Universidade Federal de Juiz de Fora
Instituto de Ciências Exatas
Bacharelado em Ciência da Computação

Calibração Câmera-Projetor de Alta Resolução Utilizando um Padrão Planar

Luiz Maurílio da Silva Maciel

JUIZ DE FORA
DEZEMBRO, 2011

Calibração Câmera-Projetor de Alta Resolução Utilizando um Padrão Planar

Luiz Maurílio da Silva Maciel

Universidade Federal de Juiz de Fora

Instituto de Ciências Exatas

Departamento de Ciência da Computação

Bacharelado em Ciência da Computação

Orientador: Marcelo Bernardes Vieira

Co-orientador: Virgínia Fernandes Mota

JUIZ DE FORA

DEZEMBRO, 2011

Calibração Câmera-Projetor de Alta Resolução Utilizando um Padrão Planar

Luiz Maurílio da Silva Maciel

MONOGRAFIA SUBMETIDA AO CORPO DOCENTE DO INSTITUTO DE CIÊNCIAS EXATAS DA UNIVERSIDADE FEDERAL DE JUIZ DE FORA, COMO PARTE INTEGRANTE DOS REQUISITOS NECESSÁRIOS PARA A OBTENÇÃO DO GRAU DE BACHAREL EM CIÊNCIA DA COMPUTAÇÃO.

Aprovada por:

Marcelo Bernardes Vieira
D. Sc. em Ciência da Computação

Virgínia Fernandes Mota
M. Sc. em Modelagem Computacional

Saul de Castro Leite
D. Sc. em Modelagem Computacional

Rodrigo Luis de Souza da Silva
D. Sc. em Engenharia

JUIZ DE FORA

09 DE DEZEMBRO, 2011

Resumo

A calibração de câmera consiste no processo de determinar atributos geométricos e ópticos de uma câmera, admitindo que sejam conhecidos um conjunto de pares de pontos bidimensionais em uma imagem e seus respectivos pontos tridimensionais no mundo real. Este trabalho realiza o estudo de um processo de calibração de câmera. Este estudo envolve as transformações geométricas de uma câmera, um método para estimar uma aproximação para os parâmetros e outro método para otimizar os parâmetros estimados. Para realizar calibrações, foi implementada uma classe para a resolução de sistemas lineares baseado na decomposição em valores singulares, o método de calibração coplanar de Tsai e o método de otimização de Levenberg-Marquardt. Foram realizados testes com dados conhecidos, a fim de determinar erros do processo de calibração. Em seguida, é feita uma análise dos resultados obtidos.

Palavras-chave: Calibração de câmera. Otimização. Sistemas Lineares.

Abstract

The camera calibration is the process of determining optical and geometric attributes of a camera, assuming it is known a set of pairs of points in a two-dimensional image and their respectives three-dimensional points in the real world. This work conducts a study of a camera calibration process. This study involves the geometric transformations of a camera, a method to estimate an approximation of the parameters and another method to optimize these parameters. To perform calibration, a class to solve linear systems based on singular value decomposition, the method of Tsai coplanar calibration and the Levenberg-Marquardt optimization method were implemented. Tests were conducted with known data in order to determine errors of the calibration process. This is followed by an analysis of the results.

Keywords: Camera calibration. Optimization. Linear Systems.

Agradecimentos

Em primeiro lugar agradeço a Deus por ter me dado a oportunidade de estar aqui. Agradeço a meus pais, Maurílio e Maria Aparecida, pelo imenso apoio e incentivo ao longo dessa trajetória.

Agradeço ao meu orientador Marcelo Bernardes Vieira e a minha co-orientadora Virgínia Fernandes Mota pela dedicação e paciência ao longo deste trabalho. Aos professores do Curso de Ciência da Computação pelos conhecimentos transmitidos ao longo do curso e a todos os demais profissionais da universidade, que de alguma forma contribuíram para que tudo desse certo.

Agradeço também a todos os meus amigos pelo companheirismo, apoio e incentivo durante esses anos. Enfim, agradeço a todos que de alguma forma contribuíram para que eu chegasse até aqui.

"A nossa maior glória não reside no fato de nunca cairmos, mas sim em levantarmo-nos sempre depois de cada queda."

Confúcio

Sumário

Lista de Figuras

Lista de Tabelas

1 Introdução

A **fotografia 3D** consiste na digitalização tridimensional de objetos do mundo real. A obtenção de modelos tridimensionais digitais, isto é, obtenção da geometria e fotometria, pode ser de grande importância em áreas como Desenho Industrial, Arte, Arqueologia, Patrimônio Histórico e Cultural e Educação (Carvalho et al., 2005).

O rápido avanço da tecnologia de fotografia e vídeo faz com que a área da fotografia 3D se torne cada vez mais promissora dentro da computação gráfica. Contudo, para que se possa obter modelos tridimensionais cada vez mais fiéis a realidade é muito importante que se tenha um sistema de calibração com alto grau de precisão, tanto de câmeras quanto de projetores.

Uma das etapas mais importantes nesta área é a calibração de câmera. Isto consiste em obter as características ópticas e geométricas de uma câmera a partir de um conjunto de pares de pontos bidimensionais de uma imagem e seus respectivos pontos tridimensionais no mundo real. Trata-se de um processo de otimização não linear.

Os parâmetros a serem calibrados são divididos em duas categorias: parâmetros intrínsecos e parâmetros extrínsecos. Os parâmetros intrínsecos fornecem as características ópticas e geométricas da câmera tais como distância focal, fatores de escala etc. Os parâmetros extrínsecos fornecem características da orientação e posição da câmera em relação a um sistema de coordenadas global.

O método escolhido para realizar a calibração foi o método de Tsai coplanar proposto em (Tsai, 1987), que é um dos métodos de calibração tradicional e amplamente utilizado. O método utilizado para a otimização dos parâmetros foi o método de Levenberg-Marquardt proposto em (Levenberg, 1944) e (Marquardt, 1963) que é o mais citado em referências sobre calibração.

1.1 Definição do Problema

O problema tratado nesta monografia é o estudo do processo de calibração de câmeras e projetores. Para compreender o processo de calibração é necessário um estudo de suas etapas que envolvem as transformações de câmera, a técnica de calibração utilizada para a obtenção dos valores iniciais dos parâmetros e técnicas de otimização para refinar os resultados obtidos.

1.2 Objetivos

Este trabalho tem como objetivo desenvolver de forma eficiente um módulo de calibração câmera-projetor através da utilização de um padrão planar de forma que a precisão dos parâmetros obtidos seja garantida. Tal módulo poderá futuramente ser utilizado no processo de fotografia 3D e em outros projetos desenvolvidos pelo Grupo de Computação Gráfica, Imagem e Visão da UFJF (GCG-UFJF).

O resultado esperado deste trabalho é um conjunto de classes capaz de realizar de forma eficiente a calibração de uma câmera. Esse conjunto é constituído de três módulos: um responsável por resolver sistemas de equações lineares através do método dos mínimos quadrados; um módulo responsável pela obtenção de uma solução inicial para os parâmetros da câmera a partir dos conjuntos de pontos da imagem e do mundo; e um módulo responsável pelo processo de otimização não linear desses parâmetros. Também será possível realizar a calibração de projetores de modo semelhante à calibração de câmeras.

Os objetivos secundários deste trabalho são:

- Validação da calibração através da comparação com resultados existentes na literatura.
- Teste da calibração para um sistema câmera projetor e verificação se os resultados obtidos correspondem aos esperados.

2 Transformações de Câmera

Para entender o problema de calibração de câmera é necessário analisar as transformações que levam os pontos do mundo real aos pontos da imagem. Este Capítulo tem por objetivo apresentar estas transformações bem como o modelo geométrico no qual elas se baseiam. O conteúdo tem como referência os trabalhos apresentados em (Marques, 2007), (Carvalho et al., 2005) e (Horn, 2000).

2.1 Modelo de Câmera

Um modelo simplificado de uma câmera consiste de uma caixa fechada contendo um pequeno orifício. Ao atravessar este orifício a luz forma uma imagem no plano no fundo da câmera. Tal imagem se forma pela interseção dos raios refletidos pelo objeto que passam pelo orifício da câmera com o fundo da câmera e se apresenta de forma invertida. Esse modelo de câmera é chamado **pinhole** e é ilustrado na Figura 2.1.

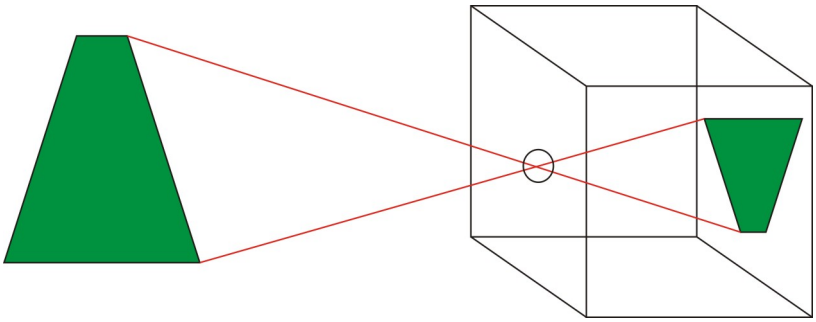


Figura 2.1: Câmera pinhole.

Convenciona-se apresentar a imagem formada na frente da câmera à mesma distância do orifício que o plano do fundo conforme mostra a Figura 2.2. Utilizando uma câmera deste tipo obtém-se um processo de calibração ideal através de transformações projetivas. Porém, para que se obtenha uma imagem nítida através de uma câmera pinhole é necessário um orifício pequeno. Por outro lado a diminuição do orifício diminui a intensidade da luz inviabilizando o uso de uma câmera deste tipo.

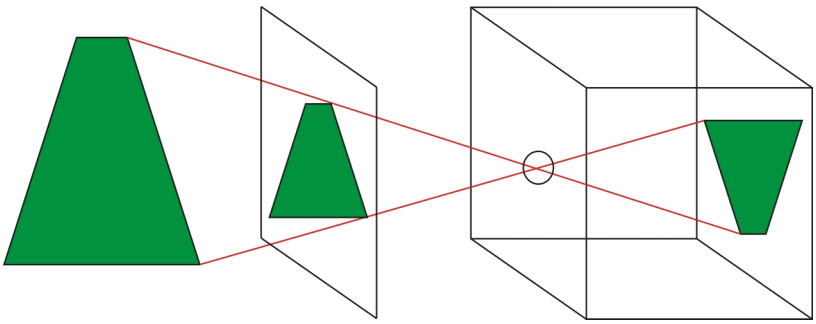


Figura 2.2: Projeção a frente da câmera.

Para solucionar tal problema utilizam-se lentes, pois essas possuem uma abertura maior, permitindo uma intensidade de luz mais elevada. A introdução de lentes, porém, faz com que a câmera apresente distorções. Como exemplo destaca-se a distorção radial que faz com que as bordas da imagem se tornem curvas. A Figura 2.3 ilustra esse tipo de distorção.

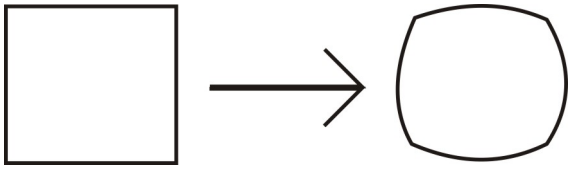


Figura 2.3: Distorção radial.

A introdução das lentes faz com que o processo de calibração de câmera passe a ser modelado por um sistema de equações não lineares. Para resolver esse sistema introduz-se um processo de otimização não linear ao final do processo de calibração.

2.2 Geometria Projetiva

Para efetuar as transformações que levam pontos 3D do mundo real em pontos 2D de uma imagem é necessário o estudo da geometria projetiva e suas transformações. Um exemplo dessas transformações muito usada em computação gráfica é a **projeção perspectiva**.

2.2.1 Projeção Perspectiva

Seja um ponto C o **centro de projeção** (ou **centro óptico**) e um plano situado a uma **distância focal** f. A projeção perspectiva de um ponto tridimensional $P = (x, y, z)$

é representada pelo ponto bidimensional $P' = (x', y')$ correspondente a interseção do plano com a reta CP. O sistema de coordenadas é definido com origem em C e eixo z perpendicular ao plano de projeção conforme a Figura 2.4.

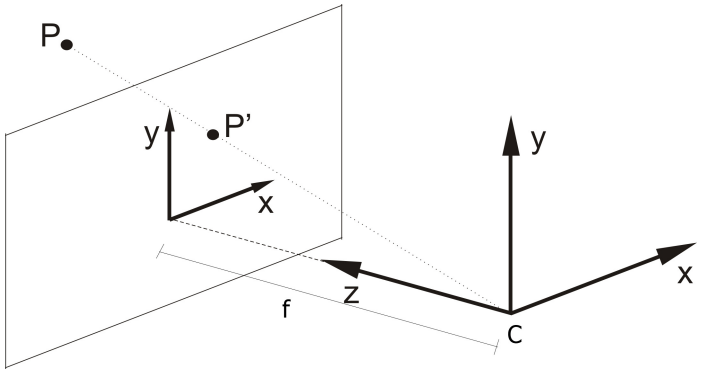


Figura 2.4: Projeção perspectiva.

Os pontos que estão sobre a reta CP são da forma $\alpha(x, y, z)$. O plano de projeção é obtido quando $z = f$. Dessa forma, o ponto P' é tal que $\alpha = f/z$. Obtém-se a seguinte relação:

$$P' = (f\frac{x}{z}, f\frac{y}{z}),$$

com $z \neq 0$.

2.2.2 Espaço Projetivo

O espaço projetivo real de dimensão n, $\mathbb{RP}^n$ é o conjunto de todas as retas de $\mathbb{R}^{n+1}$ que passam pela origem, excluindo a origem. Um ponto projetivo $P = (x_1, x_2, ..., x_{n+1})$ é equivalente a todos os pontos da forma $\lambda P = (\lambda x_1, \lambda x_2, ..., \lambda x_{n+1})$, com $\lambda \neq 0$. Pode-se fazer a associação $\mathbb{RP}^n = \mathbb{R}^{n+1} - \{(0, 0, ..., 0)\}$.

Identificam-se dois tipos de pontos em $\mathbb{RP}^n$:

- Pontos com $x_{n+1} = 1$ são chamados de **pontos afins**. Esses pontos são associados ao plano afim $\pi \equiv \mathbb{R}^n$ mergulhado em $\mathbb{RP}^n$.

- Pontos com $x_{n+1} = 0$ são chamados **pontos no infinito** ou **pontos ideais**. Esses pontos são associados as direções em $\mathbb{R}^n$.

Dessa forma $\mathbb{RP}^n = \{(x_1, x_2, ..., x_n, 1) \cup (x_1, x_2, ..., x_n, 0)\}$.

É importante notar que qualquer ponto projetivo $P \in \mathbb{RP}^n$, com $x_{n+1} \neq 0$, é equivalente a um ponto do plano afim, pois $P = (x_1, x_2, ..., x_n, x_{n+1}) \equiv (\frac{x_1}{x_{n+1}}, \frac{x_2}{x_{n+1}}, ..., \frac{x_n}{x_{n+1}}, 1)$. Dessa forma pode-se dizer que um ponto euclidiano é um ponto projetivo com $x_{n+1} = 1$ (normalizado). Pode-se operar de forma indiferente entre pontos afins e ideais. Quando $x_{n+1} \neq 0$ basta dividir todas as coordenadas por x_{n+1} para encontrar o ponto equivalente no espaço euclidiano.

