obter uma imagem mais uniforme. O passo seguinte consiste na aplicação de uma operação morfológica à imagem: é feita uma erosão da imagem. Por último, subtrai-se a imagem obtida após a erosão, à imagem que se tinha antes de fazer a erosão. A figura 2.5 tem os resultados obtidos após cada uma das etapas referidas anteriormente.

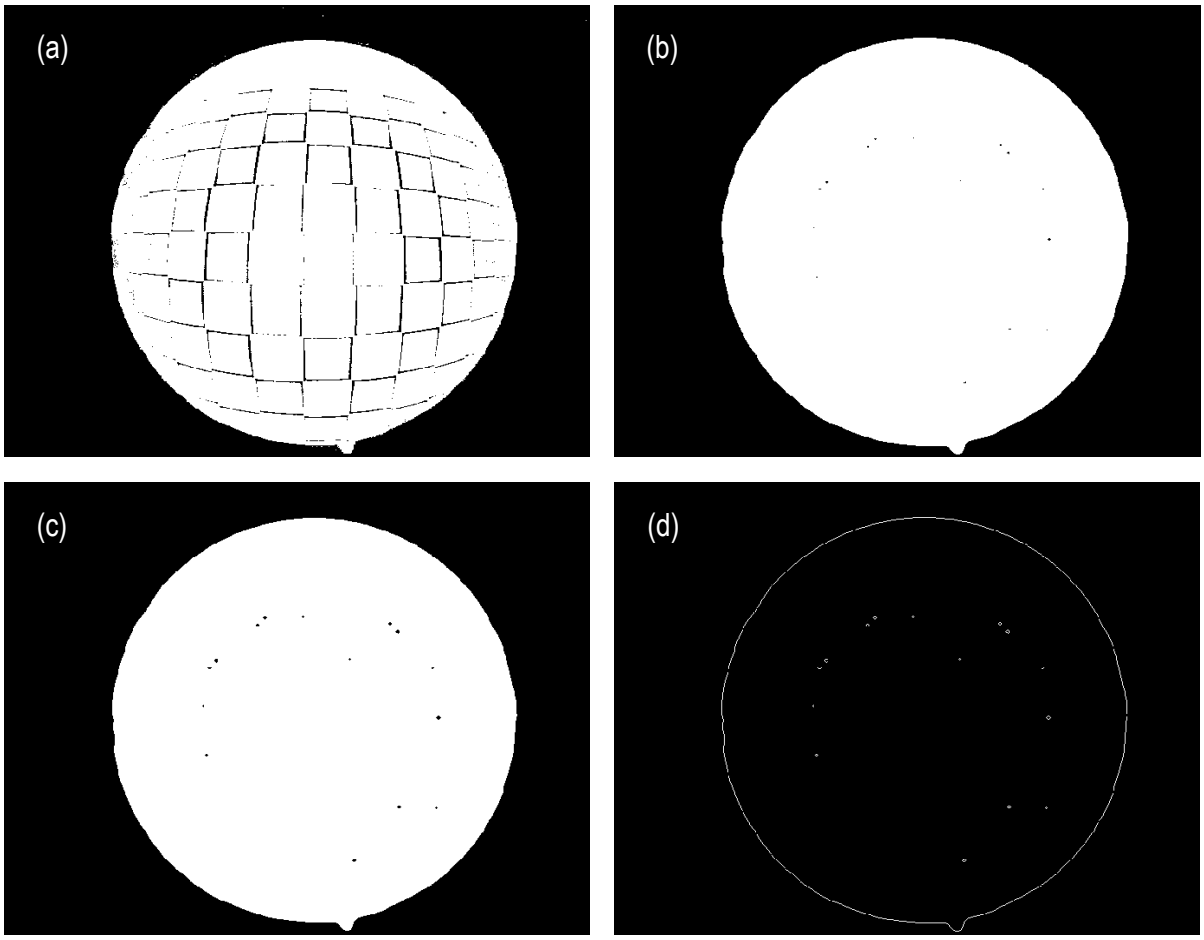

Figura 2.5: (a) Imagem obtida por binarização; **(b)** imagem obtida após filtragem da imagem (a); **(c)** imagem obtida após aplicação de erosão na imagem (b); **(d)** imagem obtida após subtracção da imagem (c) à imagem (b).

A seguinte figura mostra o resultado obtido pela aplicação deste método.

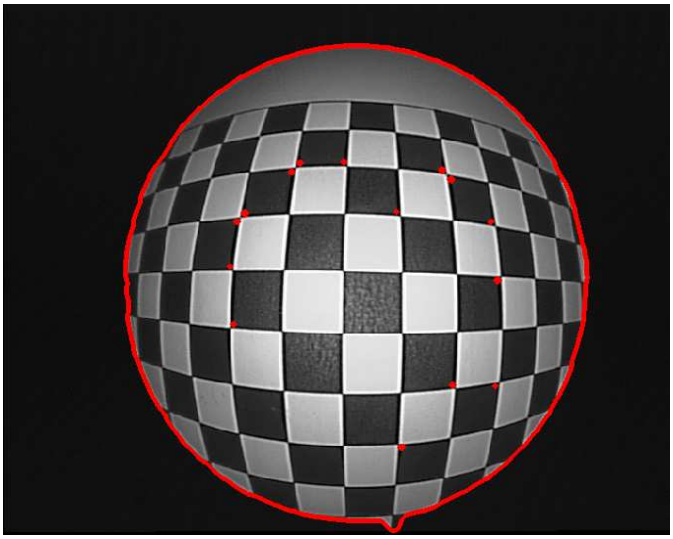

Figura 2.6: Pontos detectados (assinalados a vermelho) pelo método proposto.

Como se pode ver, o método tem um resultado satisfatório no que diz respeito aos pontos detectados. No capítulo seguinte, vão ser explicados métodos que podem tornar este método mais robusto e estável. Por agora, pode-se concluir que este método é um bom ponto de partida para ser utilizado na detecção dos pontos da fronteira. Além disso, é um método rápido, que pode ser executado em tempo real. É ainda de salientar que não se consegue obter um método que seja simples e robusto, como o pretendido, mas ao mesmo tempo, rápido.

2.7 Detecção da marca da lente e cálculo da sua rotação

As lentes artroscópicas têm uma marca, como se pode ver na figura 2.7 (a marca está assinalada a verde). Essa marca serve como uma referência de posição aos médicos. Pode ser feito algum processamento com vista a fazer a sua detecção automaticamente. A sua detecção é bastante importante pois permite saber a rotação relativa entre a câmara e a lente e, saber qual foi o movimento de rotação entre *frames*.

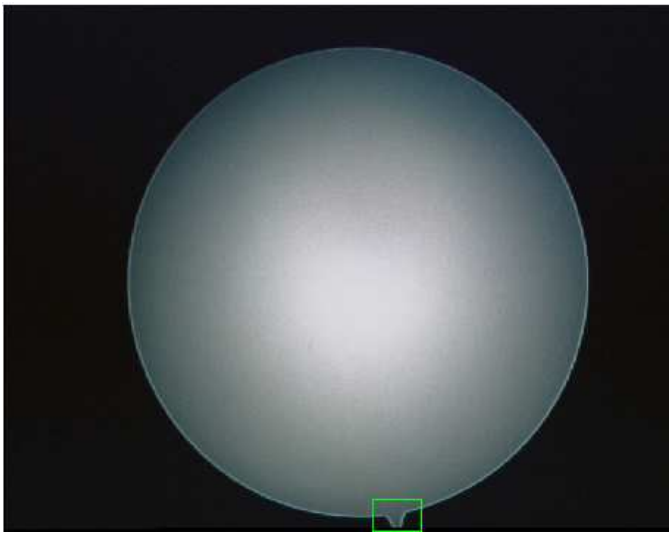

Figura 2.7: Imagem adquirida pelo artroscópio. A verde está assinalada a marca da lente.

O método utilizado para detectar a marca da lente, consiste em determinar o ponto para o qual a distância algébrica entre o ponto e a curva cónica determinada é máxima. Ou seja, supondo que foram detectados N pontos na fronteira, e que a cónica estimada foi $\omega = (a,b,c,d,e,f)$, a distância algébrica d para um determinado ponto i é dada por

$$d_i = ax_i^2 + 2bx_i y_i + cy_i^2 + 2dx_i + 2ey_i + f, i = 1,...,N$$

e a marca da lente corresponde ao ponto para o qual d é máximo.

Após ter calculado o ponto correspondente à marca da lente e sabendo o centro da cónica, pode determinar-se o declive da recta definida por estes dois pontos e consequentemente, o ângulo de rotação φ da marca. Ou seja, se o centro C da cónica tiver coordenadas $C=(C_x, C_y)$ e a marca M da lente tiver coordenadas $M=(M_x, M_y)$, é possível definir

$$m = \frac{C_y - M_y}{C_x - M_x}$$

como sendo o declive da recta definida pelo centro da cónica e a marca da lente e escrever o ângulo de rotação da lente

$$\varphi = \tan^{-1}(m) = \tan^{-1}\left(\frac{C_y - M_y}{C_x - M_x}\right).$$

Capítulo 3: Implementação do Algoritmo de Segmentação

No capítulo anterior, na secção 2.6, foi introduzido um método baseado no cálculo de um *threshold* adaptativo baseado no histograma para detectar os pontos da fronteira utilizados na segmentação da imagem. Apesar de ser rápido, este método pode ser pouco robusto. Neste capítulo, serão implementadas melhorias a este método, nomeadamente, introduzindo um método de estimação robusta (RANSAC) e um filtro de Kalman. Será também feita uma analise ao efeito do white-set na imagem artroscópica e aos histogramas de diferentes imagens, de modo a estudar a melhor forma de calcular o valor de *threshold* a utilizar. Descrever-se-á o artroscópio e o hardware adicional utilizado. Na parte final deste capítulo, irão ser apresentados alguns resultados para analisar o desempenho deste algoritmo de segmentação.

3.1 O artroscópio

Um sistema de artroscopia é composto por quatro componentes principais (mostradas na figura 3.1): o artroscópio, a câmara, a unidade de controlo e a fonte de luz. Depois, a imagem captada pode ser vista num monitor ou ser enviada para um computador para se poder fazer o tratamento.

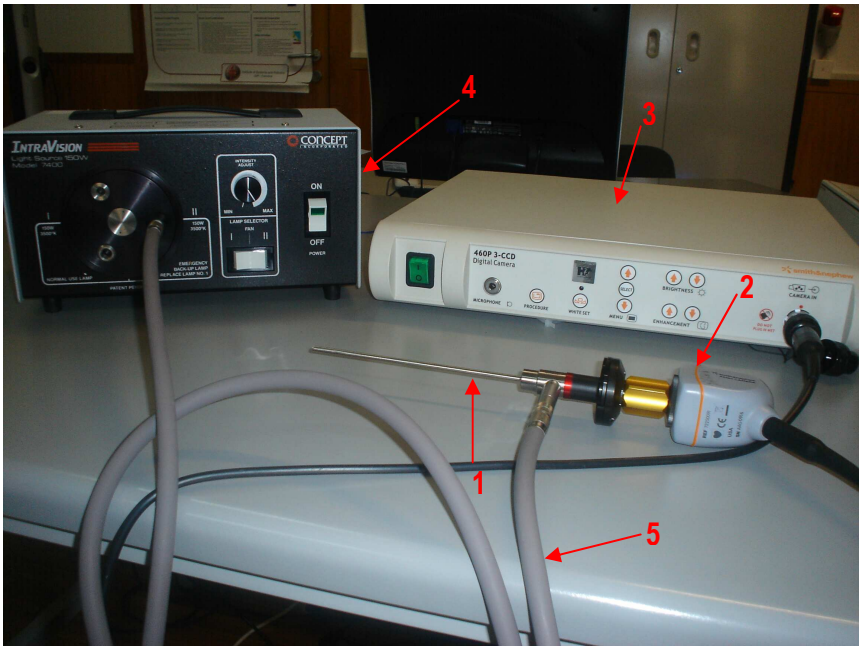

Figura 3.1: Componentes do sistema de artroscopia utilizado: 1 – artroscópio; 2 – câmara de 3 CCD; 3 – unidade de controlo; 4 – fonte de luz; 5 – fibra óptica (guia de luz).

O artroscópio utilizado foi o modelo SN QH 458406 comercializado pela Smith & Nephew. Este tem um diâmetro de 4 mm, com uma inclinação de 30º na ponta e capta um ângulo de visão de aproximadamente 180º. A câmara utilizada foi o modelo 460H da Smith & Nephew. De salientar o facto

desta câmara ter três CCD em vez de um. Tal situação leva a que esta câmara tenha maior sensibilidade, já que proporciona uma melhor separação da cor e tem uma maior eficiência quântica. Na tabela 3.1 estão descritas algumas propriedades desta câmara. A unidade de controlo utilizada foi o modelo 460P 3-CCD Digital Camera da Smith & Nephew. Esta componente do sistema é a responsável pela formação da imagem captada pela câmara e permite controlar alguns parâmetros da câmara com vista à aquisição de melhor imagens. Esta unidade tem a vantagem de exportar o vídeo artroscópico em vários formatos: analógico, digital e composto. Por último, a fonte de luz utilizada foi a Concept Model 7400, com intensidade ajustável e uma potência máxima de 150 W.

Smith & Nephew 460H	
Tamanho da imagem (pixel)	576 x 720
Área usada (%)	46
Saída de vídeo	DV-25

Tabela 3.1: Especificações técnicas da câmara Smith & Nephew 460H.

3.2 Análise ao comportamento dos histogramas

Nesta secção irá ser feita uma análise ao efeito do white-set nas imagens do artroscópio. Será também feita uma análise aos histogramas das imagens utilizadas para a segmentação da zona útil da imagem.

As imagens aqui utilizadas foram adquiridas em formato digital DV25 e gravadas em formato .tif de modo a ter-se um formato sem compressão e sem perdas.

Ao longo desta secção vai fazer-se analise a diferentes histogramas, quer de cada uma das componentes R, G, e B de imagens do tipo RGB, quer de imagens em escala de cinzentos. Importa referir que as imagens foram convertidas do formato RGB para o formato em escala de cinzentos utilizando uma média ponderada, dada pela seguinte equação

$$Y = 0.2990 * R + 0.5870 * G + 0.1140 * B.$$

As ponderações atribuídas a cada uma das componentes têm a ver com a sensibilidade do olho a cada uma das cores. Ou seja, o olho é mais sensível ao verde logo este tem um maior peso. Como o olho é menos sensível ao azul, a componente azul tem menor influência.

3.2.1 O efeito do white-set

Sempre que o artroscópio é ligado, automaticamente é pedido ao utilizador que seja efectuado o white-set. Para efectuar o white-set, é necessário envolver a ponta do artroscópio por um pano branco

e focá-lo, com a fonte de luz a irradiar na intensidade máxima. Embora não tenham sido fornecidas informações acerca da finalidade deste procedimento pelo fabricante, foi feita uma análise aos histogramas de imagens em diferentes contextos obtidas antes e depois de ser efectuado o white-set, para tentar perceber um pouco melhor qual a sua influencia nas imagens.

O primeiro conjunto de imagens consistiu em 3 imagens de uma superfície branca (uma folha de papel): uma adquirida com a fonte de luz ligada e antes de ser feito o white-set, outra adquirida com a fonte de luz ligada e depois de ser feito o white-set e uma ultima adquirida com fonte de luz desligada e depois de ser feito o white-set. As imagens adquiridas estão mostradas na figura 3.2.

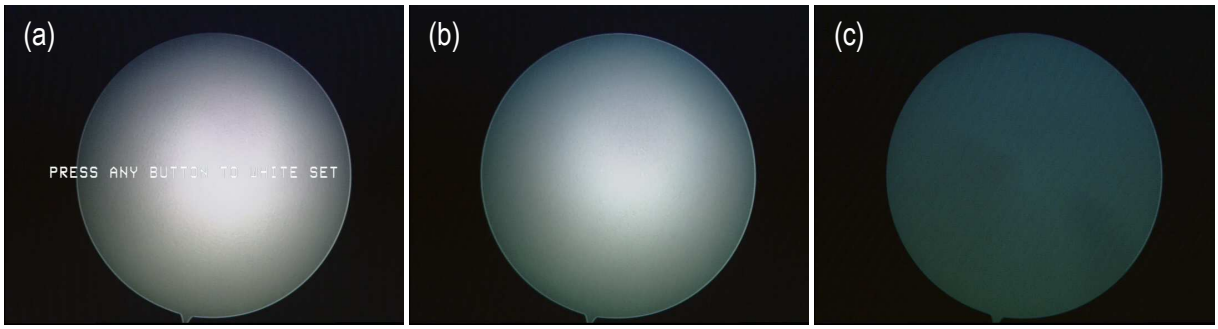

Figura 3.2: Imagem de uma superfície branca obtida **(a)** antes de efectuar o white-set e com fonte de luz, **(b)** depois de efectuar o white-set com e com fonte de luz, e **(c)** depois de efectuar o white-set e sem fonte de luz.

Importa referir que as imagens não são exactamente iguais. Embora neste conjunto de imagens não se note, é impossível de conseguir adquirir imagens exactamente iguais antes de fazer o white-set e depois de fazer o white-set, como seria idealmente desejado.

Os histogramas correspondentes estão mostrados na figura 3.3. Em cada um dos gráficos, o histograma representado a vermelho corresponde à imagem da figura 3.2(a), o histograma representado a verde corresponde à imagem da figura 3.2(b) e o histograma representado a azul corresponde à imagem da figura 3.2(c). Analisando o histograma a azul, este pode parecer diferente dos outros histogramas já que tem um segundo máximo. No entanto, este histograma corresponde à imagem adquirida sem a fonte de luz e, como se pode ver na figura 3.2(c), esta figura apresenta, sobretudo dois conjuntos de pixeis: um correspondente ao fundo, outro correspondente à zona central da imagem, daí que haja o aparecimento de um segundo pico. Nos outros histogramas não se verifica tal situação, já que a zona central da imagem apresenta maiores variações de cor. Comparando os histogramas das imagens antes e depois do white-set adquiridas com a fonte de luz percebe-se que as grandes variações do histograma estão nos extremos da escala, ou seja, nos níveis mais escuros e nos níveis mais claros. Nos níveis intermédios, as variações são pequenas.

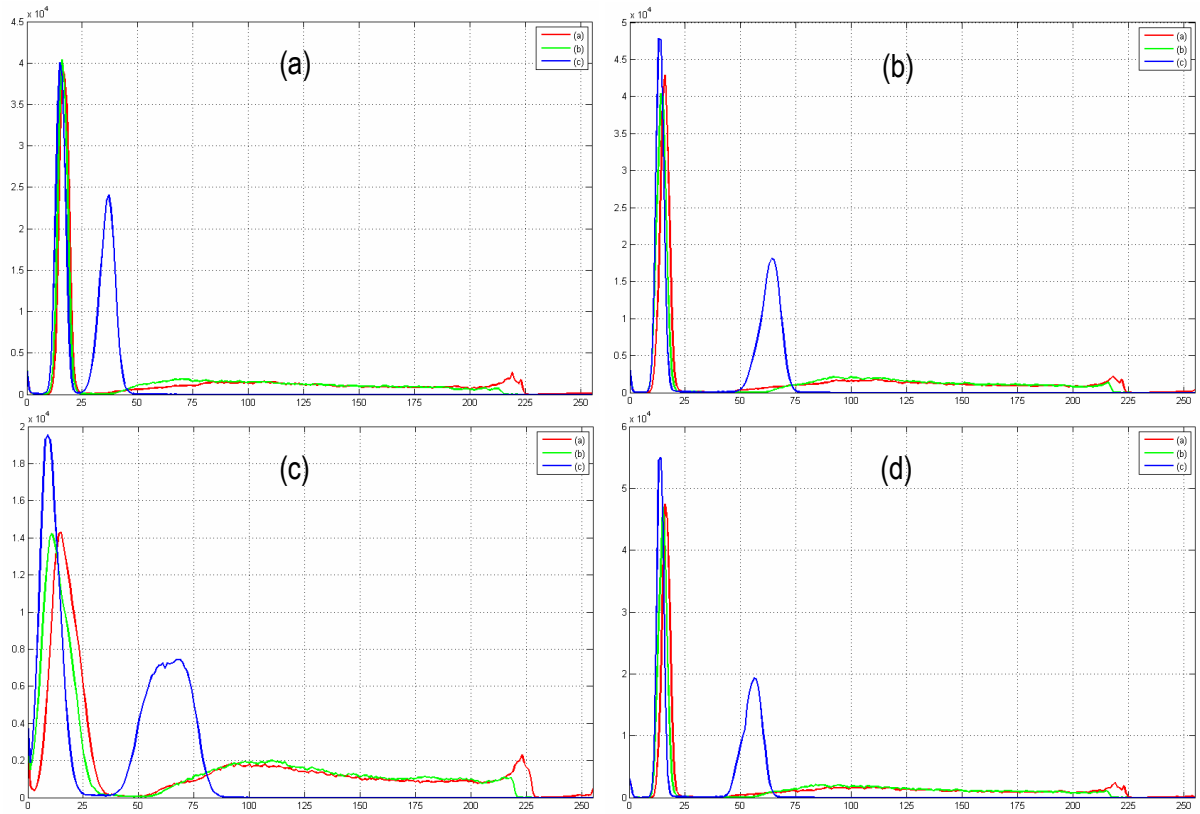

Figura 3.3: Histogramas correspondentes às imagens da figura 3.2 **(a)** histograma da componente R, **(b)** histograma da componente G, **(c)** histograma da componente B, e **(d)** histograma da imagem em escala de cinzentos.

