Pontifícia Universidade Católica
do Rio de Janeiro

Sasha Nícolas da Rocha Pinheiro

Calibração de câmera usando projeção frontal-paralela e colinearidade dos pontos de controle

Dissertação de Mestrado

Dissertação apresentada como requisito parcial para obtenção do grau de Mestre pelo Programa de Pós–graduação em Informática do Departamento de Informática da PUC–Rio

Orientador: Prof. Alberto Barbosa Raposo

Rio de Janeiro
Setembro de 2016

Pontifícia Universidade Católica
do Rio de Janeiro

Sasha Nícolas da Rocha Pinheiro

Calibração de câmera usando projeção frontal-paralela e colinearidade dos pontos de controle

Dissertação apresentada como requisito parcial para obtenção do grau de Mestre pelo Programa de Pós–graduação em Informática do Departamento de Informática do Centro Técnico Científico da PUC–Rio. Aprovada pela Comissão Examinadora abaixo assinada.

Prof. Alberto Barbosa Raposo
Orientador
Departamento de Informática — PUC–Rio

Prof. Marcelo Gattass
Departamento de Informática — PUC-Rio

Prof. Raul Queiroz Feitosa
Departamento de Engenharia Elétrica — PUC-Rio

Manuel Eduardo Loaiza Fernandéz
Departamento de Informática — PUC-Rio

Prof. Márcio da Silveira Carvalho
Coordenador Setorial do Centro Técnico Científico — PUC–Rio

Rio de Janeiro, 1 de Setembro de 2016

Sasha Nícolas da Rocha Pinheiro

Graduou-se no curso de Bacharelado em Ciência da Computação pela Universidade Federal do Maranhão em 28 de Janeiro de 2014.

Ficha Catalográfica

Pinheiro, Sasha Nicolas da Rocha

Calibração de câmera usando projeção frontal-paralela e colinearidade dos pontos de controle / Sasha Nícolas da Rocha Pinheiro; orientador: Prof. Alberto Barbosa Raposo. — Rio de Janeiro : PUC–Rio, Departamento de Informática, 2016.

v., 63 f: il. ; 29,7 cm

1. Dissertação (mestrado) - Pontifícia Universidade Católica do Rio de Janeiro, Departamento de Informática.

Inclui referências bibliográficas.

1. Informática – Tese. 2. Calibração de Câmera. 3. Refinamento Iterativo. 4. Padrão Aneliforme. 5. Segmentação Adaptativa. 6. Erro de Colinearidade. 7. Otimização paramétrica. 8. Visão Computacional. I. Raposo, Alberto Barbosa. II. Pontifícia Universidade Católica do Rio de Janeiro. Departamento de Informática. III. Título.

CDD: 004

Agradecimentos

É com assaz regozijo que demonstro meus agradecimentos:

Ao professor Alberto, meu orientador, que me recebeu no seu grupo de trabalho e posteriormente me acolheu como orientando.

Ao Manuel, meu coorientador, cujos ombros me permitiram ver além.

Aos meus amigos do Tecgraf, do Grupo de Realidade Virtual e do Grupo de Geofísica Computacional.

Aos professores do Departamento de Informática, que contribuíram para minha formação.

À CAPES, à PUC-Rio e ao Tecgraf, pelo auxílios concedidos, sem os quais este trabalho não poderia ser realizado.

Resumo

Pinheiro, Sasha Nicolas da Rocha; Raposo, Alberto Barbosa. **Calibração de câmera usando projeção frontal-paralela e colinearidade dos pontos de controle**. Rio de Janeiro, 2016. 63p. Dissertação de Mestrado — Departamento de Informática, Pontifícia Universidade Católica do Rio de Janeiro.

Imprescindível para quaisquer aplicações de visão computacional ou realidade aumentada, a calibração de câmera é o processo no qual se obtém os parâmetros intrínsecos e extrínsecos da câmera, tais como distância focal, ponto principal e valores que mensuram a distorção ótica da lente. Atualmente o método mais utilizado para calibrar uma câmera envolve o uso de imagens de um padrão planar em diferentes perspectivas, a partir das quais se extrai pontos de controle para montar um sistema de equações lineares cuja solução representa os parâmetros da câmera, que são otimizados com base no erro de reprojeção 2D. Neste trabalho, foi escolhido o padrão de calibração aneliforme por oferecer maior precisão na detecção dos pontos de controle. Ao aplicarmos técnicas como transformação frontal-paralela, refinamento iterativo dos pontos de controle e segmentação adaptativa de elipses, nossa abordagem apresentou melhoria no resultado do processo de calibração. Além disso, propomos estender o modelo de otimização ao redefinir a função objetivo, considerando não somente o erro de reprojeção 2D, mas também o erro de colinearidade 2D.

Palavras–chave

Calibração de Câmera; Refinamento Iterativo; Padrão Aneliforme; Segmentação Adaptativa; Erro de Colinearidade; Otimização paramétrica; Visão Computacional;

Abstract

Pinheiro, Sasha Nicolas da Rocha; Raposo, Alberto Barbosa (advisor). **Camera calibration using fronto parallel projection and collinearity of control points**. Rio de Janeiro, 2016. 63p. MSc Dissertation — Departamento de Informática, Pontifícia Universidade Católica do Rio de Janeiro.

Crucial for any computer vision or augmented reality application, the camera calibration is the process in which one gets the intrinsics and the extrinsics parameters of a camera, such as focal length, principal point and distortions values. Nowadays, the most used method to deploy the calibration comprises the use of images of a planar pattern in different perspectives, in order to extract control points to set up a system of linear equations whose solution represents the camera parameters, followed by an optimization based on the 2D reprojection error. In this work, the ring calibration pattern was chosen because it offers higher accuracy on the detection of control points. Upon application of techniques such as fronto-parallel transformation, iterative refinement of the control points and adaptative segmentation of ellipses, our approach has reached improvements in the result of the calibration process. Furthermore, we proposed extend the optimization model by modifying the objective function, regarding not only the 2D reprojection error but also the 2D collinearity error.

Keywords

Camera Calibration; Iterative Refinement; Ring Pattern; Adaptative Segmentation; Collinearity Error; Parameter Optimization; Computer Vision;

Sumário

Lista de figuras

Lista de Algoritmos

1
Introdução

Câmeras estão em todas as partes, em todos os lugares, muitas ubíquas, outras tantas estão na palma da nossa mão. As imagens geradas por essas câmeras, no entanto, carregam em si uma deformação muitas vezes sutil, imperceptível. Deformação essa que é inócua para uma *selfie* em uma rede social ou para um vídeo de vigilância. Entretanto, em se tratando de aplicações que requerem maior precisão, como as relacionadas com realidade aumentada e visão computacional, a menor perturbação pode ensejar um problema indesejável (Azuma *et al.*, 2001).

Essas deformações são inerentes à coisa material, às imperfeições do objeto físico e das lentes, nesse caso, ao dispositivo criado para a captura da imagem digital: a câmera. Por isso, à imagem capturada por tais dispositivos são inseridas as incertezas dos materiais e as imprecisões do processo de produção do próprio dispositivo. As principais deformações visualizadas em uma imagem são as distorções da lente e a distorção causada pelo posicionamento não perfeitamente paralelo da lente e do sensor ótico, que são conhecidas por distorção radial e tangencial (Weng *et al.*, 1992).

Na Figura 1.1 são mostrados exemplos de tipos de distorção radial e, na Figura 1.2, como o posicionamento incorreto da lente e do sensor ótico pode infligir deformação na imagem.

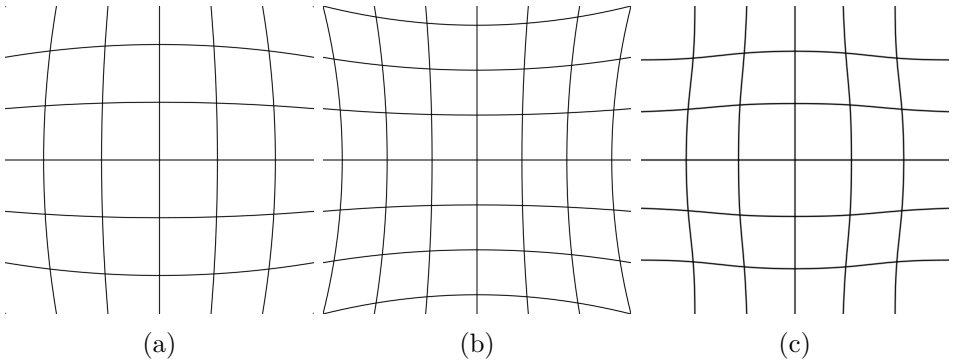


Figura 1.1: Exemplos de distorção radial: **(a)** *barrel*, **(b)** *pincushion* e **(c)** *mustache*.

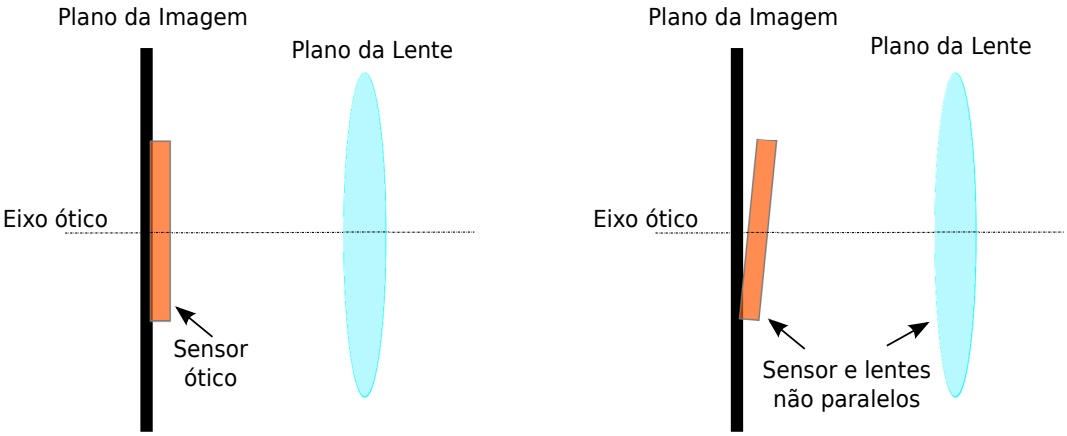


Figura 1.2: Imprecisão do posicionamento das lentes em relação ao leitor ótico causa distorção tangencial. Á esquerda: esquema ideal de posicionamento paralelo entre lente e sensor; à direita: esquema real, quando há inclinação entre lente e sensor ótico.

Além da necessidade de corrigir essas distorções, certas aplicações exigem também a capacidade de associar a cada pixel de uma imagem uma posição no mundo. Ou seja, faz-se necessário identificar uma transformação que converte posições do mundo para posição na imagem e vice-versa, de forma eficiente, com menor erro possível. É para isso que a calibração da câmera é executada.

Calibrar uma câmera significa identificar valores de parâmetros que constituem o modelo da câmera (Sturm *et al.*, 2011). A maioria das câmeras funciona semelhantemente ao modelo estenopeico, ou *pin-hole* [1], no qual a luz atravessa um pequeno orifício e gera uma imagem invertida no outro lado, no plano da imagem. Como é possível ver na Figura 1.3, o ponto P é projetado no plano da imagem no ponto Q, numa inversão em relação o eixo $x1$ e $x2$.

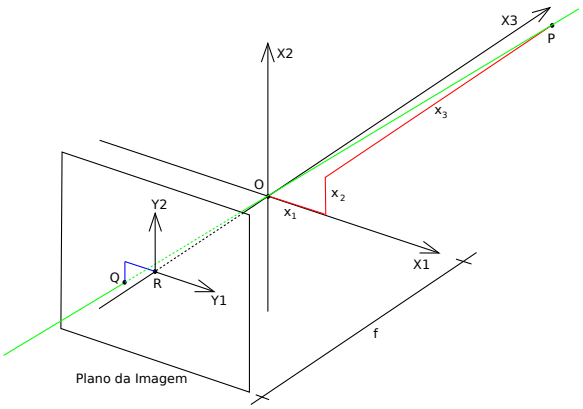


Figura 1.3: Geometria projetiva de uma câmera *pinhole*.

[1]Em tradução livre: buraco de alfinete; em analogia às câmeras primitivas sem lente em que se tem um papel fotográfico dentro de uma caixa fechada que permite a entrada de luz por um orifício com tamanho relativo à cabeça de um alfinete a fim de registrar a imagem do exterior.

Esses parâmetros que definem o modelo virtual da câmera *pinhole* podem ser intrínsecos ou extrínsecos. A distância focal, o tamanho do pixel, o ponto central da imagem, a posição da câmera no mundo são exemplos dos parâmetros de uma câmera e ajudam a descrever como uma cena 3D é projetada numa imagem 2D. São esses parâmetros que devem ser achados para se ter uma câmera calibrada.

Calibrar uma câmera é uma etapa imprescindível para aplicações que utilizam imagens para extrair informações do ambiente, ou que envolvam realidade aumentada onde objetos gráficos devam ser posicionados com precisão. Uma câmera mal calibrada gera erros nas aplicações que podem ser deveras prejudiciais. Definir abordagens e métodos que melhorem ainda mais a calibração de câmera é uma área de pesquisa de demasiada importância.

1.1 Motivação

Nos útlimos anos, tem surgido muito interesse na área de realidade aumentada. Nela, imagens de vídeo são a base para se aumentar a realidade do mundo físico inserindo elementos gráficos criados por computador. Para isso, aplicações dessa área precisam calibrar suas câmeras para funcionar com precisão, pois a calibração é útil para remover algumas distorções óticas e para posicionar elementos corretamente na imagem. Sendo assim, toda aplicação deveria realizar previamente a calibração de câmera.

Existem diversas formas de calibrar uma câmera, por exemplo temos a auto-calibração e a calibração por fotogrametria. Na primeira, não há uso de objetos de calibração. A câmera é movida por uma cena e pontos em comum são usados para extrair os parâmetros do modelo da câmera. Na segunda, são capturadas imagens de um objeto de calibração com propriedades conhecidas em diferentes posições e a detecção de pontos de controle presente nesse objeto é utilizada para calcular os parâmetros da câmera.

A calibração por fotogrametria era considerada complicada de implementar, até Zhengyou Zhang em (Zhang, 2000) apresentar um método simples e pioneiro utilizando somente um objeto de calibração planar, doravante padrão de calibração, porém confiável e eficiente. Esse método foi escolhido nesta dissertação como objeto de estudo.

Figura 1.4: Exemplo do padrão de tabuleiro de xadrez. Fonte: (Bouguet, 2014).

Zhang utilizou um padrão de calibração muito conhecido semelhante a um tabuleiro de xadrez (Figura 1.4). Seu método consiste em coletar imagens desse padrão movendo a câmera ou o próprio padrão para depois associar cada intersecção das casas do tabuleiro em cada uma das imagens e montar um sistema de equações lineares cuja solução consistia no conjunto de parâmetros da câmera.

Analisando algumas tentativas de melhorar a calibração, surgiu a proposta do presente trabalho, que consiste em unificar os métodos do estado da arte que demostraram evolução da calibração, tais como o refinamento iterativo dos pontos de controle (Datta *et al.*, 2009) e a segmentação adaptativa das elipses (Prakash e Karam, 2012), assim como a proposta de inserir nesses métodos o uso do padrão de calibração aneliforme em detrimento do tabuleiro de xadrez, haja vista seu melhor desempenho nos resultados mostrados por Ankur Datta em (Datta *et al.*, 2009).

1.2 Objetivo

1.2.1 Objetivos Gerais

O objetivo desse trabalho consiste em criar um método preciso de calibração de câmera unificando os métodos que apresentam melhores resultados em um estado da arte, assim como avaliar a adição do uso do erro

de colinearidade 2D como função objetivo à otimização dos parâmetros da câmera.

Os métodos que melhoram a abordagem de Z. Zhang apostam na premissa de que a localização dos pontos de controle do padrão deve ser feita em uma visão frontal do padrão de calibração, contrariando o que ele havia feito. Além disso, eles avaliam o uso de outros padrões como o circular e o aneliforme. Ambos são considerados mais eficazes pois a detecção de círculos e elipses sofrem menos influência do posicionamento do padrão na imagem do que a detecção de retas e dos cantos onde elas se intersectam. Aqui pretende-se unificar alguns métodos, utilizar o padrão aneliforme, avaliar combinações desses métodos com diferentes conjuntos de imagens de diferentes câmeras, e adicionar uma nova forma de otimização dos parâmetros.

1.2.2 Objetivos Específicos

Os métodos estudados mostram que os padrões circular e aneliforme apresentam grande melhoria em relação ao tabuleiro de xadrez. Por causa da precisão descrita, escolhemos manipular somente o padrão aneliforme, sem realizar comparações entre os outros padrões devido à existência de estudos que corroboram essas afirmações (Datta *et al.*, 2009).

O resultado da calibração feita utilizando a biblioteca OpenCV (Bradski *et al.*, 2005) é considerado a *baseline* nesse trabalho. Os valores obtidos com ela são comparados com: as implementações dos métodos de refinamento iterativo, onde a imagem que contém o padrão é transformada para uma visão frontal-paralela; e com o método de detecção de elipses por ajuste em detrimento da técnica de *template matching*.

Espera-se poder desenvolver um método de calibração cujo resultado de reprojeção apresente baixo erro comparado ao método base apresentado pelo OpenCV e aos trabalhos do estado-da-arte, e que seja fácil de executar, utilizando padrão planar aneliforme. Não é escopo desse trabalho alterar o cálculo dos parâmetros da câmera, criar um novo algoritmo para ajuste de elipse, nem desenvolver algoritmo para otimização. Porém, para a etapa de otimização dos parâmetros, realizamos experimentos alterando a função objetivo utilizando uma biblioteca que implementa o algoritmo de otimização Levenberg-Marquardt para esse propósito.

1.3 Estrutura

Inicialmente, o capítulo 2 apresenta a base teórica para a calibração de câmera. Mostramos como funciona o modelo de câmera *pin-hole*, ventilamos sobre como o método de Z. Zhang soluciona os parâmetros da câmera, e apresentamos um modelo de distorção de lentes. No capítulo 3, há uma mostra dos trabalhos relacionados com calibração de câmera no que tange à melhoria da localização de pontos de controle de alguns padrões de calibração. Em seguida, no capítulo 4 revelamos o método proposto, explicando o passo-a-passo para se obter uma calibração avançada. O capítulo 5 exalta os resultados deste trabalho através de dados de calibração e gráficos comparativos. A conclusão é destacada no capítulo final.

2
Fundamentação Teórica

Neste capítulo será descrito o processo de calibração de câmera convencional. Para isso é assumido o modelo de câmera pinhole (Figura 1.3) cujos parâmetros constituem a razão da calibração. Uma vez que esse modelo desconsidera as imperfeições das lentes, a ele é adicionado um modelo de distorção de lentes (Tsai, 1987).

Calibrar uma câmera significa encontrar parâmetros que associam uma imagem 2D ao mundo 3D; esses parâmetros são uma forma matemática simplificada para descrever o funcionamento dos componentes físicos da câmera. Para encontrá-los, os principais métodos de calibração utilizam objetos com características físicas conhecidas ou extraem informação do ambiente, de forma que se possa criar uma relação com o que é *visto* pela câmera e com o que se esperava ver. Essa relação pode ser definida como uma correção da imagem capturada. E essa correção é feita a partir dos parâmetros identificados.