2.2.3 Transformações Projetivas

Dados dois espaços projetivos $\mathbb{RP}^m$ e $\mathbb{RP}^n$ de dimensões m e n, as **transformações projetivas** são as transformações $T : \mathbb{RP}^m \to \mathbb{RP}^n$ dadas por transformações $T : \mathbb{R}^{m+1} \to \mathbb{R}^{n+1}$.

Considerando transformações em $\mathbb{RP}^3$ tem-se $T : \mathbb{R}^4 \to \mathbb{R}^4$ dada por uma matriz M da seguinte forma:

$$M = \begin{bmatrix} L & T \\ P & S \end{bmatrix} = \begin{bmatrix} l_1 & l_2 & l_3 & t_1 \\ l_4 & l_5 & l_6 & t_2 \\ l_7 & l_8 & l_9 & t_3 \\ p_1 & p_2 & p_3 & s \end{bmatrix}$$

O bloco L da matriz corresponde às transformações lineares e o bloco T corresponde às translações. Esses blocos mantém invariante o plano afim mergulhado e correspondem às transformações afins em $\mathbb{R}^3$.

Considere o caso particular em que L é a matriz identidade 3 x 3, $T = 0$ e $S = 1$. Aplicando essa transformação ao ponto (x, y, z, w) obtém-se:

$$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ p_1 & p_2 & p_3 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ w \end{bmatrix} = \begin{bmatrix} x \\ y \\ z \\ p_1x + p_2y + p_3z + w \end{bmatrix}$$

Dessa forma o bloco P mapeia pontos afins em ideais e vice-versa. O bloco S apenas escala a coordenada w, sendo redundante, pois $T(P) \equiv \lambda T(P)$. Utilizando

as transformações projetivas podem ser modeladas as transformações utilizadas para a visualização.

2.3 Transformações de câmera

Para realizar a correspondência entre os pontos do mundo real e os pontos da imagem capturada pela câmera são necessárias composições de transformações que levam os pontos de um sistema de coordenadas para outro conforme será visto nessa Seção.

2.3.1 Sistemas de Coordenadas

Podem ser identificados quatro sistemas de coordenadas ao longo das transformações de câmera (Tsai, 1987):

- **Sistema de Coordenadas do Mundo (SCM)**: Sistema de coordenadas tridimensional utilizado para representar a cena. A orientação desse sistema é escolhida de modo a tornar conveniente a localização de pontos na cena. Um ponto nesse sistema será representado como (x_w, y_w, z_w).

- **Sistema de Coordenadas da Câmera (SCC)**: Sistema de coordenadas tridimensional com origem no centro óptico da câmera. O eixo z é o eixo óptico e os eixos x e y são paralelos ao plano de projeção. Um ponto nesse sistema será representado como (x_c, y_c, z_c).

- **Sistema de Coordenadas da Imagem (SCI)**: Sistema de coordenadas bidimensional localizado no plano de projeção e cuja origem é a projeção ortogonal do centro óptico sobre esse plano. Um ponto nesse sistema será representado por (x_u, y_u) quando não forem consideradas as distorções da lente e (x_d, y_d) quando as distorções forem consideradas.

- **Sistema de Coordenadas em Pixels (SCP)**: Sistemas de coordenadas bidimensional representando os *pixels* da imagem armazenada. Normalmente a origem desse sistema é o canto superior esquerdo da imagem. Um ponto nesse sistema será representado por (x_f, y_f).

2.3.2 Transformação do SCM para o SCC

Primeiramente é necessário estabelecer a correspondência entre pontos da cena e pontos no sistema de coordenadas da câmera. Trata-se de uma transformação de corpo rígido representada por uma rotação em torno da origem e uma translação.

A transformação de um ponto (x_w, y_w, z_w) do SCM em um ponto (x_c, y_c, z_c) do SCC pode ser equacionada na forma matricial da seguinte forma:

$$\begin{bmatrix} x_c \\ y_c \\ z_c \end{bmatrix} = R \begin{bmatrix} x_w \\ y_w \\ z_w \end{bmatrix} + T$$

onde R é a **matriz de rotação** e T é o **vetor de translação** e correspondem aos parâmetros extrínsecos da câmera. Podem ser representados da seguinte forma:

$$R = \begin{bmatrix} r_1 & r_2 & r_3 \\ r_4 & r_5 & r_6 \\ r_7 & r_8 & r_9 \end{bmatrix}$$

$$T = \begin{bmatrix} t_x \\ t_y \\ t_z \end{bmatrix}$$

2.3.3 Transformação do SCC para o SCI

Depois de obter as coordenadas de um ponto no SCC deve-se fazer uma projeção perspectiva desse ponto sobre o plano de projeção. Considerando f a distância entre o centro óptico e o centro do SCI pode-se equacionar a transformação do SCC para o SCI da seguinte forma:

$$x_u = f\frac{x_c}{z_c}$$
$$y_u = f\frac{y_c}{z_c}.$$

A transformação acima não considera as distorções provocadas pelas lentes. O

parâmetro f é chamado de **distância focal** da câmera. Na forma matricial ela pode ser escrita como:

$$\begin{bmatrix} x_u \\ y_u \\ 1 \end{bmatrix} \backsimeq \begin{bmatrix} f & 0 & 0 & 0 \\ 0 & f & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} x_c \\ y_c \\ z_c \\ 1 \end{bmatrix} = \begin{bmatrix} fx_c & fy_c & z_c \end{bmatrix}$$

2.3.4 Modelagem das Distorções das Lentes

Conforme visto, câmeras precisam utilizar de lentes para ampliar a intensidade de luz recebida. Porém, as lentes possuem o inconveniente de introduzir distorções nas imagens obtidas.

Segundo (Horn, 2000), em sistemas ópticos feitos de superfícies esféricas, ocorre uma distorção na direção radial. Um ponto aparece na imagem em uma distância do centro de projeção maior ou menor do que a predita pela projeção perspectiva. Essa diferença cresce à medida que cresce a distância em relação ao centro. Pode-se modelar a distorção radial através das seguintes séries infinitas:

$$d_x = x_d(k_1\delta^2 + k_2\delta^4 + ...)$$
$$d_y = y_d(k_1\delta^2 + k_2\delta^4 + ...),$$

onde δ é a distância do ponto ao centro de projeção definida como $\delta = \sqrt{x_d^2 + y_d^2}$ e k_1, k_2, ... são chamados **coeficientes de distorção radial**.

Sendo (x_u, y_u) o ponto predito pela projeção perspectiva e (x_d, y_d) o ponto observado devido as distorções pode-se relacionar os dois pontos da seguinte forma:

$$x_u = x_d + d_x$$
$$y_u = y_d + d_y$$

Segundo (Tsai, 1987) somente o coeficiente k_1 deve ser considerado. Uma modelagem mais elaborada para a distorção não ajudaria e poderia causar instabilidade

numérica. Pode-se então escrever as equações finais para a distorção como:

$$x_u = x_d(1 + k_1\delta^2)$$
$$y_u = y_d(1 + k_1\delta^2)$$

2.3.5 Transformação do SCI para o SCP

Uma imagem é representada através de uma matriz de pontos (*pixels*) retangular. Imperfeições na construção dessa matriz fazem com que as linhas e colunas possam ficar desalinhadas. As linhas podem ter espaçamentos diferentes e pode ocorrer de as linhas horizontais não serem exatamente perpendiculares as verticais. A transformação do SCI para o SCP modela essas imperfeições e é ilustrada na Figura 2.5.

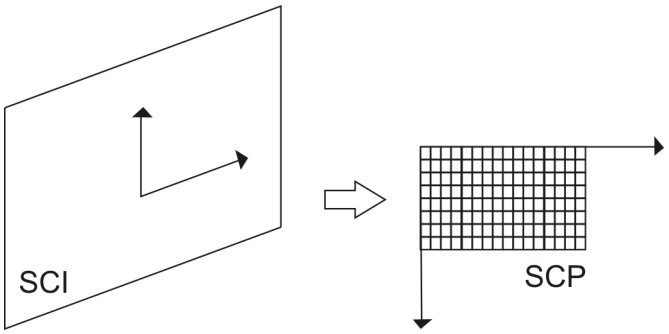


Figura 2.5: Transformação do SCI para o SCP.

Além disso, a origem do SCP é normalmente no canto superior esquerdo enquanto no SCI a origem é a projeção ortogonal do centro óptico. A seguinte transformação afim modela a transformação do SCI para o SCP:

$$\begin{bmatrix} x_f \\ y_f \\ 1 \end{bmatrix} = \begin{bmatrix} s_x & \tau & c_x \\ 0 & s_y & c_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_u \\ y_u \\ 1 \end{bmatrix},$$

onde

- s_x e s_y representam o número de *pixels* por unidade de comprimento nas direções horizontal e vertical respectivamente. Como normalmente os *pixels* são quadrados

costuma-se ter $s_x = s_y$.

- τ representa a tangente do ângulo formado pelas colunas da matriz de *pixels* com a perpendicular às linhas. Normalmente tem-se as linhas e colunas perpendiculares, ou seja, $\tau = 0$.

- c_x e c_y representam a posição na matriz de *pixels* da projeção do centro óptico. Normalmente é representado pelo centro da imagem.

2.3.6 Transformação do SCM para o SCP: Composição das Transformações

A partir da composição das transformações apresentadas nas Seções anteriores pode-se encontrar a transformação que leva do SCM para o SCP. Na forma matricial tem-se a seguinte representação:

$$\begin{bmatrix} x_f \\ y_f \\ 1 \end{bmatrix} \backsimeq \begin{bmatrix} s_x & \tau & c_x \\ 0 & s_y & c_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} f & 0 & 0 & 0 \\ 0 & f & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} R & T \\ 0 & 1 \end{bmatrix} \begin{bmatrix} x_w \\ y_w \\ z_w \end{bmatrix}$$

.

Com base nas transformações apresentadas pode-se estudar o processo de calibração, que consiste na determinação dos parâmetros associados a essas transformações.

3 Método de Tsai Coplanar

O método de calibração utilizado neste trabalho é o método proposto por Tsai (Tsai, 1987) que consiste na obtenção dos parâmetros da câmera em dois estágios. No primeiro estágio são determinados os elementos da matriz de rotação e os elementos t_x e t_y do vetor de translação. No segundo estágio, são determinados o valor de t_z, a distância focal e o coeficiente de distorção k_1.

O método de Tsai requer um padrão conhecido para a calibração. Esse padrão consiste em um objeto do mundo real cujas coordenadas no SCM são facilmente determinadas. O método coplanar utiliza um padrão em que todos os pontos estão em um mesmo plano, ou seja, $z_w = 0$. A Figura 3.1 mostra exemplos de padrões coplanares. Tsai propôs ainda a calibração utilizando um padrão não coplanar em (Tsai, 1987).

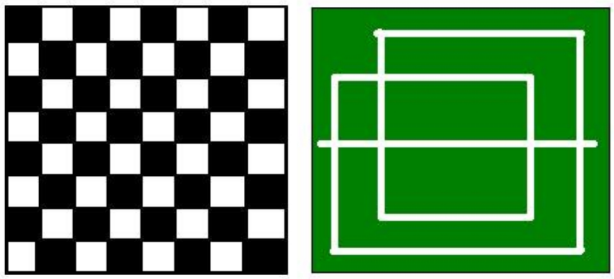


Figura 3.1: Padrão xadrez e padrão de retângulos encaixados proposto em (Szenberg, 2001).

Para realizar a calibração é necessário o prévio conhecimento dos parâmetros intrínsecos da câmera, com exceção do coeficiente de distorção k_1 e da distância focal f. Caso os valores não sejam conhecidos adotam-se os valores $s_x = s_y = 1$ e $\tau = 0$. A projeção do centro óptico (c_x, c_y) é normalmente representada pelo *pixel* do centro da imagem.

A origem do SCM deve ser definida de modo a não se projetar próximo ao centro da imagem ou do eixo x da imagem. Além disso, o SCM e o SCC devem ter a mesma orientação.

Para realizar a calibração da câmera é necessário conhecer um conjunto de n

pontos no padrão de coordenadas (x_{wi}, y_{wi}, z_{wi}) e seus respectivos pontos correspondentes na imagem (x_{fi}, y_{fi}). A seguir serão apresentados os estágios do processo de calibração.

3.1 Primeiro Estágio

O primeiro estágio da calibração consiste em determinar a matriz de rotação e as coordenadas x e y da translação. Uma vez conhecidos os pontos da imagem (x_{fi}, y_{fi}) e a projeção do centro óptico (c_x, c_y) podemos obter as coordenadas dos pontos com distorção (x_{di}, y_{di}) a partir das seguintes relações:

$$x_{di} = x_{fi} - c_x$$

$$y_{di} = y_{fi} - c_y.$$

Compondo as transformações das Seções 2.3.2 e 2.3.3 obtém-se:

$$\begin{bmatrix} x_{ui} \\ y_{ui} \\ 1 \end{bmatrix} \backsimeq \begin{bmatrix} f & 0 & 0 & 0 \\ 0 & f & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} R & T \\ 0 & 1 \end{bmatrix} \begin{bmatrix} x_{wi} \\ y_{wi} \\ z_{wi} \\ 1 \end{bmatrix} \Rightarrow$$

$$\begin{bmatrix} x_{ui} \\ y_{ui} \\ 1 \end{bmatrix} \backsimeq \begin{bmatrix} fr_1 & fr_2 & fr_3 & ft_x \\ fr_4 & fr_5 & fr_6 & ft_y \\ r_7 & r_8 & r_9 & t_z \end{bmatrix} \begin{bmatrix} x_{wi} \\ y_{wi} \\ z_{wi} \\ 1 \end{bmatrix}.$$

De onde se obtém as equações (considerando $z_{wi} = 0$):

$$x_{ui} = f\frac{r_1 x_{wi} + r_2 y_{wi} + t_x}{r_7 x_{wi} + r_8 y_{wi} + t_z} \tag{3.1}$$

$$y_{ui} = f\frac{r_4 x_{wi} + r_5 y_{wi} + t_y}{r_7 x_{wi} + r_8 y_{wi} + t_z}. \tag{3.2}$$

Como foi visto na Seção 2.3.4, pode-se relacionar os pontos com distorção e sem

distorção através das relações:

$$x_{ui} = x_{di}(1 + k_1\delta_i^2)$$

$$y_{ui} = y_{di}(1 + k_1\delta_i^2),$$

onde $\delta_i = \sqrt{x_{di}^2 + y_{di}^2}$.