O segundo conjunto de imagens consistiu em 3 imagens de uma grelha de xadrez: uma adquirida com a fonte de luz ligada e antes de ser feito o white-set, outra adquirida com a fonte de luz ligada e depois de ser feito o white-set e uma ultima adquirida com fonte de luz desligada e depois de ser feito o white-set. As imagens adquiridas estão mostradas na figura 3.4.

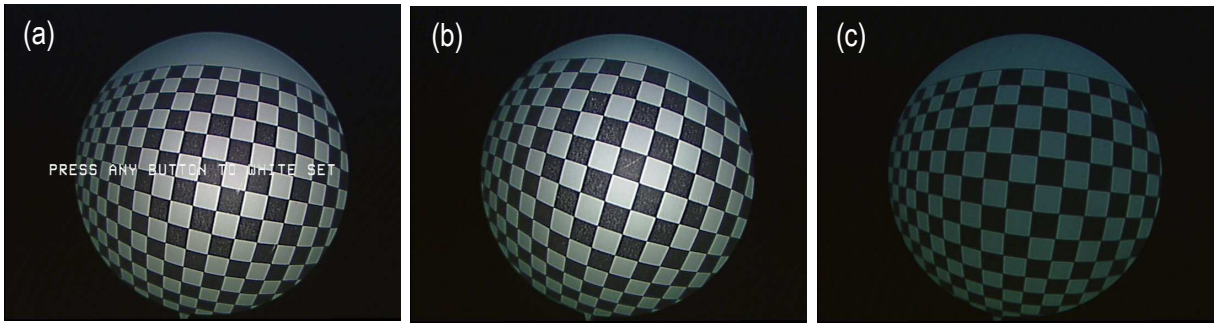

Figura 3.4: Imagem de uma grelha obtida **(a)** antes de efectuar o white-set e com fonte de luz, **(b)** depois de efectuar o white-set com e com fonte de luz, e **(c)** depois de efectuar o white-set e sem fonte de luz.

Os histogramas correspondentes estão mostrados na figura 3.5. Em cada um dos gráficos, o histograma representado a vermelho corresponde à imagem da figura 3.4(a), o histograma representado a verde corresponde à imagem da figura 3.4(b) e o histograma representado a azul

corresponde à imagem da figura 3.4(c). Comparando os histogramas, verifica-se que o histograma referente à imagem da figura 3.4(c) tem dois picos máximos mais salientes que nos restantes histogramas. Tal situação se prende com os pixeis desta imagem estarem mais agrupados em duas zonas de níveis: um que engloba os pixeis de fundo e os pixeis correspondentes às quadriculas pretas da zona central da imagem e outro referente aos pixeis correspondentes às quadrículas brancas da zona central da imagem. No caso da imagens com fonte de luz, este pico correspondentes às quadriculas brancas não se verifica, já que o branco não é tão uniforme. Contudo, os níveis de preto correspondentes às quadriculas retas da zona central estão separados dos do fundo, já que a fonte de luz os torna um pouco mais claros e por conseguinte, surge um segundo pico mais ténue correspondente aos pixeis escuros da zona central da imagem. Comparando os histogramas das imagens antes e depois do white-set adquiridas com a fonte de luz, mais uma vez se verifica que os histogramas não diferem muito. As maiores diferenças voltam a estar, mais uma vez, nos níveis extremos da escala.

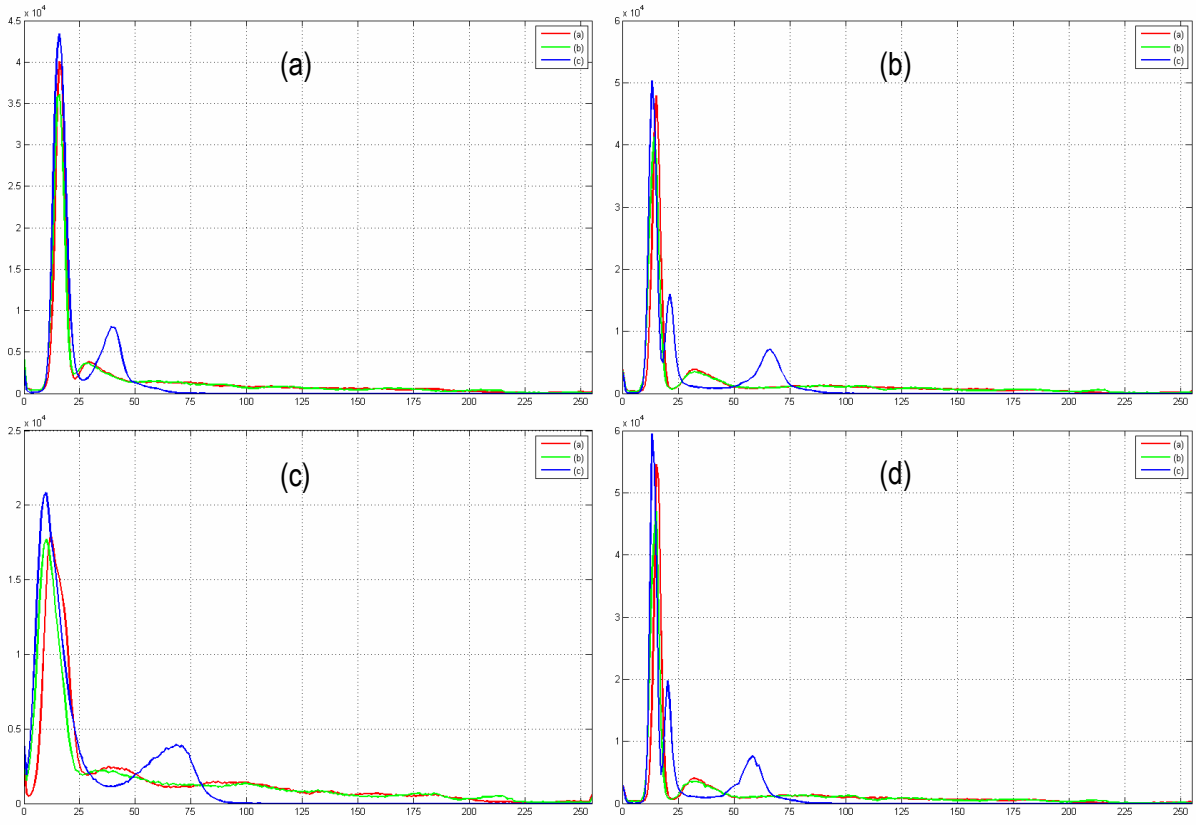

Figura 3.5: Histogramas correspondentes às imagens da figura 3.4 **(a)** histograma da componente R, **(b)** histograma da componente G, **(c)** histograma da componente B, e **(d)** histograma da imagem em escala de cinzentos.

O último conjunto de imagens consistiu em 2 imagens (adquiridas antes e depois do white-set) tiradas ao interior da mão fechada de modo a simular uma cena real. Note-se que até aqui tinha sido sempre adquirida uma imagem com luz natural (sem fonte de luz ligada). Neste caso, como não existe

luz natural no interior da mão, não foi adquirida essa imagem. Foi também adquirida uma terceira imagem de uma cena do mundo depois de se realizar o white-set. No entanto, os histogramas não têm qualquer ligação, pelo que foram feitos num gráfico à parte. As imagens adquiridas estão mostradas na figura 3.6.

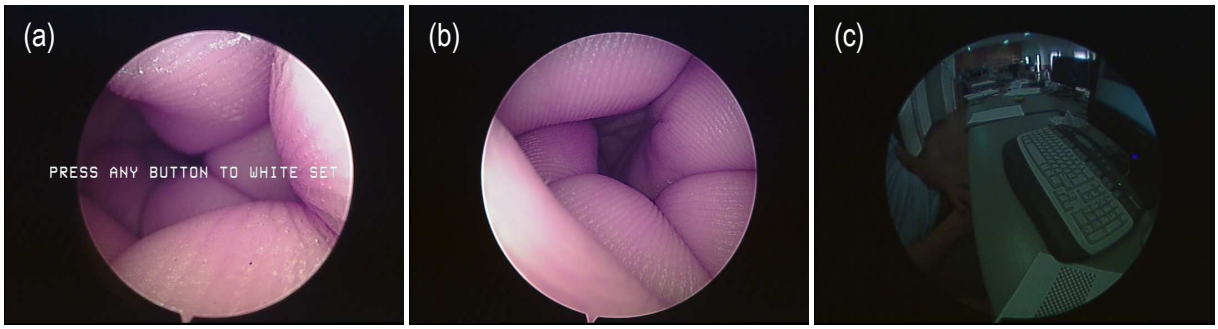

Figura 3.6: Simulação de uma imagem real obtida com fonte de luz **(a)** antes e **(b)** depois de efectuar o white-set. **(c)** Imagem de uma cena do mundo obtida após realizar o white-set.

Os histogramas correspondentes às figuras 3.6(a) e 3.6(b) estão mostrados na seguinte figura.

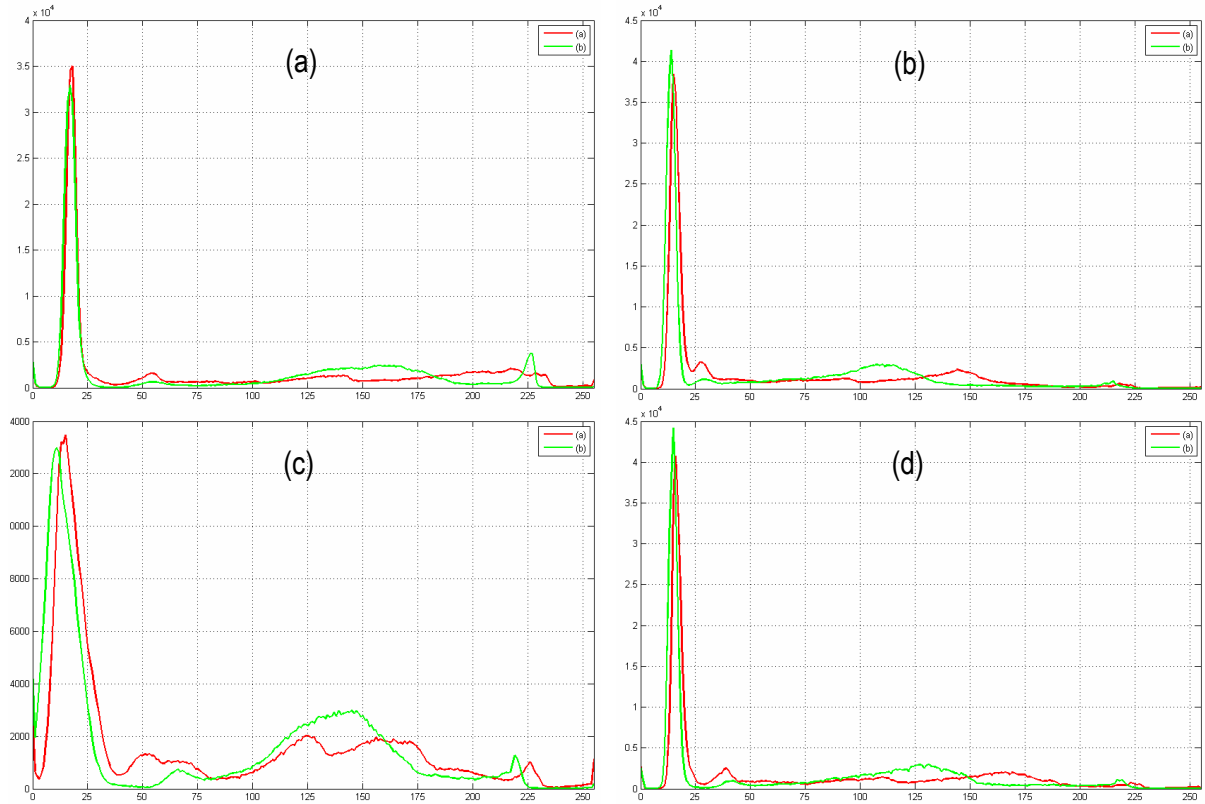

Figura 3.7: Histogramas correspondentes às imagens (a) e (b) da figura 3.6 **(a)** histograma da componente R, **(b)** histograma da componente G, **(c)** histograma da componente B, e **(d)** histograma da imagem em escala de cinzentos.

Em cada um dos gráficos, o histograma representado a vermelho corresponde à imagem da figura 3.6(a) e o histograma representado a verde corresponde à imagem da figura 3.6(b). Comparando

os histogramas das duas imagens verifica-se uma maior variação destes, quando comparada com a variação existente nos casos anteriores. No entanto, por muito que se tente adquirir imagens iguais antes e depois do white-set, na prática é impossível consegui-lo. Olhando para as figuras 3.6(a) e 3.6(b), verifica-se que as imagens são um pouco diferentes, o que pode levar às diferenças dos histogramas. Assim, percebe-se que estes histogramas não são muito conclusivos.

Por último, vai ser apresentado o histograma da imagem da figura 3.6(c). Este surge apenas para analisar a forma do histograma de uma imagem de uma cena do mundo.

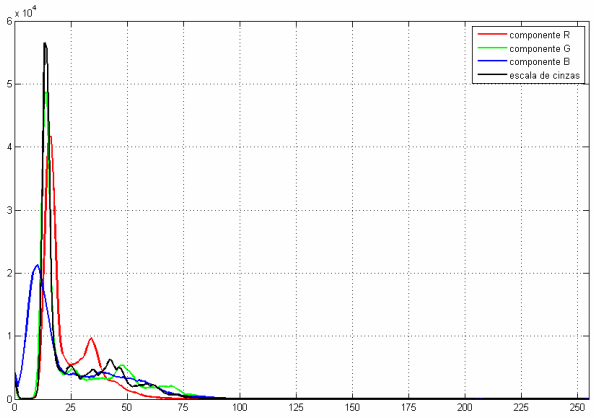

Figura 3.8: Histogramas correspondentes à imagem da figura 3.6(c).

Os gráficos seguintes mostram os ganhos que o white-set introduziu nos histogramas das imagens referentes à grelha e à da superfície branca.

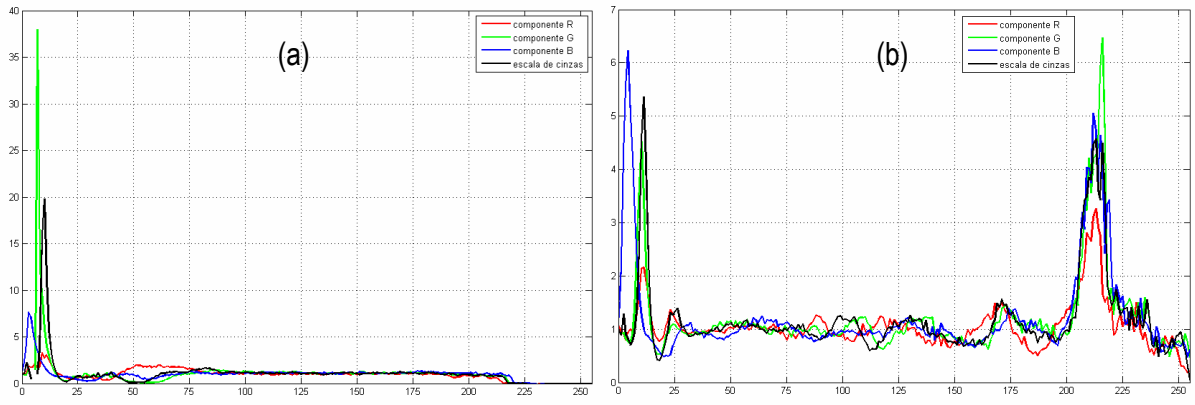

Figura 3.9: Gráficos de ganho impostos pelo white-set numa imagem **(a)** de uma superfície branca e **(b)** de uma grelha de xadrez.

Analisando os gráficos de ganho da figura 3.9, verifica-se que o white-set introduz um ganho adaptativo às imagens. Daí o facto dos gráficos de ganho diferirem tanto de um par de imagens para o outro. Por outro lado, o ganho é maior nos níveis das extremidades da escala. Conclui-se portanto que o white-set tem como objectivo introduzir um ganho adaptativo nas imagens de modo a conseguir-se

ter uma melhor definição de cor e, por conseguinte, um maior contraste. Pode dizer-se que funciona um pouco como um equalizador de histogramas.

3.2.2 Cálculo do valor de *threshold*

Como já foi referido anteriormente, o método de segmentação implementado baseia-se no cálculo de um *threshold* adaptativo baseado no histograma da imagem. Após a análise dos histogramas das imagens de uma grelha adquiridas com fonte de luz, como os que estão apresentados na subsecção anterior, verifica-se que existe um padrão nos histogramas que pode ser utilizado como forma de cálculo do valor de corte pretendido. Na figura 3.10 está representando um histograma típico de uma imagem de uma grelha.

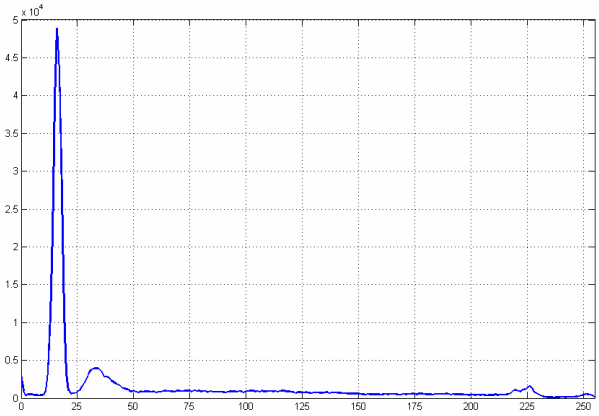

Figura 3.10: Histograma típico de uma imagem de uma grelha.

Como se verifica, existem três zonas de principais onde os pixeis estão distribuídos: o primeiro pico corresponde aos pixeis do fundo da imagem, o segundo corresponde aos pixeis das quadrículas pretas da zona útil da imagem (zona mais central da imagem) e depois, existe uma terceira zona nos níveis mais claros que corresponde às quadrículas brancas da zona útil da imagem. Assim, o valor de *threshold* é calculado com base na posição do vale entre os dois primeiros picos do histograma. Por exemplo, no caso do histograma da figura 3.10, o valor de corte iria ser no nível 23. O procedimento para determinar o valor de corte é:

- Calcular o histograma da imagem;
- Determinar a posição do pico máximo do histograma;
- Determinar a posição do primeiro nível em que há variação da monotonia do histograma, ou seja, passa de decrescente a crescente.