A principal ideia por trás da calibração se basea no fato de uma imagem ser uma projeção em um plano. A partir dessa ideia cria-se um modelo representado por uma matriz de tranformação conhecida por matriz da câmera, que é composta pelos parâmetros intrínsecos e extrínsecos da câmera, e relaciona um ponto 3D a um ponto 2D.

Como a calibração consiste no cálculo desses parâmetros, é preciso entender primeiro quais são esses parâmetros que compõem o modelo pinhole.

2.1
Modelo pinhole

O modelo de câmera pinhole representa de forma simples a formação de uma imagem. Os principais elementos desse modelo são: o plano de projeção da imagem e a distância desse plano ao centro do sistema de coordenadas. Nesse modelo, uma imagem consiste em uma projeção de pontos do espaço 3D no plano da imagem. A projeção desses pontos é feita atravéz da origem do sistema de coordenadas, que representa o sensor de captura de uma câmera, por onde a luz atravessa (Figura 2.1).

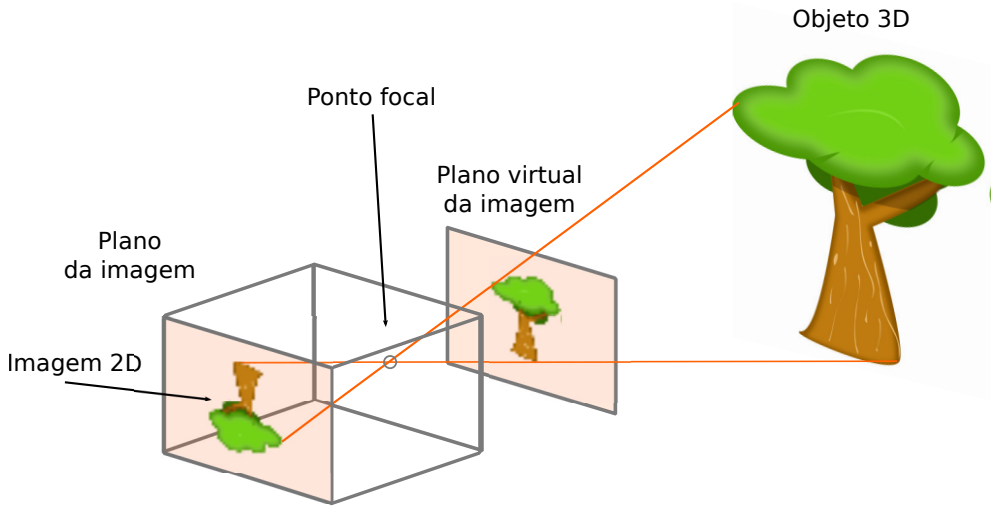


Figura 2.1: Principais componentes do modelo *pinhole*.

Precisamos considerar, nesse sistema de coordenadas, um ponto 2D no plano da imagem $\mathbf{m} = [u, v]^T$ e um ponto 3D no mundo $\mathtt{M} = [X, Y, Z]^T$. Seus vetores são aumentados ao adicionarmos 1 na última posição, sendo considerados então como $\tilde{\mathbf{m}} = [u, v, 1]^T$ e $\tilde{\mathtt{M}} = [X, Y, Z, 1]^T$, ou como alguns autores na literatura preferem dizer, são utilizadas coordenadas homogêneas.

De acordo com o modelo pinhole, a relação entre o ponto $\tilde{\mathtt{M}}$ e sua projeção na imagem $\tilde{\mathbf{m}}$ é dada pela equação

$$s\tilde{\mathbf{m}} = \mathbf{A}\begin{bmatrix}\mathbf{R} & \mathbf{t}\end{bmatrix}\tilde{\mathtt{M}} \tag{2-1}$$

onde s é um fator escalar arbitrário, $\mathbf{R}$ e $\mathbf{t}$ são os parâmetros extrínsecos da câmera, a rotação e translação da câmera. $\mathbf{A}$ é a matriz de parâmetros intrínsecos, que é definida como

$$\mathbf{A} = \begin{bmatrix}\alpha & \gamma & u_0 \\ 0 & \beta & v_0 \\ 0 & 0 & 1\end{bmatrix} \tag{2-2}$$

(u_0, v_0) é o ponto principal da imagem, que idealmente corresponde ao ponto médio da imagem, α e β são as distâncias focais no eixo x e y em pixels, e γ é a inclinação da imagem, que é considerada inexistente ou nula devido aos modernos processos de produção de sensores de imagem. [1]

A matriz 3×4 $\begin{bmatrix}\mathbf{R} & \mathbf{t}\end{bmatrix}$ é responsável pela transformação do sistema de coordenadas do mundo para o sistema de coordenadas da câmera, ao aplicar uma rotação $\mathbf{R}$ e uma translação $\mathbf{t}$. Enquanto isso, a matriz $\mathbf{A}$ é responsável pela transformação projetiva e, por conseguinte, pela transformação das coordenadas do sistema da câmera em coordenadas de pixels (Figura 2.2).

[1] CMOS (*complementary metal-oxide semiconductor*) e CCD (*charge-coupled devied*) são as duas principais tecnologias usadas na produção de sensores de captura de imagem digital.

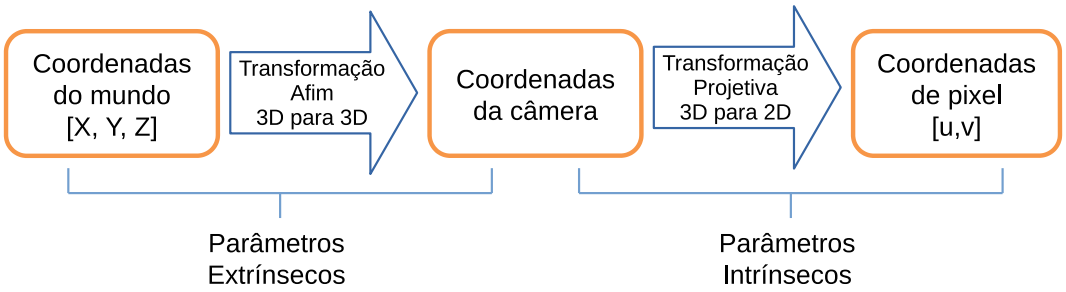


Figura 2.2: Os parâmetros extrínsecos são usados pela transformação do espaço do mundo para o espaço da câmera, enquanto que os intrínsecos projetam a cena no plano da imagem.

Ao assumirmos que o padrão de calibração está no plano $Z = 0$ do sistema de coordenadas do mundo, podemos eliminar uma coluna da matriz de rotação. Portanto a partir de 2-1 temos

$$s\begin{bmatrix} u \\ u \\ 1 \end{bmatrix} = \mathbf{A}\begin{bmatrix} \mathbf{r_1} & \mathbf{r_2} & \mathbf{r_3} & \mathbf{t} \end{bmatrix}\begin{bmatrix} X \\ Y \\ 0 \\ 1 \end{bmatrix}$$

$$= \mathbf{A}\begin{bmatrix} \mathbf{r_1} & \mathbf{r_2} & \mathbf{t} \end{bmatrix}\begin{bmatrix} X \\ Y \\ 1 \end{bmatrix}$$

Já que Z é sempre zero, temos agora o nosso ponto $\mathtt{M} = [X, Y]^T$ e $\tilde{\mathtt{M}} = [X, Y, 1]^T$. Dessa forma temos um ponto com 3 dimensões sendo transformado para outro ponto de 3 dimensões, o $\tilde{\mathbf{m}} = [u, v, 1]^T$. E aqui percebemos que essa relação nada mais é do que a matriz de homografia.

$$s\tilde{\mathbf{m}} = \mathbf{H}\tilde{\mathtt{M}} \qquad \text{com} \quad \mathbf{H} = \mathbf{A}\begin{bmatrix} \mathbf{r_1} & \mathbf{r_2} & \mathbf{t} \end{bmatrix} \tag{2-3}$$

Com a homografia estimada, calcula-se então os pârametros da câmera utilizando a solução que foi proposta por Zhengyou Zhang em (Zhang, 2000).

2.2 Método de Zhang

O método de (Zhang, 2000) é utilizado para se resolver os parâmetros lineares de uma câmera pinhole. Considerando $\mathbf{H} = \begin{bmatrix} \mathbf{h_1} & \mathbf{h_2} & \mathbf{h_3} \end{bmatrix}$, a partir da equação 2-3, temos:

$$\begin{bmatrix} \mathbf{h_1} & \mathbf{h_2} & \mathbf{h_3} \end{bmatrix} = \lambda\mathbf{A}\begin{bmatrix} \mathbf{r_1} & \mathbf{r_2} & \mathbf{t} \end{bmatrix} \tag{2-4}$$

onde λ é um escalar arbitrário. Pode-se extrair duas restrições de 2-4:

$$\mathbf{h_1^T A^{-T} A^{-1} h_2} = 0 \tag{2-5a}$$
$$\mathbf{h_1^T A^{-T} A^{-1} h_1} = \mathbf{h_2^T A^{-T} A^{-1} h_2} \tag{2-5b}$$

Tomemos então :

$$\mathbf{B} = \mathbf{A^{-T} A^{-1}} \equiv \begin{bmatrix} B_{11} & B_{12} & B_{13} \\ B_{12} & B_{22} & B_{23} \\ B_{13} & B_{23} & B_{33} \end{bmatrix}$$

$$= \begin{bmatrix} \frac{1}{\alpha^2} & -\frac{\gamma}{\alpha^2\beta} & \frac{v_0\gamma - u_0\beta}{\alpha^2\beta} \\ -\frac{\gamma}{\alpha^2\beta} & \frac{\gamma^2}{\alpha^2\beta^2} + \frac{1}{\beta^2} & -\frac{\gamma(v_0\gamma - u_0\beta)}{\alpha^2\beta^2} - \frac{v_0}{\beta^2} \\ \frac{v_0\gamma - u_0\beta}{\alpha^2\beta} & -\frac{\gamma(v_0\gamma - u_0\beta)}{\alpha^2\beta^2} - \frac{v_0}{\beta^2} & \frac{(v_0\gamma - u_0\beta)^2}{\alpha^2\beta^2} - \frac{v_0^2}{\beta^2} + 1 \end{bmatrix} \tag{2-6}$$

A matriz $\mathbf{B}$ é um matriz simétrica e, portanto, pode ser definida por um vetor 6D

$$\mathbf{b} = [B_{11}, B_{12}, B_{22}, B_{13}, B_{23}, B_{33}]^T. \tag{2-7}$$

Sendo a i-ésima coluna da matriz de homografia $\mathbf{H}$ igual a $\mathbf{h}_i = [h_{i1}, h_{i2}, h_{i3}]^T$, então é verdade que:

$$\mathbf{h}_i^T \mathbf{B} \mathbf{h}_j = \mathbf{v}_{ij}^T \mathbf{b} \tag{2-8}$$

com

$$\mathbf{v}_{ij} = [h_{i1}h_{j1}, \quad h_{i1}h_{j2} + h_{i2}h_{j1}, \quad h_{i2}h_{j2}, \\ h_{i3}h_{j1} + h_{i1}h_{j3}, \quad h_{i3}h_{j2} + h_{i2}h_{j3}, \quad h_{i3}h_{j3}]^T.$$

Assim, as restrições em 2-5 podem ser reescritas como duas equações homogêneas em $\mathbf{b}$:

$$\begin{bmatrix} \mathbf{v}_{12}^T \\ (\mathbf{v}_{11} - \mathbf{v}_{22})^T \end{bmatrix} \mathbf{b} = \mathbf{0}. \tag{2-9}$$

Se forem observadas n imagens do padrão de calibração, detectados os pontos de controle e estimatidas as homografias, podemos montar um sistema com as equações de 2-9, e teremos:

$$\mathbf{V}\mathbf{b} = \mathbf{0}, \tag{2-10}$$

e aqui $\mathbf{V}$ é uma matriz $2n \times 6$.

A solução de 2-10 depende portanto do valor de n. Para $n = 2$, pode-se considerar o valor de $\gamma = 0$ adicionando a equação $[0, 1, 0, 0, 0, 0]\mathbf{b} = 0$ à equação 2-10. Para $n \geq 3$, teremos uma solução única $\mathbf{b}$, que é o autovetor de $\mathbf{V}^T\mathbf{V}$ associado ao menor autovalor.

Quando $\mathbf{b}$ é estimado, podemos calcular os parâmetros intrínsecos da matriz $\mathbf{A}$ da seguinte forma:

$$\begin{aligned} v_0 &= (B_{12}B_{13} - B_{11}B_{23})/(B_{11}B_{22} - B_{12}^2) \\ \lambda &= B_{33} - [B_{13}^2 + v_0(B_{12}B_{13} - B_{11}B_{23})]/B_{11} \\ \alpha &= \sqrt{\lambda/B_{11}} \\ \beta &= \sqrt{\lambda B_{11}/(B_{11}B_{22} - B_{12}^2)} \\ \gamma &= -B_{12}\alpha^2\beta/\lambda \\ u_0 &= \gamma v0/\beta - B_{13}\alpha^2/\lambda. \end{aligned}$$

Em seguida, para calcular os parâmetros extrínsecos para cada imagem, fazemos:

$$\begin{aligned} \mathbf{r}_1 &= \lambda\mathbf{A}^{-1}\mathbf{h}_1 \\ \mathbf{r}_2 &= \lambda\mathbf{A}^{-1}\mathbf{h}_2 \\ \mathbf{r}_3 &= \mathbf{r}_1 \times \mathbf{r}_2 \\ \mathbf{t} &= \lambda\mathbf{A}^{-1}\mathbf{h}_3 \end{aligned}$$

Devido aos ruídos nos dados, a matriz $\mathbf{R}$ calculada não satisfaz as propriedades de uma matriz de rotação, como ortogonalidade, quando $\mathbf{R}\mathbf{R}^T = \mathbf{I}$. É necessário então corrigir essa matriz: executa-se uma decomposição em valores singulares da matriz $\mathbf{R}$, e calcula-se uma nova matriz de rotação $\mathbf{R} = \mathbf{U}\mathbf{I}\mathbf{V}^T$.

2.3 Distorção de lentes

O método de Zhang descrito na seção anterior resolve os parâmetros intrínsecos e extrínsecos da câmera do modelo pinhole. No entanto, esse modelo não representa fidedignamente uma câmera real, pois não considera a existência de lentes, outrossim ignora o sensor de captura de luz. Por essa razão, na calibração de câmera somente os parâmetros desse modelo não são suficientes. Por isso, é utilizado adicionalmente um modelo de distorção de lentes, que considera a distorção radial e tangencial.

A distorção radial provoca um efeito na imagem que é conhecido como efeito barril ou olho-de-peixe. Esse efeito faz com que a imagem fique distorcida pelas bordas, começando no centro da imagem e aumentando à medida que se chega à borda. Portanto, essa distorção depende da distância do pixel ao centro da imagem (o raio r), e é corrigida com a seguinte fórmula:

$$\breve{x} = x(1 + k_1 r^2 + k_2 r^4 + k_3 r^6)$$
$$\breve{y} = y(1 + k_1 r^2 + k_2 r^4 + k_3 r^6)$$

onde (x, y) é o ponto ideal, sem distorção, $(\breve{x}, \breve{y})$ é o ponto observado, distorcido, e k_1, k_2 e k_3 são os parâmetros da distorção radial, sendo que o terceiro é pouco utilizado.

A distorção tangencial ocorre devido ao processo de fabricação da câmera, que pode ter a lente posicionada não exatamente paralela ao sensor de captura ótico. Essa distorção é modelada pela equação:

$$\breve{x} = x + (2p_1 y + p_2(r^2 + 2x^2))$$
$$\breve{y} = y + (p_1(r^2 + 2y^2) + 2p_2 x)$$

com p_1 e p_2 como os parâmetros da distorção tangencial.

Os parâmetros da distorção da lente são não-lineares e portanto não podem ser calculados utilizando uma solução fechada. Para isso, primeiramente os parâmetros intrínsecos e extrínsecos são determinados e então é feita uma otimização dos parâmetros de distorção, que inicialmente são assumidos como zero. Nessa otimização é comum se utilizar o algoritmo de Levenberg-Marquardt para minimizar os erro de reprojeção dos pontos de controle.

2.4 Otimização dos parâmetros

Como não há uma fórmula fechada para solucionar os parâmetros de distorção das lentes, a abordagem utilizada pelas principais implementações de calibração de câmera, como *Camera Calibration With OpenCV* (Bradski e Kaehler, 2008) e *Camera Calibration Toolbox for Matlab* (Bouguet, 2014), é a minimização do erro de reprojeção 2D, dado por:

$$\sum_{i=1}^{n}\sum_{j=1}^{m}\|\mathbf{m}_{ij} - \breve{\mathbf{m}}(\mathbf{A}, \mathbf{R}_i, \mathbf{t}_i, \mathbf{k_c}, \mathtt{M}_j)\|^2 \tag{2-11}$$

onde $\mathbf{k_c} = [k_1 \quad k_2 \quad p_1 \quad p_2 \quad k_3]$ consiste nos parâmetros de distorção radial e tangencial, $\mathbf{m}_{ij}$ é o um ponto ideal e $\breve{\mathbf{m}}$ representa esse ponto real, ambos projetados no plano da imagem.

3
Trabalhos Relacionados

A abordagem introduzida por (Zhang, 2000) é o processo base usado para calibração de câmera. Como ele mesmo diz, seu método é passível de aperfeiçoamento em vários pontos. Os trabalhos que almejam melhores resultados na calibração comumente buscam um novo padrão de detecção (Vo *et al.*, 2011) que julgam apresentar características físicas melhores. Alguns podem buscar técnicas que fazem com que os pontos de controle sejam localizados com maior exatidão.

Ankur Datta, em (Datta *et al.*, 2009), percebeu que a localização dos pontos de controle era o principal problema no processo de calibração de (Zhang, 2000). Nesse processo, localizar os pontos de controle consistia em detectar as intersecções entre as casas do tabuleiro de xadrez na imagem. Para isso são utilizados métodos de processamento de imagens que filtram, binarizam, identificam bordas e fazem ajustes de linhas nas imagens. Ankur notou que a detecção do padrão de xadrez era deficiente pois era realizada em uma visão do padrão em perspectiva.

Os algoritmos que fazem detecção de bordas em imagens geralmente avaliam valores de gradiente entre pixels para determinar presença de borda. Datta notou que as linhas nas imagens em que os padrões aparecem em perspectiva – Figura 3.1(a), ou seja, inclinadas em relação aos eixos da imagem, fazem com que os algoritmos de detecção de linhas sejam prejudicados. Ele então teve a ideia de transformar a imagem que contém o padrão de calibração para fazer com que o gradiente não seja afetado.