Encontrando a razão x_{ui}/y_{ui} obtém-se

$$\frac{x_{ui}}{y_{ui}} = \frac{x_{di}(1 + k_1\delta_i^2)}{y_{di}(1 + k_1\delta_i^2)} = \frac{x_{di}}{y_{di}}$$

Portanto, as razões x_{ui}/y_{ui} e x_{di}/y_{di} são iguais e pode-se escrever:

$$\frac{x_{di}}{y_{di}} = f\frac{r_1x_{wi} + r_2y_{wi} + t_x}{r_4x_{wi} + r_5y_{wi} + t_y} \Rightarrow$$

$$x_{di}(r_4x_{wi} + r_5y_{wi} + t_y) = y_{di}(r_1x_{wi} + r_2y_{wi} + t_x) \Rightarrow$$

$$x_{di}t_y = y_{di}r_1x_{wi} + y_{di}r_2y_{wi} + y_{di}t_x - x_{di}r_4x_{wi} - x_{di}r_5y_{wi}. \tag{3.3}$$

Dividindo a Equação 3.3 por t_y obtém-se:

$$y_{di}x_{wi}\left(\frac{r_1}{t_y}\right) + y_{di}y_{wi}\left(\frac{r_2}{t_y}\right) + y_{di}\left(\frac{t_x}{t_y}\right) - x_{di}x_{wi}\left(\frac{r_4}{t_y}\right) - x_{di}y_{wi}\left(\frac{r_5}{t_y}\right) = x_{di}. \tag{3.4}$$

Fazendo a substituição $U_1 = r_1/t_y$, $U_2 = r_2/t_y$, $U_3 = t_x/t_y$, $U_4 = r_4/t_y$ e $U_5 = r_5/t_y$, chega-se a equação:

$$y_{di}x_{wi}U_1 + y_{di}y_{wi}U_2 + y_{di}U_3 - x_{di}x_{wi}U_4 - x_{di}y_{wi}U_5 = x_{di} \tag{3.5}$$

$$\begin{bmatrix} y_{di}x_{wi} & y_{di}y_{wi} & y_{di} & -x_{di}x_{wi} & -x_{di}y_{wi} \end{bmatrix} \begin{bmatrix} U_1 \\ U_2 \\ U_3 \\ U_4 \\ U_5 \end{bmatrix} = x_{di}. \tag{3.6}$$

Para cada ponto de calibração obtém-se uma equação no formato da equação

3.6, resultando no sistema linear $AU = b$. Para $n > 5$ pode-se resolver esse sistema pelo método dos mínimos quadrados e encontrar o vetor $U = [U_1\ U_2\ U_3\ U_4\ U_5]$.

Após calcular o vetor U pode-se então determinar os parâmetros da câmera. Como as linhas da matriz de rotação são ortogonais e unitárias tem-se as seguintes relações:

$$r_1^2 + r_2^2 + r_3^2 = 1 \tag{3.7}$$

$$r_4^2 + r_5^2 + r_6^2 = 1 \tag{3.8}$$

$$r_1 r_4 + r_2 r_5 + r_3 r_6 = 0. \tag{3.9}$$

Substituindo $U_1 = r_1/t_y$, $U_2 = r_2/t_y$, $U_4 = r_4/t_y$ e $U_5 = r_5/t_y$ tem-se:

$$(U_1^2 + U_2^2)t_y^2 + r_3^2 = 1 \tag{3.10}$$

$$(U_4^2 + U_5^2)t_y^2 + r_6^2 = 1 \tag{3.11}$$

$$(U_1 U_4 + U_2 U_5)t_y^2 + r_3 r_6 = 0. \tag{3.12}$$

Isolando r_3^2 em 3.10, r_6^2 em 3.11 e multiplicando as equações obtém-se:

$$r_3^2 r_6^2 = 1 - (U_1^2 + U_2^2 + U_4^2 + U_5^2)t_y^2 + (U_1^2 U_4^2 + U_1^2 U_5^2 + U_2^2 U_4^2 + U_2^2 U_5^2)t_y^4. \tag{3.13}$$

Substituindo 3.13 em 3.12 chega-se a:

$$1 - (U_1^2 + U_2^2 + U_4^2 + U_5^2)t_y^2 + (U_1 U_5 - U_2 U_4)^2 t_y^4 = 0. \tag{3.14}$$

Fazendo $S = (U_1^2 + U_2^2 + U_4^2 + U_5^2)$ e $D = (U_1 U_5 - U_2 U_4)^2$ obtém-se a seguinte equação em t_y^2:

$$D(t_y^2)^2 - S t_y^2 + 1 = 0. \tag{3.15}$$

Resolvendo 3.15 tem-se, para $D \neq 0$:

$$t_y^2 = \frac{S - \sqrt{S^2 - 4D}}{2D}. \tag{3.16}$$

Se $D = 0$, tem-se:

$$t_y^2 = \frac{1}{S}. \tag{3.17}$$

Uma vez calculado o valor de t_y^2 chega-se a $t_y = \pm\sqrt{t_y^2}$. Para determinar o sinal correto de t_y escolhe-se inicialmente o sinal como positivo e calculam-se os valores de r_1, r_2, t_x, r_4 e r_5:

$$r_1 = U_1 t_y$$

$$r_2 = U_2 t_y$$

$$t_x = U_3 t_y$$

$$r_4 = U_4 t_y$$

$$r_5 = U_5 t_y.$$

Escolhe-se então um ponto $(x_{wi}, y_{wi}, 0)$ cuja projeção (x_{di}, y_{di}) esteja suficientemente longe do centro da imagem e calcula-se o valor de $r_1 x_{wi} + r_2 y_{wi} + t_x$. Se tiver o mesmo sinal que x_{di} mantem-se o sinal de t_y. Caso contrário, o sinal de t_y e dos valores calculados a partir do seu valor devem ser trocados. Isso garante que o SCC tenha o seu eixo z orientado na direção da cena.

Pode-se agora calcular os demais elementos da matriz de rotação. Das Equações 3.7 e 3.8 obtém-se:

$$r_3 = \pm\sqrt{1 - r_1^2 - r_2^2} \tag{3.18}$$

$$r_6 = \pm\sqrt{1 - r_4^2 - r_5^2}. \tag{3.19}$$

Assumindo que r_3 seja positivo. Se $r_1 r_4 + r_2 r_5 > 0$, então, pela Equação 3.9, conclui-se que $r_6 < 0$. Caso contrário, $r_6 > 0$. Utilizando novamente a condição de

ortogonalidade de R, calcula-se os demais valores:

$$r_7 = r_2 r_6 - r_3 r_5 \tag{3.20}$$

$$r_8 = r_3 r_4 - r_1 r_6 \tag{3.21}$$

$$r_9 = r_1 r_5 - r_2 r_4. \tag{3.22}$$

3.2 Segundo Estágio

O segundo estágio da calibração consiste em determinar o parâmetro t_z do vetor de translação e a distância focal f. Considerando $k_1 = 0$, tem-se que $(x_{ui}, y_{ui}) = (x_{di}, y_{di})$. A partir das Equações 3.1 e 3.2 obtém-se as seguintes equações:

$$(r_1 x_{wi} + r_2 y_{wi} + t_x) f + x_{di} t_z = (r_7 x_{wi} + r_8 y_{wi}) x_{di} \tag{3.23}$$

$$(r_4 x_{wi} + r_5 y_{wi} + t_y) f + y_{di} t_z = (r_7 x_{wi} + r_8 y_{wi}) y_{di}. \tag{3.24}$$

Obtém-se assim $2n$ equações nas incógnitas t_z e f. O sistema assim obtido pode ser resolvido pelo método dos mínimos quadrados e então encontrar uma aproximação para os valores de t_z e f. Se o valor de f for negativo significa que o sinal de r_3 deve ser negativo. Logo, os sinais de f, r_3, r_6, r_7 e r_8 devem ser trocados.

Os valores de f e t_z calculados anteriormente não consideram a distorção das lentes. Ao introduzir a distorção, o problema torna-se não linear. Utiliza-se então um processo de otimização não linear como o método de Levenberg-Marquardt com os valores de f e t_z calculados anteriormente e $k_1 = 0$ como soluções iniciais. O próximo Capítulo trata do método de Levenberg-Marquardt.

4 Método de Levenberg-Marquardt

Para otimizar os parâmetros encontrados pelo método de calibração de Tsai é necessário um processo de otimização não linear. O método utilizado neste trabalho é o método de Levenberg-Marquardt (Lourakis, 2005; Madsen et al., 2004; Maffra et al., 2008) que é muito utilizado para a solução do problema de mínimos quadrados não linear.

O método de Levenberg-Marquardt encontra, de forma iterativa, o mínimo local de uma função de mais de uma variável expressa como a soma de quadrados de funções não lineares. Quando a solução está próxima da correta, o método se comporta como o método de **Gauss-Newton**. Caso contrário, se comporta como o método do **declive máximo** (***steepest descent***), convergindo de forma lenta.

4.1 O Problema de Mínimos Quadrados

Definição 4.1.1. Seja $x \in \mathbb{R}^n$. Dada uma função $f : \mathbb{R}^n \to \mathbb{R}^m$ com $m \geq n$, o problema de mínimos quadrados consiste em minimizar $||f(x)||$, ou equivalente, encontrar $x^* =$ mínimo local para $F(x)$, onde:

$$F(x) = \frac{1}{2}\sum_{i=1}^{m}(f_i(x))^2 = \frac{1}{2}||f(x)||^2 = \frac{1}{2}f(x)^T f(x). \tag{4.1}$$

O problema de mínimos quadrados tem como objetivo ajustar um conjunto de dados de modo a minimizar a soma dos quadrados das distâncias entre o modelo (curva ajustada) e os pontos dados. Essas distâncias são chamadas **resíduos**. A seguir será apresentado um exemplo retirado de (Madsen et al., 2004).

Considere uma coleção de pontos $(t_1, y_1), (t_2, y_2), ..., (t_m, y_m)$ obtidos de forma experimental e o modelo $M(x, t) = x_3 e^{x_1 t} + x_4 e^{x_2 t}$. Dado $x \in \mathbb{R}^4$ tem-se que os resíduos podem ser definidos como:

$$f_i(x) = y_i - M(x, t_i) = y_i - x_3 e^{x_1 t_i} - x_4 e^{x_2 t_i},$$

com $i = 1, ..., m$.

O objetivo do problema de mínimos quadrados é encontrar o valor de $x^* = (x_1^*, x_2^*, x_3^*, x_4^*)$ que minimiza a soma dos quadrados dos resíduos, o que equivale a encontrar a curva que melhor ajusta os pontos fornecidos. A Figura 4.1 mostra os pontos fornecidos e a curva obtida no ponto x^*, ou seja, $M(x^*, t)$.

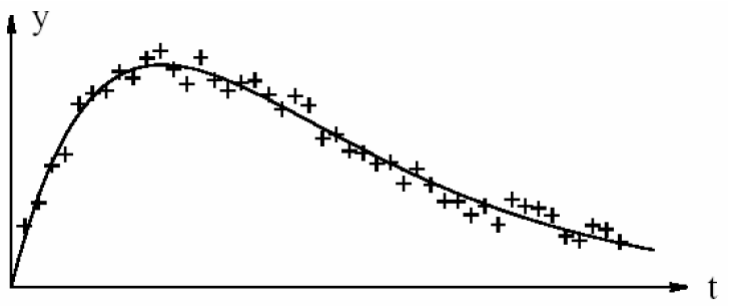


Figura 4.1: Problema de mínimos quadrados.

O método de Levenberg-Marquardt se baseia no método de Gauss-Newton e este se baseia no método de Newton. A seguir será feito um estudo desses métodos para a solução do problema de mínimos quadrados não linear.

4.2 Método de Newton

O método de Newton é baseado no fato de que se x^* é um ponto crítico de F, então $F'(x^*) = 0$. Fazendo a expansão de F em série de Taylor obtém-se:

$$F(x + h) = F(x) + hF'(x) + O(||h||^2) \approx F(x) + hF'(x),$$

para valores pequenos de $||h||$.

De modo semelhante, tem-se:

$$F'(x + h) = F'(x) + hF''(x) \tag{4.2}$$

Sabendo-se que $F'(x^*) = 0$, o método de Newton tem por objetivo encontrar a

direção de busca representada pelo vetor h_n, obtido através do seguinte sistema:

$$Hh_n = -F'(x), \tag{4.3}$$

onde $H = F''(x)$ é chamada de **matriz Hessiana**:

$$H = \begin{bmatrix} \frac{\partial^2 F}{\partial x_1^2}(x) & \frac{\partial^2 F}{\partial x_1 \partial x_2}(x) & \dots & \frac{\partial^2 F}{\partial x_1 \partial x_n}(x) \\ \frac{\partial^2 F}{\partial x_2 \partial x_1}(x) & \frac{\partial^2 F}{\partial x_2^2}(x) & \dots & \frac{\partial^2 F}{\partial x_2 \partial x_n}(x) \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial^2 F}{\partial x_n \partial x_1}(x) & \frac{\partial^2 F}{\partial x_2 \partial x_n}(x) & \dots & \frac{\partial^2 F}{\partial x_n^2}(x) \end{bmatrix}.$$

A próxima iteração do método é calculada como:

$$x \leftarrow x + h_n.$$

Partindo de um ponto inicial x_0, são gerados novos vetores $x_1, x_2, ...$ até convergir para o mínimo local x^*. O inconveniente do método de Newton é que o cálculo da matriz Hessiana pode ser muito complexo. Para resolver esse problema são propostos métodos que aproximam a matriz Hessiana.

4.3 Método de Gauss-Newton

O método de Gauss-Newton é a base para o método de Levenberg-Marquardt. Esse método é baseado em uma aproximação linear das componentes de f nas vizinhanças do ponto x (Madsen et al., 2004). Para pequenos valores de h, tem-se pela expansão em série de Taylor:

$$f(x + h) \approx l(h) \equiv f(x) + Jh \tag{4.4}$$

onde J é a **matriz Jacobiana** de f:

$$J = \begin{bmatrix} \frac{\partial f_1}{\partial x_1}(x) & \frac{\partial f_1}{\partial x_2}(x) & \dots & \frac{\partial f_1}{\partial x_n}(x) \\ \frac{\partial f_2}{\partial x_1}(x) & \frac{\partial f_2}{\partial x_2}(x) & \dots & \frac{\partial f_2}{\partial x_n}(x) \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial f_m}{\partial x_1}(x) & \frac{\partial f_m}{\partial x_2}(x) & \dots & \frac{\partial f_n}{\partial x_n}(x) \end{bmatrix}$$

Substituindo 4.4 na Equação 4.1 obtém-se:

$$F(x+h) \approx L(h) \equiv \frac{1}{2} l(h)^T l(h)$$

$$= \frac{1}{2} f(x)^T f(x) + h^T J^T f(x) + \frac{1}{2} h^T J^T J h$$

$$= F(x) + h^T J^T f(x) + \frac{1}{2} h^T J^T J h \tag{4.5}$$

Da Equação 4.5 observa-se que o gradiente $L'(h)$ e a matriz Hessiana $L''(h)$ são dados por:

$$L'(h) = J^T f(x) + J^T J h$$

$$L''(h) = J^T J$$

Usando o fato de que $F'(x) = J^T f(x)$, pode-se equacionar a direção de busca do método de Gauss-Newton como:

$$(J^T J) h_{gn} = -J^T f(x) \tag{4.6}$$

Dessa forma, a próxima iteração é dada por:

$$x \leftarrow x + h_{gn}.$$

Apesar de encontrar uma aproximação mais simples para a matriz Hessiana, o método de Gauss-Newton possui o inconveniente de que essa matriz pode não ter inversa, não sendo possível resolver o sistema da Equação 4.6. O método de Levenberg-Marquardt procura resolver esse problema.

4.4 Método de Levenberg-Marquardt

Levenberg (Levenberg, 1944), e mais tarde Marquardt (Marquardt, 1963), propuseram uma mudança no cálculo da matriz Hessiana do método de Gauss-Newton. Essa mudança consiste em somar uma parcela μI à matriz $J^T J$, onde μ é chamado **parâmetro de amortecimento** (***damping***) e I é a matriz identidade.

A direção de busca passa a ser calculada como:

$$(J^T J + \mu I)h_{lm} = -g, \tag{4.7}$$

onde $g = J^T f(x)$ é o gradiente e $\mu \geq 0$.

De acordo com (Madsen et al., 2004), o parâmetro μ tem diferentes efeitos sobre o método:

- Para todo $\mu > 0$ a matriz $(J^T J + \mu I)$ é positiva definida, o que garante que h_{lm} é uma direção de descida.
- Para valores grandes de μ tem-se

  $$h_{lm} \approx -\frac{1}{\mu} g = -\frac{1}{\mu} F'(x),$$

  que é um pequeno passo na direção máxima de descida. Isso é bom quando se está longe da solução.
- Se μ é muito pequeno, tem-se $h_{lm} \approx h_{gn}$. Isso é bom nos últimos estágios, quando se está próximo da solução.