Uma conclusão a que se chegou durante a realização deste trabalho foi o facto de poderem ser aqui utilizadas duas formas de resolver este problema: ou utilizando o histograma da componente

verde da imagem em formato RGB, ou proceder-se à conversão da imagem para a escala de cinzentos e a partir do seu histograma, calcula-se o valor de *threshold*, devido a serem mais estáveis estes histogramas quando comparados com o histogramas das componentes vermelha e azul. De seguida vão ser apresentados alguns resultados de uns estudo acerca de qual o melhor histograma para utilizar no cálculo deste valor. As imagens utilizadas serão as mesmas que foram usadas na subsecção anterior.

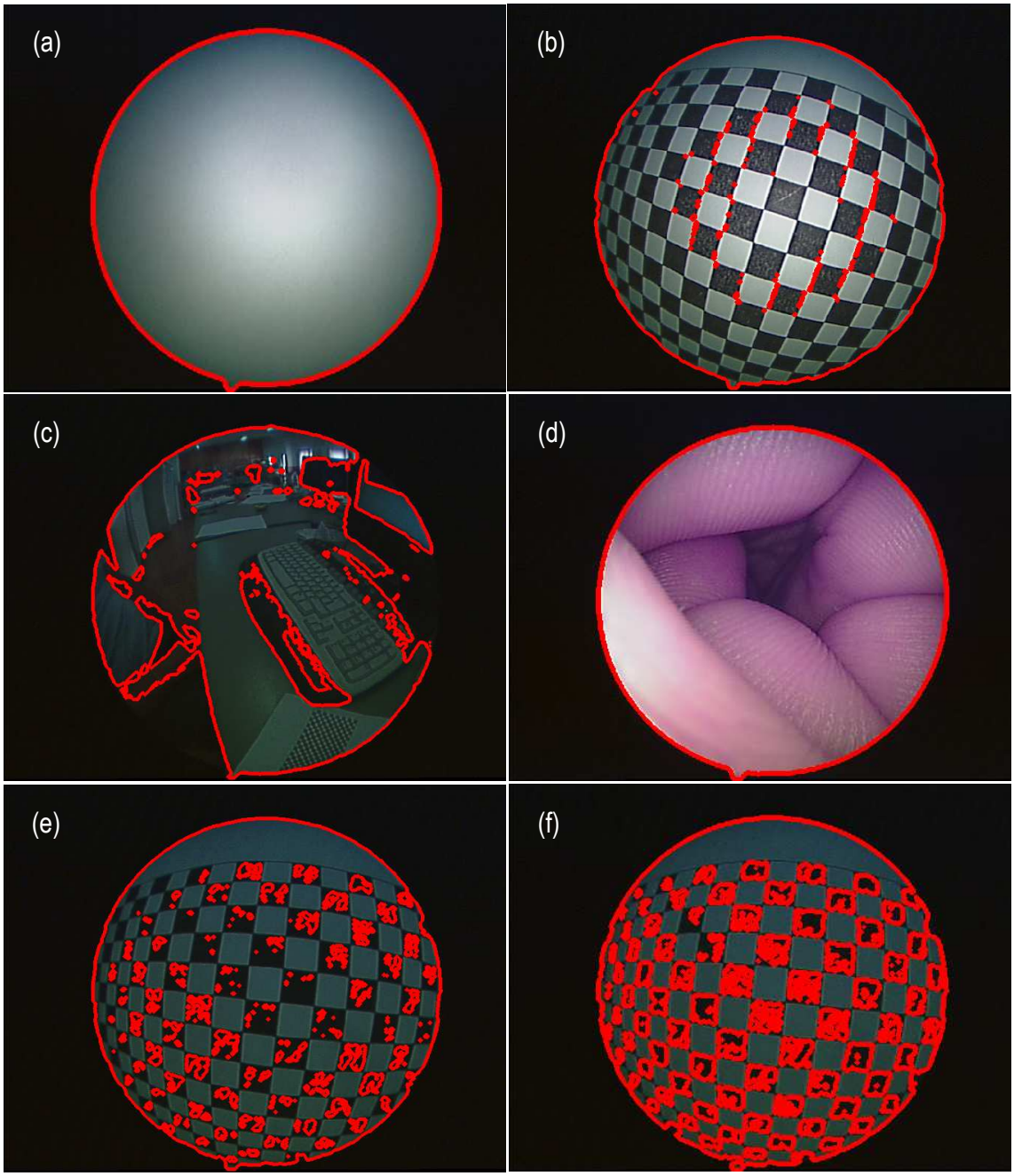

Figura 3.11: Resultados da detecção de pontos obtidos para **(a)** imagem de superfície branca, **(b)** imagem de uma grelha, **(c)** imagem de uma cena do mundo, **(d)** simulação de imagem real, **(e)** imagem de grelha obtida sem fonte de luz, onde o valor de corte foi calculado com base no histograma da imagem em escala de cinzentos, e **(f)** imagem de grelha obtida sem fonte de luz, onde o valor de corte foi calculado com base no histograma da componente verde da imagem RGB.

Nas figuras 3.11(a), 3.11(b), 3.11(c) e 3.11(d), os resultados obtidos foram os mesmos utilizando ambos os histogramas. Como se verifica, os resultados obtidos são bons, quer para a imagem da superfície branca, quer para a imagem real, já que os pontos detectados estão na zona da fronteira. Para a imagem da grelha, o resultado obtido também é razoável. Para a cena do mundo, os resultados obtidos são um pouco mais irregulares, já que o histograma da imagem não segue o mesmo comportamento padrão que nas imagens de uma grelha. Contudo, esta aplicação destina-se a servir de suporte à calibração do artroscópio, pelo que as imagens deste tipo (cena do mundo) são pouco importantes desse ponto de vista. A única divergência entre a utilização de um ou outro histograma encontra-se no caso das imagens serem adquiridas sem a fonte de luz, ou seja, apenas com a luz natural. Como se verifica nas figuras 3.11(e) e 3.11(f), os resultados obtidos utilizando o histograma da imagem após ser convertida para uma imagem em escala de cinzentos, apesar de serem maus, são melhores que os resultados obtidos pela utilização do histograma da componente verde da imagem RGB. No entanto, os resultados obtidos na ausência da fonte de luz são maus e deve-se evitar situações de iluminação reduzida.

Em suma, pode-se dizer que ambos os histogramas produzem resultados satisfatórios em imagens reais e em imagens de grelhas com iluminação.

3.3 Estimação robusta: RANSAC

Como já foi referido, o método de segmentação da imagem deve ser um método estável e robusto. Na secção anterior foi feita uma análise ao melhor histograma a utilizar para calcular o valor de *threshold*. Contudo, para uma imagem típica de calibração, uma grelha de um xadrez, o método pode detectar alguns pontos que não pertençam à zona da fronteira. Para robustecer mais o algoritmo de detecção de pontos da fronteira, introduziu-se um método de estimação robusta: o método RANSAC.

RANSAC é a abreviatura de "RANdom SAmple Concensus" e foi implementado pela primeira vez em 1981 por Fischler e Bolles. Trata-se de um método iterativo para determinar os parâmetros do modelo matemático que melhor se ajusta a determinado conjunto de dados, o qual pode conter pontos bastante inadequados ao modelo (designados de *outliers*). O aparecimento destes *outliers* pode dever-se a vários factores, como por exemplo, a existência de ruído ou de erros de medição. [6]

Um exemplo simples e que ilustra esta situação, é o modelo de aproximação a uma recta. Olhando para a figura 3.12, verifica-se que o conjunto de dados (à esquerda) tem pontos *inliers* (pontos que podem ser aproximados pelos parâmetros do modelo) e *outliers* (que, como já foi dito, são pontos inadequados ao modelo). Assim, se for utilizado, por exemplo, o método dos mínimos quadrados, a estimação é feita a partir de todos os pontos. Ou seja, os *outliers* têm influência no modelo estimado, o

que pode levar a más estimações. No entanto, utilizando o algoritmo de RANSAC, o modelo pode ser estimado utilizando apenas *inliers*, o que leva a melhores estimações.

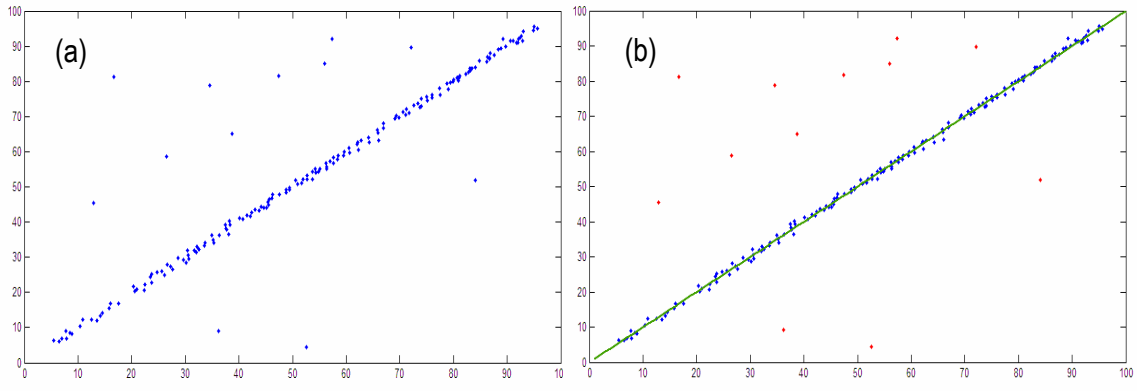

Figura 3.12: (a) Conjunto de pontos com a presença de alguns *outliers*. **(b)** Ajuste do conjunto de pontos a uma recta ignorando os pontos *outliers* (a vermelho).

O algoritmo de RANSAC tem uma estrutura simples mas é bastante poderoso. O primeiro passo consiste na extracção aleatória de um subconjunto de pontos a partir do conjunto de dados. O número mínimo de pontos a incluir neste subconjunto deve ser o necessário para estimar o modelo (que neste caso da cónica são 5 pontos). Com este subconjunto é estimado um modelo hipótese, e todos os pontos do conjunto de dados, são classificados como *inliers* ou *outliers* relativamente a este modelo. Este procedimento é repetido. Após os pontos terem sido classificados, no caso do modelo actual ter maior número de *inliers* que o até então melhor modelo, o modelo actual passa a ser o melhor modelo. No caso do número de *inliers* ser menor, o melhor modelo mantém-se inalterado, e o RANSAC passa à iteração seguinte. O procedimento é repetido até que se obtenha um modelo com uma boa percentagem de *inliers* ou até que se atinja o limite de tentativas imposto pelo utilizador. Após ser aplicado este algoritmo, é feito o ajuste do modelo utilizando todos os pontos classificados como *inliers* relativamente ao melhor modelo obtido pelo algoritmo. [6]

O valor de *threshold* para determinar se um ponto é *inlier* ao modelo estimado (parâmetro t) é ajustado experimentalmente, com base na aplicação que se pretende e no conjunto de dados. Já o número máximo de iterações permitidas pelo modelo pode ser determinado a partir de um resultado teórico.

Seja p a probabilidade de serem escolhidos apenas *inliers* para serem utilizados na estimação do modelo que vai ser classificado. Seja w a probabilidade de se escolher um ponto *inlier* de cada vez que é escolhido um único ponto e que é

$$w = \frac{n^{o}\ de\ inliers}{n^{o}\ total\ de\ pontos}.$$

Normalmente w não é conhecido, mas pode ser-lhe atribuído um valor aproximado. Admitindo que os n pontos necessários para definir o modelo são escolhidos independentemente, w^n é a probabilidade de todos os pontos escolhidos serem *inliers* e, por outro lado, $1-w^n$ é a probabilidade de pelo menos um dos pontos não ser *inlier*. $(1-w^n)^k$ é a probabilidade do algoritmo não escolher um conjunto de n pontos em que todos os pontos são inliers e deve ser igual a $1-p$. Então,

$$1-p = (1-w^n)^k$$

O método de RANSAC é muito útil já que pode fazer boas estimações, mesmo quando estão presentes pontos *outliers* ao modelo. No entanto, quando é limitado na duração de tempo, podem ser obtidas soluções que não são as melhores. Outro problema do RANSAC é o facto de um determinado conjunto de dados poder ser aproximado por mais do que um modelo, o que lhe poderá a causar alguma dificuldade na tomada de decisão sobre o modelo que deve utilizar. [6]

Na secção 2.6.2 foi apresentado um método para determinar os pontos da fronteira baseado no histograma das imagens. A figura 3.13(a) mostra o resultado obtido com este método. Como se pode verificar, existem alguns pontos detectados que não estão na zona de fronteira. Ou seja, são pontos *outliers* ao modelo que aqui se pretende estimar.

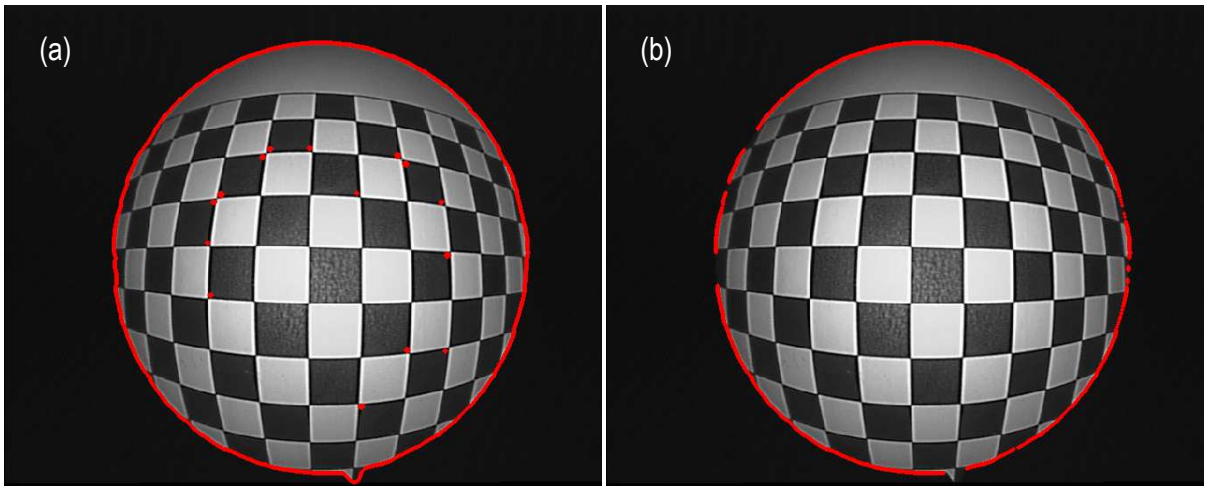

Figura 3.13: (a) Pontos detectados (assinalados a vermelho) pelo método proposto em 2.6.2. **(b)** Pontos detectados (assinalados a vermelho) após ter sido aplicado o RANSAC aos pontos estimados pelo método descrito em 2.6.2.

Na figura 3.13(b), está mostrado o resultado obtido depois de aplicar a estimação de RANSAC aos pontos do conjunto de dados. Como se pode, todos os pontos obtidos após a aplicação do método

de RANSAC estão na zona de fronteira. A estimação do modelo de ajuste a partir dos pontos da figura 3.13(b) será melhor já que não tem a presença de pontos *outliers*.

3.4 Filtro de Kalman

Após a aplicação do método de RANSAC, o algoritmo de segmentação tornou-se mais robusto. No entanto, e como foi dito na secção anterior, este método tem a desvantagem de poder não dar a estimação mais ajustada a certo conjunto de pontos quando é limitado no tempo. E dado que se retende construir um algoritmo rápido, que possa ser implementado no processamento de imagem em tempo real, este pode ser um problema. Na tentativa de tornar o método mais estável, foi introduzido um filtro de Kalman.

O filtro de Kalman consiste num conjunto de equações matemáticas que permitem estimar de forma recursiva o estado de um processo, de modo a minimizar a média do erro quadrático. Com o filtro de Kalman é possível fazer estimativas de estados passados, presentes ou até mesmo prever estados futuros. [7]

Este tipo de filtro tenta calcular o estado x de um processo descrito pela seguinte equação

$$x_n = \phi.x_{n-1} + w_{n-1}$$

com uma medição y que é

$$y_n = C.x_n + v_n.$$

As variáveis aleatórias w e v representam o ruído do processo e da medição, respectivamente. Estas assumem-se como sendo independentes, de ruído branco e com distribuições de probabilidade normais

$$p(w) \sim N(0, Q),$$
$$p(v) \sim N(0, R).$$

Q é a matriz de covariância de ruído do processo e R é a matriz de covariância de ruído da medição. A matriz ϕ relaciona o estado no instante de tempo anterior, $n-1$, com o estado do instante de tempo actual, n. A matriz C relaciona o estado no instante actual com a medição no instante actual. [7]

Definindo $\hat{x}_n^-$ como sendo a estimativa do estado à *priori* no instante n, dado o conhecimento do processo anterior ao instante n e $\hat{x}_n$ como sendo a estimativa do estado à *posteriori* no instante n

dada a medição y_n. Podem definir-se erros de estimativa à *priori* e à *posteriori* como sendo, respectivamente,

$$e_n^- \equiv x_n - \hat{x}_n^-,$$
$$e_n \equiv x_n - \hat{x}_n.$$

Assim sendo, as covariâncias dos erros das estimativas à *priori* e à *posteriori* são, respectivamente,

$$P_n^- = E\left[e_n^- e_n^{-T}\right],$$
$$P_n = E\left[e_n e_n^T\right].$$

A equação que calcula a uma estimativa de estado à *posteriori* $\hat{x}_n$ como combinação linear de uma estimativa à *priori* $\hat{x}_n^-$ e uma diferença pesada entre a medição actual y_n e uma predição de medição $C\hat{x}_n^-$ é

$$\hat{x}_n = \hat{x}_n^- + K(y_n - C\hat{x}_n^-),$$

onde a diferença $(y_n - C\hat{x}_n^-)$ é a inovação da medição e representa a discrepância entre a medição actual e a predição da medição e tende a ser zero, à medida que estão em concordância. K é a matriz de ganho que minimiza a covariância do erro da estimativa à *posteriori*. [7]

O filtro de Kalman estima um processo, usando um controlo de *feedback*: o filtro estima o estado do processo em determinado instante e obtém um *feedback* na forma de medições. As equações deste tipo de filtro dividem-se em dois grupos: equações de actualização de tempo e equações de actualização de medida. As equações de actualização de tempo projectam o estado actual e a estimativa da covariância do erro para obter estimativas à *priori* no instante seguinte. As equações de actualização de medida são responsáveis pelo *feedback*: incorporam uma nova medição na estimativa à *priori* com a finalidade de obter uma estimativa à *posteriori* melhorada. [7]

A figura 3.14 representa o algoritmo de um filtro de Kalman. Como se pode ver, nas equações de actualização de tempo, é feita a projecção das estimativas do estado e covariância do estado do instante actual para ser utilizado no instante seguinte. Na actualização da medida, o primeiro passo consiste em calcular o ganho de Kalman, K_n, o seguinte passo é gerar uma estimativa do estado à *posteriori* recorrendo à medição e por fim, calcular uma estimativa da covariância do erro à *posteriori*.