Para isso ele executa uma calibração inicial com os padrões em perspectiva, e com o resultado da calibração, ou seja com os parâmetros intrínsecos e extrínsecos, ele transforma a imagem para uma visão canônica que ele chamou de frontal paralela – Figura 3.1(b). Nessa imagem é removida a distorção e a projeção em perspectiva, processos que ele chama de *undistort* e *unproject*, e faz com que as linhas do padrão fiquem paralelas em relação às linhas das imagem, seguindo a disposição dos pixels, assim como o gradiente fica ortogonal em relação às bordas da imagem como se pode ver na Figura 3.1.

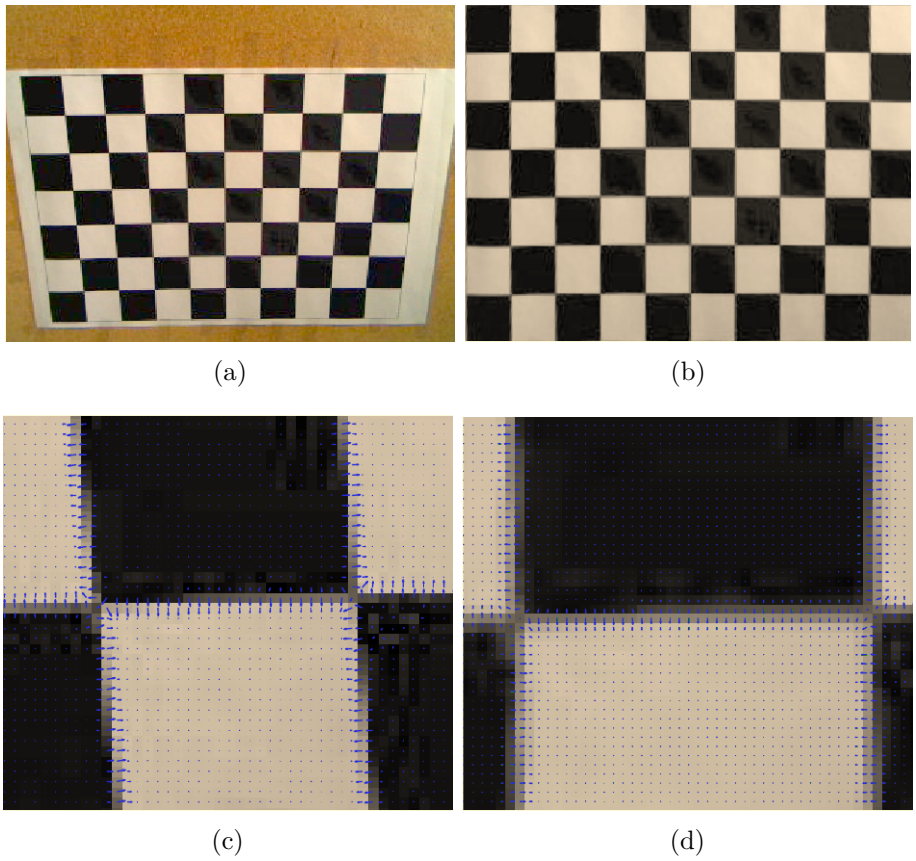


Figura 3.1: (a) Padrão de calibração em perspectiva. (b) Padrão transformado para visão canônica frontal paralela. (c) Na perspectiva, o gradiente (representado em azul) não satisfaz os algoritmos de detecção de borda. (d) No fronto-paralelo, o gradiente é ortogonal às bordas da imagem. Fronte: (Datta *et al.*, 2009).

Transformado para a visão frontal-paralelo, o padrão é novamente identificado. Datta utiliza três tipos de padrões de calibração: tabuleiro de xadrez, circular, e aneliforme. No caso do tabuleiro de xadrez, o fato de o padrão estar paralelo aos eixos da imagem melhora o resultado da localização dos pontos de controle pois o gradiente fica alinhado com esses eixos. Por outro lado, com os padrões circular e aneliforme, a localização dos centros do círculo e do anel tem maior acurácia porque nessa visão se parecem mais com circunferências enquanto que na perspectiva se parecem com elipses. E por se parecerem com circunferências, ele aplicou um algoritmo de *template matching*, utilizando soma dos quadrados das diferenças — *sum of square differences SSD* — para achar os centros dos círculos e dos anéis. Porém, é um equívoco achar que por estarem no fronto-paralelo o casamento de um modelo de círculo e de um modelo de anel seria eficiente para achar o centro, pois no fundo continuam sendo elipses. E foi pensando nisso que (Prakash, 2012) decidiu fazer ajuste de elipses (Fitzgibbon *et al.*, 1999).

Ao trocar o *template matching* pelo ajuste de elipses, Prakash atinge melhor resultado comparado a Ankur. Porém no trabalho de Prakash foi utilizado somente o padrão circular e imagens de alta resolução. Entre seus experimentos ele utiliza padrões circulares com grande dimensão, com 25x25 pontos de controle por cada imagem. Ele não avalia padrões com menores dimensões nem imagens de resolução baixa.

Prakash adiciona ao processo de calibração o que ele chama de segmentação adaptativa na localização dos pontos de controle. Além disso, ele utiliza a visão frontal-paralela proposta por Ankur, porém ao invés de fazer *template matching*, ele faz segmentação dos círculos usando um algoritmo que calcula o valor de *threshold* de forma iterativa e adaptativa. Esse processo também é incorporado nessa dissertação.

Na abordagem proposta por (Vo *et al.*, 2011), o padrão de calibração aneliforme foi modificado: os anéis passaram a ter três círculos concêntricos e foram adicionados marcadadores para facilitar a detecção do padrão. Além disso, o modelo de distorção de lentes também foi alterado: eles utilizaram cinco parâmetros de distorção radial, quatro para distorção tangencial, e quatro para distorção prismática. Não incorporamos o uso do padrão com marcadores pois ele não contribui diretamente para melhoria da calibração. Outrossim, não alteramos o modelo de distorção de lentes pois (Tsai, 1987) e (Wei e De Ma, 1994) mostram que um modelo mais elaborado não somente não ajudaria, mas também poderia causar instabilidade numérica.

Além do fronto-paralelo, Ankur propôs também fazer o processo de calibração de forma iterativa. O passo inicial da calibração é feito nas imagens em perspectiva, e os passos seguintes utilizam os pontos de controle achados na visão frontal-paralela. Dessa forma ele faz o que ele chamou de refinamento iterativo, onde cada novo passo utiliza os valores do passo anterior. Esse método também é abordado nesse trabalho.

Na etapa de otimização dos parâmetros da câmera, os trabalhos citados utilizam o erro de reprojeção 2D como função objetivo. No trabalho de (Loaiza *et al.*, 2011), por sua vez, ele utiliza outras métricas para otimizar os parâmetros da câmera. Na sua calibração múltipla, ele faz testes inserindo características de colinearidade na otimização e obtém resultado promissor ao fazê-lo. No presente trabalho foi adicionado o erro de colinearidade 2D na etapa da otimização dos parâmetros da câmera.

Na calibração de (Loaiza *et al.*, 2007), é apresentado um algoritmo capaz de agrupar, etiquetar, identificar e realizar rastreamento ótico de um conjunto de marcadores. Eles utilizam a estratégia de "dividir e conquistar" na segmentação da imagem, utilizando *quadtree* para criar relação de vizinhança

entre os padrões de calibração. No presente trabalho, temos um padrão de calibração mais simples, e criamos um algoritmo próprio para sua detecção automática.

Diante disso, percebe-se a oportunidade de integrar técnicas que melhoram o resultado da calibração. Além disso permitir a utilização do padrão aneliforme, que não está disponível nas principais bibliotecas de calibração de câmera.

4
Metodologia Proposta

Neste capítulo será apresentado o processo de calibração de câmera proposto por este trabalho. Na figura 4.1, é possível ter uma visão geral de como é o processo de calibração proposto.

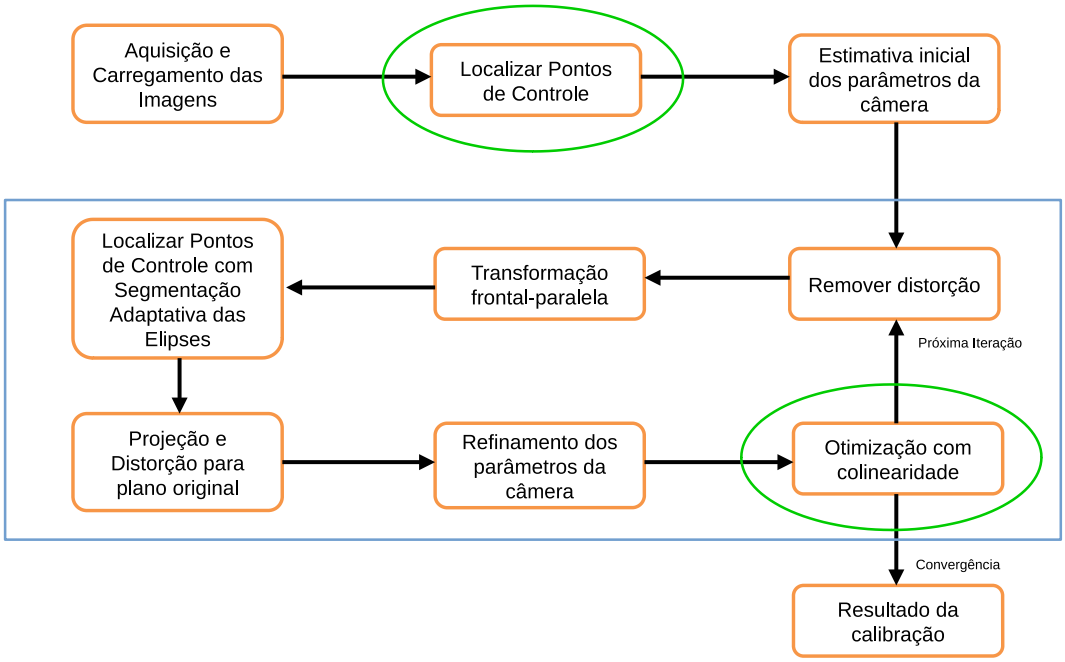


Figura 4.1: Diagrama do método proposto. Dentro do retângulo azul estão as etapas do processo iterativo. As elipses verdes indicam as principais contribuições desse trabalho.

Primeiramente, é feita a aquisição e o carregamento de várias imagens, doravante referidas como conjunto de imagens, em que cada imagem contenha o padrão de calibração aneliforme visto de diferentes perspectivas. O conjunto de imagens deve ser submetido à etapa de localização inicial do padrão de calibração, que resultará na identificação dos pontos de controle, que são os centros dos anéis do padrão. Após termos os pontos de controle localizados, é feita a primeira calibração utilizando a função de calibração de câmera da biblioteca *OpenCV* (Bradski e Kaehler, 2008).

Em seguida, inicia-se o processo iterativo da calibração, similar ao método proposto por (Datta *et al.*, 2009). Aqui, será removida a distorção de cada uma das imagens usadas desde o início do processo de calibração.

Em seguida, será realizada a transformação frontal-paralela do conjunto de imagens, processo esse que remove a projeção em perspectiva do padrão, de forma que ele fique de frente para quem o vê, alinhado com os eixos da imagem. A remoção da distorção e a transformação frontal-paralela são feitas utilizando os parâmetros intrínsecos e extrínsecos da câmera e os parâmetros de distorção das lentes que foram estimados inicialmente.

Então, os pontos de controle são novamente identificados para cada imagem do conjunto, utilizando segmentação adaptativa das elipses (Prakash e Karam, 2012). Dessa forma, tem-se novas posições dos centros dos anéis, que são projetados e distorcidos para a visão de perspectiva inicial, com o uso dos mesmos parâmetros usados para remover a distorção e aplicar a transformação frontal-paralela.

Em posse dos novos pontos de controle, cujas posições são consideradas mais precisas pelo fato de os anéis terem sido localizados na visão frontal-paralela, é feito um refinamento dos parâmetros da câmera. Em seguida, é executada uma otimização desses parâmetros utilizando o erro de colinearidade 2D.

O passo iterativo pode terminar quando se atingir convergência dos parâmetros da câmera ou quando os erros de reprojeção e de colinearidade atingirem limiares predefinidos.

4.1 Detecção inicial do padrão aneliforme

A detecção do padrão de calibração aneliforme (figura 4.2) no conjunto de imagens de calibração é a primeira contribuição deste trabalho. As bibliotecas de calibração de câmeras mais utilizadas, como *OpenCV* e *Camera Calibration Toolbox for Matlab*, não oferecem detecção automática desse padrão.

Similarmente, o código disponível *online* da implementação de (Datta *et al.*, 2009) em (Higuchi *et al.*, 2012) disponibiliza uma detecção parcial do padrão, exigindo do usuário uma marcação manual dos limites do padrão em cada imagem do conjunto, para então criar uma grade que serve para segmentar de forma grosseira os anéis. Obviamente, essa abordagem não é muito eficiente e, por isso, criamos um algoritmo para automatizar essa etapa. O algoritmo 1 mostra como isso é feito.

Nele, basicamente, a ideia consiste em fazer um processamento na imagem para detectar as elipses, identificar os anéis e, por conseguinte, os seus centros. Com esses centros, que são pontos 2D, é feita sucessivamente a organização desses pontos em uma grade. A organização desses pontos é importante para a calibração pois é preciso saber exatamente qual ponto de

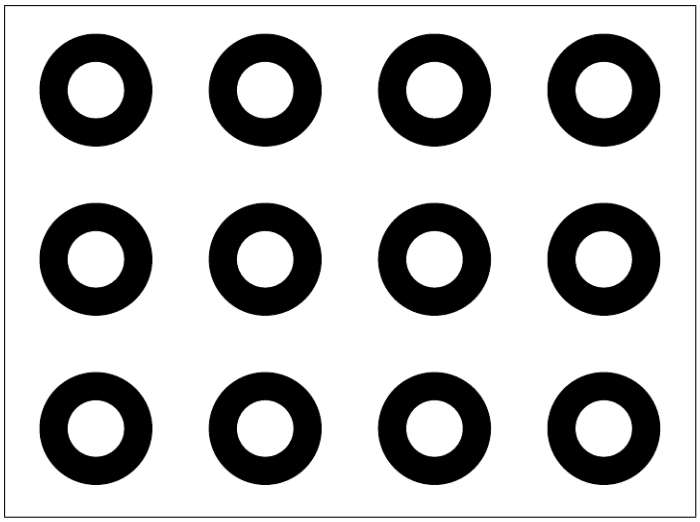
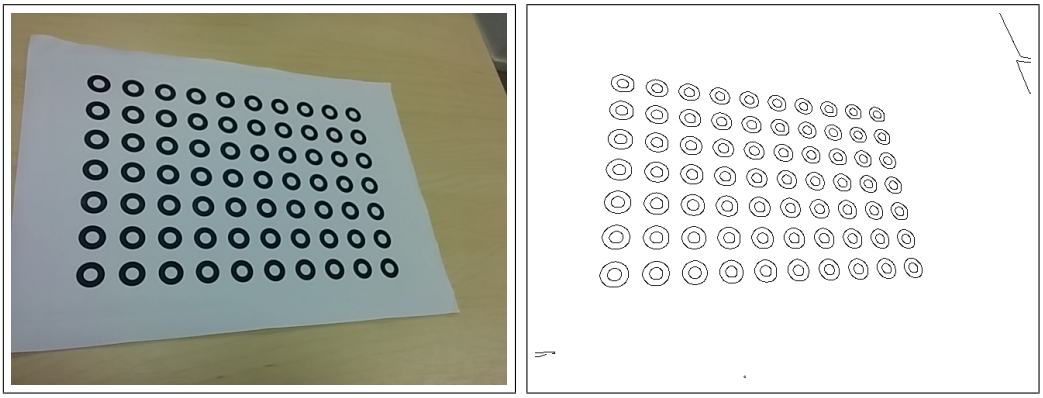


Figura 4.2: Exemplo do padrão aneliforme: distribuição colinear de anéis em uma grade. Um anel consiste em dois círculos concêntricos em um fundo branco.

controle na imagem corresponde a qual ponto no modelo do padrão, como visto na seção 2.1, cuja localização é bem definida no espaço do mundo.

A biblioteca *OpenCV* disponibiliza várias funções de processamento de imagens. Uma delas é a *findContours*, que disponibiliza os contornos achados em uma imagem e uma árvore de hierarquia entre esses contornos. Essa árvore pode ser usada para relacionar os contornos de elipses que são concêntricas, a fim de identificar os anéis. Cada anel na imagem gera dois contornos, que são conjuntos de pixels que serão usados no ajuste de elipse. Depois de as elipses serem calculadas, o ponto médio dos seus centros serve como o centro do anel. Por termos duas elipses, a localização dos pontos de controle no padrão aneliforme se torna mais precisa em detrimento do padrão circular, que só gera uma elipse.



(a) (b)

Figura 4.3: Em *a*) imagem de entrada com padrão de calibração; *b*) imagem resultante do processamento de detecção inicial dos anéis.