O valor inicial do parâmetro μ é calculado com base no tamanho dos elementos da matriz $A = J^T J$ calculada no ponto x_0:

$$\mu = \tau max\{a_{ii}\},$$

onde τ é definido pelo usuário. Quando x está próximo de x^* recomenda-se $\tau = 10^{-6}$. Caso contrário pode-se adotar $\tau = 10^{-3}$ ou $\tau = 1$ (Maffra et al., 2008).

O valor de μ é atualizado com base no valor de um outro parâmetro ρ que controla o ganho obtido por aquela iteração do método. Esse parâmetro é calculado como:

$$\rho = \frac{F(x) + F(x + h_{lm})}{L(0) - L(h_{lm})}, \tag{4.8}$$

onde $L(0) - L(h_{lm})$ é o ganho obtido pelo modelo linear e é calculado como:

$$L(0) - L(h_{lm}) = \frac{1}{2} h_{lm}^T (\mu h_{lm} - g)$$

Se o valor de ρ for grande indica que $L(h_{lm})$ é uma boa aproximação para $F(x + h_{lm})$. Então o valor de μ pode ser diminuído. Se o valor de ρ for pequeno (talvez até negativo) deve-se aumentar o valor de μ para buscar a direção máxima de descida (Madsen et al., 2004).

A regra para alterar o valor de μ é:

se $\rho > 0$, então $v = 2$;

senão, $\mu = v\mu$ e $v = 2v$;

onde v é inicializado com 2.

Segundo (Madsen et al., 2004), os critérios de parada do método de Levenberg-Marquardt são:

- A norma infinita do gradiente seja menor do que um ϵ_1 pequeno escolhido pelo usuário, isto é, $||g||_\infty \leq \epsilon_1$.
- Pequena variação em x, ou seja, $||x_{novo} - x|| \leq \epsilon_2(||x|| + \epsilon_2)$, onde ϵ_2 é determinado pelo usuário.
- O número máximo de iterações é atingido, ou seja, $k \geq k_{max}$.

5 Modelo Computacional

Neste capítulo será apresentado o modelo computacional implementado neste trabalho. Esse modelo consiste em três módulos:

- **gcgLinearSolver**: módulo para resolver sistemas de equações lineares;
- **gcgLevMarq**: módulo que implementa o método de Levenberg-Marquardt;
- **gcgCalibrator**: módulo para realizar a calibração de câmeras e projetores.

O diagrama de classes da Figura 5.1 mostra as classes desenvolvidas e seus relacionamentos.

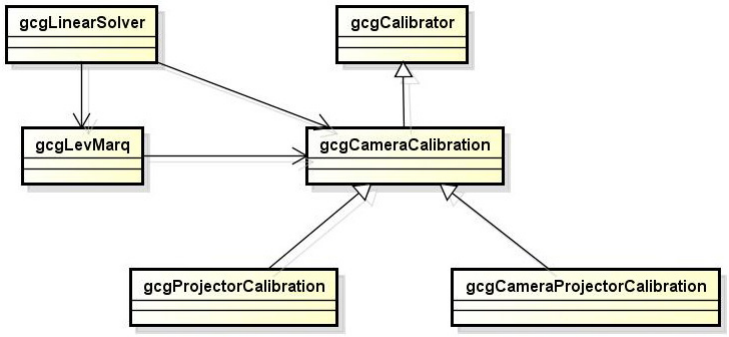


Figura 5.1: Classes e relacionamentos.

5.1 Módulo gcgLinearSolver

A classe **gcgLinearSolver** foi implementada para a solução de sistemas de equações lineares da forma $Ax = b$ com m equações e n incógnitas, onde $m \geq n$. Para a implementação foi utilizada a biblioteca **gcgLib** que contém as classes **gcgDISCRETE1D** e **gcgDISCRETE2D**, utilizadas para representar vetores e matrizes, respectivamente. A Figura 5.2 mostra os atributos e métodos dessa classe.

O método **PYTHAG** tem a função de calcular $(a^2 + b^2)^{1/2}$ sem *underflow* ou *overflow*. É utilizada no método **dsvd**. Esse método calcula a **decomposição em valores singulares** (SVD) da matriz A.

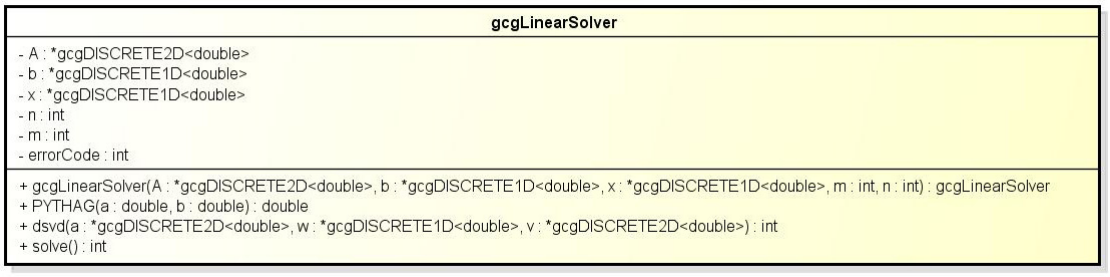


Figura 5.2: Classe gcgLinearSolver.

A decomposição em valores singulares é um dos métodos mais utilizados para resolver problemas de mínimos quadrados lineares. Por este motivo, foi o escolhido para o desenvolvimento deste trabalho. O método implementado foi baseado na proposta de (Press et al., 1992). O método **solve** determina a solução do sistema a partir da decomposição calculada pelo método **dsvd**.

Para resolver um sistema de equações m x n, com $m \geq n$, basta criar uma instância da classe **gcgLinearSolver** e invocar o método **solve**. A solução do sistema ficará armazenada no atributo **x** da instância.

Duas situações podem gerar erro na solução do sistema. A primeira é o caso em que o número de equações informado é menor do que o número de incógnitas. A segunda é quando a rotina que determina a decomposição em valores singulares da matriz A não converge após atingido o número máximo de iterações.

5.2 Módulo gcgLevMarq

O método de Levenberg-Marquardt foi implementado através da classe abstrata **gcgLevMarq**. A Figura 5.3 mostra os atributos e métodos dessa classe. O algoritmo implementado foi retirado de (Lourakis, 2005) e é descrito no Algoritmo 1.

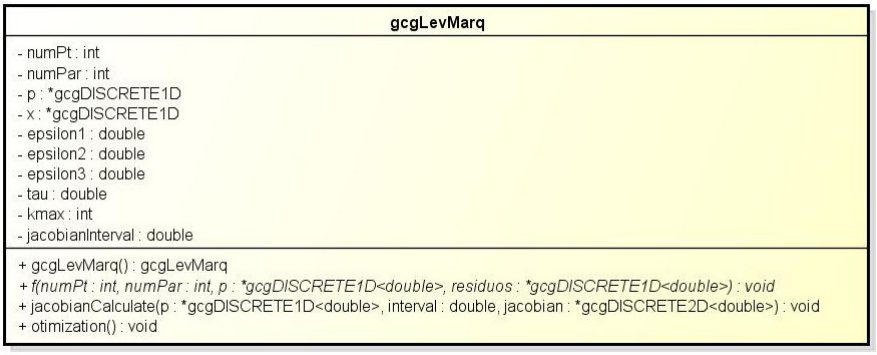


Figura 5.3: Classe gcgLevMarq.

A matriz J é a matriz jacobiana e é calculada através da fórmula de diferenças finitas central conforme indica a Equação 5.1.

$$f'(x) \approx \frac{f(x+h) - f(x-h)}{2h}, \tag{5.1}$$

para pequenos valores de h. O valor de h é um dos atributos da classe (**jacobianInterval**) e seu valor padrão é 10^{-3}. As tolerâncias do método são inicializados da seguinte forma $\epsilon_1 = \epsilon_2 = \epsilon_3 = 10^{-15}$, segundo sugestão de (Lourakis, 2005). O valor de τ depende da proximidade do ponto inicial conforme discutido na Seção 4.4. O valor de $kmax$ também dependerá do problema a ser tratado.

Para resolver o sistema $((A + \mu I)\delta_p) = g$ foi criada uma instância da classe **gcgLinearSolver** descrita na seção anterior. A classe **gcgLevMarq** possui um método virtual e abstrato correspondente a função que se deseja minimizar. Para utilizar a classe é necessário criar uma classe derivada de **gcgLevMarq** que implemente o método abstrato **void f(int numPt, int numPar, gcgDISCRETE1D<double> *p, gcgDISCRETE1D <double> *residuos)**, onde:

- **numPt**: tamanho do vetor imagem de f. Normalmente é o número de pontos a serem ajustados.
- **numPar**: número de parâmetros a serem determinados. Deve-se ter **numPt** > **numPar**.
- **p**: vetor com os parâmetros.

- **residuos**: vetor para armazenar os resíduos calculados.

A classe derivada de **gcgLevMarq** pode então ser instanciada e a otimização pode ser feita invocando o método **otimization** que corresponde a implementação do Algoritmo 1.

Algoritmo 1: Método de Levenberg-Marquardt.

Entrada: uma função vetorial $f : \mathbb{R}^m \rightarrow \mathbb{R}^n$ com $n \geq m$, um vetor de medida $x \in \mathbb{R}^n$ e uma estimativa inicial dos parâmetros $p_0 \in \mathbb{R}^m$.
Saída: $p^+ \in \mathbb{R}^m$ minizando $||x - f(p)||^2$.
início
  $k := 0$; $v := 2$; $p := p_0$;
  $A := J^T J$; $x_p := x - f(p)$; $g := J^T x_p$;
  $pare := (||g||_\infty \leq \epsilon_1)$; $\mu := \tau * max_{i=1,2,...m}(A_{ii})$;
  enquanto *!pare* ***e*** $k \leq k_{max}$ **faça**
    $k := k + 1$;
    repita
      resolva $((A + \mu I)\delta_p) = g$;
      se $||\delta_p|| \leq \epsilon_2 ||p||$ **então**
        $pare :=$ **verdadeiro**;
      senão
        $p_{novo} := p + \delta_p$;
        $\rho := (||x_p||^2 - ||x - f(p_{novo})||^2)/(\delta_p^T(\mu\delta_p + g))$;
        se $\rho > 0$ **então**
          $p := p_{novo}$;
          $A := J^T J$; $x_p := x - f(p)$; $g := J^T x_p$;
          $pare := (||g||_\infty \leq \epsilon_1)$ **ou** $(||x_p||^2 \leq \epsilon_3)$;
          $\mu := \mu * max(\frac{1}{3}, 1 - (2\rho - 1)^3)$; $v := 2$;
        senão
          $\mu := \mu * v$; $v := 2 * v$;
        fim se
      fim se
    até $\rho > 0$ ***ou*** *pare*;
  fim enqto
  $p^+ := p$;
fim

5.3 Módulo gcgCalibrator

Para a realização da calibração de câmeras e projetores foi criada uma classe base **gcgCalibrator**. Essa classe contém os parâmetros da câmera como atributos, conforme mostra a Figura 5.4.

Foram criadas classes derivadas da classe **gcgCalibrator** para a calibração de

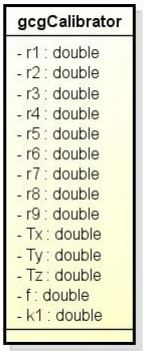


Figura 5.4: Classe gcgCalibrator.

câmeras e projetores.

5.3.1 Calibração de Câmeras

Para realizar a calibração de uma câmera foi criada a classe **gcgCameraCalibration** derivada de **gcgCalibrator**. A Figura 5.5 mostra os atributos e métodos dessa classe.

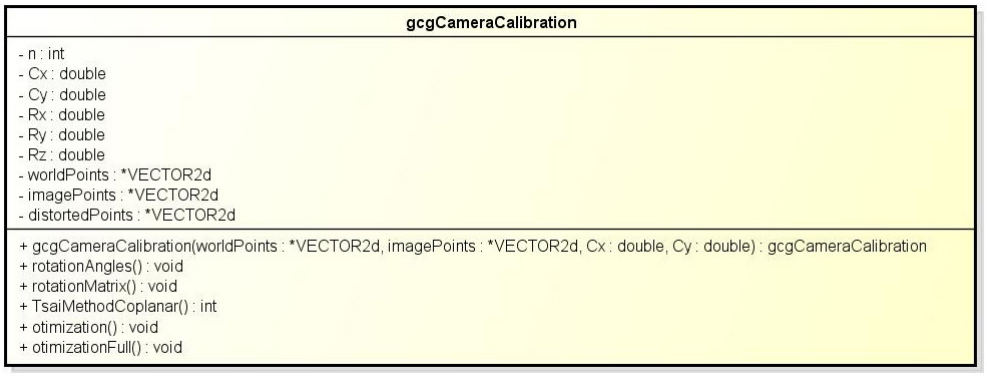


Figura 5.5: Classe gcgCameraCalibration.

Essa classe possui como atributos próprios:

- **n**: número de pontos para a calibração.
- **Cx e Cy**: coordenadas da projeção do centro óptico.
- **Rx, Ry, Rz**: ângulos da rotação.
- **worldPoints**: arranjo para armazenar os pontos do mundo.

- **imagePoints**: arranjo para armazenar os pontos da imagem.

- **distortedPoints**: arranjo para armazenar os pontos com distorção.

A estrutura **VECTOR2d** pertence a biblioteca **gcgLib** e corresponde a um arranjo de duas posições do tipo **double**. Essa estrutura é utilizada para representar um ponto no espaço bidimensional. Os pontos do mundo são considerados bidimensionais pois a calibração implementada usa um padrão planar ($z_w = 0$).

O método **TsaiMethodCoplanar** implementa o método de Tsai coplanar descrito no Capítulo 3. A solução dos sistemas lineares do método é feita utilizando a classe **gcgLinearSolver** descrita na Seção 5.1. Para a otimização dos parâmetros foi criada uma classe derivada da classe **gcgLevMarq** descrita na seção 5.2. Essa classe derivada implementa a seguinte função para otimização:

$$f_i = (x_{ui} - x'_{ui})^2 + (y_{ui} - y'_{ui})^2. \tag{5.2}$$

onde (x_{ui}, y_{ui}) é o ponto sem distorção calculado a partir da projeção dos pontos do mundo (x_{wi}, y_{wi}) e (x'_{ui}, y'_{ui}) é o ponto sem distorção calculado a partir dos pontos com distorção (x_{di}, y_{di}). Essa equação para a otimização foi proposta em (Wilson, 1994).

Pode ser feita a otimização dos parâmetros T_z, f e k_1 conforme proposto por (Tsai, 1987) ou pode ser feita otimização de todos os parâmetros conforme propõe (Wilson, 1994). Nesse útimo caso, é necessário calcular os ângulos de rotação, realizar a otimização utilizando esses ângulos e depois recalcular a matriz de rotação a partir dos ângulos otimizados.

Para realizar a calibração de uma câmera é necessário criar uma instância da classe **gcgCameraCalibration** passando o número de pontos, os pontos do mundo, os pontos da imagem e a projeção do centro óptico. Ao invocar o método **TsaiMethodCoplanar** os parâmetros são calculados e armazenados nos atributos da classe.

5.3.2 Calibração de Projetores

Para calibrar um projetor é necessário que se tenha uma câmera previamente calibrada. Como o projetor projeta uma imagem no mundo real, deve-se capturar essa imagem

projetada. Pode-se projetar a imagem sobre o mesmo plano utilizado para a calibração da câmera e fotografar essa projeção.