O ajuste do filtro de Kalman passa pela sincronização das matrizes R e Q, onde R é a matriz de covariância da medição e Q é a matriz de covariância do modelo. Assim, cabe ao utilizador decidir o peso que dá a cada um dos parâmetros: se o modelo é fiável e se pretende basear-se mais no modelo, então, devem ser atribuídos valores baixos à matriz de covariância do modelo. Ao invés, se o modelo

for pouco preciso e se pretende dar maior peso às medições, deve atribuir valor baixos à matriz de covariância da medição. [7]

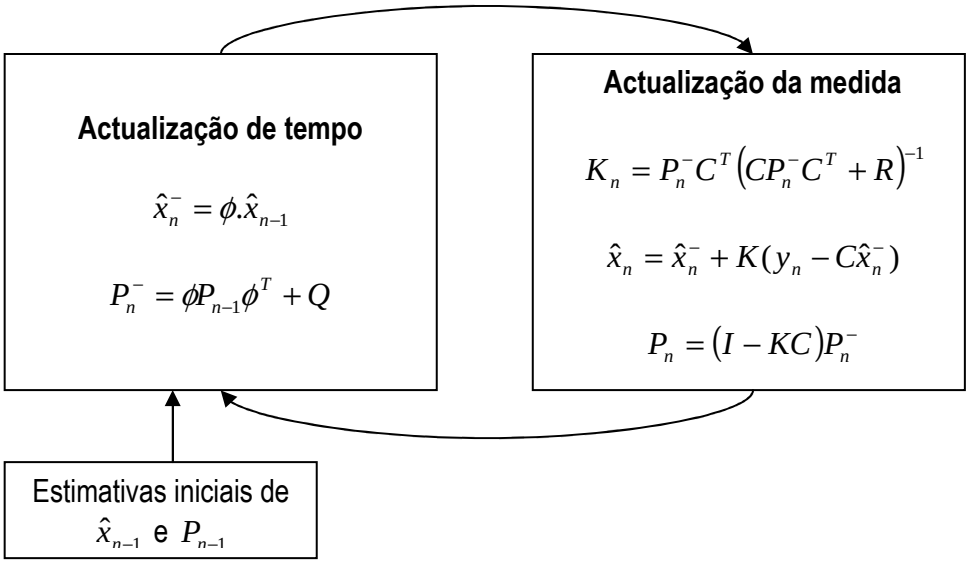

Figura 3.14: Esquema das equações calculados durante o algoritmo do filtro de Kalman.

Ao longo do processo, caso o modelo e as matrizes de covariância do modelo e da medição sejam constantes, então a matriz de ganho e a covariância do erro da estimativa vão convergir rapidamente para um valor e manterem-se constantes. Neste caso, os parâmetros do filtro podem ser pré-calculados. Extrai-se o valor da matriz de ganho e apenas há a necessidade de calcular, em cada iteração, a estimativa de estado à *posteriori* $\hat{x}_n$ dada por

$$\hat{x}_n = \hat{x}_n^- + K(y_n - C\hat{x}_n^-).$$

Neste caso, o vector de medições y é composto por seis parâmetros:

$$y = (C_x, C_{y,} M, m, \theta, \varphi)^T$$

onde C_x e C_y representam as coordenadas x e y do centro da cónica, respectivamente, M e m representa os semi-eixos maior e menor da cónica, respectivamente, θ é o ângulo de rotação do eixo maior da cónica e φ é o ângulo de rotação da marca da lente artroscópica.

Para os semi-eixos da cónica, o modelo assume que devem manter-se constantes ao longo do tempo. Em relação aos restantes parâmetros, o modelo assume que estes se deslocam com um movimento rectilíneo uniforme, o qual se traduz na equação

$$x_n = x_{n-1} + v.\Delta t,$$

onde $\Delta t = 40ms$ (tempo de uma *frame*). O modelo considera ainda que o deslocamento do ângulo de rotação da cónica deve ser igual ao deslocamento do ângulo da marca da lente, já que ambos dependem da rotação do artroscópio relativamente à câmara.

O ajuste do filtro foi feito, ajustando as matrizes R e Q. Dado que o modelo do filtro é constante, a matriz de ganho tende para um valor e não se altera ao longo do tempo. Não existe, portanto, a necessidade de esta ser calculada em cada ciclo de processamento da imagem. O procedimento adoptado consistiu em fazer a determinação da matriz de ganho em MATLAB e utilizá-la directamente no algoritmo de segmentação implementado em C. A matriz de ganho K obtida foi

$$K = \begin{bmatrix} 0.6644 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0.6644 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0.2000 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0.2000 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0.1489 & 0.0706 \\ 0 & 0 & 0 & 0 & 0.0706 & 0.1489 \\ 0.4096 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0.4096 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0.3018 & 0.3018 \end{bmatrix}.$$

3.5 Cálculo da percentagem de pixeis de fundo

Por vezes os histogramas apresentam comportamentos que podem não se adequar ao método de segmentação utilizado. Por exemplo, no caso do histograma da figura 3.15(a) existe um pico secundário para pixeis de valor 19. Assim, ao procurar o valor de *threshold* para binarizar a imagem, este pico iria ser tido em conta e o valor que devia de ser de 23, passaria a ser 19, pelo que se iria estar a aplicar um valor muito baixo de corte.

Como se verifica, o valor de corte seria muito baixo e o resultado seria mau, já que iriam ser apanhados pontos que pertencem ao fundo da imagem (zona não útil da imagem). Após a aplicação do RANSAC e do filtro de Kalman está situação pode ser atenuada, no entanto, continua a apresentar problemas de segmentação, sobretudo no RANSAC, por dois motivos: por um lado, o RANSAC pode agarrar-se a estes pontos *outliers* e efectuar estimativas erradas e, por outro lado, como existem muitos pontos *outliers*, o algoritmo de RANSAC pode tornar-se mais lento a encontrar uma boa estimativa, o que pode ser mau no sentido de não se conseguir fazer o processamento em tempo real, como se pretende. Neste sentido, foi implementada uma condição induzida pela percentagem de pixeis de

fundo. Ou seja, inicialmente calcula-se a percentagem de pixeis de fundo, e em cada iteração, no cálculo do valor de corte, é tido em conta o valor desta percentagem.

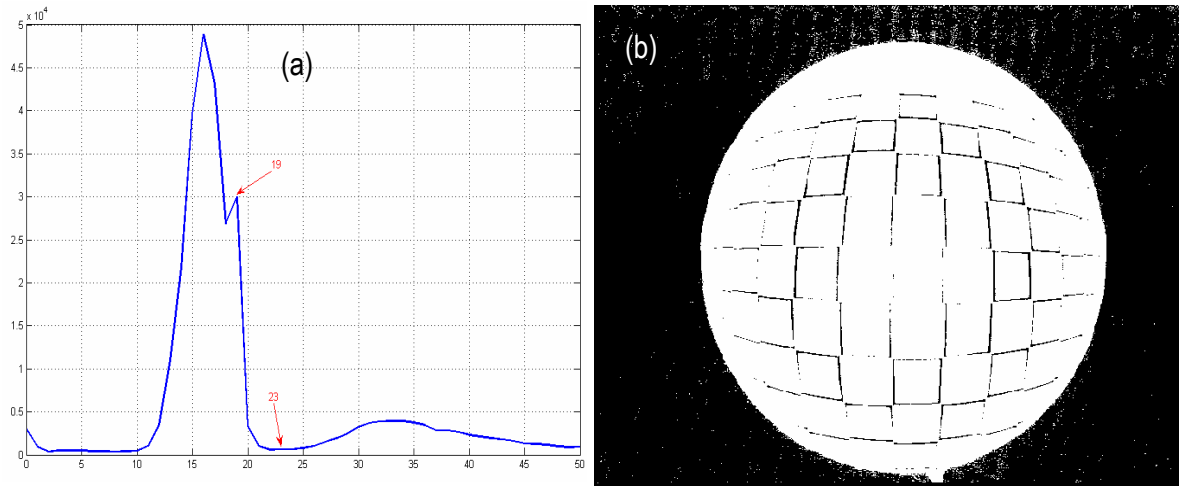

Figura 3.15: (a) Exemplo de histograma com artefacto, o qual pode levar a erros de segmentação. **(b)** Resultado da binarização da imagem obtido.

O procedimento para calcular o valor da percentagem de pixeis de fundo é o seguinte:

- Adquire-se uma imagem de uma superfície branca, neste caso, uma folha de papel (figura 3.16);
- Binariza-se a imagem com um valor de corte fixo de valor 40;
- Faz-se a contagem do número de pixeis brancos na imagem (vai ser o número de pixeis correspondente à zona útil da imagem);
- A partir do valor do número de pixeis obtido, é possível determinar a percentagem de pixeis do fundo.

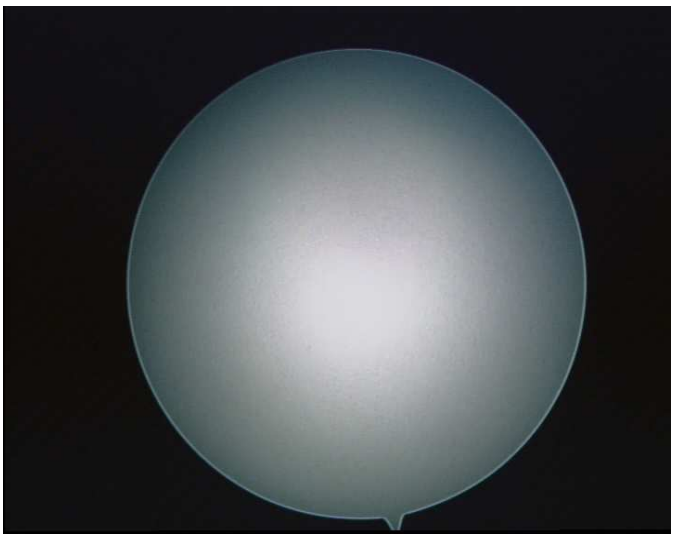

Figura 3.16: Imagem típica de superfície branca (folha de papel) utilizada para calcular percentagem de pixeis de fundo.

A seguinte figura representa um histograma de uma imagem deste tipo.

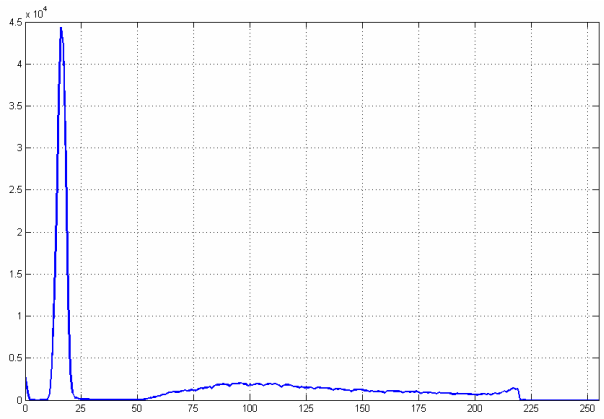

Figura 3.17: Histograma de uma imagem de uma superfície branca.

Como se pode verificar na imagem 3.17, os pixeis estão divididos em duas zonas distintas: uma que vai do nível 0 ao nível 25, correspondente à zona de fundo e outra que vai do nível 54 ao nível 220 e que correspondente à zona útil da imagem. Assim, após a análise do histograma de uma imagem deste tipo, concluiu-se que o valor de corte igual a 40 seria adequado, já que se encontra sensivelmente a meio da separação das duas zonas. Na seguinte figura, está mostrado o resultado da binarização de uma imagem com o valor de corte de 40.

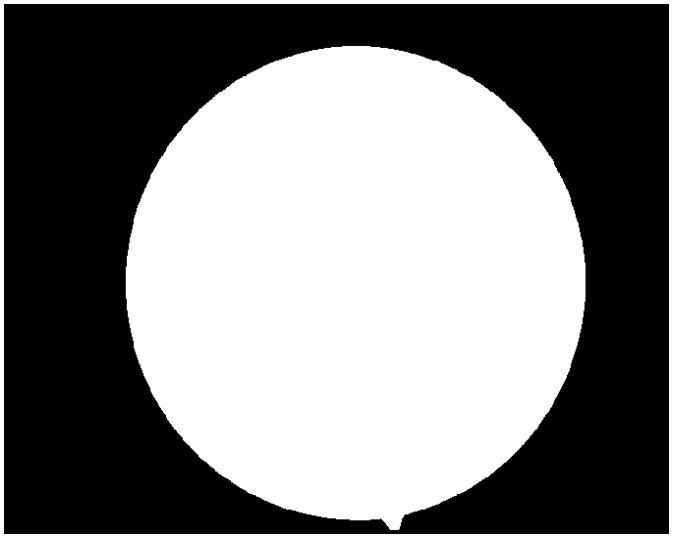

Figura 3.18: Resultado da binarização da imagem da figura 3.16, utilizando um valor de corte de 40.

Como se pode verificar, com valor de corte escolhido, obtém-se um bom resultado de binarização. Assim, é possível determinar o número de pixeis do fundo e calcular a sua percentagem, relativamente ao número total de pixeis. Para facilitar, esta percentagem será denominada *%PF* (percentagem de pixeis de fundo).

O valor obtido vai servir como referência no cálculo do valor de *threshold* adaptativo a utilizar na segmentação da imagem. Ou seja, quando se calcula o valor de *threshold*, é calculada a percentagem total de pixeis referentes a todos os níveis entre 0 e o nível referente ao valor de corte, inclusive. Esta percentagem será denominada de *%TN* (Percentagem de pixeis de todos os níveis).

No caso de *%TN* ser menor que *%PF* e a diferença entre as duas ser menor ou igual a 5% ou no caso de *%TN* ser maior que *%PF*, então, admite-se que o valor de corte está correcto e é aplicado o valor determinado. No caso de *%TN* ser menor que *%PF* e a diferença entre as duas ser maior que 5%, nesse caso, o valor de corte determinado não é o mais ajustado. O procedimento é calcular o vale imediatamente a seguir ao valor actual de corte, passando este a ser o novo valor de corte. Este valor é então testado novamente e o procedimento descrito é efectuado até que se encontre um valor de *%PF* - *%TN* < 5% ou então, um valor de *%TN* maior que o valor de *%PF*. O valor da diferença de 5% foi obtido experimentalmente.

3.6 Testes ao algoritmo de segmentação implementado

Após ser efectuada a segmentação, o resultado é mostrado no ecrã. Este pode ser feito em duas modalidades: apenas se mostra a cónica obtida e o ângulo de rotação da marca da lente (figura 3.19(a)) ou mostra-se a cónica obtida, o ângulo de rotação da marca da lente e todos os parâmetros da cónica, os pontos determinados e a box da cónica (figura 3.19(b)).

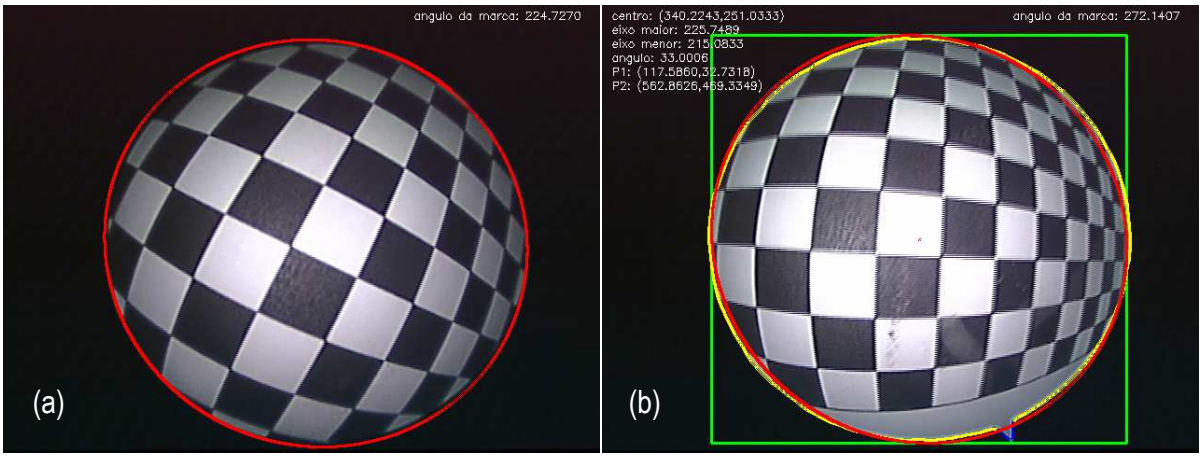

Figura 3.19: Formas possíveis de mostrar os resultados **(a)** apenas se mostra a cónica e o ângulo da marca da lente, e **(b)** mostra todos os parâmetros da cónica, o ângulo da marca da lente, a cónica (a vermelho), os pontos *inliers* (a amarelo) e os *outliers* (a azul) estimados no RANSAC e a box da cónica (a verde).

Foram feitos alguns testes ao algoritmo de segmentação. Afim de testar o tempo médio de processamento de uma *frame*, determinou-se o tempo de processamento de cada *frame* num conjunto

de *263 frames*. O tempo médio de processamento de uma *frame* obtido foi de *35.7414 ms*, com um desvio padrão de *2.4992 ms*.

De seguida, apresentam-se alguns gráficos com os resultados da determinação de alguns parâmetros da cónica em três situações diferentes: com a câmara fixa, a aplicar tensão na lente e com a câmara em rotação.

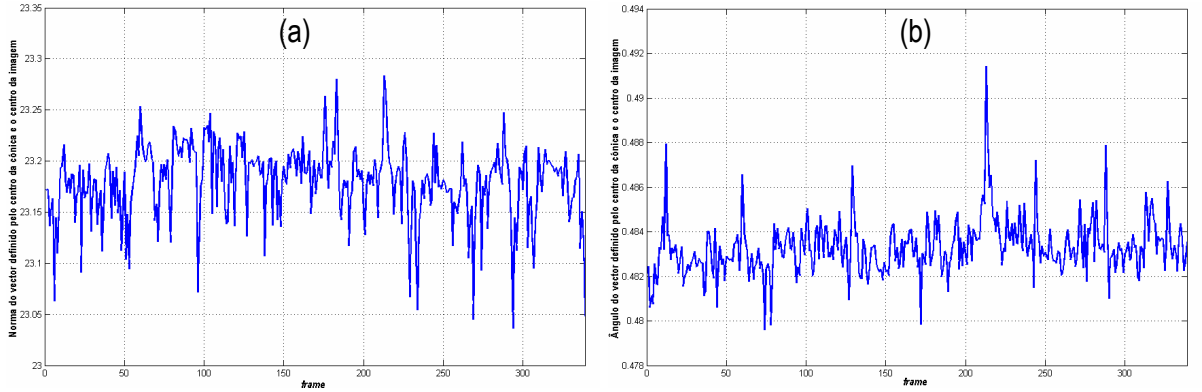

Figura 3.20: (a) Norma (em pixeis) e **(b)** ângulo (em radianos) do vector definido pelo centro da cónica e o centro da imagem no caso da câmara estar fixa.

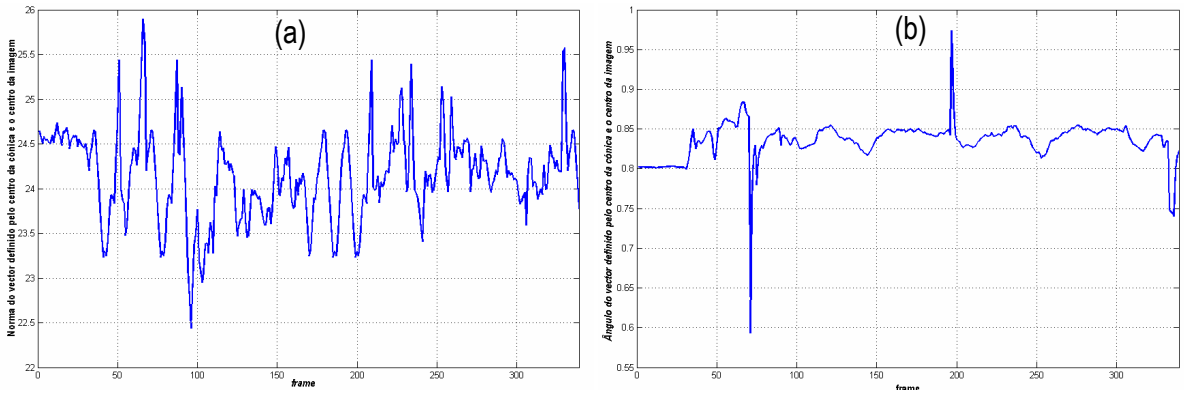

Figura 3.21: (a) Norma (em pixeis) e **(b)** ângulo (em radianos) do vector definido pelo centro da cónica e o centro da imagem no caso da câmara rodar em relação à lente.