Algoritmo 1 Detecção automática do padrão aneliforme

Função DETECTAR PADRAO ANELIFORME(*imagem, dimensaoPadrao*)

imagemGrey ← converter *imagem* para níveis de cinza;
imagemBlur ← suavizar *imagemGrey* com gaussiana;
imagemEdges ← aplicar Canny em *imagemBlur*;
contornos, *hierarquia* ← identificar contornos em *imagemEdges*;
gridDeAneis ← identificar anéis usando *contornos* e *hierarquia*;

centros ← *gridDeAneis.centros* ▷ Organizar Grid
gridDeAneisOrganizado ← *array*(*gridDeAneis.centros.size*())
para *i* ← 1 . . . *dimensaoPadrao*[1]/2
    hull ← convexHull(gridDeAneis.centros);
    corners_id ← encontra os 4 pontos dos cantos;
    corners ← subSet(*hull*, *corners_id*);
    pontosRetaInferior ← pontosMaisProximosDaLinhaInferior();
    pontosRetaSuperior ← pontosMaisProximosDaLinhaSuperior();
    gridDeAneisOrganizado ← adicionar *pontosRetaInferior*;
    gridDeAneisOrganizado ← adicionar *pontosRetaSuperior*;
    remove pontos do gridDeAneis já adicionados no grid organizado;
fim para

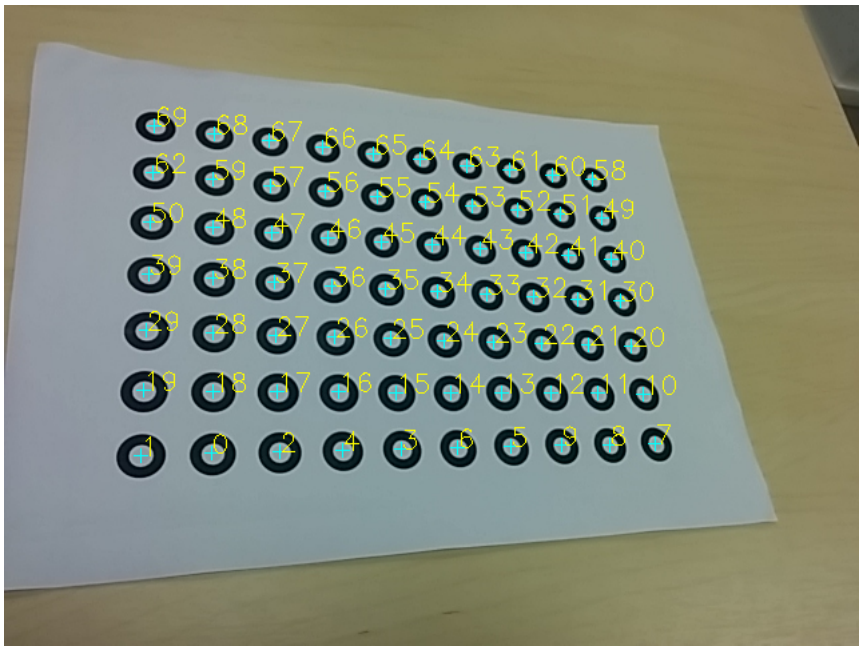
se dimensaoPadrao[1] é ímpar **então** ▷ Linha ímpar central de pontos
    pontosRestantes ← recupera pontos restantes de gridDeAneis;
    pontos ← organiza *pontosRestantes*;
    gridDeAneisOrganizado ← adicionar *pontos*;
fim se

retornar *gridDeAneisOrganizado.centros*;

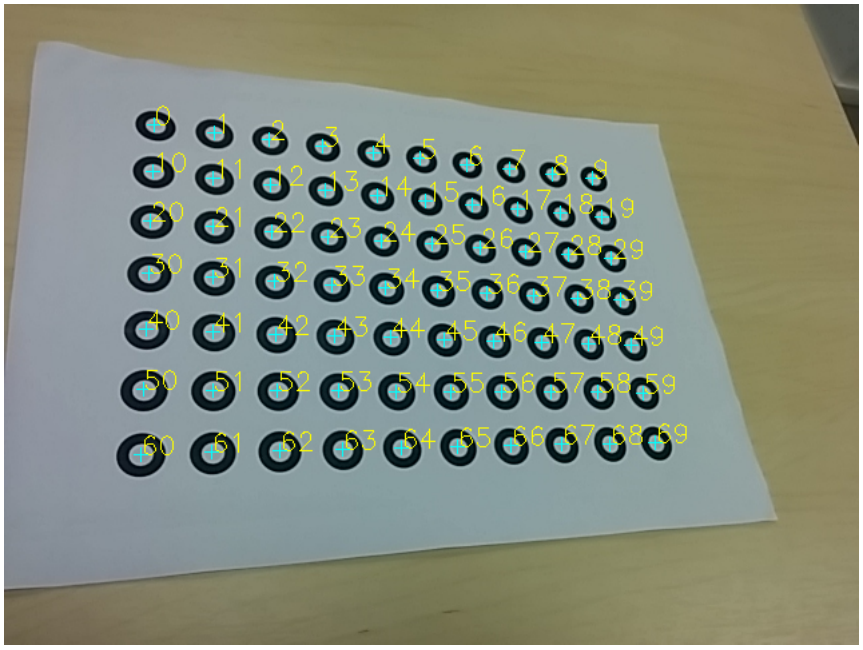
Na figura 4.3 pode-se ver a imagem de entrada que contém o padrão de calibração e a imagem resultante do processamento inicial que localiza os anéis. Depois de identificado o padrão, ainda podemos ter os pontos de controle desorganizados. Então executamos a organização dos pontos de controle a fim de manter relação lógica com o modelo de calibração.

Nessa organização, tomamos os pontos de controle e calculamos o fecho convexo. Então precisamos identificar os quatro pontos de canto do fecho convexo, que não necessariamente nos retorna somente quatro pontos. Utilizamos o ângulo formado entre três pontos consecutivos no fecho convexo para determinar se é canto ou não. Depois de termos os cantos do padrão, procuramos os pontos próximos das linhas formadas por eles tomados dois a dois, e removemos do conjunto de pontos de controle desorganizados. Repetimos esse processo com os pontos restantes até restar somente uma linha, no caso de um padrão com linhas ímpares, ou até acabar os pontos desorganizados.

Na figura 4.4 é mostrado como os pontos de controle são organizados a fim de manter uma relação lógica entre os pontos do modelo do padrão de calibração e os pontos da imagem capturada com o padrão.



(a)



(b)

Figura 4.4: Numeração dos pontos de controle *a*) anterior e *b*) posterior à organização lógica dos pontos de controle.

4.2
Estimativa inicial dos parâmetros da câmera

Depois de termos localizado os pontos de controle no conjunto de imagens, é executado o método de (Zhang, 2000), como descrito em 2.2. Com os pontos ordenados é possível associar cada ponto do modelo a seu respectivo ponto localizado na imagem.

A biblioteca OpenCV disponibiliza uma implementação de calibração que inclui além do método de Zhang, um modelo de distorção de lentes com três parâmetros radiais e 2 tangenciais, além de executar uma otimização dos parâmetros como visto em 2.4. Para cálculo da estimativa inicial dos parâmetros da câmera utilizamos essa implementação, cujo resultado é utilizado como *baseline*.

4.3
Remoção das distorções óticas das lentes

Após estimarmos os valores iniciais dos parâmetros da câmera, utilizamos os valores das distorções das lentes para remover a distorção radial e tangencial do conjunto de imagens. Para cada imagem do conjunto, uma nova imagem é criada sem as distorções das lentes. As imagens originais são sempre mantidas inalteradas pois são sempre usadas em cada passo do processo iterativo de refinamento. Na figura 4.5 podemos ver um exemplo do resultado dessa etapa.

A imagem resultante sem a distorção é criada aplicando uma transformação geométrica para cada pixel da imagem e calculando a interpolação bilinear para as posições de pixels que não receberem nenhum valor.

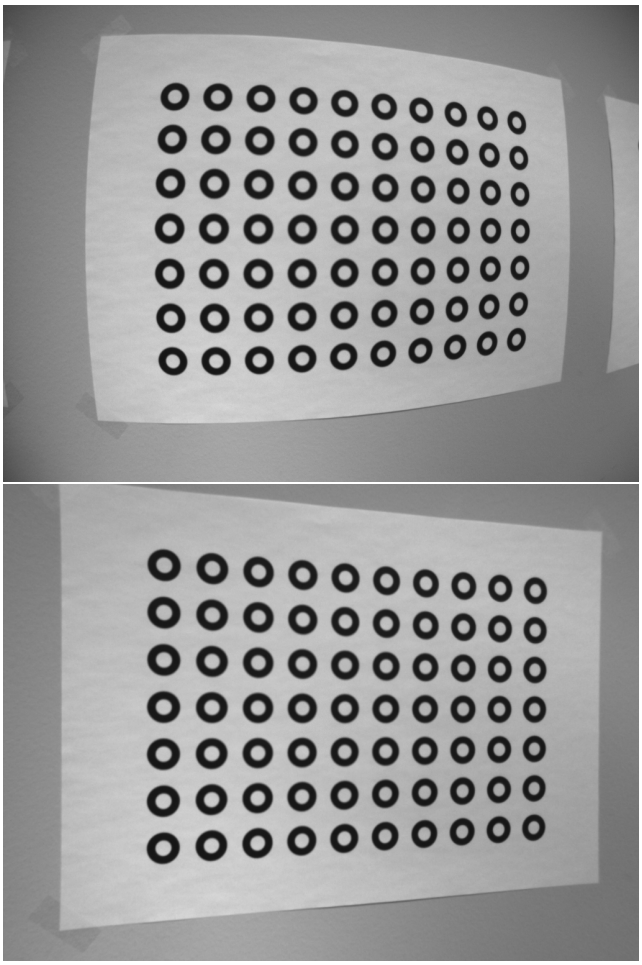


Figura 4.5: Comparação entre imagem com distorção e imagem sem distorção.

4.4 Transformação frontal-paralela

Nesta etapa, o conjunto de imagens sem distorção de lentes é submetido a uma transformação geométrica projetiva, que é executada independentemente para cada imagem, levando em consideração a posição do padrão de calibração em cada imagem do conjunto.

Essa transformação (figura 4.6) é feita calculando-se uma matriz de homografia que relaciona os quatro pontos de controle dos cantos do padrão com quatro pontos da visão frontal-paralela. Então, essa homografia é usada para gerar uma nova imagem, resultante de um processo de *warping*. Depois disso, o padrão de calibração passa a ficar isolado do *background* e alinhado com os eixos da imagem.

Como consequência disso, espera-se que as imagens resultantes desse processo sobre cada imagem do conjunto sejam bem similares, como se pode ver na figura 4.7, pois somente a parte do padrão é usada nessa fase, excluindo-se o restante da imagem.

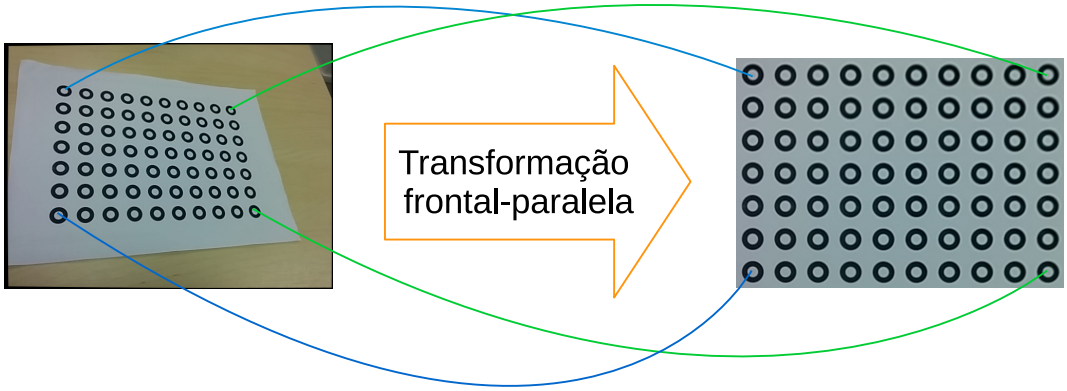


Figura 4.6: Transformação frontal-paralela.

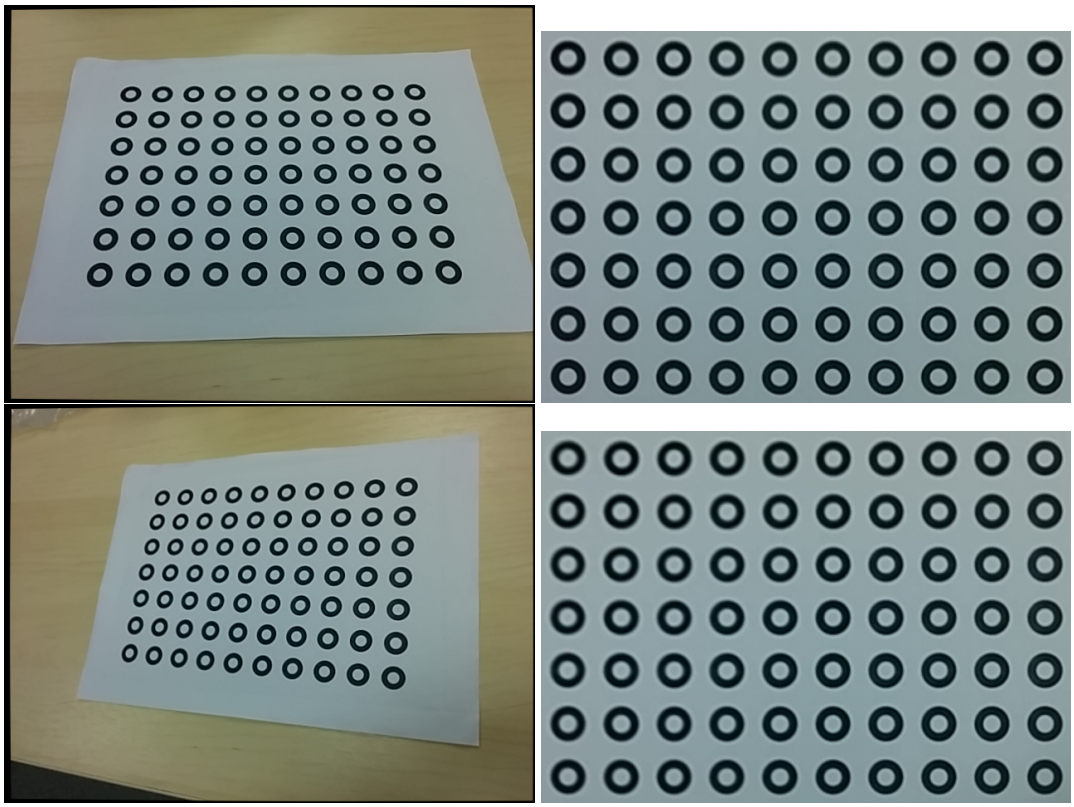


Figura 4.7: Resultado da transformação frontal-paralela.

4.5 Refinamento dos pontos de controle na visão frontal-paralela

Depois da transformação frontal-paralela, o padrão de calibração continua completamente visível e isolado na imagem. Essa próxima etapa serve para refinar a localização dos anéis, já que na etapa inicial, os anéis são localizados em perspectiva, prejudicando o ajuste das elipses. Na visão frontal-paralela, as elipses se aproximam de círculos, e os seus centros são mais precisamente localizados.

O processamento na imagem frontal-paralela é similar ao processamento para localização inicial do padrão. Mas adicionalmente é executado o refinamento da posição do ponto de controle. Nesse refinamento, cada anel da grade é isolado numa região de interesse para que se localize novamente o anel, a partir de uma segmentação baseada em um *threshold* adaptativo. Ao isolarmos cada anel, reduzimos a variação de luz presente na imagem como um todo, aumentando a precisão da localização.

O algoritmo 2 descreve como é feito o cálculo do *threshold* local.

Algoritmo 2 Algoritmo de threshold adaptativo usado na segmentação do anel. (Prakash e Karam, 2012)

Função THRESHOLD ADAPTATIVO($imagem$)
  $\mu \leftarrow$ média dos valores dos pixel das $imagem$
  $hist \leftarrow$ calcula o histograma de $imagem$
  $hist_\mu \leftarrow$ separa os μ primeiros valores de $hist$
  $T \leftarrow$ média dos valores de $hist_\mu$
  enquanto T não converge
    $\mu_1 \leftarrow$ média dos valores de $hist[0...T]$
    $\mu_2 \leftarrow$ média dos valores de $hist[T...n]$
    $T \leftarrow (\mu_1 + \mu_2)/2$
  fim enquanto
  retornar T;

Nas figuras 4.8 e 4.9 são apresentados exemplos da aplicação desse algoritmo em dois conjuntos de imagens. Cada exemplo mostra um anel isolado do padrão de calibração na visão frontal-paralela em níveis de cinza, suas segmentações intermediárias do passo iterativo do algoritmo com diferentes valores de *threshold*, e uma comparação da mudança da área do anel detectado ao longo das iterações.

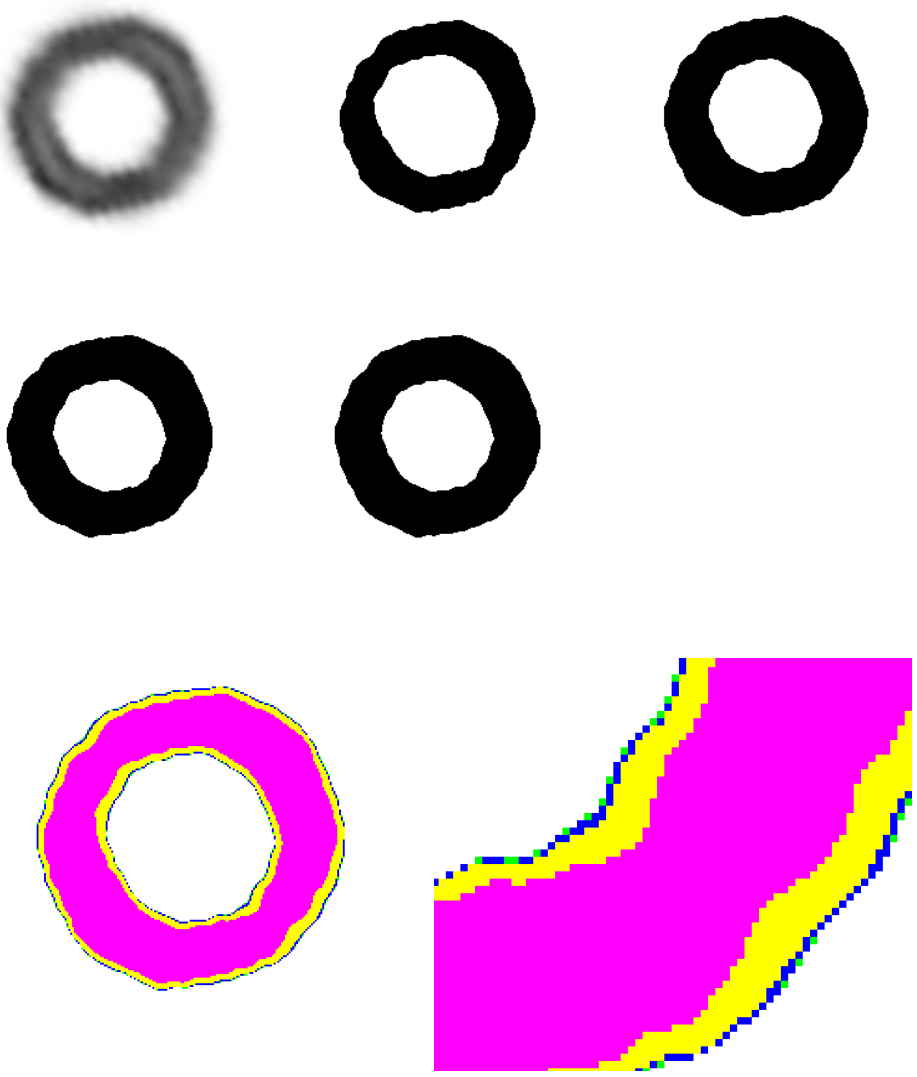


Figura 4.8: Exemplo 1 do algoritmo de thresold adaptativo com imagem de baixa qualidade. No canto superior esquerdo temos um anel original isolado da imagem. Em seguida, suas sucessivas segmentações adaptativas. Em cor-de-rosa, a primeira iteração. Nela, a segmentação não consegue capturar o anel corretamente. Nos passos seguintes, principalmente no segundo, em amarelo, temos um grande ganho na segmentação correta do anel, que é ligeiramente melhorado nos passos seguintes, representados pelas cores azul e verde. O algoritmo teve 4 iterações até convergência do valor de *threshold*.

