Projeto Demonstrativo 2
Calibração de Câmeras

Carlos Adir Ely Murussi Leite
150121059
carlos.adir.leite@gmail.com
Rebeca Helen Silva de Sousa
130145068
rebeca.hellenss@gmail.com

Departamento de Ciência da Comptutação
Universidade de Brasília
Campus Darcy Ribeiro, Asa Norte
Brasília-DF, CEP 70910-900, Brazil,

Abstract

Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Ut purus elit, vestibulum ut, placerat ac, adipiscing vitae, felis. Curabitur dictum gravida mauris. Nam arcu libero, nonummy eget, consectetuer id, vulputate a, magna. Donec vehicula augue eu neque. Pellentesque habitant morbi tristique senectus et netus et malesuada fames ac turpis egestas. Mauris ut leo. Cras viverra metus rhoncus sem. Nulla et lectus vestibulum urna fringilla ultrices. Phasellus eu tellus sit amet tortor gravida placerat. Integer sapien est, iaculis in, pretium quis, viverra ac, nunc. Praesent eget sem vel leo ultrices bibendum. Aenean faucibus. Morbi dolor nulla, malesuada eu, pulvinar at, mollis ac, nulla. Curabitur auctor semper nulla. Donec varius orci eget risus. Duis nibh mi, congue eu, accumsan eleifend, sagittis quis, diam. Duis eget orci sit amet orci dignissim rutrum.

1 Introdução

O ramo da visão computacional é bastante diverso e possui variadas aplicações. Para tais aplicações, frequentemente tem-se o problema de relacionar o cenário digital(as imagens bidimensionais obtidas por câmeras) com o cenário real

À medida com que uma pessoa cresce, adquire-se a capacidade de relacionar a imagem obtida pelos olhos com o seu significado real e isso é feito principalmente através de aprendizado: aprende-se o tamanho e forma dos objetos do dia a dia e através da inferência, por comparação pode-se ter uma estimativa de posicionamento de objetos bem como seu tamanho.

Já para câmeras digitais que se comportam de forma diferente ao olho humano na captura de imagens, existem incertezas relacionadas à câmera que se dividem em dois grupos:

Intrínseco Inerentes ao processo de fabricação da câmera.

Extrinseco Relativos à posição da câmera em relação ao espaço.

Por esse motivo, utiliza-se conceitos de algebra linear, geometria e ótica para relacionar o espaço da imagem obtida pela câmera com a posição real do ponto observado. Essa relação pode ser dada pela Equação (1) utilizando-se coordenadas homogêneas, em que a matriz *In*

relaciona os parâmetros intrínsecos e a matriz *Ex* relaciona os parâmetros extrinsecos. A Figura (1) mostra alguns dos parâmetros apresentados na Equação (1).

$$s\begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \underbrace{\begin{bmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix}}_{In} \underbrace{\begin{bmatrix} r_{11} & r_{12} & r_{13} & t_1 \\ r_{21} & r_{22} & r_{23} & t_2 \\ r_{31} & r_{32} & r_{33} & t_3 \end{bmatrix}}_{Ex} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix} \quad (1)$$

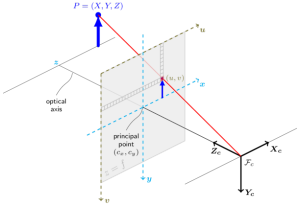

Figure 1: Modelo da câmera Pinhole

Segundo Fulano, é necessário apenas 6 pontos para determinar todos os parâmetros intrínsecos e extrínsecos da câmera. Mas para aumentar a precisão dos parâmetros e diminuir a incerteza associada, pode-se utilizar mais pontos conhecidos.

Contudo, como toda a teoria exposta para calibração de câmera considera o modelo de câmera Pinhole e as câmeras digitais são feitas de diversas formas como mostra Raskar[1]. Devido a essa variação entre câmeras, existem câmeras que autoajustam seu foco e portanto os parâmetros intrínsecos variam.

2 Metodologia

A metodologia consistiu em seguir os passos dos requisitos necessários.

Para o primeiro requisito, a metodologia consistiu em adaptar o código implementado em C++ no Projeto Demonstrativo 1.