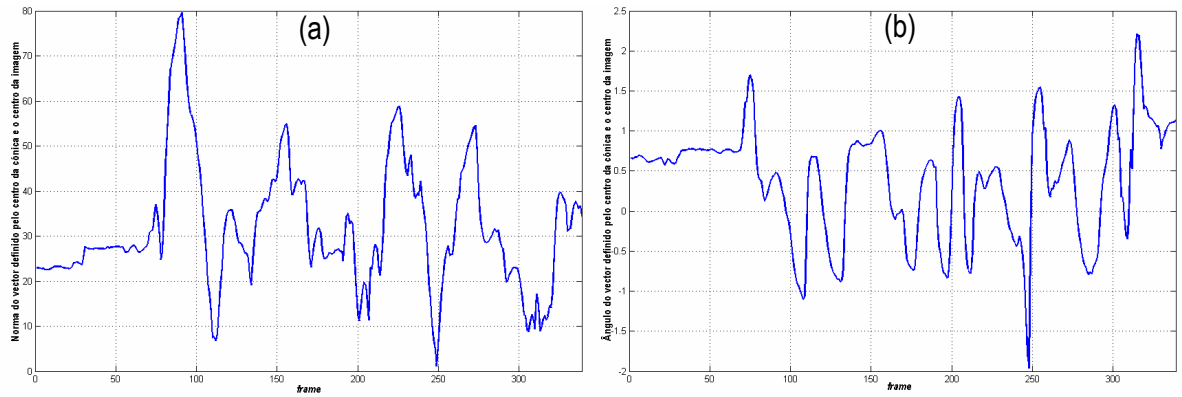

Figura 3.22: (a) Norma (em pixeis) e **(b)** ângulo (em radianos) do vector definido pelo centro da cónica e o centro da imagem no caso de ser aplicada tensão à lente.

Como se verifica nos gráficos relativamente à posição do centro da cónica estimada, a posição deste mantém-se bastante estável no caso da câmara estar fixa, como se previa, já que não deverão existir variações acentuadas da posição da zona útil da imagem e, por conseguinte, a cónica deverá manter-se numa posição mais ou menos constante. No caso da câmara estar em rotação, também se verifica que a posição do centro tem pouca variação. No caso de ser aplicada uma tensão na lente, a posição do centro tem uma maior variação. Tal situação era previsível já que ao ser aplicada uma tensão na lente, a posição da zona útil da imagem varia e dado que a cónica estimada tende a fazer o seguimento desta zona, deverá variar a sua posição, o que implica que a posição do centro da cónica varie.

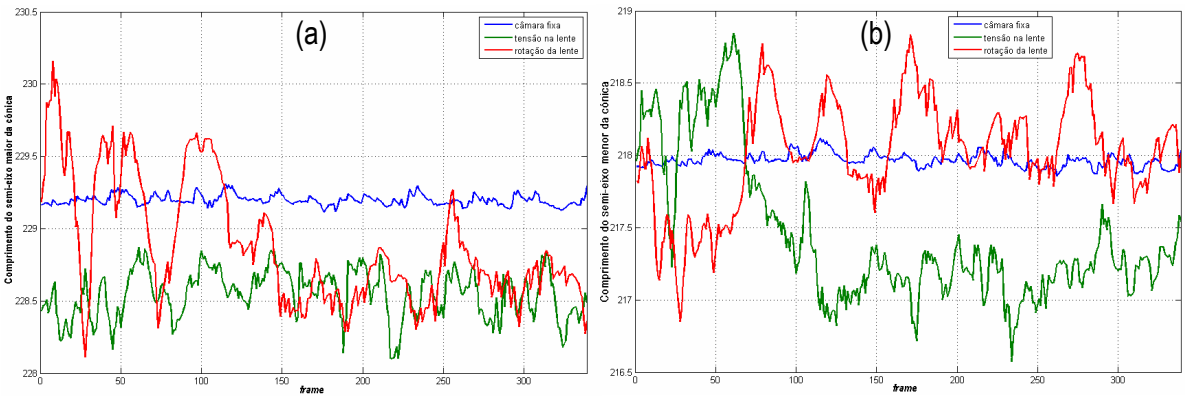

Figura 3.23: Comprimento dos semi-eixos **(a)** maior e **(b)** menor (em pixeis) em cada um dos casos estudados

No que diz respeito as variações do comprimento dos semi-eixos maior e menor da cónica, estes mantêm-se bastante estáveis em ambos os casos, como seria de esperar. O facto de haver rotação da lente ou de lhe ser aplicada uma tensão implica a variação da posição da cónica, mas as suas dimensões devem manter-se constantes, o que implica que os semi-eixos não se alterem.

Quanto ao ângulo de rotação da cónica no caso da câmara estar numa posição fixa ou de lhe ser aplicada uma tensão, mantém-se constante, como se verificam nos gráficos correspondentes. No caso da câmara rodar relativamente à lente, verifica-se que o ângulo de rotação da cónica também varia, ou seja, tende a acompanhar a rotação imposta pela câmara. Tal situação era previsível já que existe a variação da posição da zona útil da imagem em torno do centro da mesma. Ou seja, apesar de não existirem grandes variações da posição do centro da cónica neste caso, esta apresenta variações na sua rotação, o que leva a variações do seu ângulo de rotação.

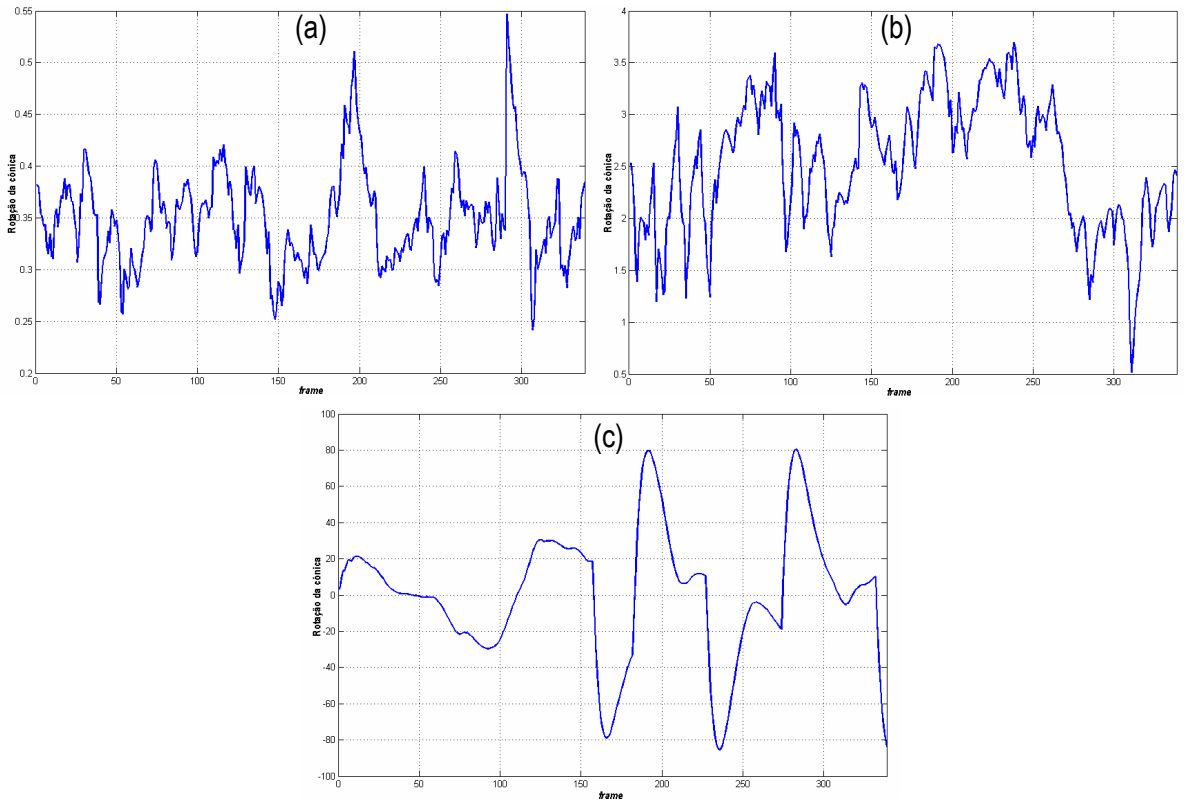

Figura 3.24: Ângulo de rotação da cónica, em graus, no caso **(a)** da câmara estar fixa, **(b)** de ser aplicada tensão na lente, e **(c)** da câmara rodar em relação à lente.

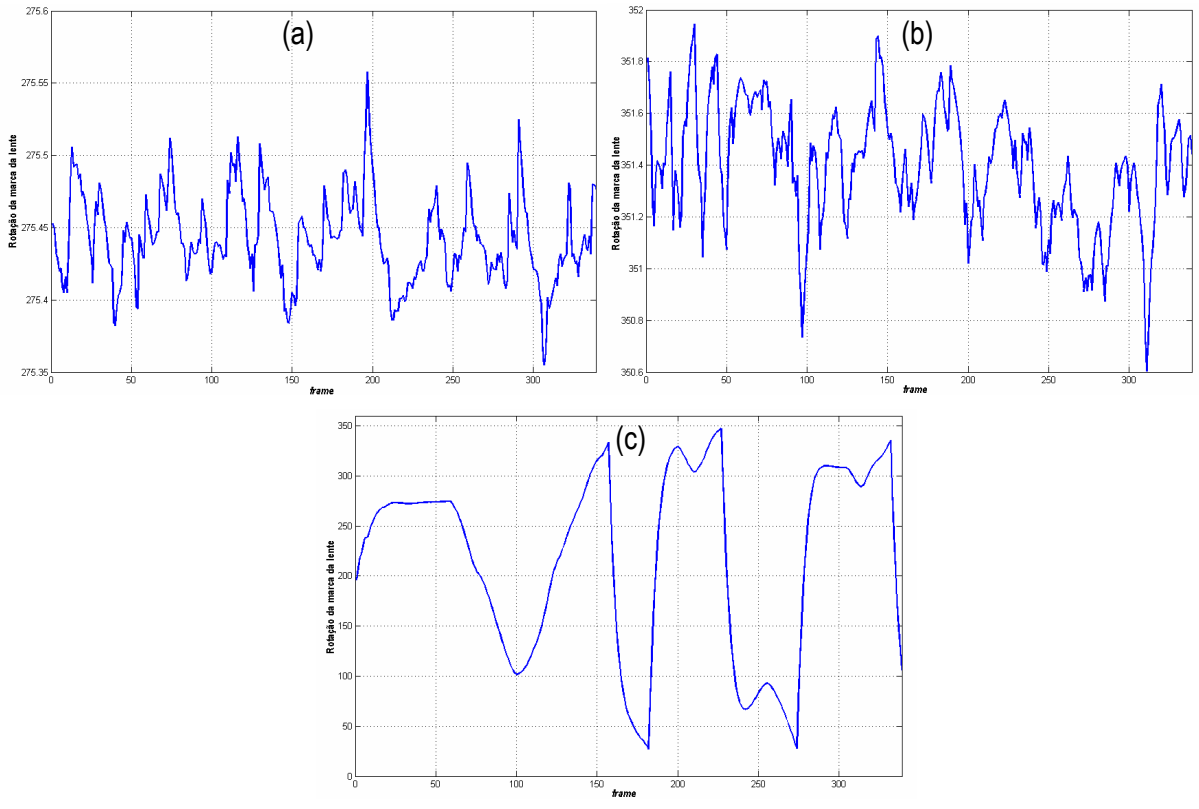

Figura 3.25: Ângulo de rotação da marca da lente, em graus, no caso **(a)** da câmara estar fixa, **(b)** de ser aplicada tensão na lente, e **(c)** da câmara rodar em relação à lente.

Como se verifica, o ângulo de rotação da marca da lente mantém-se aproximadamente constante no caso da câmara estar fixa, ou quando é aplicada tensão na lente. No caso da câmara rodar em torno da lente, o algoritmo tende a fazer o seguimento da posição da marca, o que implica que o ângulo de rotação desta varie, como seria de esperar.

3.7 Conclusões

Ao longo deste capítulo, foi explicada a implementação do algoritmo de segmentação utilizado para determinar a zona da área útil das imagens artroscópicas. Houve a necessidade de introduzir um método de estimação robusta, o RANSAC, e um filtro de Kalman, de modo a estabilizar os parâmetros da cónica estimados. Conclui-se que o método implementado é estável e robusto, tal como se pretendia e, além disso, é rápido, o que permite a sua utilização em tempo real.

Capítulo 4: Extracção de Cantos da Grelha

O segundo objectivo deste trabalho consistiu em tentar implementar uma forma de fazer a detecção automática dos cantos de uma grelha de xadrez de modo a poder-se utilizar na calibração da câmara. Embora esta seja uma área bastante explorada na literatura não se aplica no caso do vídeo artroscópico, devido a elevada distorção radial que estas imagens possuem. Foram feitos estudos a dois métodos existentes para a detecção de cantos. No entanto, verificou-se que para imagens deste tipo, estes falham.

Pretende-se portanto um método rápido, eficiente e com boa precisão, que consiga bons resultados para imagens artroscópicas. Foram implementados alguns métodos: um método baseado no detector de Harris e um método baseado na entropia dos ângulos do gradiente. A sua implementação irá ser discutida neste capítulo.

Será também, feita uma breve introdução à calibração de câmaras.

4.1 Calibração de câmaras

Como já foi referido anteriormente, a calibração pode ser definida como sendo a determinação da matriz de projecção da câmara, a qual representa o mapeamento entre as coordenadas de um ponto no mundo e as coordenadas de um ponto na imagem. Para realizar este modelo, é necessário ter em conta a existência de parâmetros intrínsecos, como por exemplo a dimensão dos pixeis e a posição do centro da imagem, que estabelecem uma relação entre o plano imagem e o plano da câmara e os factores extrínsecos, como por exemplo a orientação da câmara, que relacionam o plano da câmara com o plano do mundo. A seguinte figura mostra estas relações.

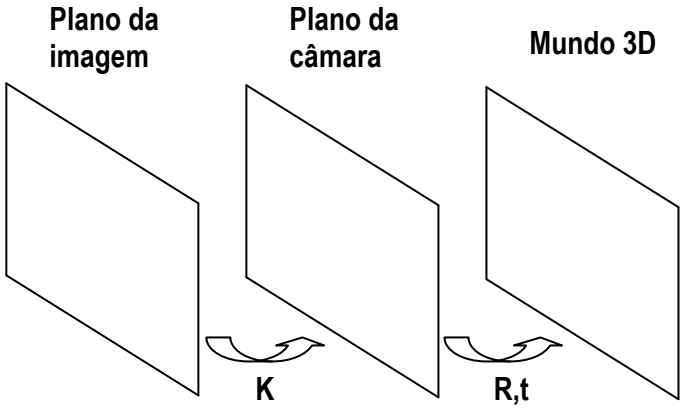

Figura 4.1: Relação entre o plano imagem e o plano da câmara e o plano da câmara e o mundo 3D.

Na figura, as matrizes R e t representam, respectivamente, rotação e translação (impostas pela orientação da câmara) e a matriz K diz respeito à matriz dos parâmetros intrínsecos da câmara e define-se como sendo

$$K = \begin{bmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix}$$

onde f_x e c_x representam a distância focal em pixeis e a posição do centro da câmara segundo a direcção x, respectivamente e f_y e c_y representam a distância focal em pixeis e a posição do centro da câmara segundo a direcção y, respectivamente.

A seguinte figura mostra a relação entre o plano da imagem e mundo 3D.

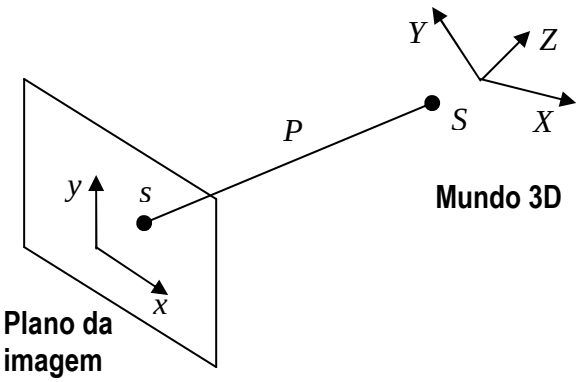

Figura 4.2: Relação entre o plano da imagem e o mundo 3D.

A matriz P define a relação entre um ponto $S(X,Y,Z)$ no mundo e um ponto $s(x,y)$ no plano da imagem e engloba os parâmetros intrínsecos e extrínsecos da câmara. Esta transformação define-se como sendo

$$\lambda \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = P \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix}, P = \begin{bmatrix} p_{11} & p_{12} & p_{13} & p_{14} \\ p_{21} & p_{22} & p_{23} & p_{24} \\ p_{31} & p_{32} & p_{33} & p_{34} \end{bmatrix} \quad (4.1)$$

onde os pontos S e s estão definidos em coordenadas homogéneas. Da equação (4.1) tem-se que

$$x = \frac{p_{11}X + p_{12}Y + p_{13}Z + p_{14}}{p_{31}X + p_{32}Y + p_{33}Z + 1} \text{ e } y = \frac{p_{21}X + p_{22}Y + p_{23}Z + p_{24}}{p_{31}X + p_{32}Y + p_{33}Z + 1}.$$

Uma vez que a matriz P tem 12 elementos, são necessários pelo menos 6 pontos para resolver este sistema. Para tal, utiliza-se uma grelha de calibração (padrão de xadrez) com dimensões conhecidas à priori. Detectam-se os cantos da imagem, dos quais se conhece a sua posição no mundo e obtém-se assim um sistema sobredimensionado. A seguinte figura resume o procedimento de calibração de uma câmara.

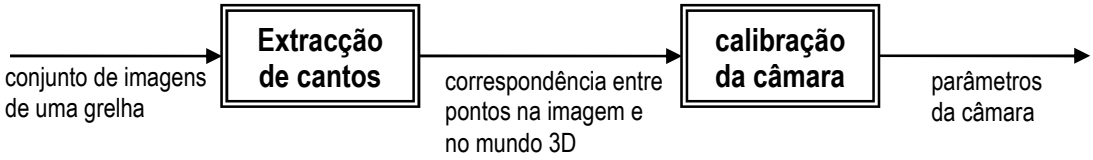

Figura 4.3: Resumo do procedimento de calibração de uma câmara.

4.2 Alguns algoritmos de extracção de cantos existentes

Nesta secção vais ser feita referência ao modo como duas importantes bibliotecas de funções de calibração já existentes efectuam a extracção de cantos de uma grelha de modo a proceder à calibração de câmaras: a toolbox de calibração de câmaras de Bouguet (implementada em Matlab) e a função *FindChessBoardCorners* do OpenCV (implementada em C/C++).

4.2.1 Toolbox de calibração de câmaras de Bouguet

Bouguet desenvolveu uma toolbox de calibração de câmaras em Matlab. De seguida vai ser explicado o funcionamento do algoritmo de detecção de cantos de uma grelha utilizado por esta toolbox.

- Marcam-se os quatro pontos correspondentes aos cantos das extremidades da grelha de calibração (a figura 4.4 ilustra esta situação). O primeiro ponto a ser marcado, deve ser a origem. Com estes quatro pontos é definida a fronteira da zona da grelha de calibração.

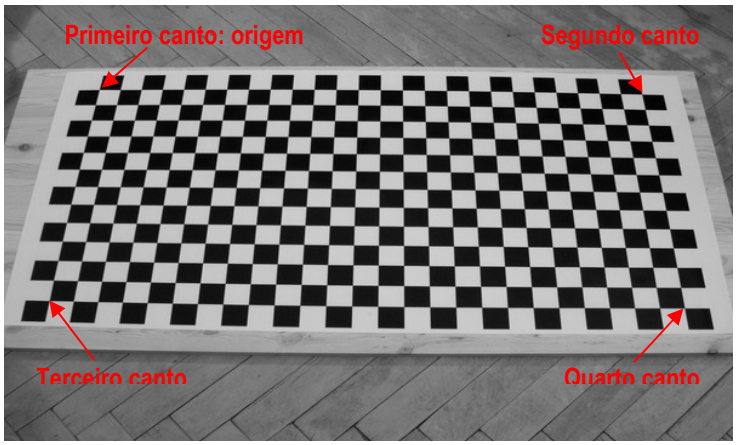

Figura 4.4: Exemplo de como se marcam os quatro cantos externos da grelha de calibração. O primeiro canto que se marca é a origem.