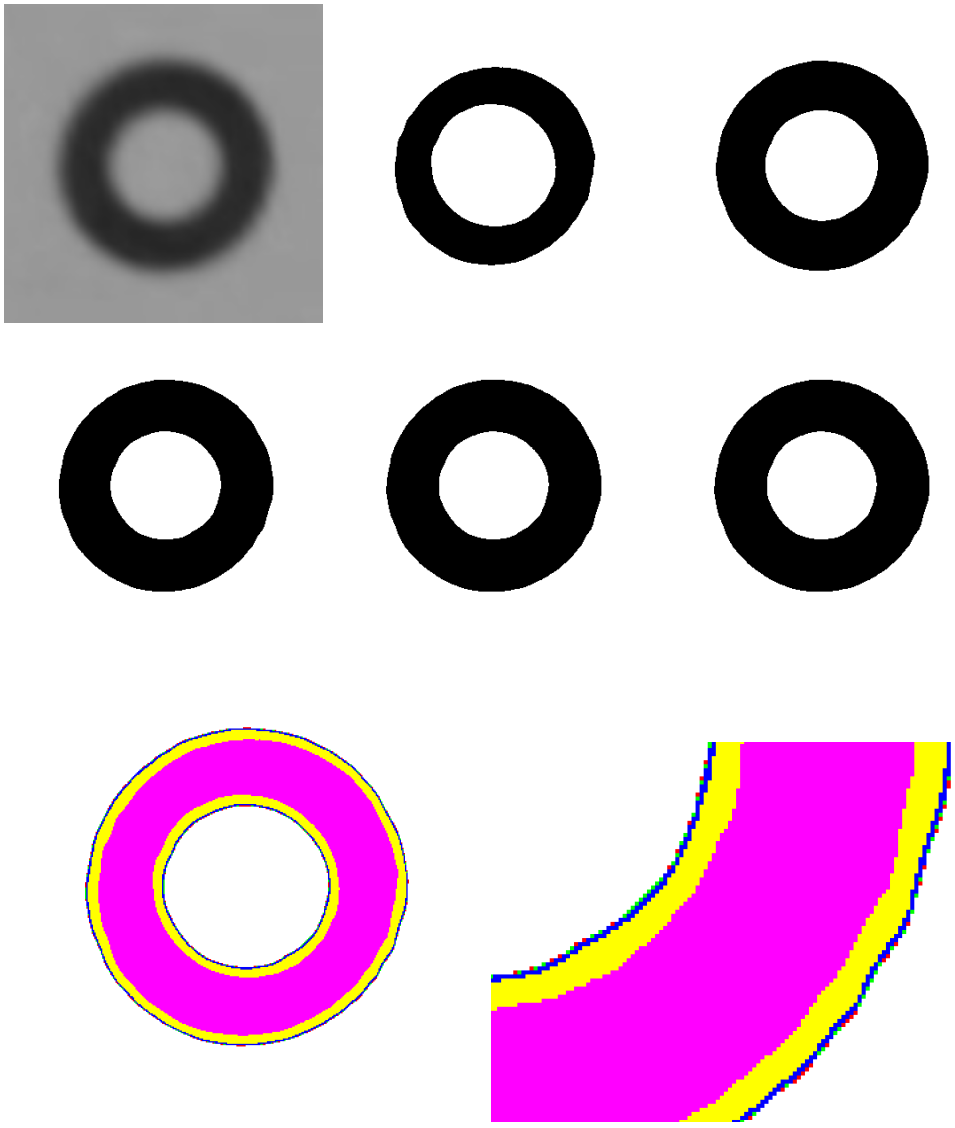


Figura 4.9: Exemplo 2 do algoritmo de threshold adaptativo com imagem de qualidade mediana. Igualmente ao exemplo anterior, temos o anel original e suas segmentações. Na comparação do ganho na área do anel, temos um alargamento do anel logo na segunda iteração, em amarelo, que é ligeiramente melhorado nos passos seguintes, representados pelas cores azul, verde e vermelho. O algoritmo teve 5 iterações até convergência do valor de *threshold*.

4.6 Projeção e Distorção para o plano original

Nessa etapa, os pontos de controle já foram localizados e refinados com a segmentação adaptativa das elipses na visão frontal-paralela. Agora, esses pontos são transformados, a fim de serem reprojetados na imagem original. Esse procedimento é o procedimento reverso à transformação feita para a visão frontal-paralela.

Inicialmente é aplicada para cada ponto uma transformação projetiva utilizando a inversa da matriz de homografia. Com isso, os pontos se apresentam numa em perspectiva sem distorção. Então, os pontos de controle são distorcidos, como mostra o algoritmo 3, e agora substituem os pontos que inicialmente foram achados com processamento de imagem na imagem original.

Algoritmo 3 Algoritmo utilizado para aplicar distorção radial e tangencial no ponto de controle.

Função DISTORCE PONTOS DE CONTROLE(*centrosProjetados, K, distCoeficientes*)
  Extrair c_x, c_y, f_x, f_y da matrix K
  Extrair k_1, k_2, p_1, p_2, k_3 de $distCoeficientes$
  $centrosDistorcidos \leftarrow \emptyset$
  para todo $c_i \in centrosProjetados$
    $x \leftarrow (c_i[0] - c_x)/f_x$; ▷ Coordenadas relativas
    $y \leftarrow (c_i[1] - c_y)/f_y$;
    $xDistort \leftarrow x(1 + k_1r^2 + k_2r^4 + k_3r^6)$; ▷ Distorção radial
    $yDistort \leftarrow y(1 + k_1r^2 + k_2r^4 + k_3r^6)$;
    $xDistort+ \leftarrow x + (2p_1y + p_2(r^2 + 2x^2))$; ▷ Distorção tangencial
    $yDistort+ \leftarrow y + (p_1(r^2 + 2y^2) + 2p_2x)$;
    $xDistort \leftarrow xDistort * f_x + c_x$; ▷ Coordenadas absolutas
    $yDistort \leftarrow yDistort * f_y + c_y$;
    $centrosDistorcidos \leftarrow centrosDistorcidos \cup \{(xDistort, yDistort)\}$;
  fim para
  retornar $centrosDistorcidos$;

Esses novos pontos são então utilizados para uma nova calibração, que refina os parâmetros da calibração do passo anterior. Para demonstração desse refinamento, na figura 4.10 foram desenhadas cruzes representando pontos de controle. As cruzes vermelhas representam os pontos de controle achados na visão frontal-paralela que foram projetados e distorcidos para a imagem original. As cruzes verdes indicam os pontos de controle anteriormente achados.

Os pontos que em não aparecem cruzes verdes significam que a cruz vermelha foi desenhada na mesma posição (x, y) em pixel da cruz verde. Não significando porém que a coordenada do ponto de controle seja a mesma.

Ou seja, mesmo que as coordenadas nas imagens tenham valores inteiros, as posições dos pontos de controle independem dos pixel, e comumente têm valores reais de coordenada.

Na figura 4.10 podemos perceber isso. Ainda podemos notar que as maiores diferenças entre os pontos anteriores e os pontos reprojetado ocorre nos anéis mais distantes do centro da imagem. Na figura 4.11, temos dois anéis ampliados para melhor mostrarmos a diferença entre os pontos de controle.

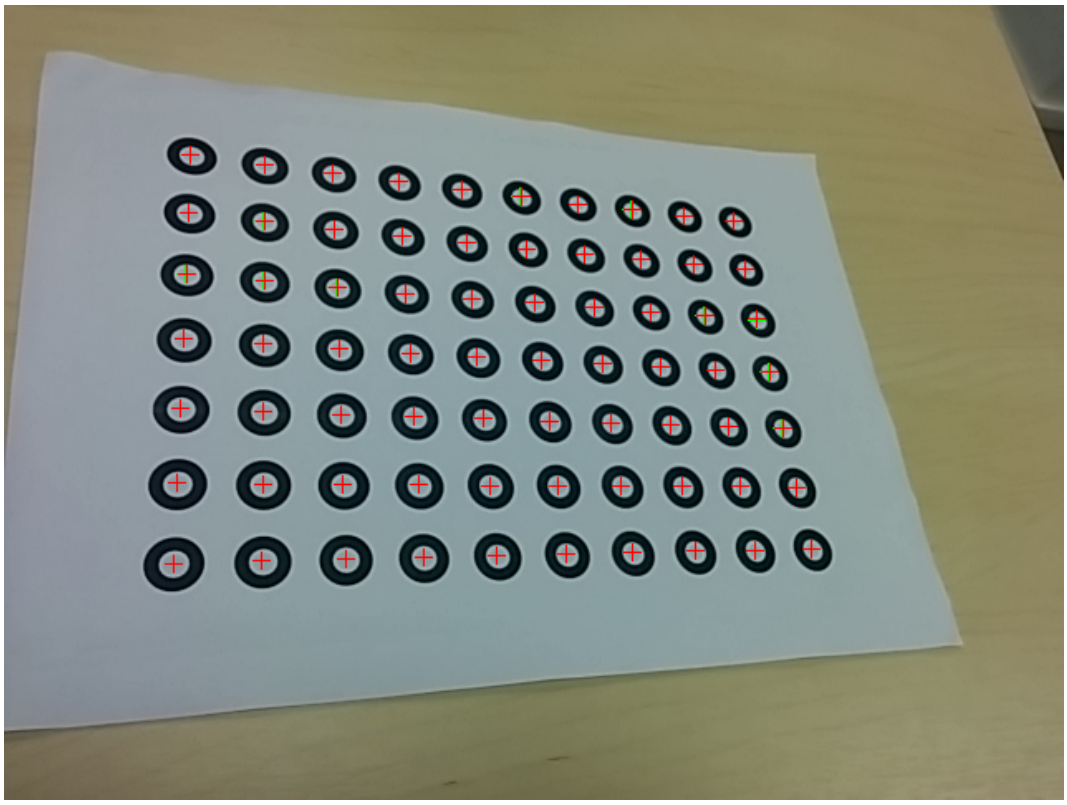


Figura 4.10: Comparação entre pontos reprojetados e distorcidos advindos da visão frontal-paralela e pontos de controle do passo anterior.

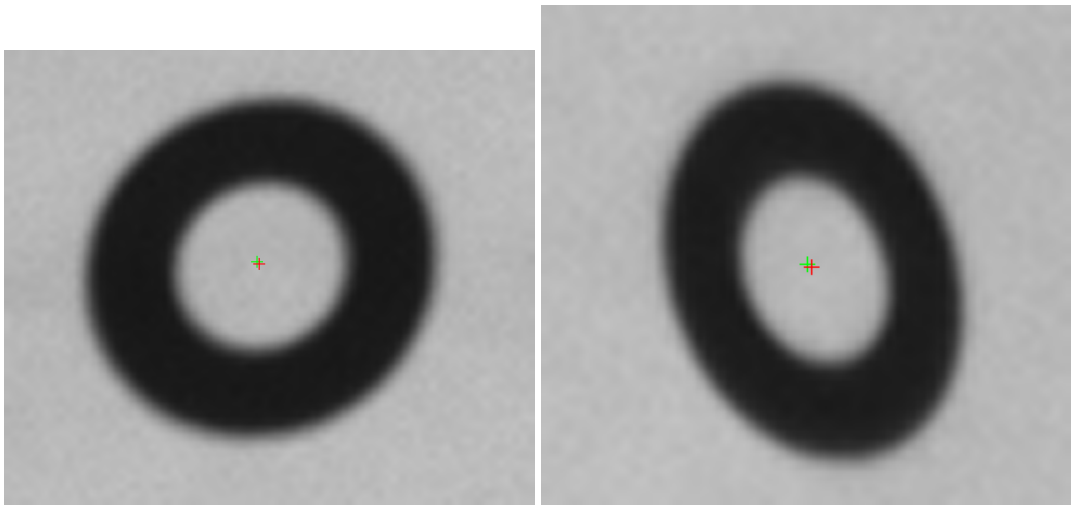


Figura 4.11: Comparação entre pontos reprojetados e distorcidos advindos da visão frontal-paralela e pontos de controle do passo anterior, em destaque.

4.7 Otimização com erro de colinearidade

Sabemos que os pontos de controle no padrão de calibração são colineares entre si, em cada linha. Partindo dessa premissa, modelamos uma função objetivo que considera o erro de reprojeção 2D e o erro de colinearidade para a otimização dos parâmetros da câmera.

Como vimos no capítulo 2, normalmente se otimiza somente o erro de reprojeção 2D dos pontos de controle, dado pela equação:

$$\sum_{i=1}^{n}\sum_{j=1}^{m}\|\mathbf{m}_{ij} - \breve{\mathbf{m}}(\mathbf{A}, \mathbf{R}_i, \mathbf{t}_i, \mathbf{k_c}, \mathtt{M}_j)\|^2 \tag{4-1}$$

ou seja, minimiza-se a distância euclidiana no espaço da imagem entre os pontos de controle achados pelo processamento de imagem e os pontos de controle reprojetados com os parâmetros do modelo da câmera achados com o processo de calibração.

Na nossa abordagem, adicionamos o erro de colinearidade, que considera os pontos projetados do modelo do padrão de calibração como colineares entre si. Isso significa que após aplicarmos a transformação $[\mathbf{R} \quad \mathbf{t}]$ nos pontos do modelo, os pontos transformados devem permanecer colineares.

A nova função objetivo portanto deseja minimizar o seguinte somatório:

$$\sum_{i=1}^{n}\sum_{j=1}^{m}\|\mathbf{m}_{ij} - \breve{\mathbf{m}}\|^2 + \|\mathbf{m}_{ij} - \mathbf{m}'\|^2 \tag{4-2}$$

que representa a adição do erro de colinearidade à função objetivo 4-1. Aqui, $\mathbf{m}'$ representa o ponto projetado perpendicularmente na linha ajustada.

Para cada sequência horizontal de pontos de controle do padrão de calibração, fazemos um ajuste de linha. Então projetamos os pontos usados

para o ajuste da linha. Esses novos pontos projetados na linha são os pontos que esperávamos que os parâmetros estimados nos desse. Por isso, tentamos otimizar a soma dos erros desses dois pontos em relação ao ponto original.

A figura 4.12 mostra uma linha ajustada nos pontos em vermelho. Os pontos em azul representam a projeção dos pontos em vermelho nessa linha. Dessa forma os dois pontos são utilizados para otimizar os parâmetros da câmera.

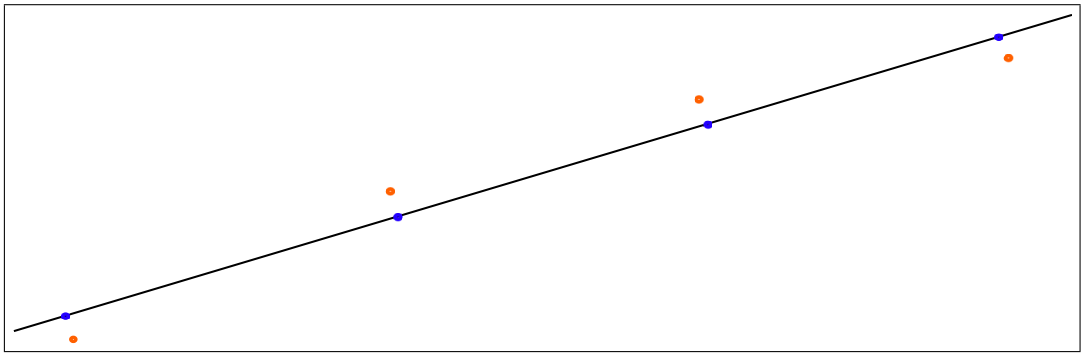


Figura 4.12: Linha ajustada sobre os pontos vermelhos. Em azul, os pontos projetados na linha.

5
Resultados Obtidos

Nessa seção serão apresentados os resultados obtidos com a nossa abordagem em relação à implementação do OpenCV e às abordagens sugeridas por (Datta *et al.*, 2009) e (Prakash e Karam, 2012).

Para os testes, capturamos conjuntos de imagens de diversas câmeras e com padrões de calibração de dimensões diferentes. Para o conjunto 1, figura 5.1, utilizamos a câmera RGB do Kinect® v1.6 para coletar 6 imagens com 640x480 pixels cada. Para o conjunto 2, figura 5.2, utilizamos a câmera de 8 megapixel do smartphone Nexus 5® da LG Electronics para capturar 5 imagens. Ambas as câmeras supracitadas geraram imagens que contêm 640x480 pixels.

Em (Higuchi *et al.*, 2012) é disponibilizado o software em Matlab® referente à abordagem proposta em (Datta *et al.*, 2009). Além disso, são disponibilizadas imagens sintéticas e reais para teste de calibração. Coletamos algumas e montamos o conjunto 3 com 10 imagens de calibração de 1024x768 pixels cada. Na figura 5.3 mostramos algumas imagens desse conjunto.

Adicionalmente, capturamos *frames* de vídeos contendo o padrão anelifome a fim de testar a calibração com imagens que tenham *blur* mais acentuado, usando uma câmera PlayStation Eye®. Na figura 5.4 mostramos alguns frames utilizados.

Nas seções seguintes, iremos comparar o RMS (*root mean square*) das abordagens desenvolvidas aqui com o valor de referência do OpenCV. Inicialmente comparamos o uso de SSD (*Sum of Squared Differences*) proposto por (Datta *et al.*, 2009). Em seguida, comparamos o uso da segmentação adaptativa proposto por (Prakash e Karam, 2012). Em ambas as comparações utilizamos o refinamento iterativo dos pontos de controle, também proposto por (Datta *et al.*, 2009), processo que consiste em refinar os pontos de controle ao transformar o padrão para a visão frontal-paralela em sucessivas iterações.

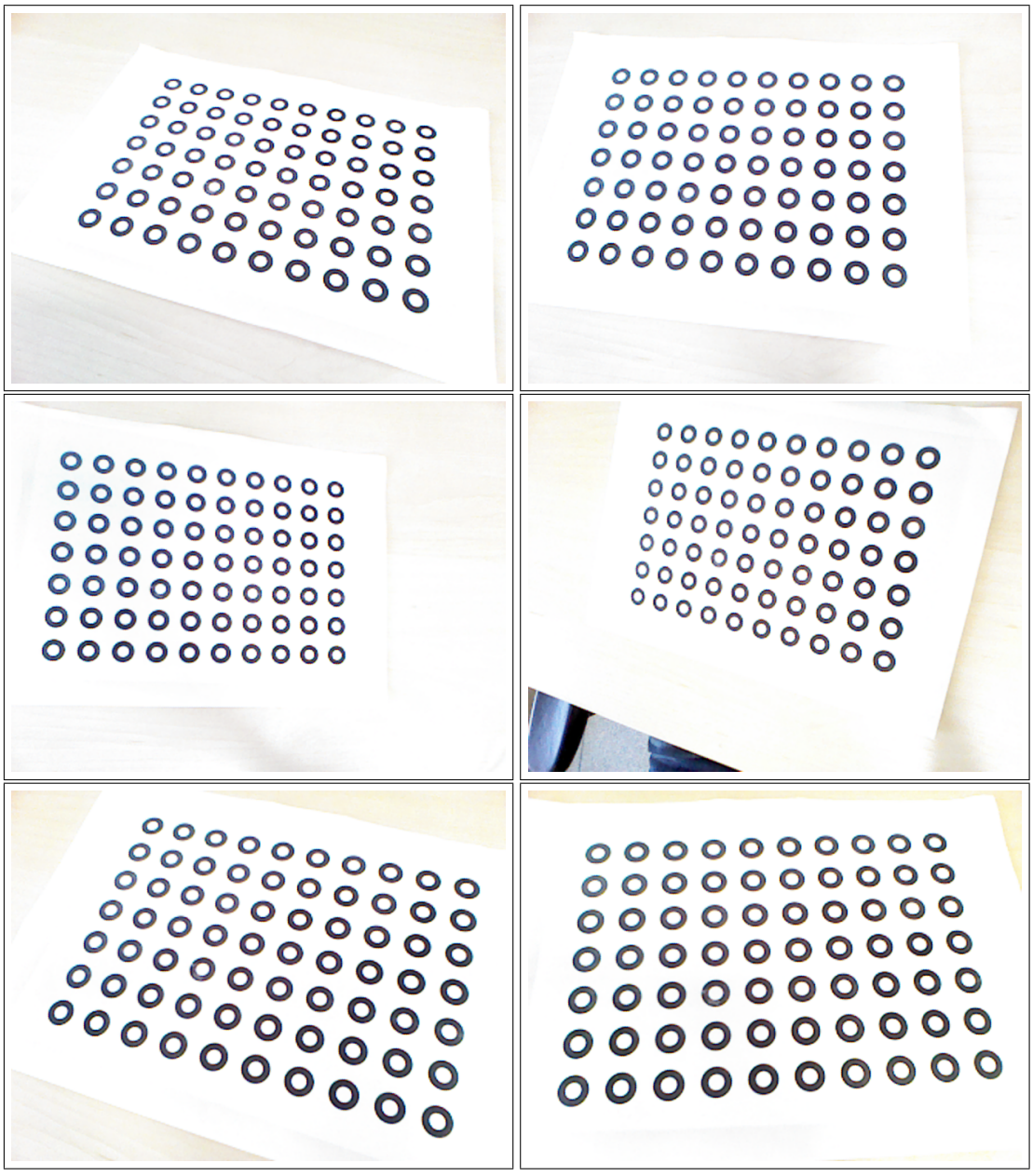


Figura 5.1: Conjunto de imagens 1.