É conveniente expressar as coordenadas dos pontos projetados no SCC para facilitar a transformação entre os sistemas da câmera e do projetor (Carvalho et al., 2005). A relação entre pontos da imagem do padrão projetado (x_{fi}, y_{fi}) e o SCC (que será considerado o SCM para a calibração do projetor) é dada pela equação:

$$\begin{bmatrix} x_{wi} \\ y_{wi} \\ z_{wi} \end{bmatrix} = \lambda \begin{bmatrix} r_1 & r_4 & r_7 \\ r_2 & r_5 & r_8 \\ r_3 & r_6 & r_9 \end{bmatrix} \begin{bmatrix} x_{fi} \\ y_{fi} \\ f \end{bmatrix} - \begin{bmatrix} r_1 & r_4 & r_7 \\ r_2 & r_5 & r_8 \\ r_3 & r_6 & r_9 \end{bmatrix} \begin{bmatrix} T_x \\ T_y \\ T_z \end{bmatrix}. \tag{5.3}$$

Como se consideram pontos com $z_{wi} = 0$, o valor de λ é dado por:

$$\lambda = \frac{r_3 T_x + r_6 T_y + r_9 T_z}{r_3 x_{fi} + r_6 y_{fi} + r_9 f}. \tag{5.4}$$

Após obter os pontos do mundo a partir da Equação 5.3, o projetor pode ser calibrado da mesma forma que uma câmera.

Para realizar a calibração de um projetor foi implementada a classe **gcgProjectorCalibration** derivada da classe **gcgCameraCalibration**. A classe **gcgProjectorCalibration** possui como atributos próprios os atributos da câmera previamente calibrada que será utilizada para calibrar o projetor. A Figura 5.6 mostra essa classe.

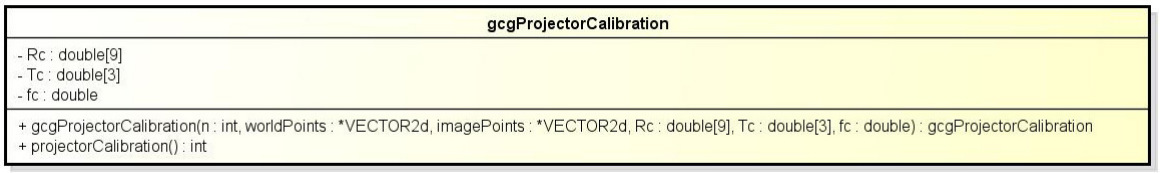


Figura 5.6: Classe gcgProjectorCalibration.

Ao criar uma instância da classe **gcgProjectorCalibration** devem ser informados: o número de pontos, os pontos da imagem da cena do padrão projetado capturada pela câmera (pontos do mundo), os pontos da imagem projetada e o centro de projeção do projetor. A calibração é realizada invocando o método **projectorCalibration**. Esse

método realiza a transformação da Equação 5.3 e calibra o projetor da mesma forma que uma câmera.

Também foi criada a classe **gcgCameraProjectorCalibration** derivada da classe **gcgCameraCalibration** que permite calibrar câmera e projetor juntos. A classe contém como atributo próprio uma instância da classe **projectorCalibration** utilizada para calibrar o projetor.

Ao criar uma instância da classe **gcgCameraProjectorCalibration** devem ser passados como parâmetros os dados necessários para a calibração da câmera e os dados necessários para a calibração do projetor. O método **cameraProjectorCalibration** realiza a calibração da câmera e utiliza os parâmetros obtidos para a determinação dos parâmetros do projetor. Ao final, tem-se câmera e projetor calibrados.

É importante destacar que como considerou-se os valores $s_x = s_y = 1$, a distância focal calculada pelas calibrações é expressa em *pixels*. Para determinar o valor real é necessário conhecer a dimensão de cada *pixel* (Carvalho et al., 2005).

6 Resultados Experimentais

Neste capítulo serão descritos os experimentos realizados para testar e avaliar os módulos desenvolvidos.

6.1 Módulo gcgLinearSolver

A classe **gcgLinearSolver** foi testada para diversos sistemas de equações lineares cuja solução era previamente conhecida e comparou-se o resultado obtido pela função **solve**. A seguir serão apresentados dois desses sistemas.

O primeiro sistema testado foi:

$$\begin{bmatrix} 1 & 1 & 1 \\ 1 & 2 & 4 \\ 1 & 3 & 9 \\ 1 & 4 & 16 \\ 1 & 5 & 25 \\ 1 & 6 & 36 \\ 1 & 7 & 49 \\ 1 & 8 & 64 \\ 1 & 9 & 81 \\ 1 & 10 & 100 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} 444 \\ 458 \\ 478 \\ 493 \\ 506 \\ 516 \\ 523 \\ 531 \\ 543 \\ 571 \end{bmatrix}$$

A Tabela 6.1 mostra a solução esperada para esse sistema e a solução obtida com pela classe **gcgLinearSolver**.

Variável	Valor esperado	Valor encontrado	Erro
x_1	431.7833	431.7833	3.3333e-5
x_2	14.9538	14.9538	1.2121e-5
x_3	-0.2008	-0.2008	4.0156e-5

Tabela 6.1: Resultado primeiro sistema.

A seguir tem-se outro exemplo:

$$\begin{bmatrix} 1 & -7.8 & -7.6 \\ 1 & -3.2 & -7.2 \\ 1 & -4 & 2.8 \\ 1 & 2.1 & 1.4 \\ 1 & 1.1 & -2 \\ 1 & -6.1 & 5.2 \\ 1 & 0.4 & -6.8 \\ 1 & -6.8 & -7.1 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} -4.2 \\ -5.3 \\ -1 \\ 4.85 \\ 1.85 \\ -1.75 \\ -2.7 \\ -7.95 \end{bmatrix}$$

O resultado obtido para esse sistema é mostrado na Tabela 6.2.

Variável	Valor esperado	Valor encontrado	Erro
x_1	1.1957	1.1957	1.8892e-5
x_2	0.7001	0.7001	2.7062e-5
x_3	0.4110	0.4110	2.9273e-5

Tabela 6.2: Resultado segundo sistema.

A análise desses resultados e de outros sistemas testados demonstrou que o módulo **gcgLinearSolver** funciona de maneira adequada, pois encontra a solução do sistema com erros pequenos, na ordem de 10^{-5}, em relação ao valor esperado.

6.2 Módulo gcgLevMarq

A classe **gcgLevMarq** foi testada com diversas funções não lineares cuja solução ótima era conhecida e comparou-se com o resultado obtido pela função **otimization**. A seguir são apresentadas duas dessas funções.

O primeiro exemplo é a função de Bard descrita na Equação 6.1.

$$f(x) = \frac{1}{2} \sum_{k=1}^{15} f_k^2(x), \tag{6.1}$$

onde $x = (x_1, x_2, x_3)$, $f_k(x) = y_k - (x_1 + [(u_k)/(v_k x_2 + w_k x_3)])$, com $u_k = k$, $v_k = 16 - k$, $w_k = min(u_k, v_k)$ e $y = (0.14, 0.18, 0.22, 0.25, 0.29, 0.32, 0.35, 0.39, 0.37, 0.58, 0.73, 0.96, 1.34,$

2.10, 4.39).

Utilizando como ponto inicial o ponto $x_0 = (1, 1, 1)$ verificou-se que o método convergiu em 7 iterações, para $\tau = 10^{-6}$. Para testar a capacidade do método de encontrar a solução a partir de um ponto inicial distante, outro teste foi realizado para o ponto $x_0 = (100, 100, 100)$ com $\tau = 10^{-3}$. Nesse caso, o método convergiu em 20 iterações. Em ambos os casos a convergência foi para o mesmo valor. A Tabela 6.3 mostra a solução esperada e os resultados obtidos.

Variável	Valor esperado	Valor encontrado	Erro
x_1	0.0824	0.0824	4.0809e-8
x_2	1.1330	1.1330	3.9132e-6
x_3	2.3437	2.3437	4.8163e-6

Tabela 6.3: Resultado função de Bard.

Como segundo exemplo tem-se a função de Wood, definida como:

$$\begin{aligned} F(x_0, x_1, x_2, x_3) &= 100(x_1 - x_0^2)^2 + (x_0 - 1)^2 + 90(x_3 - x_2^2)^2 + (1 - x_2)^2 \\ &\quad +10.1[(x_1 - 1)^2 + (x_3 - 1)^2] + 19.8(x_1 - 1)(x_3 - 1) \end{aligned} \tag{6.2}$$

Foi utilizado como ponto inicial $x = (-3, -1, -3, -1)$. Utilizando $\tau = 10^{-6}$ o método convergiu em 66 iterações e com $\tau = 10^{-3}$ o método convergiu em 64 iterações. A Tabelas 6.4 e 6.5 mostram os resultados obtidos para esse problema:

	Valor esperado	Valor encontrado	Erro
x_0	1.0000	1.0000	5.5885e-9
x_1	1.0000	1.0000	1.3200e-8
x_2	1.0000	1.0000	6.5891e-9
x_3	1.0000	1.0000	1.3200e-8

Tabela 6.4: Resultado função de Wood ($\tau = 10^{-6}$).

	Valor esperado	Valor encontrado	Erro
x_0	1.0000	1.0000	1.3286e-8
x_1	1.0000	1.0000	2.6621e-8
x_2	1.0000	1.0000	1.3290e-8
x_3	1.0000	1.0000	2.6621e-8

Tabela 6.5: Resultado função de Wood ($\tau = 10^{-3}$).

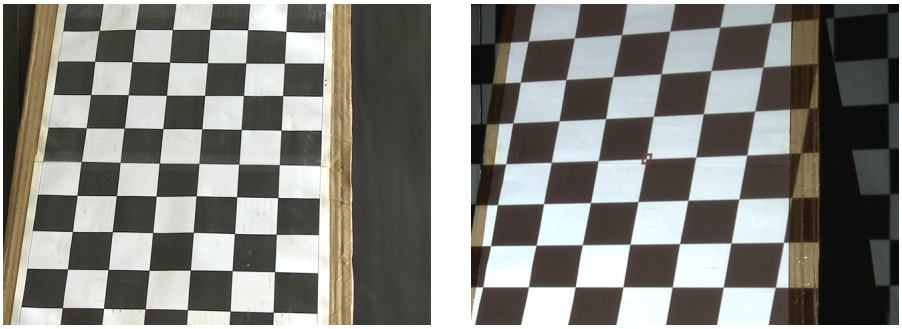
A análise dos resultados obtidos para essas e outras funções permitem concluir

que o módulo **gcgLevMarq** funciona adequadamente, pois encontra a solução do sistema com erros pequenos, variando na ordem de 10^{-6} a 10^{-9}, em relação ao valor esperado.

6.3 Módulo gcgCalibrator

Para testar o módulo de calibração foram realizados diversos testes com dados de câmeras e projetores. Foram utilizados tanto conjuntos de pontos previamente obtidos quanto pontos extraídos de imagens obtidas experimentalmente. A seguir serão descritos dois experimentos.

O padrão utilizado para calibrar a câmera é formado por quadrados de 4 cm de lado. A imagem projetada pelo projetor é formada por retângulos de 80 x 60 *pixels*. A Figura 6.1 mostra as imagens utilizadas para a primeira calibração.



(a) Câmera (b) Projetor

Figura 6.1: Imagens para primeira calibração.

Após extrair os pontos das imagens, foram realizadas três tipos de calibração utilizando a implementação desenvolvida: sem otimização, com otimização nos parâmetros Tz, f e k_1, e com otimização em todos os parâmetros. Para fins de comparação, foi realizada, também, a calibração utilizando um código desenvolvido por (Wan, 2004) em Matlab[1]. Esse código realiza otimização nos parâmetros T_z, f e k_1.

A Tabela 6.6 mostra os pontos utilizados nesse exemplo para a câmera e a Figura 6.7 mostra os pontos utilizados para o projetor. Os pontos (x_{wi}, y_{wi}) correspondem aos pontos extraídos da imagem da Figura 6.6. Para realizar a calibração do projetor com esses pontos, é aplicada a transformação da Equação 5.3. A projeção do centro óptico foi

[1]Matlab é uma marca registrada de *The MathWorks, Inc.*

assumida como sendo o centro o ponto (320, 240) para a câmera e o ponto (400, 300) para o projetor correspondente ao centro das imagens. No caso em que todos os parâmetros são otimizados, também ocorre otimização do centro.

x_{wi}	y_{wi}	x_{fi}	y_{fi}	x_{wi}	y_{wi}	x_{fi}	y_{fi}
0.000000	0.000000	145.000000	45.000000	12.000000	16.000000	293.217407	238.913925
4.000000	0.000000	198.100632	38.143116	16.000000	16.000000	352.757538	239.978745
8.000000	0.000000	253.468887	38.384041	20.000000	16.000000	413.299042	240.705643
12.000000	0.000000	309.505524	38.183723	0.000000	20.000000	110.925919	286.384521
16.000000	0.000000	366.320892	38.885590	4.000000	20.000000	169.495071	286.728333
20.000000	0.000000	422.953033	41.175308	8.000000	20.000000	229.256287	287.529480
0.000000	4.000000	141.500000	84.500000	12.000000	20.000000	288.893860	287.006226
4.000000	4.000000	193.100739	86.525887	16.000000	20.000000	349.613831	288.419983
8.000000	4.000000	249.439102	87.452782	20.000000	20.000000	411.030029	289.499329
12.000000	4.000000	305.995148	88.117996	0.000000	24.000000	104.032295	340.504883
16.000000	4.000000	363.091187	88.480873	4.000000	24.000000	163.424179	340.833191
20.000000	4.000000	421.064423	88.620064	8.000000	24.000000	223.866684	341.947235
0.000000	8.000000	131.487305	135.605484	12.000000	24.000000	284.832733	341.493317
4.000000	8.000000	187.435364	136.158707	16.000000	24.000000	346.419769	343.446136
8.000000	8.000000	244.435638	136.690857	20.000000	24.000000	408.853821	344.434113
12.000000	8.000000	301.862946	136.906006	0.000000	28.000000	96.644432	396.509125
16.000000	8.000000	359.639496	136.895004	4.000000	28.000000	157.492615	397.210419
20.000000	8.000000	418.479401	137.299423	8.000000	28.000000	218.503372	398.360413
0.000000	12.000000	124.725021	185.354492	12.000000	28.000000	280.335785	398.243896
4.000000	12.000000	181.483063	185.635223	16.000000	28.000000	342.848816	399.050262
8.000000	12.000000	239.545273	186.489960	20.000000	28.000000	406.373352	400.458008
12.000000	12.000000	297.490784	186.903564	0.000000	32.000000	88.950905	454.458099
16.000000	12.000000	356.613831	187.879364	4.000000	32.000000	150.525543	455.052185
20.000000	12.000000	415.987610	188.847824	8.000000	32.000000	212.603134	456.072021
0.000000	16.000000	118.000000	232.000000	12.000000	32.000000	276.059021	457.330811
4.000000	16.000000	174.000000	230.500000	16.000000	32.000000	339.700012	459.388885
8.000000	16.000000	233.000000	235.500000	20.000000	32.000000	404.095581	461.020752

Tabela 6.6: Exemplo 1: pontos para calibração da câmera.

x_{wi}	y_{wi}	x_{fi}	y_{fi}	x_{wi}	y_{wi}	x_{fi}	y_{fi}
86.500000	54.000000	160.000000	120.000000	262.500000	232.000000	400.000000	300.000000
155.000000	52.500000	240.000000	120.000000	336.000000	229.500000	480.000000	300.000000
225.000000	46.000000	320.000000	120.000000	44.000000	304.000000	160.000000	360.000000
299.000000	42.000000	400.000000	120.000000	109.000000	298.500000	240.000000	360.000000
374.000000	36.000000	480.000000	120.000000	179.000000	297.000000	320.000000	360.000000
73.000000	116.000000	160.000000	180.000000	249.500000	296.000000	400.000000	360.000000
141.000000	113.500000	240.000000	180.000000	324.000000	294.000000	480.000000	360.000000
212.000000	109.500000	320.000000	180.000000	35.000000	356.000000	160.000000	420.000000
282.500000	106.000000	400.000000	180.000000	97.500000	361.000000	240.000000	420.000000
360.000000	101.000000	480.000000	180.000000	167.000000	360.500000	320.000000	420.000000
64.500000	176.500000	160.000000	240.000000	238.000000	359.500000	400.000000	420.000000
131.500000	175.000000	240.000000	240.000000	312.000000	358.500000	480.000000	420.000000
201.000000	173.000000	320.000000	240.000000	19.000000	424.000000	160.000000	480.000000
273.500000	169.000000	400.000000	240.000000	87.000000	424.000000	240.000000	480.000000
346.000000	165.500000	480.000000	240.000000	155.500000	424.000000	320.000000	480.000000
57.000000	234.000000	160.000000	300.000000	227.000000	423.000000	400.000000	480.000000
121.500000	236.500000	240.000000	300.000000	300.500000	424.000000	480.000000	480.000000
191.500000	236.500000	320.000000	300.000000				

Tabela 6.7: Exemplo 1: pontos para calibração do projetor.