- A posição dos quatro pontos marcados é refinada com base com base em características da imagem: é procurada numa vizinhança do ponto (definida pelo utilizador) pixeis que sejam mais prováveis de serem cantos da grelha do que o ponto marcado. No caso de não ser encontrado nenhum ponto que verifique esta condição, o próprio ponto é considerado o canto.

- É calculado o número de quadrículas que a grelha do padrão têm em cada uma das direcções. Este procedimento é efectuado com base em variações de cor preto/branco na imagem compreendida dentro da fronteira da grelha de calibração.

- É calculada uma homografia com os quatro pontos marcados: estes são forçados a pertencer aos vértices de um quadrado com uma unidade de lado. São marcados pontos intermédios igualmente espaçados segundo cada uma das direcções. O número de pontos intermédios marcados segundo cada uma das direcções é o mesmo que o número de quadrículas determinadas no passo anterior. Calcula-se novamente uma homografia, mas inversa à anterior: com base na posição real dos quatro pontos iniciais, determina-se a posição real de todos os pontos intermédios.

- É aplicado novamente o refinador de cantos da grelha a todos os pontos determinados para calcular o pixel que mais se adequa a ser considerado canto.

- Pode ainda ser feito o ajuste de um parâmetro para tentar corrigir as posições dos cantos, quando estas falham devido à distorção radial que a imagem possa ter.

A figura 4.3 mostra o resultado da extracção dos cantos de uma grelha obtido, utilizando o algoritmo de extracção de cantos da toolbox de calibração do Bouguet.

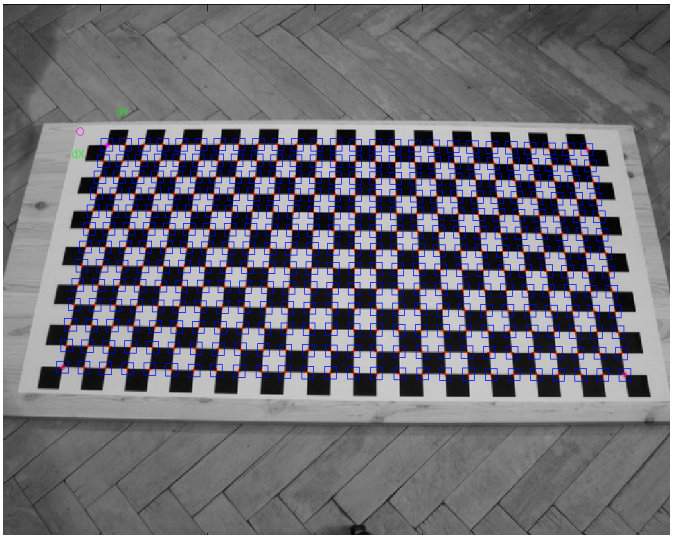

Figura 4.5: Resultado da detecção de cantos de uma grelha obtido pela toolbox de calibração de câmaras de Bouguet.

Em conclusão, o algoritmo de detecção de cantos da toolbox de calibração de câmaras de Bouguet é bastante preciso para imagens com pouca distorção. No entanto, é não é completamente automático, já que é necessário marcar quatro pontos iniciais para que o algoritmo possa determinar a posição dos cantos da grelha.

4.2.2 Função *FindChessBoardCorners* do OpenCV

A Intel desenvolveu uma biblioteca de funções para trabalhar, sobretudo, com problemas de visão por computador e, por conseguinte, existem funções associadas à calibração de câmaras. Interessa aqui perceber o funcionamento da função *FindChessBoardCorners*, a qual é a responsável pela detecção dos cantos da grelha. Na seguinte figura, estão representados os parâmetros de entrada e de saída da função.

Figura 4.6: Parâmetros de entrada e de saída da função *FindChessBoardCorners*.

De seguida e de modo sucinto, vai-se apresentar o método de detecção de cantos da grelha utilizado pela função *FindChessBoardCorners*. [8]

- É passada à função uma imagem de calibração e as dimensões (numero de coluna e de linhas) da grelha que está contida nessa imagem.

- A imagem é convertida para imagem em escala de cinzentos (no caso de ainda não estar nesse formato), é feita uma dilatação de modo a separar as quadrículas da grelha de calibração e efectua-se uma binarização para efectuar a segmentação das quadrículas da grelha.

- Inicia-se a procura de cantos na imagem: são procurados contornos de regiões a preto na imagem e são seleccionados aqueles que apresentem uma forma quadrangular, fazendo a sua aproximação por polígonos de 4 vértices. Destes polígonos, escolhem-se aqueles que se assemelham a quadrados.

- São extraídos os cantos dos polígonos semelhantes a quadrados que tenham pelo menos um canto na sua proximidade.

- Os cantos são agrupados em linhas de acordo com o tamanho do objecto de calibração e com determinada ordenação.

- No caso dos cantos da grelha não serem todos detectados e ordenados (daí a necessidade de introduzir como parâmetros as dimensões da grelha), é formada uma região de interesse na zona da imagem onde tenham sido detectados alguns pontos e é repetido o procedimento.

- Após terem sido detectados todos os cantos, é chamada uma outra função, a função *FindCornerSubPix*, a qual refina a posição do canto numa vizinhança de 5x5 pixeis, refinamento este que é feito com base em características da imagem.

A figura 4.5 mostra o resultado da extracção dos cantos de uma grelha obtido, utilizando o algoritmo de extracção de cantos da função *FindChessBoardCorners*.

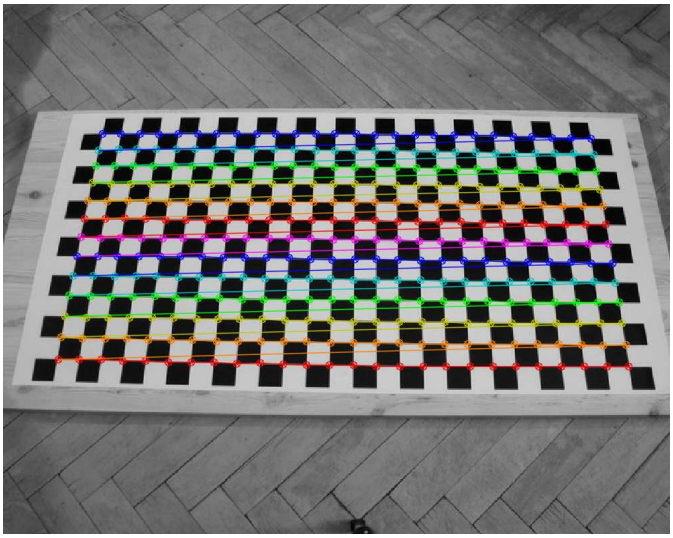

Figura 4.7: Resultado da detecção de cantos de uma grelha obtido pela função *FindChessBoardCorners*.

Em conclusão, o algoritmo de detecção de cantos da função *FindChessBoardCorners* é bastante preciso para imagens com pouca distorção. Além disso, é um método totalmente automático.

4.3 Aplicação dos algoritmos de detecção de cantos descritos às imagens artroscópicas

Como já foi referido anteriormente, as imagens artroscópicas são imagens com grande distorção, sobretudo, distorção radial. A distorção radial degrada a qualidade de uma imagem. Por exemplo, para uma imagem de uma grelha, as quadriculas que são todas do mesmo tamanho e estão separadas por linhas rectas, na imagem adquirida, as linhas são projectadas em linhas curvas e, além disso, as quadrículas não têm um tamanho uniforme. Estes artefactos são introduzidos pela distorção radial da lente artroscópica, a qual dá a ideia de a imagem ser "esférica" e não planar, como o é verdadeiramente. Na figura 4.8, são mostradas imagens de uma grelha que reflectem esta situação.

Na secção anterior foram apresentados de forma sucinta dois algoritmos de detecção de cantos de uma grelha de calibração. E concluiu-se que ambos os métodos têm um bom desempenho quando são aplicados em imagens com pouca distorção radial. No entanto, em imagens artroscópicas, uma vez que há muita distorção radial, os algoritmos descritos são incapazes de determinar os cantos das grelhas de calibração.

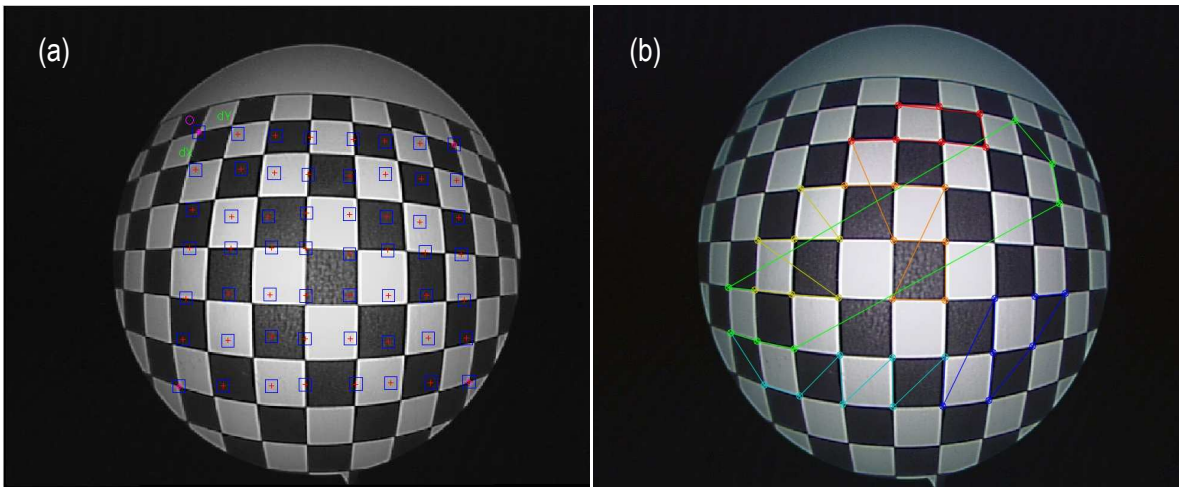

Figura 4.8: (a) Resultado da detecção de cantos da grelha numa imagem artroscópica obtido com a toolbox de calibração de câmaras de Bouguet. **(b)** Resultado da detecção de cantos da grelha numa imagem artroscópica obtido com a função *FindChessBoardCorners.*

Como se pode verificar nas figuras 4.8(a) e 4.8 (b), os algoritmos de detecção de cantos falham para imagens artroscópicas. Dado que estas têm uma grande distorção radial, as quadrículas tendem a não ter o mesmo tamanho. No caso da figura 4.8(a), quando se calculam as homografias de pontos, o algoritmo assume que os pontos se encontram todos à mesma distância entre si, o que não se verifica para estas imagens e, por conseguinte, o algoritmo não consegue encontrar as coordenadas próximas dos cantos da grelha para que depois possa ser feito o seu refinamento. No caso da figura 4.8(b), o algoritmo de detecção não consegue ordenar correctamente os pontos, já que as linhas rectas são projectadas em linhas curvas. Assim, conclui-se que nenhum destes métodos pode ser utilizado neste caso.

4.4 Implementação de novos métodos de detecção de cantos de grelhas

Neste relatório, já foram descritos alguns métodos de detecção de cantos de uma grelha. No entanto, como se verificou na secção anterior, estes métodos falham em imagens com muita distorção radial. Nesta secção vão ser descritos dois métodos implementados para efectuar a detecção dos cantos da grelha: método baseado no detector de Harris e método baseado nas entropias dos ângulos do gradiente.

4.4.1 Método baseado no detector de Harris

O detector de Harris é uma ferramenta muito utilizada na detecção de pontos de interesse numa imagem, pois este não varia com rotações, escalas variações de iluminação ou ruído da imagem. Este é baseado na função de auto-correlação local de um sinal. A função de auto-correlação local mede variações locais do sinal quando uma determinada janela é deslocada por uma quantidade pequena em diferentes direcções. [9] Na seguinte figura é apresentada a ideia base do detector de Harris.

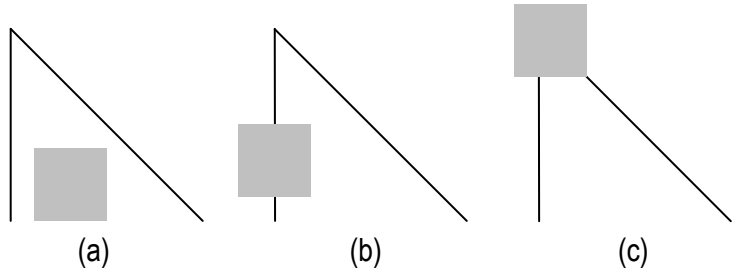

Figura 4.9: Ideia base do detector de Harris. (a) zona constante: não existe variações ao longo das todas as direcções; (b) aresta: não existem variações ao longo da direcção da aresta; e (c) canto: variações significantes em todas as direcções.

Seja $(\Delta x, \Delta y)$ um deslocamento e (x, y) um ponto. A função de auto-correlação é dada por

$$c(x,y) = \sum_{W} \left[I(x_i, y_i) - I(x_i + \Delta x, y_i + \Delta y) \right]^2$$

onde I representa a função imagem e (x_i, y_i) são os pontos situados sob a janela gaussiana W centrada em (x, y).

Em [9] foi demonstrado que a função de auto-correlação pode ser descrita por

$$c(x,y) = \begin{bmatrix} \Delta x & \Delta y \end{bmatrix} C(x,y) \begin{bmatrix} \Delta x \\ \Delta y \end{bmatrix}$$

onde a matriz $C(x,y)$ é definida por

$$C(x,y) = \begin{bmatrix} \sum_{W} (I_x(x_i, y_i))^2 & \sum_{W} I_x(x_i, y_i) I_y(x_i, y_i) \\ \sum_{W} I_x(x_i, y_i) I_y(x_i, y_i) & \sum_{W} (I_y(x_i, y_i))^2 \end{bmatrix}$$

em que I_x e I_y representam as derivadas parciais (gradientes) segundo a direcção x e y, respectivamente.

Sejam λ_1 e λ_2 os valores próprios de $C(x, y)$. Então:

- Se λ_1 e λ_2 têm ambos um valor baixo, a zona da imagem em questão tem intensidade aproximadamente constante;

- Se um valor próprio tem valor elevado e outro tem valor baixo, a zona da imagem diz respeito a uma aresta;

- Se λ_1 e λ_2 têm valores elevados, a zona da imagem diz respeito a um canto.

R é o operador de Harris e define-se como sendo

$$R = \det C - k\left(traço(C)\right)$$

onde $\det C = \lambda_1 \lambda_2$, $traço(C) = \lambda_1 + \lambda_2$ e k é um parâmetro constante que pode ser ajustado (o valor ideal deverá ser $k = 0.04$). Quando $R > R_{thr}$, onde R_{thr} é um parâmetro de *threshold*, está-se na presença de um canto.

Na figura 4.10 está representado o resultado obtido por este método. Neste caso, a janela utilizada foi uma janela gaussiana com dimensão 49x49 pixeis e o valor de R_{thr} foi de 0.5.

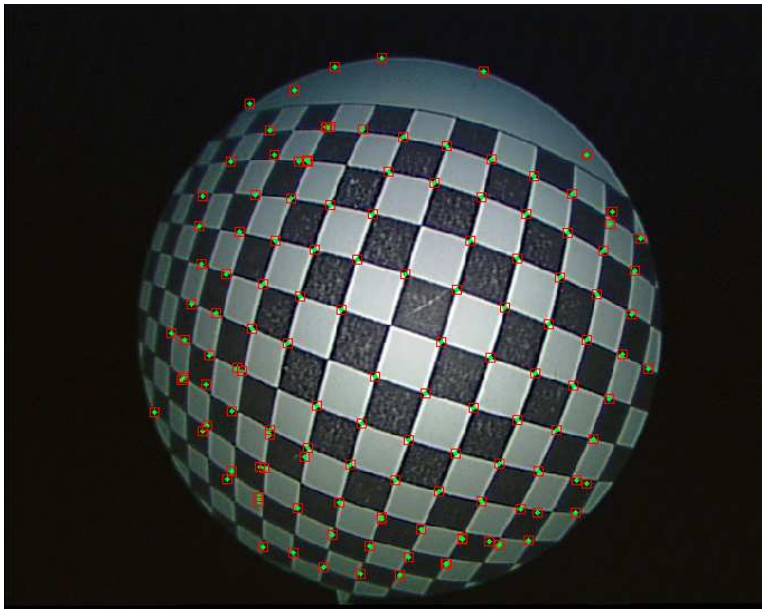

Figura 4.10: Resultado obtido pelo método de Harris.

Analisando a figura, verifica-se um bom desempenho do método na zona central da imagem, já que de um modo geral os cantos que foram marcados têm boa precisão. Houve, no entanto, alguns

cantos que não foram marcados. Para os cantos mais situados à periferia, o desempenho já não é tão bom, pois é pouco preciso. E também aqui, existem cantos que não são marcados.

Em suma, concluiu-se que seria muito difícil definir uma janela que se ajustasse a ambas as zonas da imagem, porque se na zona central os quadrados estão pouco distorcidos, na zona mais periférica, estes são afectados por muita distorção e o tamanho das quadrículas é muito menor, o que leva a maior imprecisão na detecção de cantos.

4.4.2 Método baseado na entropia dos ângulos do gradiente

A entropia é uma medida de desordem de um sistema. Neste caso, este conceito foi aplicado aos ângulos do gradiente da imagem. A figura 4.11 esquematiza os quatro conjuntos principais de ângulos dos gradientes da imagem de uma grelha. Nas zonas assinaladas a vermelho, uma vez que a que a cor se mantém constante, o gradiente deverá ser nulo. Tal situação implica a que o gradiente não tenha orientação nestas zonas e, portanto, a entropia dos gradientes nestas zonas deverá ser reduzida. Nas zonas de fronteira assinaladas a azul e a violeta existem variações fortes de cor pelo que existirá um gradiente com uma orientação bem definida nestas zonas. Contudo, será predominante numa direcção (horizontal no caso da zona assinalada a azul, já que não existem variações de cor na direcção vertical e vertical na zona assinalada a violeta, pois não existem variações na direcção horizontal). Uma vez que estas direcções do gradiente estão bem definidas, também nestas zonas, a entropia dos ângulos do gradiente deverá ser reduzida. Na zona de um canto, assinalada a verde, o gradiente tenderá a ser nulo, pois os gradientes são compensados pelos sentidos opostos. Contudo, estas zonas possuem grande variação dos ângulos de gradiente, o que implica que sejam zonas de muita entropia.

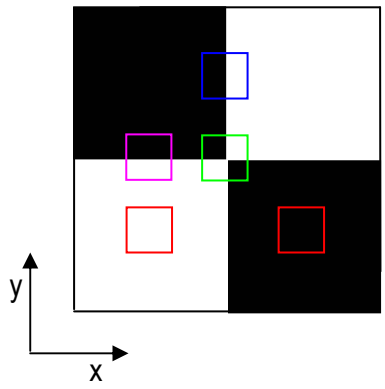

Figura 4.11: Esquematização dos quatro conjuntos principais de ângulos dos gradientes da imagem de uma grelha.

Assim, o primeiro passo consiste na detecção dos contornos da imagem de modo a construir-se uma máscara de modo a utilizar-se apenas as zonas de interesse da imagem. O método utilizado na

detecção de contornos foi o método de Canny. Este é algoritmo muito poderoso neste tipo de aplicações e tem por base os gradientes da imagem. Primeiro, é feito um alisamento da imagem com um filtro gaussiano. É então calculado a amplitude do gradiente da imagem em cada uma das direcções, bem como a sua orientação. Aplica-se um detector de máximos locais de modo a tornar mais finos os contornos detectados. Por último, é aplicada uma histerese de modo a eliminar alguns falsos contornos detectados. [10]

De seguida calcula-se a entropia das orientações do gradiente para os pixeis definidos pela máscara e determinam-se os máximos locais de entropia. Este procedimento permite obter posições de zonas onde possam estar definidos cantos da grelha.