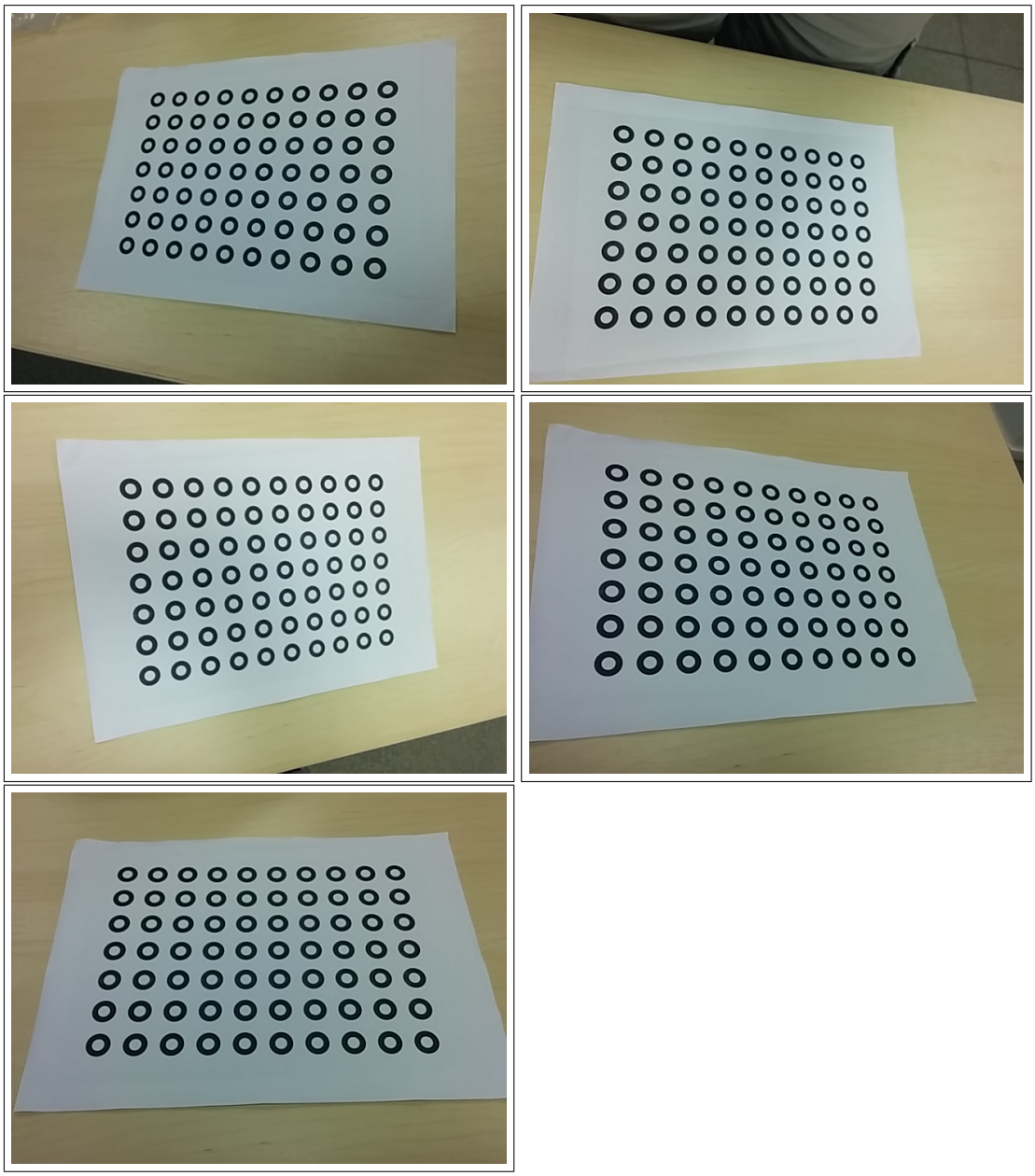


Figura 5.2: Conjunto de imagens 2.

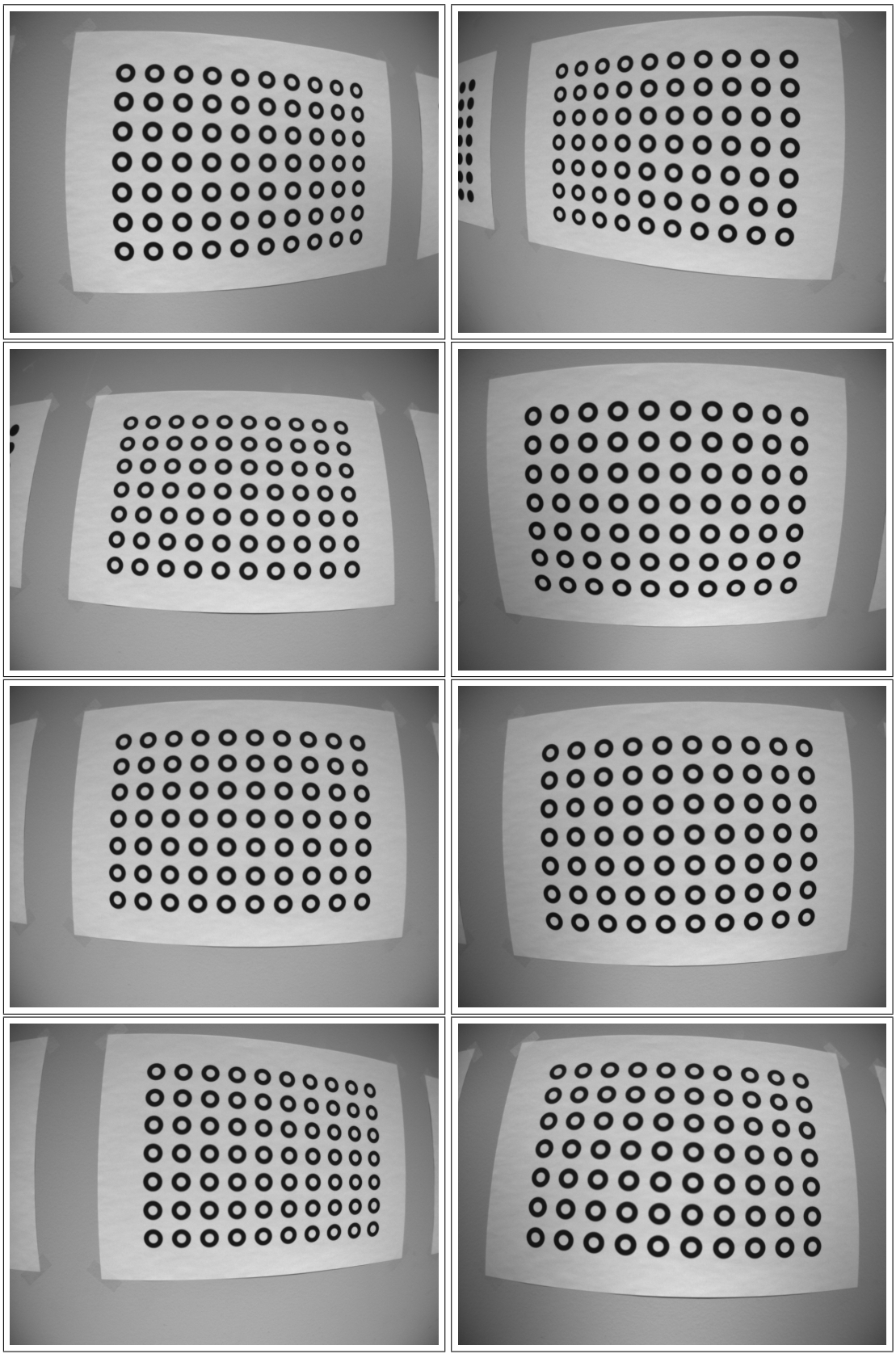


Figura 5.3: Conjunto de imagens 3. Fonte: (Higuchi *et al.*, 2012).

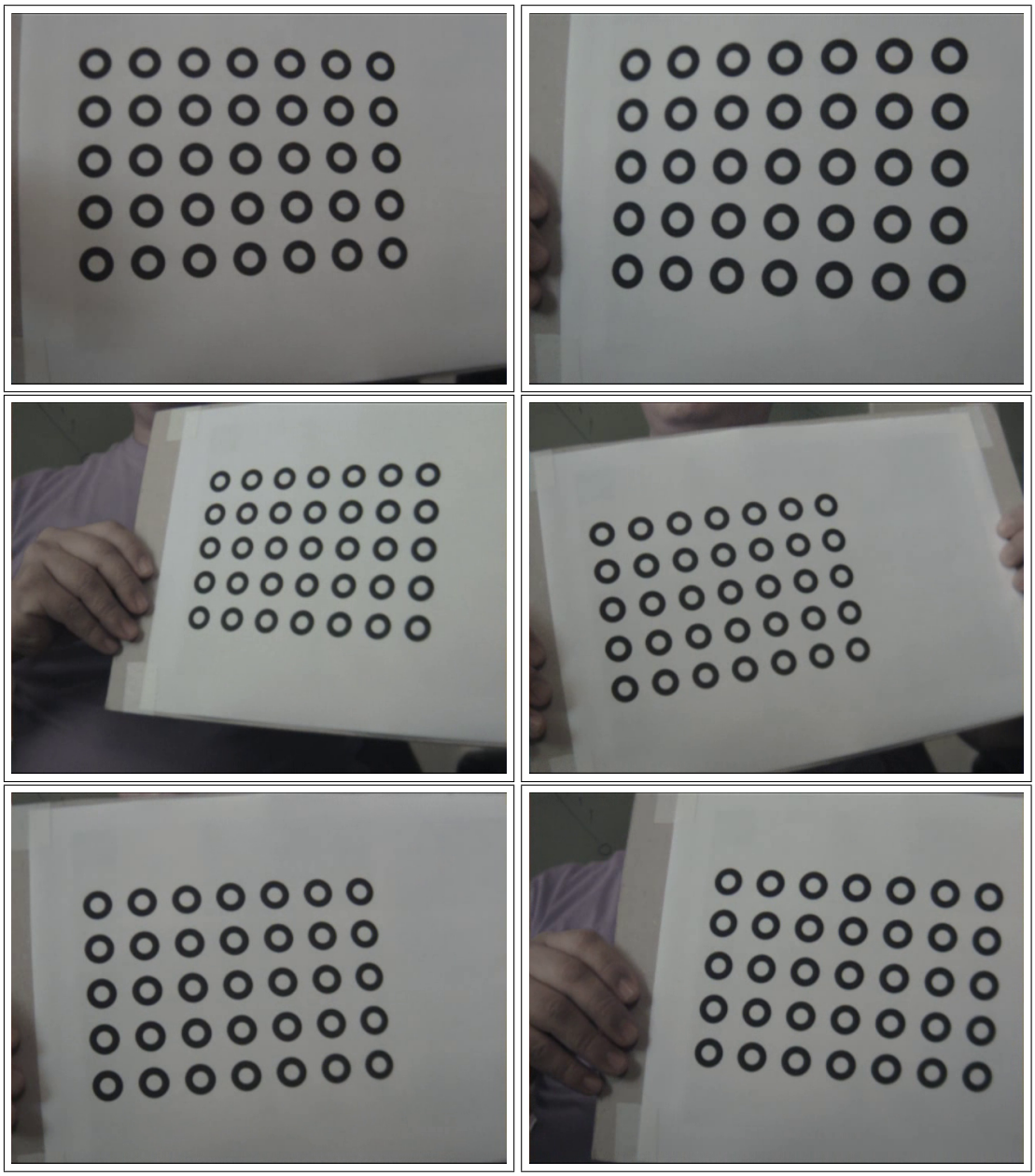


Figura 5.4: Conjunto de imagens 4.

5.1
Comparação do uso da segmentação adaptativa

O refinamento do ponto de controle ocorre numa região de interesse que isola cada anel na visão frontal-paralela. Nessa seção mostraremos como se comporta o RMS ao longo do refinamento iterativo e compararemos o erro de reprojeção com o erro produzido pelo OpenCV, para cada conjunto de imagem.

Conjunto 1

Para o conjunto 1, o RMS é reduzido, como se pode ver na figura 5.5. O valor inicial representa a execução pelo OpenCV sem qualquer melhoria. Nos passos seguintes ocorre o refinamento iterativo. Em azul temos o refinamento sem a segmentação adaptativa, enquanto que em verde temos o uso da segmentação adaptativa.

Nas figuras seguintes, temos os gráficos de nuvem de pontos que mostram o erro de reprojeção dos pontos de controle para os valores produzidos pelo OpenCV (figura 5.6), para os valores produzidos sem e com a segmentação adaptativa, respectivamente, nas figuras 5.7 e 5.8.

Nos gráficos de nuvem, as cruzes representam o erro de reprojeção de cada ponto de controle de cada imagem do conjunto. Uma cruz em $(0.5, -0,3)$ significa que o ponto reprojetado ficou à distância de 0.5 pixel à direita e à 0.3 pixel à esquerda do ponto de controle original. As cruzes recebem cores diferentes para cada imagem do conjunto.

Para o conjunto 1, o uso da segmentação adaptativa melhora o resultado da calibração, como se vê na figura 5.5, em que o RMS é menor. A melhoria no entanto ocorre na terceira casa decimal.

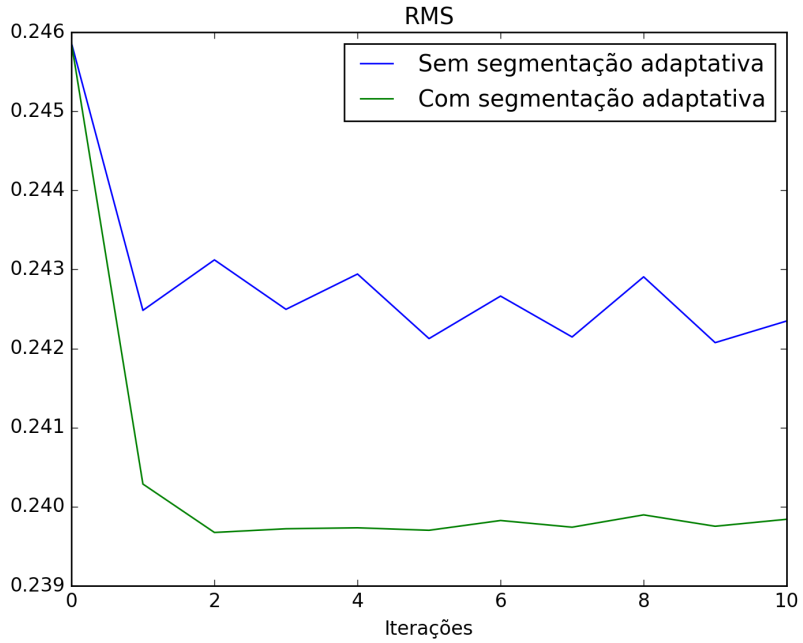


Figura 5.5: Comparação RMS com e sem segmentação adaptativa para conjunto 1.

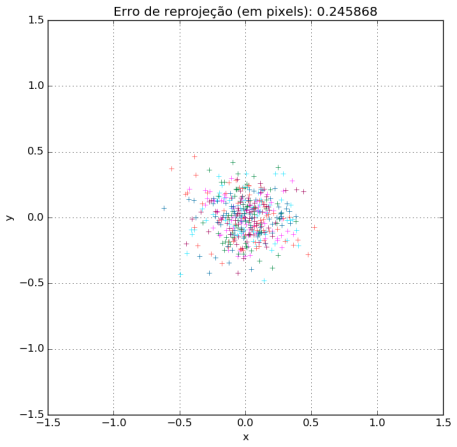


Figura 5.6: Gráfico de nuvem para erro de reprojeção pelo OpenCV para conjunto 1.

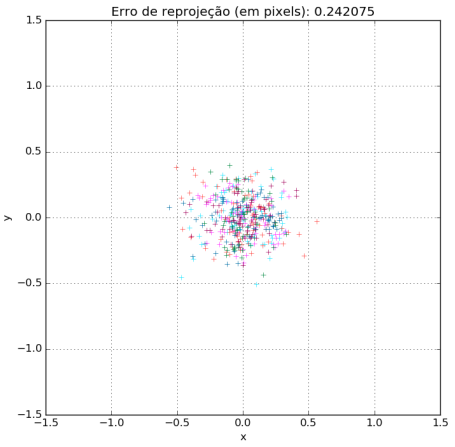


Figura 5.7: Gráfico de nuvem para erro de reprojeção sem segmentação adaptativa para conjunto 1.

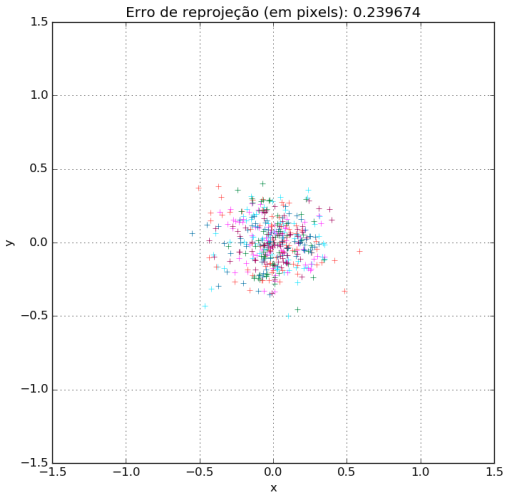


Figura 5.8: Gráfico de nuvem para erro de reprojeção com segmentação adaptativa para conjunto 1.