Os valores calculados para os parâmetros da câmera e do projetor utilizando o **gcgCalibrator** sem otimização foram:

$$f_c = 1811.5631 \qquad f_p = 2680.8656$$

$$T_c = \begin{bmatrix} -12.6481 & -14.3206 & 130.4052 \end{bmatrix} \qquad T_p = \begin{bmatrix} -34.9659 & -25.6669 & 151.1080 \end{bmatrix}$$

$$R_c = \begin{bmatrix} 0.9945 & -0.0631 & 0.0837 \\ 0.0137 & 0.8699 & 0.4930 \\ -0.1039 & -0.4891 & 0.8660 \end{bmatrix} \qquad R_p = \begin{bmatrix} 0.9530 & 0.1677 & -0.2527 \\ 0.0431 & 0.7499 & 0.6601 \\ 0.3002 & -0.6399 & 0.7074 \end{bmatrix}$$

$$k_{1c} = 0.0000 \qquad k_{1p} = 0.0000$$

$$C_{xc} = 320.0000 \qquad C_{xp} = 400.0000$$

$$C_{yc} = 240.0000 \qquad C_{yp} = 300.0000$$

Observa-se que o valor de $k_1 = 0$, pois a calibração sem otimização não considera a distorção da lente. Os gráficos das Figuras 6.2 e 6.3 mostram uma comparação entre os pontos reconstruídos com os parâmetros calculados e os pontos da imagem original.

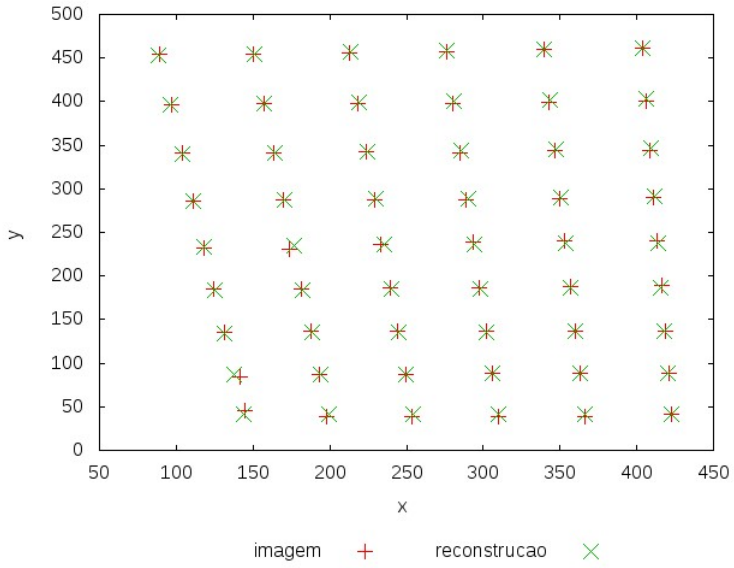


Figura 6.2: Resultado da reconstrução dos pontos da câmera a partir da calibração com **gcgCalibrator** sem otimização.

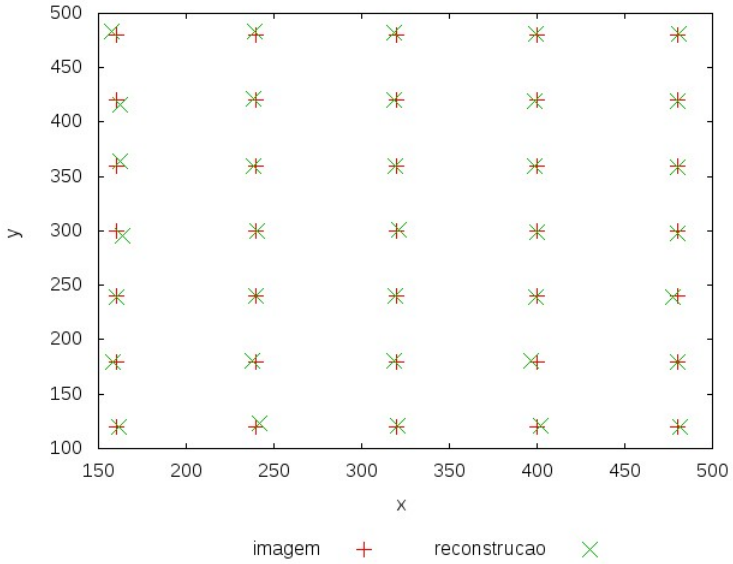


Figura 6.3: Resultado da reconstrução dos pontos do projetor a partir da calibração com **gcgCalibrator** sem otimização.

Para o **gcgCalibrator** com otimização nos parâmetros T_z, f e k_1 foram obtidos os seguintes valores para os parâmetros otimizados:

$$f_c = 1811.5631 \qquad f_p = 2872.4744$$
$$T_{zc} = 130.4052 \qquad T_{zp} = 160.7190$$
$$k_{1c} = 2.9449e-010 \quad k_{1p} = -3.1224e-008$$

Os gráficos das Figuras 6.4 e 6.5 mostram uma comparação entre os pontos reconstruídos com os parâmetros calculados e os pontos da imagem original.

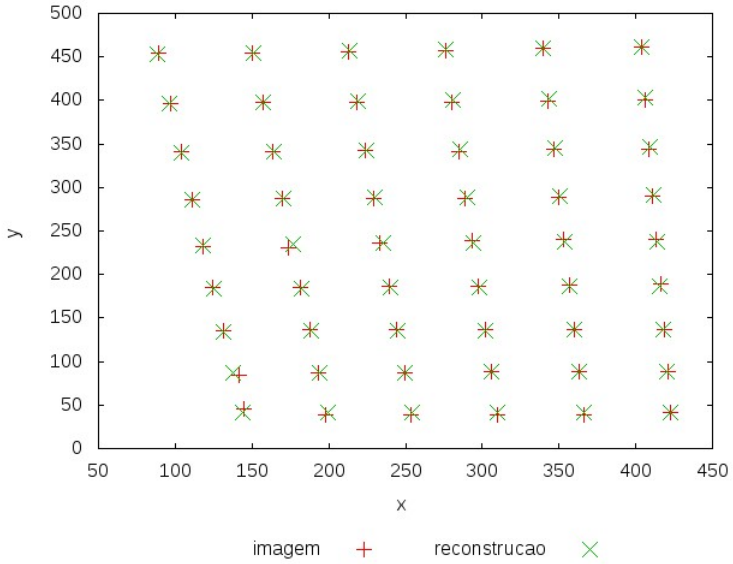


Figura 6.4: Resultado da reconstrução dos pontos da câmera a partir da calibração com **gcgCalibrator** com otimização em T_z, f e k_1.

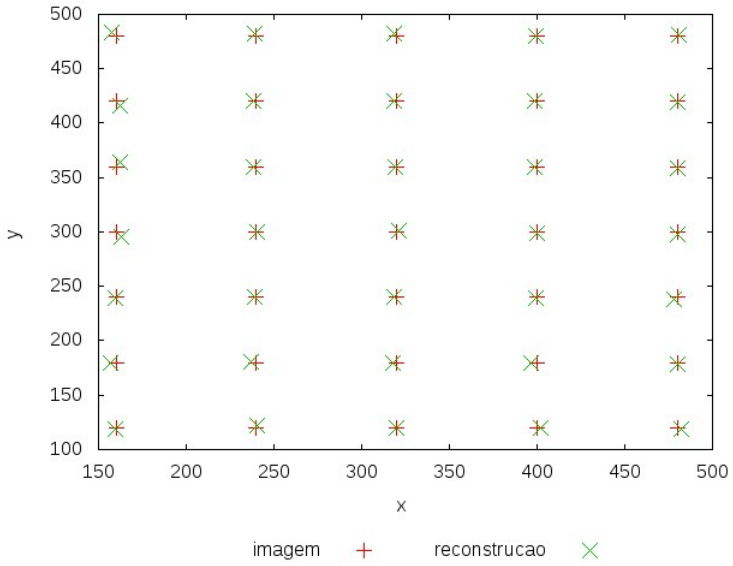


Figura 6.5: Resultado da reconstrução dos pontos do projetor a partir da calibração com **gcgCalibrator** com otimização em T_z, f e k_1.

Para o **gcgCalibrator** com otimização em todos os parâmetros foram obtidos os seguintes resultados:

$$f_c = 1811.5631 \qquad f_p = 2675.5143$$

$$T_c = \begin{bmatrix} -12.6478 & -14.3223 & 130.4052 \end{bmatrix} \qquad T_p = \begin{bmatrix} -35.0490 & -25.4611 & 149.9057 \end{bmatrix}$$

$$R_c = \begin{bmatrix} 0.9941 & -0.0650 & 0.0869 \\ 0.0135 & 0.8687 & 0.4952 \\ -0.1076 & -0.4911 & 0.8644 \end{bmatrix} \qquad R_p = \begin{bmatrix} 0.9544 & 0.1681 & -0.2468 \\ 0.0415 & 0.7438 & 0.6671 \\ 0.2957 & -0.6469 & 0.7029 \end{bmatrix}$$

$$k_{1c} = 6.9137e-009 \qquad k_{1p} = 5.4738e-008$$

$$C_{xc} = 320.0000 \qquad C_{xp} = 399.9721$$

$$C_{yc} = 239.9999 \qquad C_{yp} = 300.0437$$

Os gráficos das Figuras 6.6 e 6.7 mostram uma comparação entre os pontos reconstruídos com os parâmetros calculados e os pontos da imagem original.

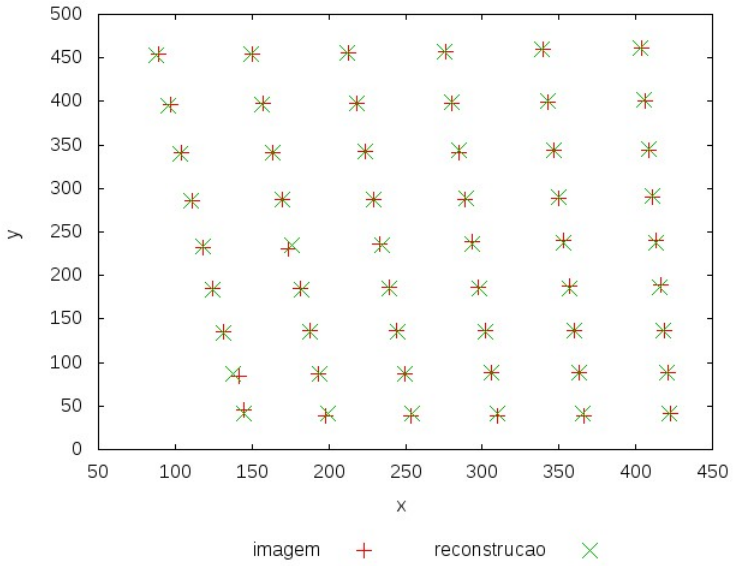


Figura 6.6: Resultado da reconstrução dos pontos da câmera a partir da calibração com **gcgCalibrator** com otimização em todos os parâmetros.

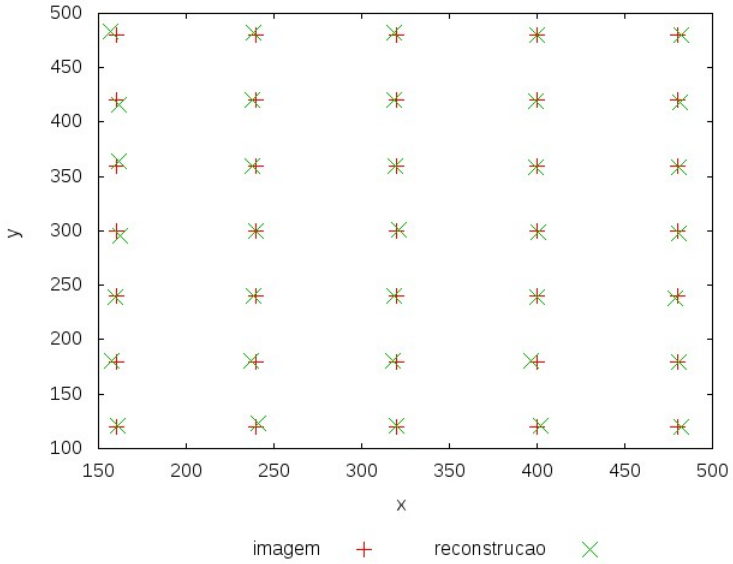


Figura 6.7: Resultado da reconstrução dos pontos do projetor a partir da calibração com **gcgCalibrator** com otimização em todos os parâmetros.

Para o código Matlab com otimização nos parâmetros T_z, f e k_1 foram obtidos os seguintes valores para os parâmetros otimizados:

$$f_c = 1869.7863 \qquad f_p = 2810.4890$$
$$T_{zc} = 135.7275 \qquad T_{zp} = 157.6129$$
$$k_{1c} = -2.2485e-007 \quad k_{1p} = 4.8007e-008$$

Os gráficos das Figuras 6.8 e 6.9 mostram uma comparação entre os pontos reconstruídos com os parâmetros calculados e os pontos da imagem original.

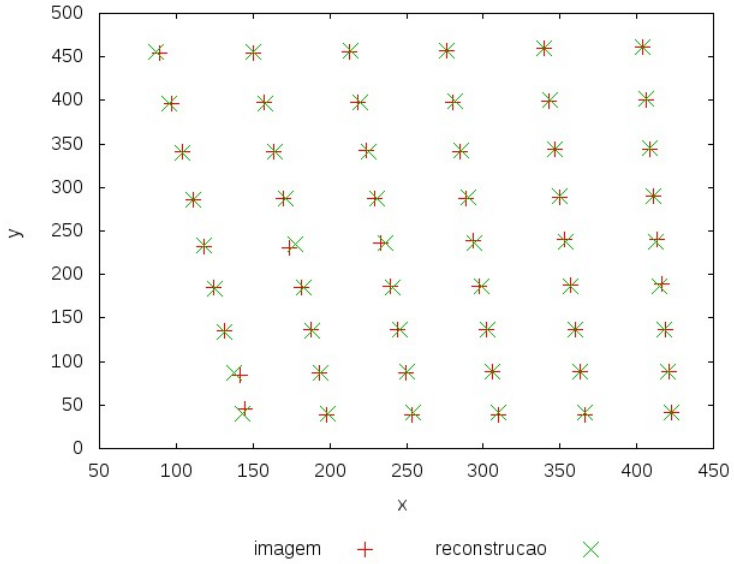


Figura 6.8: Resultado da reconstrução dos pontos da câmera a partir da calibração com código Matlab com otimização em T_z, f e k_1.