Por ultimo, é feito um refinamento da posição exacta do canto, baseada em características da imagem. De seguida, Irá ser apresentada de forma sucinta a teoria utilizada para efectuar este refinamento. A equação

$$\nabla I^T u + I_t = 0 \qquad (4.1)$$

expressa o facto de que para determinado ponto numa imagem I, existe um ponto diferente numa outra imagem com a mesma intensidade. $\nabla I = \left(I_x, I_y\right)^T$ e I_t são as derivadas espacial e temporal, respectivamente, da imagem e o vector u é a velocidade de movimento da imagem (que esta relacionada com a forma como um ponto se move numa imagem). Em [11], mostra-se que o vector u é dado, no caso de G ser invertível, por

$$u = G^{-1} b$$

onde

$$G = \begin{bmatrix} \sum I_x^2 & \sum I_x I_y \\ \sum I_x I_y & \sum I_y^2 \end{bmatrix} \text{ e } b = \begin{bmatrix} \sum I_x I_t & \sum I_y I_t \end{bmatrix}^T .$$

Como se verifica, nem sempre a matriz G é invertível. No caso de $I_x = 0$ e/ou $I_y = 0$, a matriz G não é invertível. Para pontos da imagem em que a correspondente matriz G definida para a sua vizinhança for invertível, estes pontos são marcados como sendo pontos característicos, neste caso, cantos da grelha de calibração.

A seguinte figura mostra o resultado obtido, após ser aplicado o método de detecção de cantos baseado na entropia dos ângulos do gradiente.

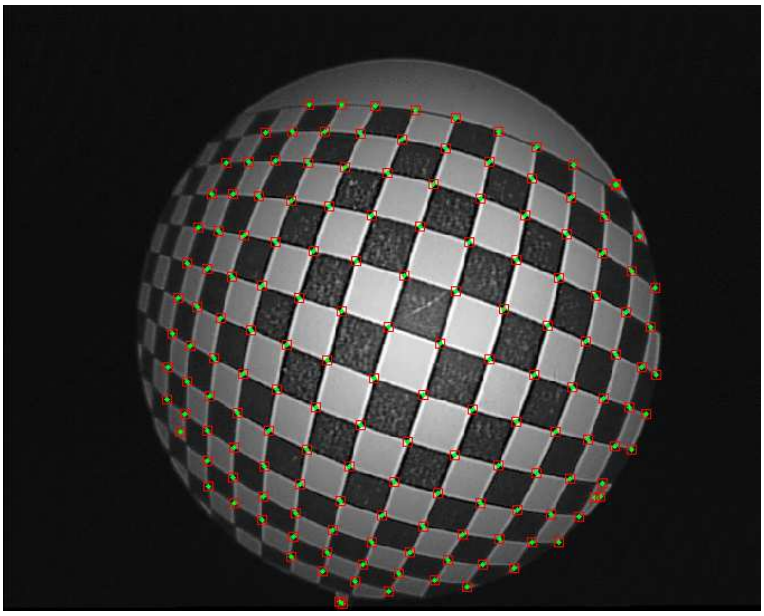

Figura 4.12: Resultado obtido pelo método de entropia dos ângulos do gradiente.

Como se verifica na figura 4.12, este método é bastante preciso e de um modo geral, os cantos são marcados com boa precisão. No entanto, na zona mais periférica da imagem, existem alguns cantos que não são detectados e outros que são marcados mas em posições incorrectas. Tal situação se deve ao elevado efeito da distorção radial na periferia. Ainda assim, conclui-se que a aproximação utilizada permite obter bons resultados.

4.5 Contagem das quadrículas

Como se verificou, o método baseado na entropia dos ângulos do gradiente permite obter posições precisas dos cantos da grelha na imagem, sobretudo, na zona central da imagem. No entanto, na zona mais periférica os cantos da grelha são pouco precisos e inúteis do ponto de vista da calibração. Pretende-se portanto determinar a melhor zona da imagem, zona onde todos os cantos tenham sido marcados com boa precisão. Para tal, foi pensado um método iterativo de contagem de quadrículas. Este consiste na expansão a partir do centro da cónica e baseia-se nas normas dos vectores definidos por cantos opostos das quadrículas.

A figura 4.13 mostra a primeira iteração do processo de contagem. É definido um novo centro (ponto marcado a azul) que consiste no ponto médio dos três pontos mais próximos do centro da cónica (representado a vermelho). Calculam-se os quatros cantos da quadrícula mais próximos do novo centro (assinalados a verde). É contada a primeira quadrícula. Os cantos são ordenados em relação ao centro através do ângulo da recta definida por cada um dos cantos e o centro. É calculado um novo centro para a iteração seguinte nas quatro quadrículas adjacentes (marcados a roxo). Esta nova posição

consiste em somar a norma do vector definido por cantos opostos da quadrícula ao centro utilizado na iteração anterior.

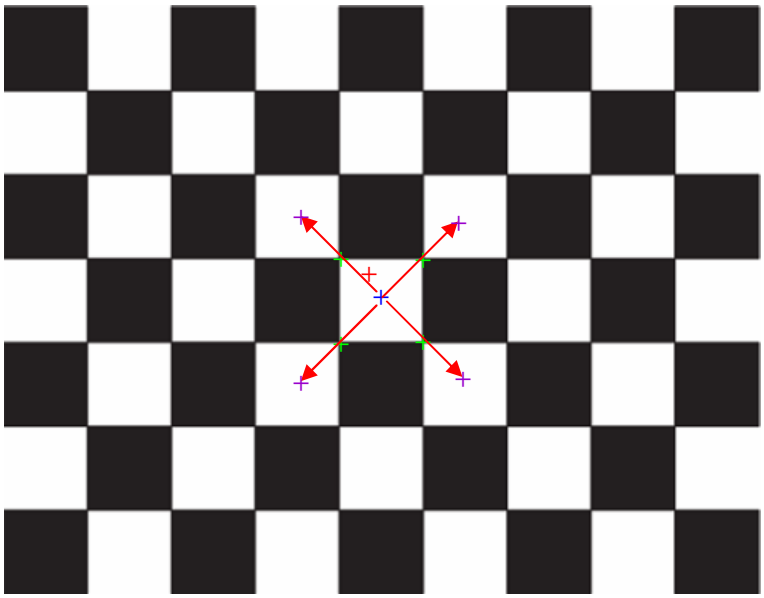

Figura 4.13: Primeira iteração do processo de contagem de quadrículas.

De seguida, passa-se à segunda iteração. A figura 4.14 esquematiza a segunda iteração do processo de contagem. Neste caso, é recalculada a posição do novo centro (a azul) com base nos três pontos mais próximos do centro calculado na iteração anterior (a roxo). Calculam-se os quatros cantos da quadrícula mais próximos do novo centro (assinalados a verde). Os cantos são ordenados em relação ao centro através do ângulo da recta definida por cada um dos cantos e o centro. É calculado um novo centro para a iteração seguinte na quadrícula adjacente (marcados a vermelho), utilizando a norma do vector definido por cantos opostos da quadrícula. Neste momento, já se contam três quadrículas em cada direcção.

Na terceira iteração, o procedimento será igual ao da segunda iteração dois e no final, têm contadas cinco quadrículas em cada direcção. E assim sucessivamente. A expansão irá terminar quando já não for possível determinar mais quadrículas em nenhuma direcção.

Este método de contagem já foi estudado, mas ainda não foi totalmente implementado. Em [12] foi descrito um método para estimar a homografia entre uma grelha planar e uma imagem catadióptrica, a partir de 12 pontos clicados na imagem. Após a ser efectuada a contagem de quadriculas que foi proposta, é possível calcular a homografia através de método proposto em [12] e proceder à calibração do artroscópio.

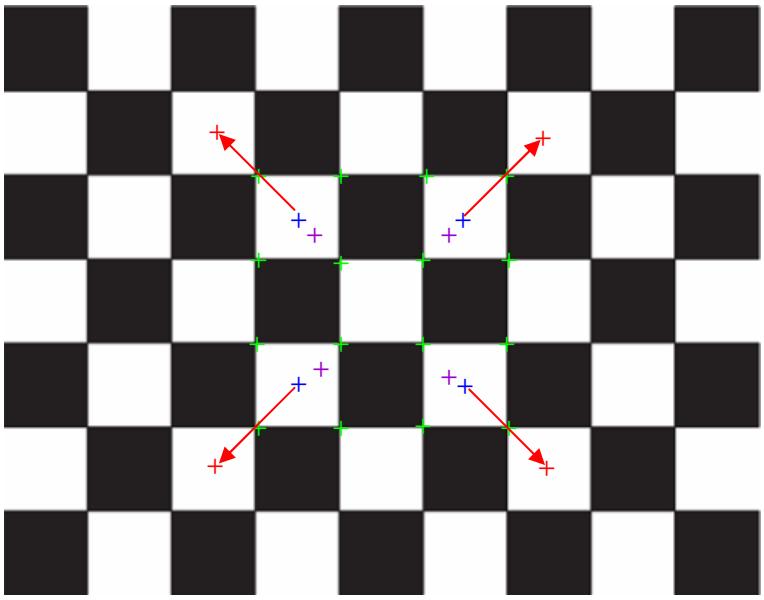

Figura 4.14: Segunda iteração do processo de contagem de quadrículas.

4.6 Conclusões

Neste capítulo foi definido um método de detecção de cantos baseado na entropia dos ângulos do gradiente, bem como um método de contagem de quadrículas. Embora seja necessário tornar estes métodos mais robustos, verifica-se que são uma boa aproximação para abordar o problema da detecção dos cantos da grelha que se pretende resolver, de modo a conseguir-se efectuar a calibração da câmara.

Capítulo 5: Considerações Finais

Ao longo deste ano de trabalho foram adquiridos conhecimentos em diversas áreas, o que fez com que este tenha contribuído para uma melhor formação a nível pessoal e profissional.

Para a realização deste projecto, foram propostos dois objectivos como já foi referido. O primeiro objectivo foi bem sucedido. Conseguiu-se implementar um método de segmentação da zona útil das imagens artroscópicas robusto, estável e rápido, o que permite a sua implementação em tempo real. Importa no entanto referir que a realização do white-set ao artroscópio é fundamental para o bom desempenho do algoritmo e a sua má execução pode levar a que o algoritmo falhe. Como se verificou, este método pode falhar para vídeos de cenas do mundo, dado que os histogramas não têm uma forma padrão que se ajuste ao modelo.

Na realização do segundo objectivo, na detecção de cantos da grelha conseguiram-se bons resultados. Obteve-se uma boa precisão, até em imagens com muita distorção radial, ao contrário de alguns algoritmos já existentes. No entanto, será necessário ajustar o algoritmo de detecção, de modo a torná-lo mais estável e robusto. Por último, desenvolveu-se uma solução para proceder à contagem das quadrículas da grelha de calibração. No entanto, esta ainda não foi totalmente implementada na prática, mas esta poderá uma solução possível para resolver este problema.

Assim, algum do trabalho futuro que pode ser realizado no âmbito deste projecto passa por terminar a implementação da contagem de quadrículas. E tornar todo o algoritmo de detecção mais estável e robusto, de modo a que este possa ser útil para se proceder à calibração da câmara artroscópica.

Anexos

A. Esquema de funcionamento do algoritmo de segmentação implementado em C

A seguinte figura esquematiza o funcionamento do algoritmo de segmentação implementado em C. Do lado direito, estão escritas as principais funções criadas para cada uma das etapas da segmentação.

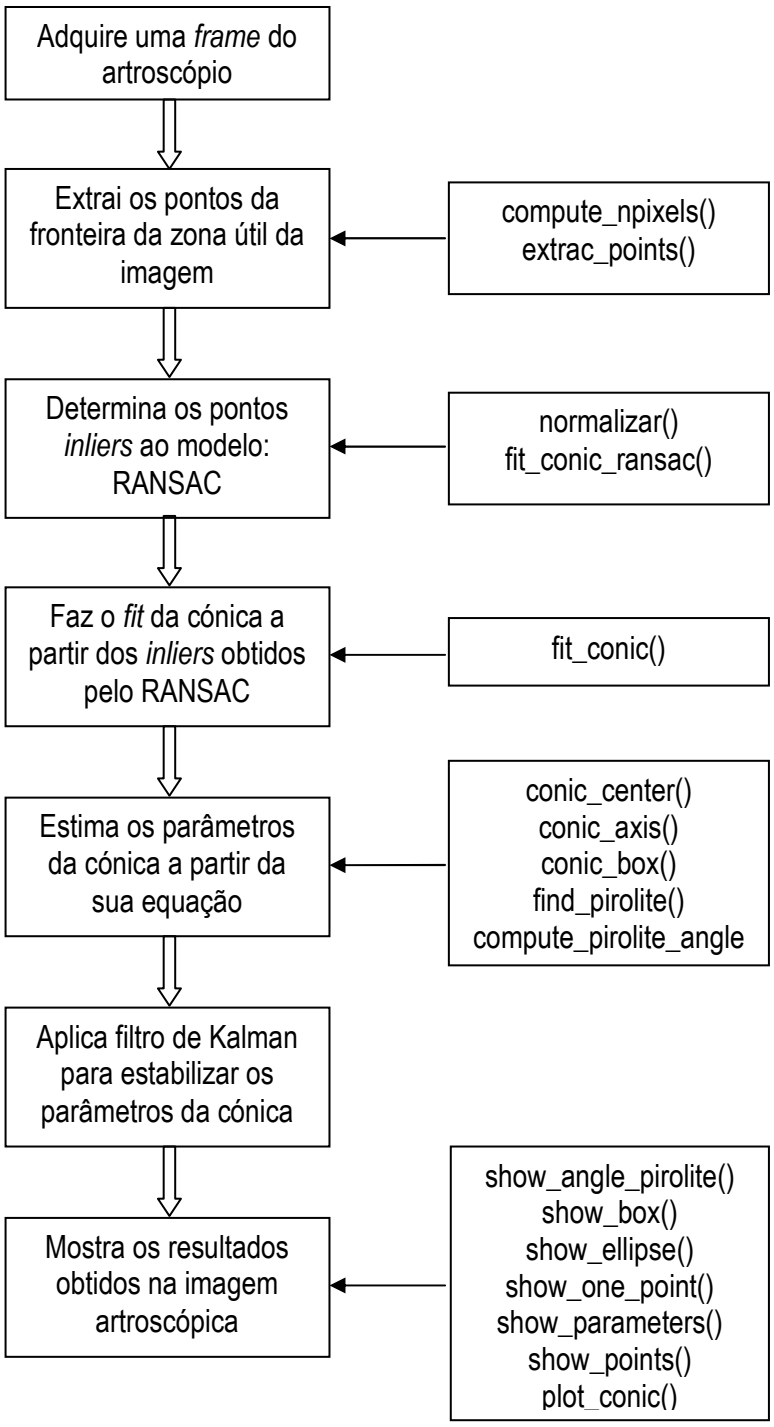

Figura A.1: Esquema do funcionamento do algoritmo de segmentação implementado em C.

B. Configuração do algoritmo de segmentação: *configuracao.h*

O utilizador tem a possibilidade de controlar alguns parâmetros do algoritmo de segmentação. O cabeçalho *configuracao.h* tem um conjunto de *flags*. O utilizador escolhe quais pretende usar. A seguir, vão ser apresentadas as *flags* possíveis, bem como a sua funcionalidade.

#define DEBBUG

Define o modo de visualização dos resultados: caso esteja definida esta *flag*, são mostrados os parâmetros da cónica, o ângulo da marca da lente, a cónica, os pontos *inliers* e os *outliers* estimados no RANSAC e a box da cónica. Caso contrário, apenas mostra a cónica e o ângulo da marca da lente.

#define KALMAN

O utilizador define se pretende utilizar o filtro de Kalman no algoritmo de segmentação ou não. Caso pretenda, deve definir esta *flag*.

```
#define VIDEO
#ifdef VIDEO
#define FICHEIRO "/home/Hugo/Desktop/arthrovideo.avi"
#endif
```

Esta *flag* refere-se ao vídeo utilizado. Caso não seja definida, o vídeo adquirido é o vídeo proveniente do artroscópio. Caso esteja definida, deve definir-se a *flag* FICHEIRO com a localização do vídeo a partir do qual se pretende fazer a aquisição.

```
#define REC_VIDEO_INICIAL
#ifdef REC_VIDEO_INICIAL
#define VIDEO_INICIAL "/home/hugo/Desktop/test_inicial.avi"
#endif
```

O utilizador deve definir esta *flag* caso pretenda gravar o vídeo adquirido (sem que estejam mostrados os resultados da segmentação). Caso esta esteja definida, deve definir-se a *flag* VIDEO_INICIAL com a localização do ficheiro para o qual se vai gravar o vídeo.

```
#define REC_VIDEO_FINAL
#ifdef REC_VIDEO_FINAL
#define VIDEO_FINAL "/home/hugo/Desktop/test_final.avi"
#endif
```

Caso o utilizador pretenda gravar o vídeo adquirido com os resultados da segmentação mostrados, deve definir esta *flag*. Neste caso, deve definir-se também a *flag* VIDEO_FINAL com a localização do ficheiro para o qual se vai gravar o vídeo.

```
#define RESULTADOS
#ifdef RESULTADOS
#define TXT_RESULTADOS "/home/hugo/Desktop/resultados_tensao.txt"
#endif
```

Caso esteja definida, parâmetros da cónica (posição do centro, comprimentos dos semi-eixos, ângulo de rotação da cónica e ângulo de rotação da marca da lente) são guardados num ficheiro. Definindo esta *flag*, deve também definir-se a *flag* TXT_RESULTADOS com a localização do ficheiro para o qual se vão gravar os valores dos parâmetros da cónica.

```
#define REC_IMAGES_INICIAL
#ifdef REC_IMAGES_INICIAL
#define DEST_IMAGES_INICIAL "/home/hugo/Desktop/imagens_gravadas/imagem"
#define FORMAT_IMAGE_INICIAL "tif"
#endif
```

Esta *flag* deve ser definida no caso do utilizador pretender guardar as imagens adquiridas (sem que estejam mostrados os resultados da segmentação). Caso esta esteja definida, deve definir-se a *flag* DEST_IMAGES_INICIAL com a localização dos ficheiros para os quais se vão gravar as imagens e a *flag* FORMAT_IMAGE_INICIAL com o formato no qual se pretende que as imagens sejam guardadas.

```
#define REC_IMAGES_FINAL
#ifdef REC_IMAGES_FINAL
#define DEST_IMAGES_FINAL "/home/hugo/Desktop/imagens_gravadas/imagem_final"
#define FORMAT_IMAGE_FINAL "bmp"
#endif
```

Esta *flag* deve ser definida no caso do utilizador pretender guardar as imagens com os resultados da segmentação mostrados. Caso esta esteja definida, deve definir-se a *flag* DEST_IMAGES_FINAL com a localização dos ficheiros para os quais se vão gravar as imagens e a *flag* FORMAT_IMAGE_FINAL com o formato no qual se pretende que as imagens sejam guardadas.

C. Documentação das funções criadas em C

Neste anexo, estão documentadas as funções que foram criadas em C. A documentação aqui escrita foi gerada automaticamente pelo Doxygen.

void compute_npixels (float* *percent_pixels*, IplImage * *Img*)

Calcula a percentagem de pixeis de fundo numa imagem de uma superfície branca.

Parâmetros:

percent_pixels - Percentagem de pixeis de fundo na imagem
Img - Imagem artroscópica de uma superfície branca

void compute_pirolite_angle (float * *angle*, float * *angle_rad*, CvPoint2D32f *Pirolito*, CvPoint2D32f *coordenadas_centro*)

Calcula o ângulo da marca da lente artroscópica.

Parâmetros:

Pirolito - Ponteiro para uma estrutura do tipo *CvPoint2D32f* com as coordenadas x e y da marca da lente
pontos - Conjunto de dados do tipo *vector<CvPoint2D32f>* com as coordenadas x e y dos pontos
result - Vector do tipo *Matrix* com seis elementos com os parâmetros da equação da cónica

void conic_axis(float * *eixo_maior*, float * *eixo_menor*, float * *angulo*, Matrix *result*)

Calcula o comprimento dos semi-eixos e o ângulo de rotação de uma cónica.