Conjunto 2

Para o conjunto 2, o uso do passo iterativo melhora consideravelmente o RMS logo no início do refinamento, como podemos ver na figura 5.9. A melhoria em relação ao uso ou não da segmentação adaptativa não é tão expressiva, ocorre na quarta casa decimal. O resultado do refinamento iterativo sem a segmentação adaptativa chega bem próximo do resultado com a segmentação adaptativa nos passos seguintes. Na figura 5.10, isolamos o RMS resultante dos passos iterativos, removendo o valor inicial do OpenCV, a fim de visualizarmos como o RMS não se estabiliza nesse conjunto.

O RMS obtido pelo OpenCV pode ser visto na figura 5.11. Para o passo iterativo com e sem a segmentação adaptativa, podemos ver o resultado nas figuras 5.7 e 5.8.

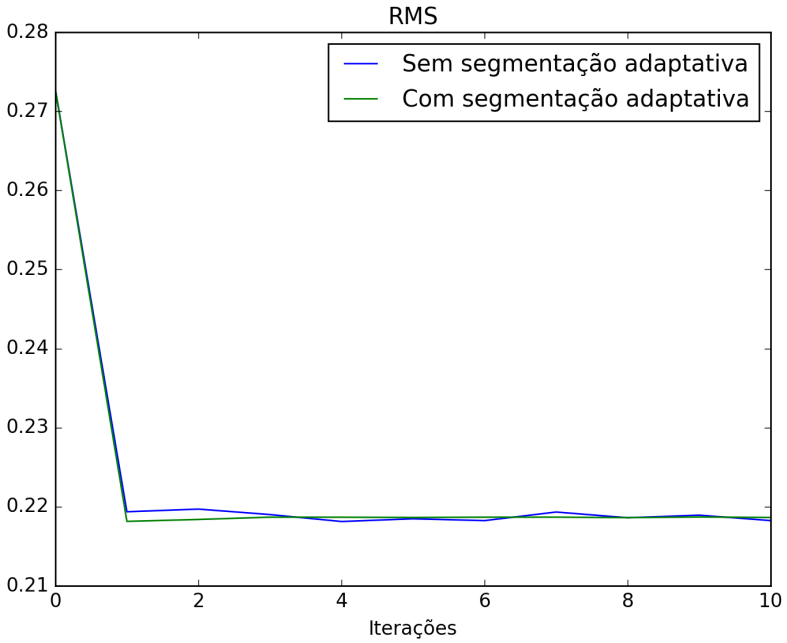


Figura 5.9: Comparação RMS com e sem segmentação adaptativa para conjunto 2.

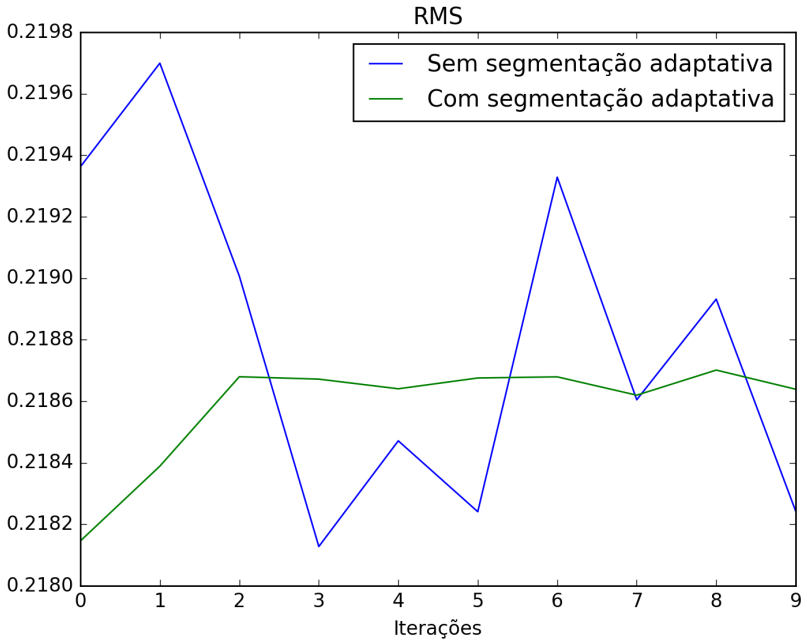


Figura 5.10: Comparação RMS com e sem segmentação adaptativa para conjunto 2 somente passo iterativo.

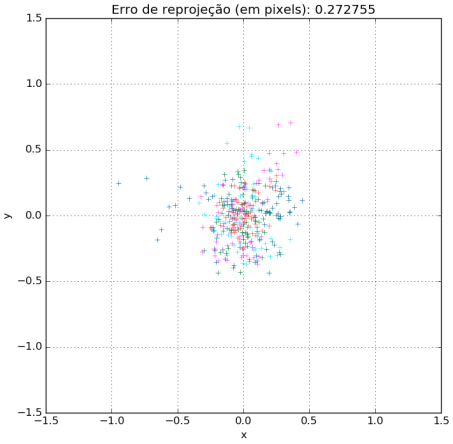


Figura 5.11: Gráfico de nuvem para erro de reprojeção pelo OpenCV para conjunto 2.

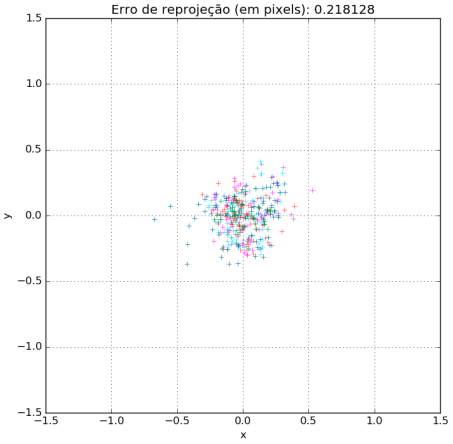


Figura 5.12: Gráfico de nuvem para erro de reprojeção sem segmentação adaptativa para conjunto 2.

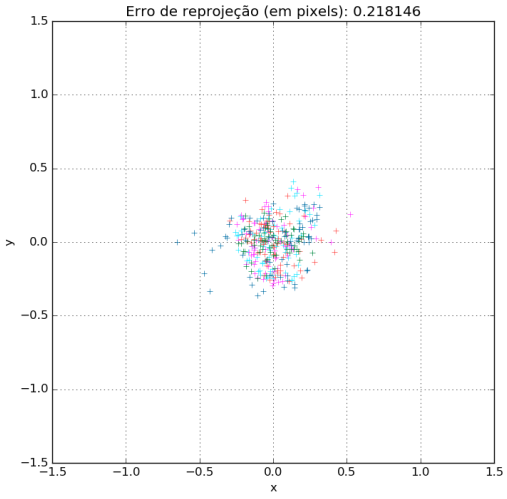


Figura 5.13: Gráfico de nuvem para erro de reprojeção com segmentação adaptativa para conjunto 2.

Conjunto 3

No conjunto 3, tivemos o melhor resultado, figura 5.14. O RMS inicial pelo OpenCV é reduzido de 0,35 para 0,06. Isolando o resultado do refinamento iterativo, percebemos que a segmentação adaptativa das elipses também ajuda para diminuir o RMS, figura 5.15.

O RMS obtido pelo OpenCV para o conjunto 3 pode ser visto na figura 5.16. Para o passo iterativo com e sem a segmentação adaptativa, podemos ver o resultado nas figuras 5.17 e 5.18.

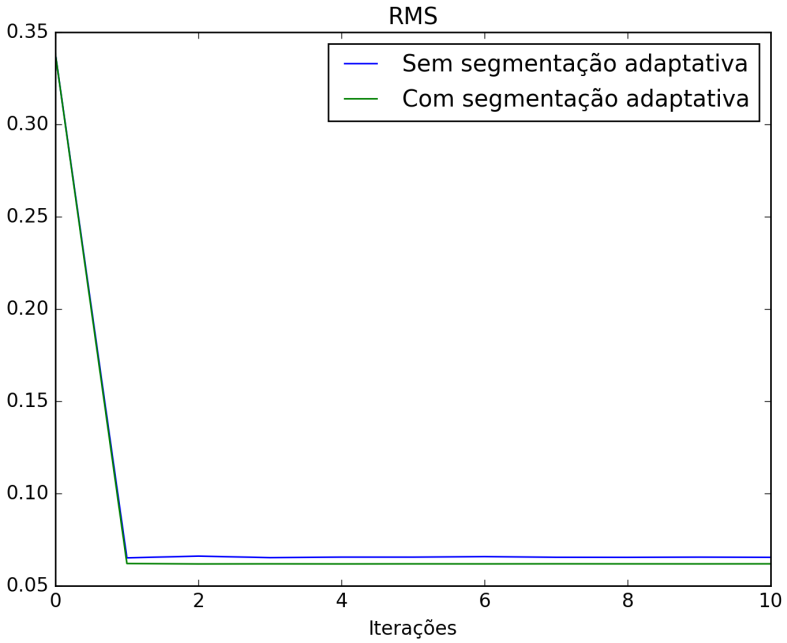


Figura 5.14: Comparação RMS com e sem segmentação adaptativa para conjunto 3.

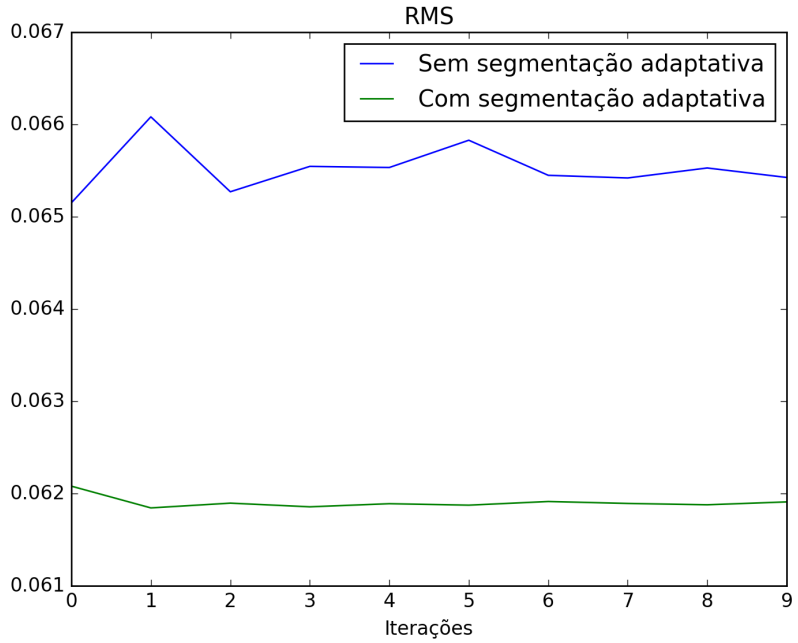


Figura 5.15: Comparação RMS com e sem segmentação adaptativa para conjunto 3 somente passo iterativo.

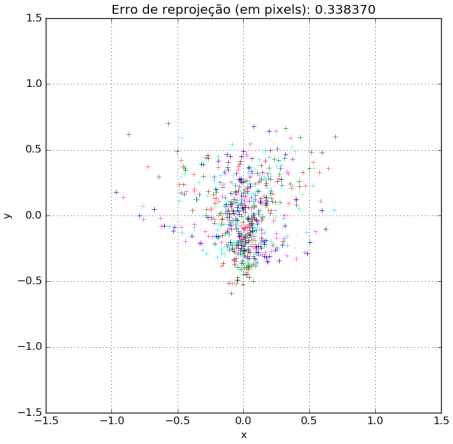


Figura 5.16: Gráfico de nuvem para erro de reprojeção pelo OpenCV para conjunto 3.

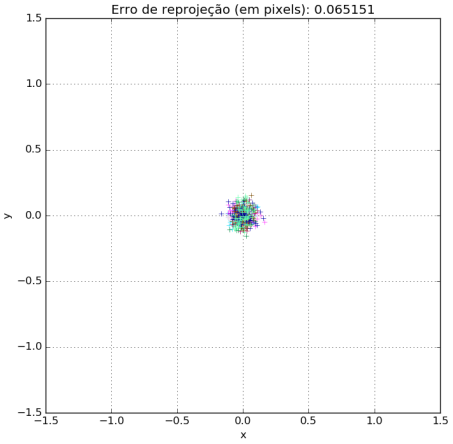


Figura 5.17: Gráfico de nuvem para erro de reprojeção sem segmentação adaptativa para conjunto 3.

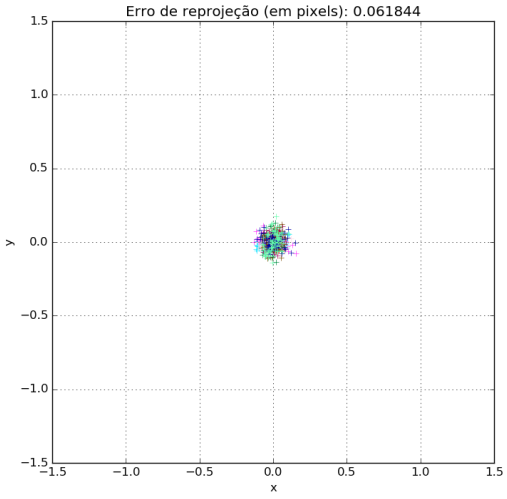


Figura 5.18: Gráfico de nuvem para erro de reprojeção com segmentação adaptativa para conjunto 3.

Conjunto 4

O conjunto 4 também apresentou melhoria na terceira casa decimal com o uso da segmentação adaptativa e do refinamento iterativo, figura 5.19.

O RMS obtido pelo OpenCV pode ser visto na figura 5.20. Para o passo iterativo com e sem a segmentação adaptativa, podemos ver o resultado nas figuras 5.21 e 5.22.

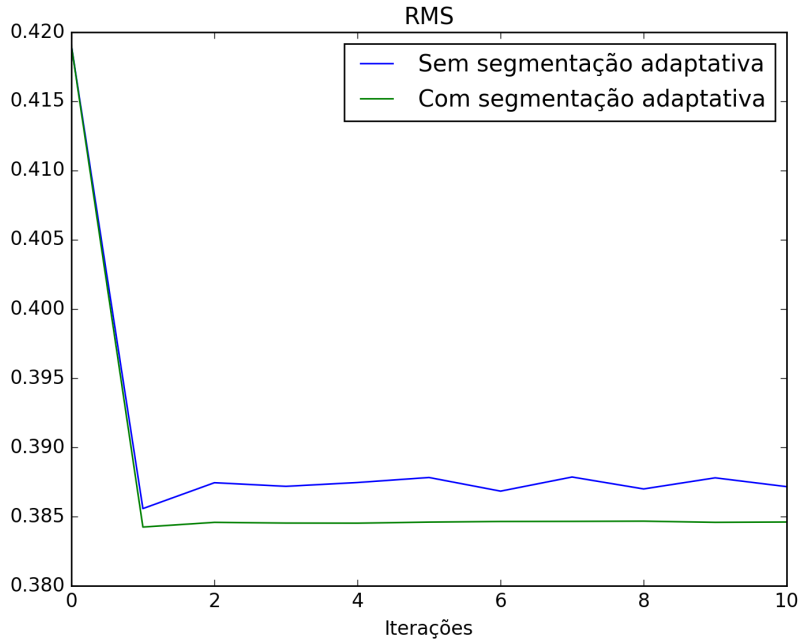


Figura 5.19: Comparação RMS com e sem segmentação adaptativa para conjunto 4.

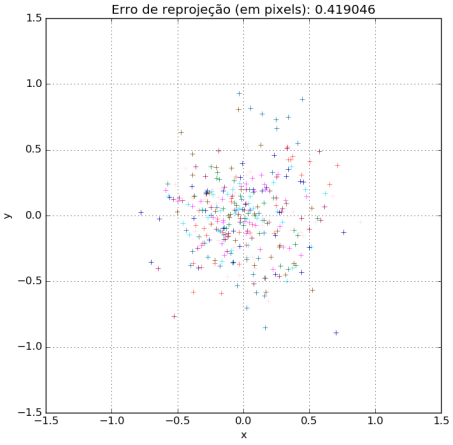


Figura 5.20: Gráfico de nuvem para erro de reprojeção pelo OpenCV para conjunto 4.

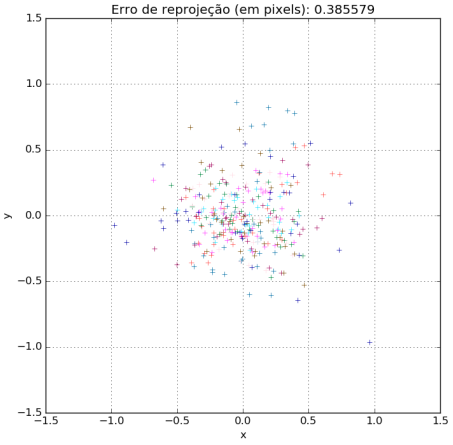


Figura 5.21: Gráfico de nuvem para erro de reprojeção sem segmentação adaptativa para conjunto 4.

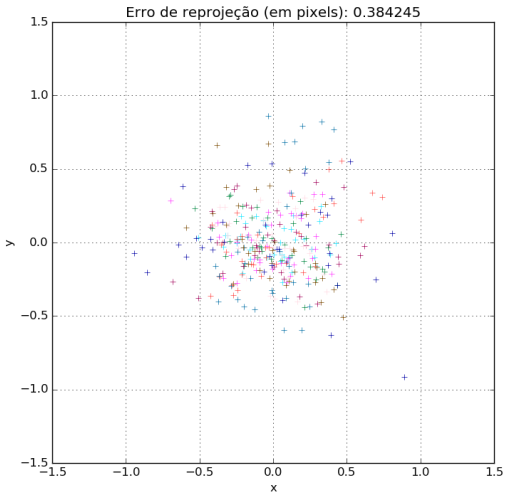


Figura 5.22: Gráfico de nuvem para erro de reprojeção com segmentação adaptativa para conjunto 4.

5.2 Comparação da otimização com erro de colinearidade

Nessa seção, mostraremos o resultados da execução da nossa abordagem comparando o uso da otimização com erro de colinearidade. Aqui, novamente, apresentaremos para cada conjunto, a evolução do RMS pelo processo iterativo. Foi removido de todos os gráficos, o valor inicial da calibração pelo OpenCV, por ser um valor alto que faz com que os valores do passo iterativo não sejam bem percebidos. Na alternância entre usar e não usar a otimização com erro de colinearidade, foi mantido o uso da segmentação adaptativa.

Conjunto 1

Como podemos ver na figura 5.23, o valor de menor RMS ocorre na iteração 11, pertecente à iteração com uso do erro de colinearidade. Já com os gráficos das figuras 5.24 e 5.25, percebemos que a diferença entre os erros de reprojeção alcançados é de $5x10^{-5}$.