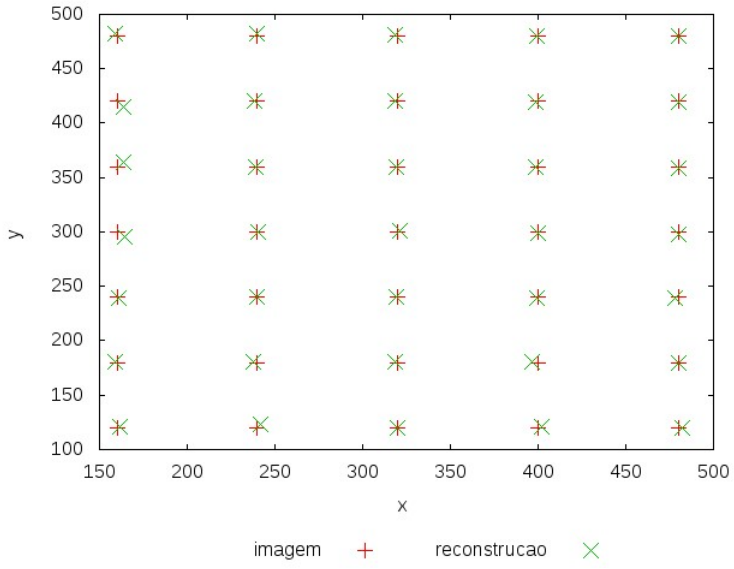
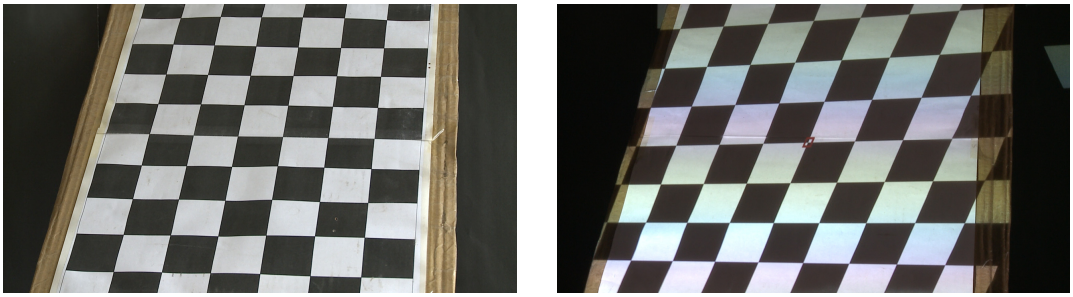


Figura 6.9: Resultado da reconstrução dos pontos do projetor a partir da calibração com código Matlab com otimização em T_z, f e k_1.

Para o segundo exemplo foi utilizado o mesmo padrão da calibração em uma posição mais inclinada. A Figura 6.10 mostra as imagens utilizadas para essa calibração. Foram realizadas para esse exemplo as mesmas calibrações do exemplo anterior. A Ta-



(a) Câmera (b) Projetor

Figura 6.10: Imagens para segunda calibração.

bela 6.8 mostra os pontos utilizados nesse exemplo para a câmera e a Figura 6.9 mostra os pontos utilizados para o projetor. A projeção do centro óptico foi assumida como sendo o centro o ponto (680, 360) para a câmera e o ponto (400, 300) para o projetor correspondente ao centro das imagens.

x_{wi}	y_{wi}	x_{fi}	y_{fi}	x_{wi}	y_{wi}	x_{fi}	y_{fi}
0.000000	0.000000	442.650757	35.075863	12.000000	16.000000	696.722839	330.998962
4.000000	0.000000	541.993225	31.652330	16.000000	16.000000	809.913574	333.368378
8.000000	0.000000	643.616760	32.772057	20.000000	16.000000	924.818359	335.441223
12.000000	0.000000	746.312988	33.185860	0.000000	20.000000	345.162811	401.795715
16.000000	0.000000	850.404724	35.254791	4.000000	20.000000	456.971313	403.377686
20.000000	0.000000	955.724792	39.982494	8.000000	20.000000	569.909546	405.338531
0.000000	4.000000	424.230408	100.431541	12.000000	20.000000	684.188232	403.739716
4.000000	4.000000	526.427917	102.666504	16.000000	20.000000	799.718628	407.344879
8.000000	4.000000	630.435913	105.394707	20.000000	20.000000	917.323792	410.883087
12.000000	4.000000	735.358032	107.806992	0.000000	24.000000	323.080994	485.671967
16.000000	4.000000	841.325195	109.205872	4.000000	24.000000	437.502014	486.501709
20.000000	4.000000	948.511719	110.031746	8.000000	24.000000	553.478516	489.553284
0.000000	8.000000	406.177948	175.321518	12.000000	24.000000	669.908447	487.530518
4.000000	8.000000	510.326111	176.392456	16.000000	24.000000	788.645996	492.654541
8.000000	8.000000	616.284912	178.302505	20.000000	24.000000	909.102173	495.115082
12.000000	8.000000	722.815735	178.921249	0.000000	28.000000	300.158356	574.553955
16.000000	8.000000	831.280029	179.566833	4.000000	28.000000	417.293091	575.838196
20.000000	8.000000	941.031250	180.860458	8.000000	28.000000	535.678345	578.507507
0.000000	12.000000	386.396423	249.084656	12.000000	28.000000	655.200745	576.413757
4.000000	12.000000	492.767792	249.587112	16.000000	28.000000	776.617493	578.363525
8.000000	12.000000	601.101074	251.806870	20.000000	28.000000	900.363953	581.672791
12.000000	12.000000	710.241638	252.750092	0.000000	32.000000	275.478027	666.967590
16.000000	12.000000	820.990356	255.560425	4.000000	32.000000	395.521454	668.365723
20.000000	12.000000	933.274536	258.305237	8.000000	32.000000	516.698242	670.358765
0.000000	16.000000	364.557800	323.740967	12.000000	32.000000	640.149536	672.491882
4.000000	16.000000	474.186279	327.275360	16.000000	32.000000	765.121704	677.566345
8.000000	16.000000	584.947266	329.523804	20.000000	32.000000	891.709412	680.582581

Tabela 6.8: Exemplo 2: pontos para calibração da câmera.

x_{wi}	y_{wi}	x_{fi}	y_{fi}	x_{wi}	y_{wi}	x_{fi}	y_{fi}
419.000000	58.000000	160.000000	120.000000	297.000000	352.500000	160.000000	300.000000
526.000000	49.000000	240.000000	120.000000	405.000000	350.000000	240.000000	300.000000
640.500000	41.500000	320.000000	120.000000	515.000000	347.500000	320.000000	300.000000
757.000000	34.000000	400.000000	120.000000	626.000000	345.000000	400.000000	300.000000
879.000000	25.500000	480.000000	120.000000	743.000000	341.500000	480.000000	300.000000
1004.000000	16.500000	560.000000	120.000000	863.000000	339.000000	560.000000	300.000000
374.000000	156.500000	160.000000	180.000000	259.000000	449.000000	160.000000	360.000000
482.000000	151.000000	240.000000	180.000000	365.500000	448.000000	240.000000	360.000000
594.000000	145.500000	320.000000	180.000000	473.000000	447.000000	320.000000	360.000000
710.000000	139.500000	400.000000	180.000000	582.000000	446.000000	400.000000	360.000000
830.000000	133.000000	480.000000	180.000000	698.000000	444.500000	480.000000	360.000000
955.000000	126.000000	560.000000	180.000000	817.500000	442.500000	560.000000	360.000000
336.000000	255.000000	160.000000	240.000000	227.000000	541.000000	160.000000	420.000000
444.500000	251.500000	240.000000	240.000000	325.000000	545.000000	240.000000	420.000000
554.000000	247.500000	320.000000	240.000000	430.500000	545.000000	320.000000	420.000000
667.000000	243.500000	400.000000	240.000000	540.500000	545.000000	400.000000	420.000000
786.000000	238.500000	480.000000	240.000000	654.000000	545.500000	480.000000	420.000000
909.000000	233.500000	560.000000	240.000000	771.500000	545.500000	560.000000	420.000000

Tabela 6.9: Exemplo 2: pontos para calibração do projetor.

Os valores calculados para os parâmetros da câmera e do projetor utilizando o **gcgCalibrator** sem otimização foram:

$$f_c = 3557.6986 \qquad f_p = 2869.4270$$

$$T_c = \begin{bmatrix} -7.7181 & -12.5200 & 138.2977 \end{bmatrix} \quad T_p = \begin{bmatrix} -38.8125 & -20.3948 & 125.5400 \end{bmatrix}$$

$$R_c = \begin{bmatrix} 0.9868 & -0.1267 & 0.1009 \\ 0.0167 & 0.6992 & 0.7148 \\ -0.1611 & -0.7037 & 0.6920 \end{bmatrix} \quad R_p = \begin{bmatrix} 0.9249 & 0.2839 & -0.2529 \\ 0.0434 & 0.5821 & 0.8119 \\ 0.3777 & -0.7619 & 0.5261 \end{bmatrix}$$

$$k_{1c} = 0.0000 \qquad k_{1p} = 0.0000$$

$$C_{xc} = 640.0000 \qquad C_{xp} = 400.0000$$

$$C_{yc} = 360.0000 \qquad C_{yp} = 300.0000$$

Os gráficos das Figuras 6.11 e 6.12 mostram uma comparação entre os pontos reconstruídos com os parâmetros calculados e os pontos da imagem original.

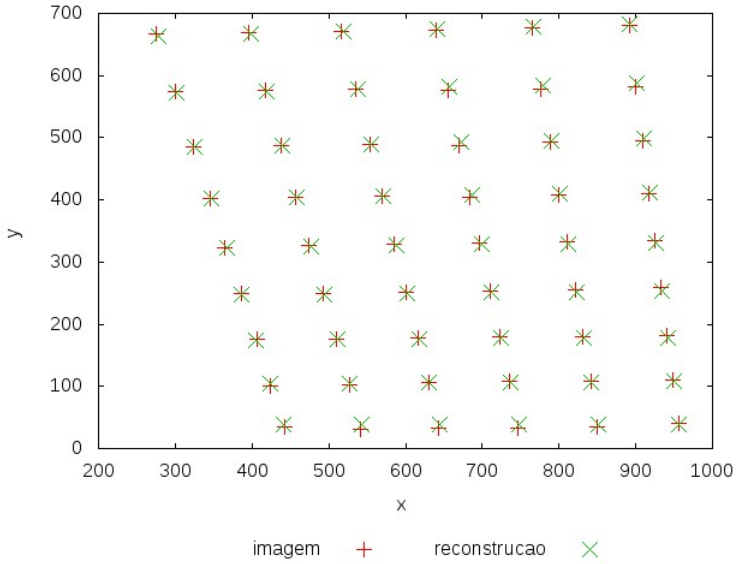


Figura 6.11: Resultado da reconstrução dos pontos da câmera a partir da calibração com **gcgCalibrator** sem otimização.

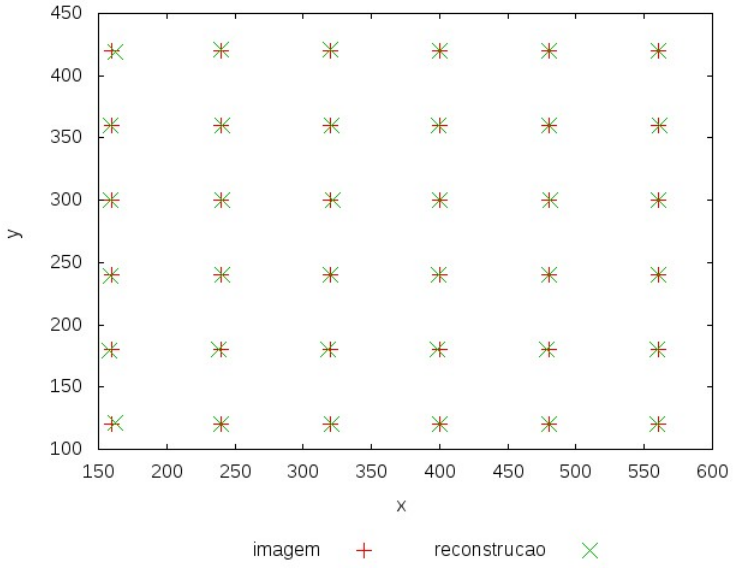


Figura 6.12: Resultado da reconstrução dos pontos do projetor a partir da calibração com **gcgCalibrator** sem otimização.

Para o **gcgCalibrator** com otimização nos parâmetros T_z, f e k_1 foram obtidos os seguintes valores para os parâmetros otimizados:

$$f_c = 3706.1989 \qquad f_p = 2864.4610$$

$$T_{zc} = 144.6665 \qquad T_{zp} = 127.9767$$

$$k_{1c} = -6.9732e-008 \quad k_{1p} = -8.2347e-008$$

Os gráficos das Figuras 6.13 e 6.14 mostram uma comparação entre os pontos

reconstruídos com os parâmetros calculados e os pontos da imagem original.

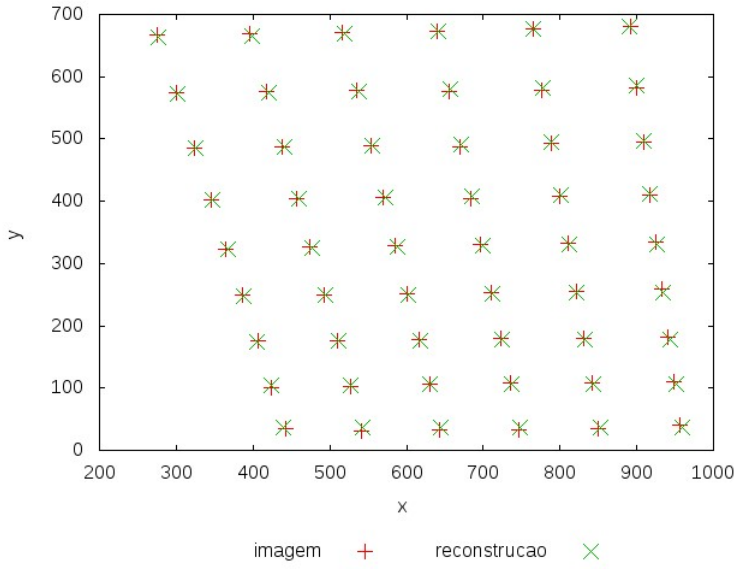


Figura 6.13: Resultado da reconstrução dos pontos da câmera a partir da calibração com **gcgCalibrator** com otimização em T_z, f e k_1.