Parâmetros:

eixo_maior,eixo_menor - Ponteiro para uma variável do tipo *float* com o comprimento dos semi-eixos maior e menor, respectivamente
angulo - Ponteiro para uma variável do tipo *float* com o ângulo de rotação da cónica
result - Vector do tipo *Matrix* com seis elementos com os parâmetros da equação da cónica

void conic_box (CvPoint2D32f * *Ponto1*, CvPoint2D32f * *Ponto2*, Matrix *result*, IplImage * *Img*)

Calcula as coordenadas do canto superior esquerdo e do canto inferior direito da box da cónica.

Parâmetros:

Ponto1,Ponto2 - Ponteiro para uma estrutura do tipo *CvPoint2D32f* com as coordenadas x e y do canto superior esquerdo e do canto inferior direito da box da cónica, respectivamente
result - Vector do tipo *Matrix* com seis elementos com os parâmetros da equação da cónica
Img - Imagem utilizada para encontrar os pontos para estimar a cónica

void conic_center (CvPoint2D32f * *coordenadas_centro*, Matrix *result*)

Calcula o centro da cónica.

Parâmetros:

coordenadas_centro - Ponteiro para uma estrutura do tipo *CvPoint2D32f* com as coordenadas x e y do centro da cónica

result - Vector do tipo *Matrix* com seis elementos com os parâmetros da equação da cónica

void convert_Matrix (Matrix &*mat_pontos*, vector< CvPoint2D32f> *Pontos*)

Converte um conjunto de dados do tipo *vector<CvPoint2D32f>* para uma matriz do tipo *Matrix*.

Parâmetros:

mat_points - Retorna um ponteiro para uma matriz do tipo *Matrix* (compatível com o tipo de dados utilizado noutras funções)

Pontos - Conjunto de dados do tipo *vector<CvPoint2D32f>* com as coordenadas x e y dos pontos de estimação da cónica

void distancias (vector< CvPoint2D32f > & *inliers*, Matrix *result*, Matrix *mat_pontos*, float *t*)

Classificação de um conjunto de pontos como *inliers* ou *outliers*, de acordo com um valor de *threshold*.

Parâmetros:

inliers - Conjunto de dados do tipo *vector<CvPoint2D32f>* com as coordenadas x e y dos pontos que foram classificados como *inliers*

result - Vector do tipo *Matrix* com seis elementos com os parâmetros da equação da cónica

mat_points - Matriz de tamanho 2xn: a primeira linha tem as coordenadas x e a segunda as coordenadas y dos pontos

t - valor de *threshold*

void extrac_points (vector< CvPoint2D32f > & *Pontos*, IplImage * *Img*, float *npixels*)

Extrai os pontos da fronteira entre a zona útil e o fundo de uma imagem artroscópica.

Parâmetros:

Pontos - Ponteiro para uma estrutura do tipo *vector<CvPoint2D32f>* com as coordenadas x e y dos pontos na fronteira

Img - Imagem artroscópica

npixels - Percentagem de pixeis de fundo da imagem

void find_pirolite (CvPoint2D32f * *Pirolito*, vector< CvPoint2D32f > *pontos*, Matrix *result*)

Calcula as coordenadas da marca da lente em imagens artroscópicas.

Parâmetros:

Pirolito - Ponteiro para uma estrutura do tipo *CvPoint2D32f* com as coordenadas x e y da

marca da lente

pontos - Conjunto de dados do tipo *vector<CvPoint2D32f>* com as coordenadas x e y dos pontos

result - Vector do tipo *Matrix* com seis elementos com os parâmetros da equação da cónica

void fit_conic (Matrix & *result*, float & *AproxError*, Matrix *mat_pontos*, Matrix *K*)

Calcula os parâmetros da equação da cónica ajustada por um conjunto de pontos.

Parâmetros:

result - Vector do tipo *Matrix* com seis elementos com os parâmetros da equação da cónica

AproxError - Ponteiro para uma variável do tipo *float* com o valor do erro obtido na estimativa

mat_points - Matriz de tamanho 2xn: a primeira linha tem as coordenadas x e a segunda as coordenadas y dos pontos

K - Matriz de normalização dos dados

void fit_conic_ransac (vector< CvPoint2D32f > & *bestinliers*, Matrix & *bestmodel*, vector< CvPoint2D32f > *Pontos*, int *s*, float *t*, int *maxDataTrials*, int *maxTrials*)

Calcula os parâmetros da equação da cónica ajustada por um conjunto de pontos usando a estimação de RANSAC.

Parâmetros:

final_inliers - Ponteiro para um vector do tipo *vector<CvPoint2D32f>* com as coordenadas x e y dos pontos que foram classificados como *inliers* no melhor modelo obtido.

bestmodel - Vector do tipo *Matrix* com seis elementos com os parâmetros da equação da cónica estimada no melhor modelo (modelo com mais *inliers*)

Pontos - Conjunto de dados do tipo *vector<CvPoint2D32f>* com as coordenadas x e y dos pontos

s - Número mínimo de pontos que são necessários para estimar o modelo (no caso da cónica, s=5)

t - Valor de *threshold* utilizado na classificação dos pontos como *inliers* ou *outliers*

maxDataTrials - Número de tentativas para determinar um modelo não degenerado

maxTrials - Número de tentativas para determinar o melhor modelo

void normalizar (Matrix & *K*, Matrix *mat_pontos*)

Calcula a matriz de normalização de dados.

Parâmetros:

Pontos - Ponteiro para uma matriz de normalização do tipo *Matrix*

mat_pontos - Matriz de tamanho 2xn: a primeira linha tem as coordenadas x e a segunda as coordenadas y dos pontos

void plot_conic (IplImage * *Img*, Matrix *result*, int *R*, int *G*, int *B*)

Desenha a cónica numa imagem, utilizando os parâmetros da sua equação.

Parâmetros:

Img - Ponteiro para a imagem onde se pretende desenhar a cónica
result - Vector do tipo *Matrix* com seis elementos com os parâmetros da equação da cónica
R,G,B - Níveis de cor

void return_conic (Matrix & *result*, CvPoint2D32f *centro*, float *eixo_maior*, float *eixo_menor*, float *angulo*)

Calcula a equação de uma cónica a partir dos seus parâmetros (comprimento dos semi-eixos, centro e ângulo de rotação da cónica).

Parâmetros:

result - Ponteiro para um vector do tipo *Matrix* com seis elementos com os parâmetros da equação da cónica
centro - Estrutura do tipo *CvPoint2D32f* com as coordenadas x e y do centro da cónica
eixo_maior - Comprimento do semi-eixo maior da cónica
eixo_menor - Comprimento do semi-eixo menor da cónica
angulo - Ângulo de rotação da cónica

void show_angle_pirolite (IplImage * *Img*, float *angulo*)

Mostra o ângulo de rotação da marca da lente.

Parâmetros:

Img - Ponteiro para a imagem onde se pretende escrever o ângulo da marca da lente
angulo - Ângulo da marca da lente

void show_box (IplImage * *Img*, CvPoint2D32f *Ponto1*, CvPoint2D32f *Ponto2*)

Desenha a box da cónica numa imagem.

Parâmetros:

Img - Ponteiro para a imagem onde se pretende desenhar a box da cónica
Ponto1,Ponto2 - Estrutura do tipo *CvPoint2D32f* com as coordenadas x e y do canto superior esquerdo e do canto inferior direito da box da cónica, respectivamente

void show_ellipse (IplImage * *Img*, float *angulo*, float *maior*, float *menor*, CvPoint2D32f *coordenadas_centro*, int *R*, int *G*, int *B*)

Desenha a cónica numa imagem, utilizando os seus parâmetros.

Parâmetros:

Img - Ponteiro para a imagem onde se pretende desenhar a cónica
angulo - Ângulo de rotação da cónica
maior,menor - Comprimento dos semi-eixos maior e menor da cónica, respectivamente
coordenadas_centro - Estrutura do tipo *CvPoint2D32f* com as coordenadas x e y do centro da cónica
R,G,B - Níveis de cor

void show_one_point (IplImage * *Img*, CvPoint2D32f *Ponto*, int *R*, int *G*, int *B*)

Desenha um ponto numa imagem.

Parâmetros:

Img - Ponteiro para a imagem onde se pretende desenhar o ponto

Ponto - Estrutura do tipo *CvPoint2D32f* com as coordenadas x e y do ponto que se pretende desenhar

R,G,B - Níveis de cor

void show_parameters (IplImage * *Img*, float *angulo*, float *maior*, float *menor*, CvPoint2D32f *Ponto1*, CvPoint2D32f *Ponto2*, CvPoint2D32f *coordenadas_centro*, Matrix *result*)

Escreve os parâmetros da cónica numa imagem.

Parâmetros:

Img	- Ponteiro para a imagem onde se pretende escrever os parâmetros da conica
angulo	- Ângulo de rotação da cónica
maior,menor	- Comprimento dos semi-eixos maior e menor da cónica, respectivamente
Ponto1,Ponto2	- Estrutura do tipo *CvPoint2D32f* com as coordenadas x e y do canto superior esquerdo e do canto inferior direito da box da cónica, respectivamente
coordenadas_centro	- Estrutura do tipo *CvPoint2D32f* com as coordenadas x e y do centro da cónica
result	- Vector do tipo *Matrix* com seis elementos com os parâmetros da equação da cónica

void show_points (IplImage * *Img*, vector< CvPoint2D32f > *Pontos*, int *R*, int *G*, int *B*)

Desenha um conjunto de pontos numa imagem

Parâmetros:

Img - Ponteiro para a imagem onde se pretendem desenhar os pontos

Pontos - Conjunto de dados do tipo *vector<CvPoint2D32f>* com as coordenadas x e y dos pontos que se pretendem desenhar

R,G,B - Níveis de cor

D. Documentação das funções criadas em MATLAB

Neste anexo, estão documentadas as funções que foram criadas em MATLAB

Função	*[ang,ang_rad] = cal_angle (cent,marca)*
Descrição	Calcula o ângulo da marca da lente em graus e radianos
Parâmetros de entrada	**cent:** centro da cónica **marca:** marca da lente artroscópica
Parâmetros de saída	**ang:** ângulo de rotação da marca da lente em graus **ang_rad:** ângulo de rotação da marca da lente em radianos

.

Função	*[gradient, or] = canny (im, sigma)*
Descrição	Função para determinar as arestas de uma imagem pelo método de Canny (função desenvolvida por Peter Kovesi)
Parâmetros de entrada	i**m:** imagem para ser processada **sigma:** desvio padrão do filtro gaussiano de alisamento
Parâmetros de saída	**gradient:** modulo do gradiente da imagem **or:** orientação do gradiente da imagem

Função	*[P1,P2] = conic_box (result)*
Descrição	Função para determinar o canto superior esquerdo e do canto inferior direito da box da cónica
Parâmetros de entrada	**result:** vector com os parâmetros da equação de uma cónica
Parâmetros de saída	**P1:** coordenadas do canto superior esquerdo da box da cónica **P2:** coordenadas do canto inferior direito da box da cónica

Função	*[result, AproxError, K] = conic_equation (data)*
Descrição	Faz a estimação de uma cónica a partir de um conjunto de dados
Parâmetros de entrada	**data:** conjunto de dados para o qual se pretende estimar o modelo
Parâmetros de saída	**result:** vector com os parâmetros da equação da cónica estimada **AproxError:** Valor do erro obtido na estimativa **K:** matriz de normalização dos dados

Função	*[centro,A,B,Phi] = conic_parameters (result)*
Descrição	Calcula os parâmetros de uma cónica a partir da sua equação
Parâmetros de entrada	**result:** vector com os parâmetros da equação da cónica estimada
Parâmetros de saída	**centro:** coordenadas do centro da cónica **A:** comprimento semi-eixo maior da cónica **B:** comprimento semi-eixo menor da cónica **Phi:** ângulo de rotação da cónica

Função	*[result, inliers] = conic_ransac (x, s, t)*
Descrição	Faz o ajuste de um conjunto de pontos a uma cónica, utilizando a estimação de RANSAC
Parâmetros de entrada	**x:** conjunto dos pontos para os quais se pretende efectuar a estimativa **s:** numero de pontos mínimos necessário para efectuar a estimação

	t: valor de *threshold* utilizado para efectuar a classificação dos pontos em *inliers* ou *outliers*
Parâmetros de saída	**result:** vector com os parâmetros da equação da cónica estimada **inliers:** vector com os índices dos pontos classificados como *inliers*

Função	*[xc] = corner_refine (xt,I,wintx,winty)*
Descrição	Faz o refinamento da posição dos cantos de uma grelha numa imagem.
Parâmetros de entrada	**xt:** coordenadas dos pontos que se pretendem refinar **I:** imagem sobre a qual se pretende fazer o refinamento **Wintx:** dimensão da janela de vizinhança na direcção x **Winty:** dimensão da janela de vizinhança na direcção y
Parâmetros de saída	**xc:** matriz com as novas posições dos pontos após o refinamento

Função	*[Pontos] = det_pontos (imagem)*
Descrição	Determina os pontos da fronteira entre a zona útil e o fundo da imagem artrsocópica.
Parâmetros de entrada	**imagem:** imagem artroscópica que se pretende processar
Parâmetros de saída	**Pontos:** coordenadas dos pontos da fronteira determinados

Função	*draw_hist(imagem)*
Descrição	Desenha os histogramas de cada uma das componentes de uma imagem do tipo RGB e desenha os histogramas da correspondente imagem em escala de cinzentos convertida através da fórmula do MATLAB ou através de uma media das trás componentes.
Parâmetros de entrada	imagem: imagem da qual se pretende calcular os histogramas
Parâmetros de saída	

Função	*[pixeis] = find_background_pixels (imagem)*
Descrição	Determina o número aproximado de pixeis da zona não útil da imagem artroscópica
Parâmetros de entrada	**imagem:** imagem da qual se pretende determinar o número de pixeis do fundo
Parâmetros de saída	**pixeis:** número de pixeis correspondentes à zona não útil da imagem

Função	[*boundary] = find_cont (image,tr)*
Descrição	Determina a fronteira de separação entre a zona útil e não útil da imagem através da binarização da imagem com um valor de *threshold* calculado com base no número de pixeis do fundo
Parâmetros de entrada	**imagem:** imagem da qual se pretende determinar o número de pixeis do fundo **tr:** valor de *threshold* utilizado na binarização da imagem (calculado com o numero de pixeis da zona não util da imagem)
Parâmetros de saída	**boundary:** coordenadas dos pontos da fronteira determinados

Função	*[coor] = find_pirolite(x,result)*
Descrição	Calcula as coordenadas x e y da posição da marca da lente artroscópica.
Parâmetros de entrada	**result:** vector com os parâmetros da equação da cónica estimada **x:** coordenadas dos pontos da fronteira determinados
Parâmetros de saída	**coor:** coordenadas da posição x e y da marca da lente artroscópico

Função	*[thresh] = find_tresh_level (imagem,npixels)*
Descrição	Calcula o valor de *threshold* baseado no número de pixeis da zona não útil da imagem, para que este seja aplicado na função *find_cont*.
Parâmetros de entrada	**imagem:** imagem da qual se pretende determinar os pontos da fronteira **npixels:** numero de pixeis pertecnentes à zona não util da imagem.
Parâmetros de saída	**thresh:** valor de threshold a aplicar para binarizar a imagem, na determinação dos pontos da fronteira.

Função	*[cx,cy] = harris_detector (imagem,sigma)*
Descrição	Determina a posição dos cantos de um grelha de calibração pelo método de detecção de Harris.
Parmetros de entrada	**imagem:** imagem da qual se pretende determinar as posições dos cantos da grelha de calibração **sigma:** valor do desvio padrão do filtro gaussiano
Parâmetros de saída	**cx:** vector com a coordenada x das posições dos cantos determinados **cy:** vector com a coordenada y das posições dos cantos determinados

Função	*[bw] = hysthresh (im, T1, T2)*
Descrição	Aplica um threshold com histerese a uma imagem. Todos os pixeis com valores superiores a um valor T1 são marcados como arestas. Todos os pixeis que estão conectados com pontos que tenham sido marcados como arestas e tenham valor superior a T1, também são marcados como arestas (função desenvolvida por Peter Kovesi).
Parâmetros de entrada	**im:** imagem à qual se pretende aplicar o *threshold* com histerese **T1:** valor de *threshold* superior **T1:** valor de *threshold* inferior
Parâmetros de saída	**bw:** imagem obtida após ser aplicado o *threshold*

Função	*[points,img] = plot_conic_curve (C,img,h,color)*
Descrição	Função para desenhar um cónica numa imagem.
Parâmetros de entrada	**C:** vector com os parâmetros da equação da cónica que se pretende desenhar **img:** imagem onde se pretende desenhar a cónica **h:** sistema de eixos utilizado **color:** vector com os níveis de cor com que se pretende desenhar a conica
Parâmetros de saída	**points:** pontos definidos para desenhar a cónica. Estes pontos pertencem à cónica. **img:** imagem onde foi desenhada a cónica

Função	*[result] = return_conic(center,A,B,Phi)*
Descrição	Determina a equação de uma cónica a partir dos seus parâmetros (centro, comprimento dos semi-eixos e ângulo de rotação)
Parâmetros de entrada	**center:** coordenadas do centro da cónica **A:** comprimento do semi-eixo maior da cónica **B:** comprimento do semi-eixo menor da cónica **Phi:** ângulo de rotação da cónica
Parâmetros de saída	**result:** vector com os parâmetros da equação da cónica que se obteve

Bibliografia

[1] P. D. Gusmao, *Artroscopia*, disponível em http://www.iof.com.br/int_default.php?p=artigos/art_artroscopia, consultado no dia 18 de Agosto de 2008

[2] *ArthroNav - Computer Assisted Navigation in Orthopedic Surgery using Endoscopic Images*, disponível em http://www.deec.uc.pt/~jpbar/ArthroNav, consultado no dia 18 de Agosto de 2008

[3] J. P. Barreto, *General Central Projection Systems Modeling, Calibration and Visual Servoing*, PhD Thesis, Coimbra, 2003

[4] P. Menezes, *Multi-Cue Visual Tracking for Human-Robot Interaction*, PhD Thesis, Coimbra, 2007

[5] A. Fitzgibbon, M Pilu, and R. Fisher. *Direct least square fitting of ellipses*. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 21(5), May 1999

[6] H. Cantzler, Random Sample Consensus (RANSAC), disponível em , consultado no dia 19 de Fevereiro de 2008

[7] G. Welch, G. Bishop, *An Introduction to the Kalman Filter*, disponível em http://www.cs.unc.edu/~welch/kalman/kalmanIntro.html, consultado no dia 18 de Junho de 2008

[9] K. G. Derpanis, *The Harris Corner Detector, disponível em* www.cse.yorku.ca/~kosta/CompVis_Notes/**harris_detector**.pdf, consultado no dia 22 de Julho de 2008

[8] L. P. Ribeiro, *CellRuller II – Medição de Espaços e Objectos Usando o Telemóvel*, Relatório de Licenciatura, Coimbra, 2007

[10] Edge detection, disponível em http://en.wikipedia.org/wiki/Edge_detection, consultado no dia 18 de Agosto de 2008.

[11] Y. Ma, S. Soatto, J. Kosecka, S. Sastry, An Invitation to 3-D Vision: From Images to Geometric Models, Springer, 2004

[12] Anonymous ECCV submission, *General Catadioptric Imaging Geometry*, 2008

[13] D. A. Forsyth, J. Ponce, *Computer Vision: A Modern Approach*, Prentice Hall, 2002

[14] R. Hartley e A. Zisserman. *Multiple View Geometry*. Cambridge University Press, 2003

[15] Jean-Yves Bouguet. *Camera Calibration Toolbox for Matlab*, disponível em http://www.vision.caltech.edu/bouguetj/calib_doc/, consultado em 15 de Outubro de 2007

[16] Intel - OpenCV (Open Source Computer Vision Library), disponível em http://www.intel.com/technology/computing/opencv/, consultado em 15 de outubro de 2007 de outubro de 2007