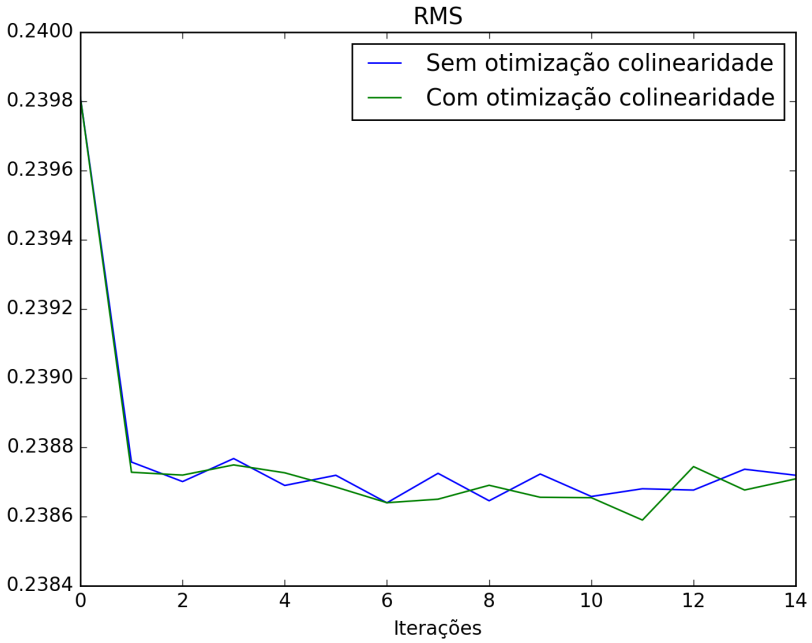


Figura 5.23: Comparação RMS com e sem otimização com erro de colinearidade para conjunto 1.

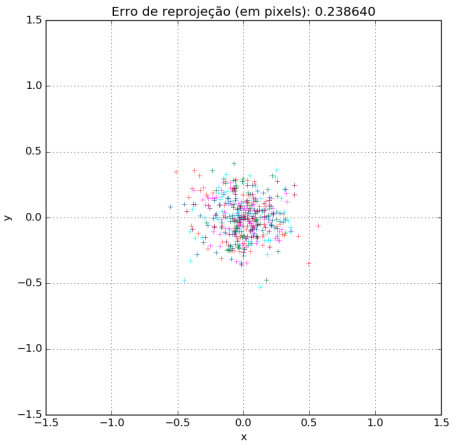


Figura 5.24: Gráfico de nuvem para erro de reprojeção sem otimização com erro de colinearidade para conjunto 1.

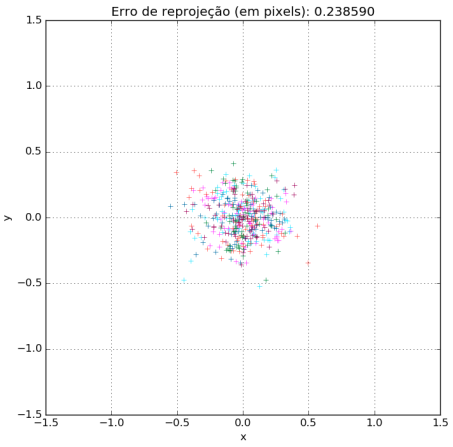


Figura 5.25: Gráfico de nuvem para erro de reprojeção com otimização com erro de colinearidade para conjunto 1.

Conjunto 2

No conjunto 2, já não havia melhoria nos passos iterativos utilizando somente a segmentação adaptativa. O comportamento se repetiu com o uso da otimização com erro de colinearidade, como vemos na figura 5.23.

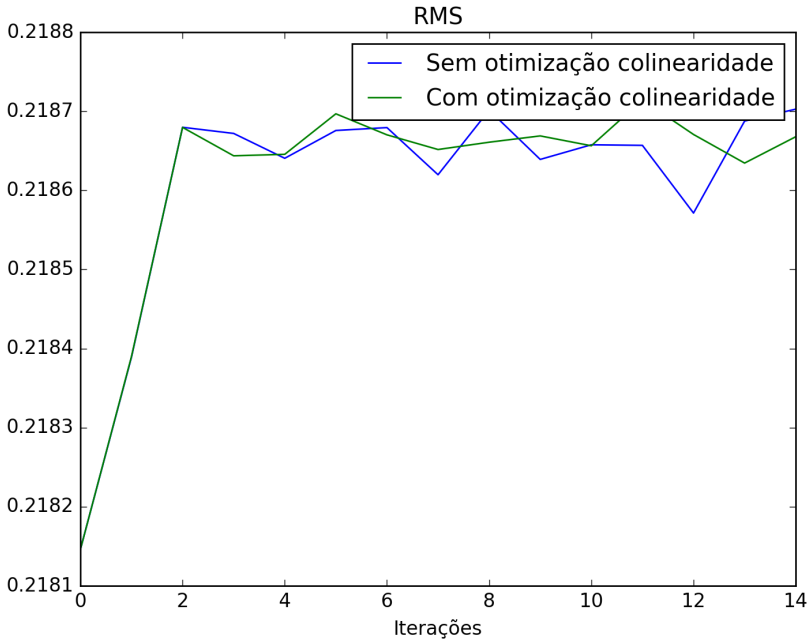


Figura 5.26: Comparação RMS com e sem otimização com erro de colinearidade para conjunto 2.

Conjunto 3

Para o conjunto 3, podemos ver que os melhores valores de RMS (figura 5.23) nas iterações variam na quinta casa decimal, assim como no conjunto 1. Nas figuras 5.28 e 5.29, vemos que a diferença entre os erros de reprojeção alcançados é de $2.8x10^{-5}$.

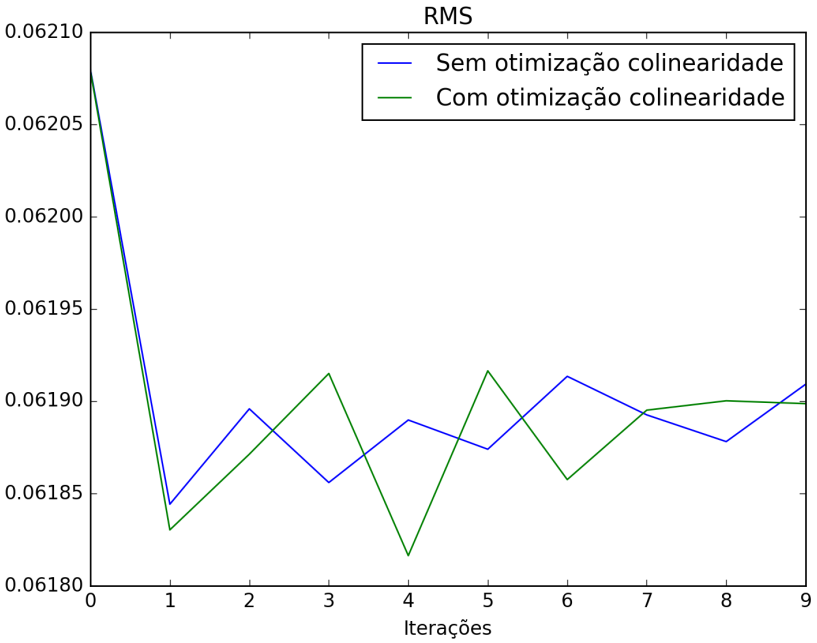


Figura 5.27: Comparação RMS com e sem otimização com erro de colinearidade para conjunto 3.

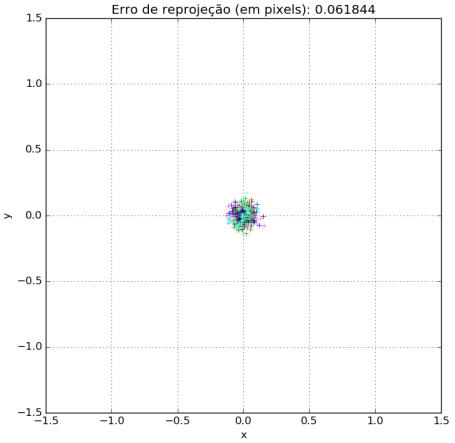


Figura 5.28: Gráfico de nuvem para erro de reprojeção sem otimização com erro de colinearidade para conjunto 3.

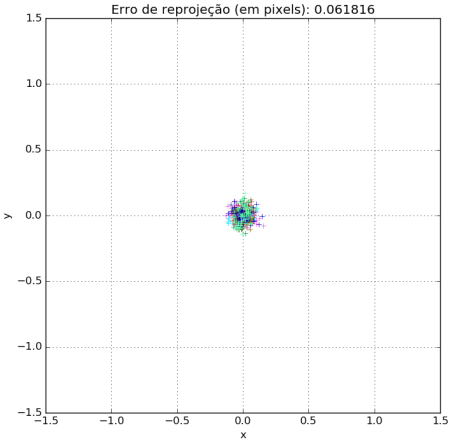


Figura 5.29: Gráfico de nuvem para erro de reprojeção com otimização com erro de colinearidade para conjunto 3.

Conjunto 4

Para o conjunto 4, assim como não houve melhoria a partir da segunda iteração para o uso da segmentação adaptativa, também não houve melhoria para o uso da otimização com erro de colinearidade. Aqui lembramos que a

iteração 0 representa a primeira iteração depois da calibração pelo OpenCV. Na figura 5.30, mostramos o resultado do RMS das iterações.

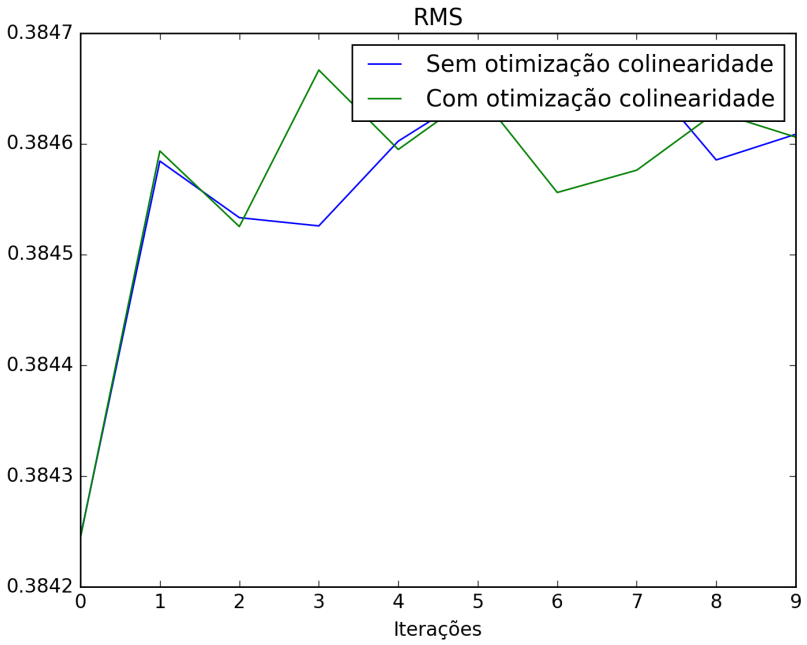


Figura 5.30: Comparação RMS com e sem otimização com erro de colinearidade para conjunto 4.

5.2.1 Sobre o uso da otimização com erro de colinearidade

Podemos perceber que a melhoria adquirida com a otimização com erro de colinearidade foi muito pequena, e ocorreu somente para metade dos casos estudados. Com os testes que executamos, o modelo de distorção das lentes parece representar com muita precisão as distorções, fazendo com que as linhas ajustadas com os pontos de controle tivessem erro muito baixo. Isso quer dizer que ao removermos a distorção dos pontos de controle, e calcularmos o quão não colineares os pontos estavam, vimos que isso era muito baixo, pois os pontos estavam praticamente colineares. Dessa forma, o erro de colineariade não poderia ajudar muito na otimização dos parâmetros.

Acreditamos que outras propriedades físicas do padrão de calibração possa ser levado em consideração na otimização, como as distâncias entre os anéis. Um outro viés sugere a possibilidade de termos atingido o mínimo de erro possível para esse modelo, dadas as incertezas inerentes à imagem.

No trabalho de (Prakash e Karam, 2012), foi utilizado câmeras de alta resolução. No nosso, usamos câmeras comuns. Uma outra especulação reside no fato de termos usado imagens de resolução baixa.

6
Conclusões e Trabalhos Futuros

Neste trabalho de dissertação, nos propomos a desenvolver uma abordagem de calibração de câmera avançada que unisse diferentes técnicas do estado-da-arte que, por sua vez, demonstraram melhoria no processo de calibração.

As técnicas envolviam o uso de segmentação adaptativa dos pontos de controle: processo que utiliza os parâmetros do passo anterior para refinar a localização dos pontos de controle; ajuste de elipse para achar o centro dos anéis; uso do padrão aneliforme, por oferecer maior precisão na localização dos pontos de controle; e alteração da função objetivo da otimização dos parâmetros da câmera.

Este último se justificou por conta da propriedade física do padrão de calibração, em que os pontos de controle estão dispostos de forma colinear entre si. Diante disso, modelamos uma nova função que considera essa propriedade na hora de otimizar os parâmetros do modelo *pinhole*. Nessa otimização, tenta-se minimizar as distâncias entre os pontos achados pelo processamento de imagem e os pontos reprojetados do modelo de calibração, além dos pontos projetados na linha ajustada.

O uso do padrão aneliforme foi justificado com base nas pesquisas anteriores que já consolidavam esse padrão como mais preciso. A transformação para a visão frontal-paralela, parte do processo iterativo de refinamento dos pontos de controle, se mostrou muito eficaz na redução do erro quadrático médio (RMS). O que se observa no entanto é que não são necessárias muitas iterações para ocorrer essa redução. Muitas vezes na primeira transformação já se obtém o melhor resultado.

O uso da segmentação adaptativa conseguiu reduzir o erro de reprojeção na maioria dos casos na ordem de 10^{-3}. Já a otimização com erro de colinearidade reduziu o erro de reprojeção na ordem de 10^{-5}, para metade dos conjuntos de imagens testados. Para um estudo mais aprofundado dos impactos das sugestões propostas, poderiam ser feitos testes com reconstrução 3D e estereoscopia a fim de avaliar se essas reduções do RMS têm impacto em aplicações de visão computacional ou de robótica.

É interessante notar que utilizamos diferentes câmeras, desde acessíveis, como a de um smartphone, assim como comerciais, como a do Kinect® v1.6, e câmeras com grande distorção, do tipo olho-de-peixe. Ou seja, nossa metodologia se mostrou suficientemente robusta para diversos tipos de câmera.

Conseguimos reproduzir e unificar as metodologias do estado-da-arte numa abordagem que mostrou resultado modesto em relação ao que é considerado o estado-da-arte em calibração, porém promissor. Numa futura progressão desse trabalho, pode-se experimentar novas funções objetivo que consideram outras propriedades do padrão de calibração, como distância física dos pontos de controle, assim como alterar o modelo de distorção de lentes, adicionando por exemplo parâmetros da distorção prismática.

Em se tratando de calibração de câmera e aplicações de visão computacional e realidade aumentada, sabemos que um erro minúsculo de reprojeção no espaço da imagem pode significar centímetros ou decímetros de distância no espaço do mundo. O menor passo que dermos em direção à redução desse erro será válido.

Referências Bibliográficas

[Azuma *et al.*, 2001] AZUMA, R.; BAILLOT, Y.; BEHRINGER, R.; FEINER, S.; JULIER, S. ; MACINTYRE, B.. **Recent advances in augmented reality**. Computer Graphics and Applications, IEEE, 21(6):34–47, 2001. 1

[Bouguet, 2014] BOUGUET, J.-Y.. **Camera calibration toolbox for matlab**. 2004. 1.4, 2.4

[Bradski *et al.*, 2005] BRADSKI, G.; KAEHLER, A. ; PISAREVSKY, V.. **Learning-based computer vision with intel's open source computer vision library.** Intel Technology Journal, 9(2), 2005. 1.2.2

[Bradski e Kaehler, 2008] BRADSKI, G.; KAEHLER, A.. **Learning OpenCV: Computer vision with the OpenCV library**. "O'Reilly Media, Inc.", 2008. 2.4, 4

[Datta *et al.*, 2009] DATTA, A.; KIM, J.-S. ; KANADE, T.. **Accurate camera calibration using iterative refinement of control points**. In: COMPUTER VISION WORKSHOPS (ICCV WORKSHOPS), 2009 IEEE 12TH INTERNATIONAL CONFERENCE ON, p. 1201–1208. IEEE, 2009. 1.1, 1.2.2, 3, 3.1, 4, 4.1, 5

[Fitzgibbon *et al.*, 1999] FITZGIBBON, A.; PILU, M. ; FISHER, R. B.. **Direct least square fitting of ellipses**. Pattern Analysis and Machine Intelligence, IEEE Transactions on, 21(5):476–480, 1999. 3

[Higuchi *et al.*, 2012] HIGUCHI, M.; DATTA, A. ; KANADE, T.. **Software Package for Precise Camera Calibration**, 2012. 4.1, 5, 5.3

[Loaiza *et al.*, 2007] LOAIZA, M.; RAPOSO, A. ; GATTASS, M.. **A novel optical tracking algorithm for point-based projective invariant marker patterns**. In: INTERNATIONAL SYMPOSIUM ON VISUAL COMPUTING, p. 160–169. Springer, 2007. 3

[Loaiza *et al.*, 2011] LOAIZA, M. E.; RAPOSO, A. B. ; GATTASS, M.. **Multi-camera calibration based on an invariant pattern**. Computers & Graphics, 35(2):198–207, 2011. 3

[Prakash e Karam, 2012] PRAKASH, C. D.; KARAM, L. J.. **Camera calibration using adaptive segmentation and ellipse fitting for localizing control points**. In: IMAGE PROCESSING (ICIP), 2012 19TH IEEE INTERNATIONAL CONFERENCE ON, p. 341–344. IEEE, 2012. (document), 1.1, 4, 2, 5, 5.2.1

[Prakash, 2012] PRAKASH, C. D.. **Camera calibration using adaptive segmentation and ellipse fitting for localizing control points**. Master's thesis, ARIZONA STATE UNIVERSITY, 2012. 3

[Sturm *et al.*, 2011] STURM, P.; RAMALINGAM, S.; TARDIF, J.-P.; GASPARINI, S. ; BARRETO, J.. **Camera models and fundamental concepts used in geometric computer vision**. Foundations and Trends® in Computer Graphics and Vision, 6(1–2):1–183, 2011. 1

[Tsai, 1987] TSAI, R.. **A versatile camera calibration technique for high-accuracy 3d machine vision metrology using off-the-shelf tv cameras and lenses**. IEEE Journal on Robotics and Automation, 3(4):323–344, 1987. 2, 3

[Vo *et al.*, 2011] VO, M.; WANG, Z.; LUU, L. ; MA, J.. **Advanced geometric camera calibration for machine vision**. Optical Engineering, 50(11):110503–110503, 2011. 3, 3

[Wei e De Ma, 1994] WEI, G.-Q.; DE MA, S.. **Implicit and explicit camera calibration: Theory and experiments**. IEEE Transactions on Pattern Analysis and Machine Intelligence, 16(5):469–480, 1994. 3

[Weng *et al.*, 1992] WENG, J.; COHEN, P. ; HERNIOU, M.. **Camera calibration with distortion models and accuracy evaluation**. IEEE Transactions on Pattern Analysis & Machine Intelligence, (10):965–980, 1992. 1

[Zhang, 2000] ZHANG, Z.. **A flexible new technique for camera calibration**. Pattern Analysis and Machine Intelligence, IEEE Transactions on, 22(11):1330–1334, 2000. 1.1, 2.1, 2.2, 3, 4.2