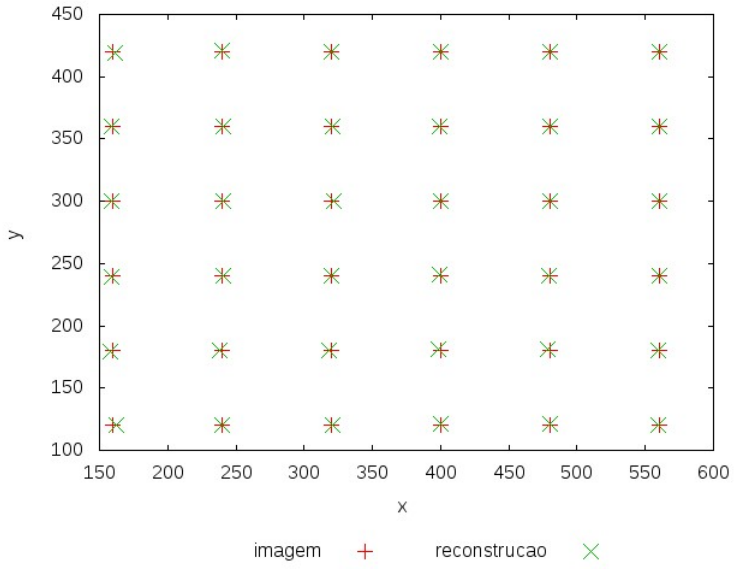


Figura 6.14: Resultado da reconstrução dos pontos do projetor a partir da calibração com **gcgCalibrator** com otimização em T_z, f e k_1.

Para o **gcgCalibrator** com otimização em todos os parâmetros foram obtidos os seguintes resultados:

$$f_c = 3557.6986 \qquad f_p = 2864.4655$$

$$T_c = \begin{bmatrix} -7.7232 & -12.5206 & 138.2984 \end{bmatrix} \qquad T_p = \begin{bmatrix} -38.5678 & -20.2941 & 126.1424 \end{bmatrix}$$

$$R_c = \begin{bmatrix} 0.9864 & -0.1291 & 0.1022 \\ 0.0171 & 0.6976 & 0.7163 \\ -0.1637 & -0.7048 & 0.6903 \end{bmatrix} \qquad R_p = \begin{bmatrix} 0.9249 & 0.2840 & -0.2528 \\ 0.0427 & 0.5829 & 0.8114 \\ 0.3778 & -0.7613 & 0.5270 \end{bmatrix}$$

$$k_{1c} = -2.0933e-008 \qquad k_{1p} = -7.8964e-008$$

$$C_{xc} = 639.9998 \qquad C_{xp} = 400.0076$$

$$C_{yc} = 360.0000 \qquad C_{yp} = 300.0018$$

Os gráficos das Figuras 6.15 e 6.16 mostram uma comparação entre os pontos reconstruídos com os parâmetros calculados e os pontos da imagem original.

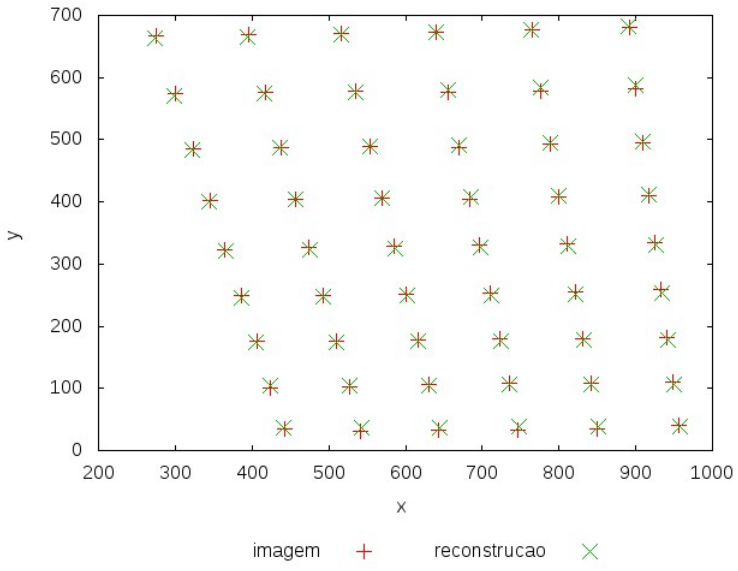


Figura 6.15: Resultado da reconstrução dos pontos da câmera a partir da calibração com **gcgCalibrator** com otimização em todos os parâmetros.

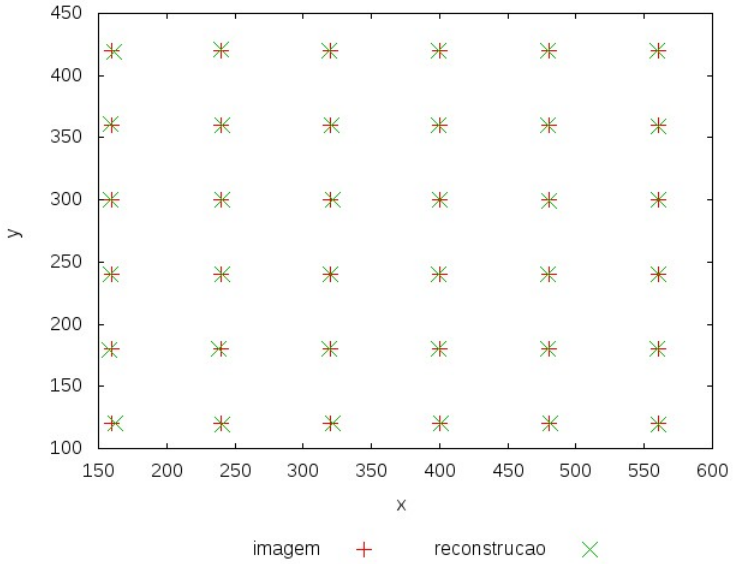


Figura 6.16: Resultado da reconstrução dos pontos do projetor a partir da calibração com **gcgCalibrator** com otimização em todos os parâmetros.

Para o código Matlab com otimização nos parâmetros T_z, f e k_1 foram obtidos os seguintes valores para os parâmetros otimizados:

$$f_c = 3666.4141 \qquad f_p = 2954.3524$$
$$T_{zc} = 144.2225 \qquad T_{zp} = 131.2783$$
$$k_{1c} = -1.3770e-007 \quad k_{1p} = -4.3702e-008$$

Os gráficos das Figuras 6.17 e 6.18 mostram uma comparação entre os pontos reconstruídos com os parâmetros calculados e os pontos da imagem original.

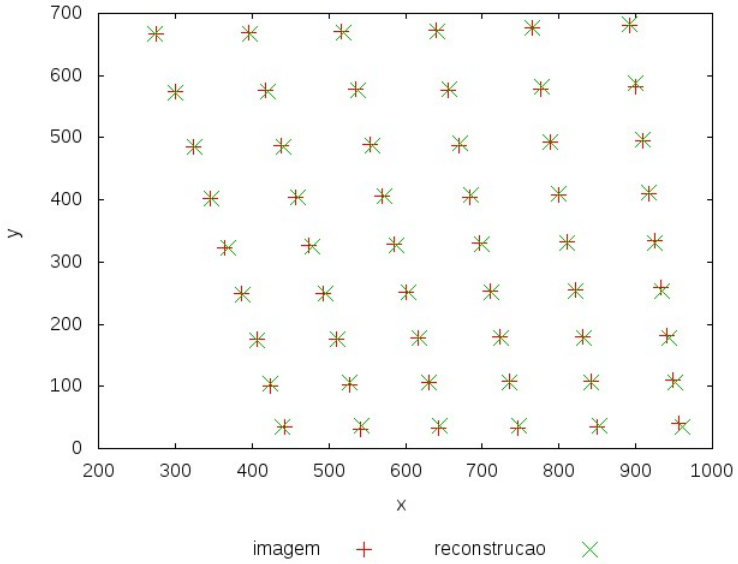


Figura 6.17: Resultado da reconstrução dos pontos da câmera a partir da calibração com código Matlab com otimização em T_z, f e k_1.

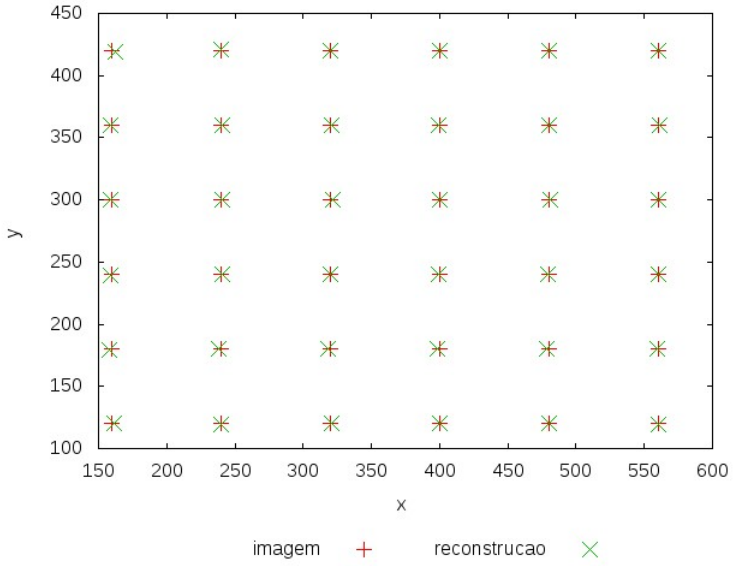


Figura 6.18: Resultado da reconstrução dos pontos do projetor a partir da calibração com código Matlab com otimização em T_z, f e k_1.

A Figura 6.19 mostra os erros médios obtidos para as diversas calibrações realizadas. No caso, o erro é calculado utilizando a distância euclidiana (em *pixels*) entre os pontos da imagem original (x_{fi}, y_{fi}) e os pontos reconstruídos (x'_{fi}, y'_{fi}) conforme indica a Equação 6.3.

$$erro_i = \sqrt{(x_{fi} - x'_{fi})^2 + (y_{fi} - y'_{fi})^2} \tag{6.3}$$

Para cada calibração realizada foram determinadas as seguintes medidas de erro:

- Erro absoluto máximo:

$$max(erro_i)$$

- Erro absoluto médio:

$$\frac{1}{n}\sum erro_i$$

- Erro quadrático médio:

$$\frac{1}{n}\sum (erro_i)^2$$

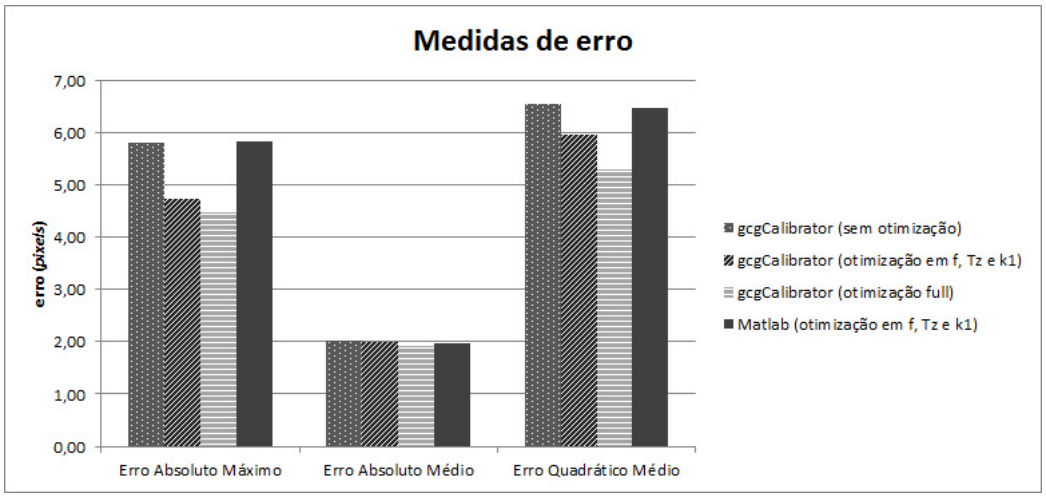


Figura 6.19: Valor médio dos erros.

A análise do gráfico da Figura 6.19 permite concluir que o processo de calibração desenvolvido obteve resultados melhores que o código utilizado para comparação no que diz respeito a otimização para os parâmetros T_z, f e k_1. Mesmo para o caso sem otimização, observou-se erros próximos aos obtidos pelo código em Matlab. Essa diferença é causada pelo fato de que o cálculo dos valores de T_z, f e k_1 no código em Matlab considera apenas o ajuste da coordenada y, enquanto o **gcgCalibrator** considera o ajuste de x e y.

É importante destacar que o erro diminui a medida que se otimizam os parâmetros da calibração, porque passam a ser consideradas as distorções da lente. Ao otimizar todos os parâmetros verifica-se que o resultado foi ainda melhor, porque podem ser corrigidos mais erros encontrados na aproximação inicial. Porém, isso torna o processo mais lento por ter mais parâmetros a serem corrigidos e necessitar de mais iterações para convergir.

7 Conclusão

Neste trabalho foi apresentado um processo de calibração de câmeras e projetores. Foram mostradas as transformações geométricas envolvidas no processo, o método de calibração coplanar desenvolvido por Tsai e o método de otimização de Levenberg-Marquardt.

Como modelo computacional, foi apresentada uma classe desenvolvida para a solução de sistemas lineares. Essa resolução é feita baseada na decomposição em valores singulares. Foram implementadas classes para realizar a calibração de câmeras e projetores pelo método de Tsai coplanar e para realizar otimização de funções não lineares pelo método de Levenberg-Marquardt.

As classes desenvolvidas foram testadas para diversos exemplos e obtiveram resultados com erros aceitáveis. Uma das dificuldades encontradas na realização de testes foi a obtenção de conjuntos de pontos para realizar calibrações e extrair os cantos de imagens. Um dos trabalhos futuros é a criação de um módulo que realize essa extração dos pontos. Outra possibilidade é a implementação de outros métodos para realizar a calibração.

Referências Bibliográficas

Carvalho, P. C.; Velho, L.; Montenegro, A. A.; Peixoto, A.; Sá, A.; Soares, E. ; Escriba, L. A. R. **Fotografia 3D**. Associação Instituto de Matemática Pura e Aplicada, IMPA, 2005.

Horn, B. K. Tsai's camera calibration method revisited. **http://www.ai.mit.edu/people/bkph/papers/tsaiexplain.pdf**, 2000.

Levenberg, K. A method for the solution of certain non-linear problems in least squares. **Quarterly of Applied Mathematics**, 1944.

Lourakis, M. I. A. A brief description of the levenberg-marquardt algorithm implemented by levmar. **http://www.ics.forth.gr/∼lourakis/levmar/levmar.pdf**, 2005.

Madsen, K.; Nielsen, H. B. ; Tingleff, O. **Methods for Non-Linear Least Squares Problems (2nd ed.)**. Informatics and Mathematical Modelling, Technical University of Denmark, (DTU), 2004.

Maffra, F. A.; Gattass, M. Método de levenberg-marquardt. **http://www.tecgraf.puc-rio.br/∼mgattass/LM_Fabiola/LM_Teoria.pdf**, 2008.

Marquardt, D. W. An algorithm for the least-squares estimation of nonlinear parameters. **SIAM Journal of Applied Mathematics**, 1963.

Marques, C. C. S. C. **Um sistema de calibração de câmera**. 2007. Dissertação de Mestrado - Universidade Federal de Alagoas.

Press, W. H.; Teukolsky, S. A.; Vetterling, W. T. ; Flannery, B. P. **Numerical Recipes in C**. 2. ed., Cambridge University Press, 1992.

Szenberg, F. **Acompanhamento de Cenas com Calibração Automática de Câmeras**. 2001. Tese de Doutorado - Pontifícia Universidade Católica do Rio de Janeiro.

Tsai, R. Y. A versatile camera calibration technique for high-accuracy 3d machine vision metrology using off-the-shelf tv cameras and lenses. **Ieee Journal Of Robotics And Automation**, 1987.

Wan, S. Calibrating a Camera Using a Monoview Coplanar Set of Points. **http://read.pudn.com/downloads61/sourcecode/graph/texture_mapping/211785/Tsai/Tsai.m__.htm**, 2004.

Willson, R. G. **Modeling and Calibration of Automated Zoom Lenses**. 1994. Tese de Doutorado - Carnegie Mellon University.