Para o segundo e terceiro requisito, consistiu primeiramente em selecionar um tabuleiro de xadrez com 6×8 interseções e cada quadrado com 30 mm de lado, imprimiu-se e colou-se em um papelão espesso.

Com o tabuleiro, faz 5 pacotes de medições com 25 imagens do tabuleiro em cada pacote. Cada imagem em um pacote bem similares entre si, enquanto imagem entre pacotes ficam em regiões separadas da imagem para haver uma abrangência maior. Para evitar perda de tempo capturando o padrão frequentemente, planejou-se fazer um código para armazenar as imagens com padrão do tabuleiro assim que fossem capturadas, o qual foi chamado de *get_images*.

Uma vez com 5×25 imagens do tabuleiro, utilizou-se a função *findChessboardCorners*[**?**] para encontrar os 48 pontos para cada imagem. Com o conjunto de 25×48 pontos em cada pacote, utilizou-se a função *calibrateCamera*[**?**] para capturar a matriz intrínseca e os coeficientes de distorção como parâmetros intrínsecos; e os vetores de rotação e translação referentes aos parâmetros extrínsecos.

Com os parâmetros intrínsecos, pegou-se a média e desvio padrão das 5 matrizes intrínsecas(uma para cada pacote) e dos 5 coeficientes de distorção.

Então, para verificar o efeito da distância do padrão sobre os parâmetros extrínsecos, fez-se o mesmo processo de 5 pacotes para 3 distâncias: a mais distante que ainda capturava o padrão, a menor distância que se reconhecia todos os pontos do tabuleiro, e uma distância intermediária. Apenas para a distância mais próxima era possível apenas 1 pacote de medição, visto que o tabuleiro ocupa a tela totalmente. Para cada distância, mediu-se com uma trena a distância entre a câmera

Para o requisito 4, foi mesclar os resultados obtidos nos requisitos 2 e 3 com o requisito 1 pois a interface, os parâmetros intrínsecos e extrínsecos foram obtidos nos requisitos anteriores.

Para implementação do código, seguiu-se a metodologia de utilizar funções já implementadas pela OpenCV para encontrar vertices do tabuleiro de xadrez, calcular os parâmetros das matrizes *In* e *Ex* da Equação (1). Para implementação primeiramente pensou-se em utilizar C++, mas por dificuldade na utilização de funções especificas da OpenCV optou-se por utilizar Python. Para escolha da câmera, optou-se por uma webcam, visto que outra câmera auto ajustava o seu foco e isso mudava os parâmetros intrínsecos frequentemente.

3 Resultados

Os requisitos foram separados e divididos pelos requisitos através dos códigos *r1* a *r4*. Para o código *r1*, obteve-se simplesmente uma interface do usuário em que com dois cliques do mouse era possível medir a distância em píxeis entre dois pontos.

Com a captura das imagens pelo algoritmo *get_images*, obteve-se a média e desvio padrão das 5 matrizes intrínsecas e coefientes de distorção. Os valores dos parâmetros das matrizes intrínsecas são mostrados pela Tabela (3).

Table 1: Parâmetros intrínsecos da matriz *In* para diferentes distâncias com medida central

	In_1	In_2	In_3	In_4	In_5	$In\ (\bar{x} \pm \sigma)$
f_x	721	1154	1739	1052	1955	1324 ± 455
f_y	728	1164	1846	1079	3219	1607 ± 884
c_x	634	714	639	649	633	654 ± 30
c_y	374	316	369	334	353	349 ± 22

4 Discussão e Conclusões

Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Ut purus elit, vestibulum ut, placerat ac, adipiscing vitae, felis. Curabitur dictum gravida mauris. Nam arcu libero, nonummy eget, consectetuer id, vulputate a, magna. Donec vehicula augue eu neque. Pellentesque habitant morbi tristique senectus et netus et malesuada fames ac turpis egestas. Mauris ut leo. Cras viverra metus rhoncus sem. Nulla et lectus vestibulum urna fringilla ultrices. Phasellus eu tellus sit amet tortor gravida placerat. Integer sapien est, iaculis in, pretium quis, viverra ac, nunc. Praesent eget sem vel leo ultrices bibendum. Aenean faucibus. Morbi dolor nulla, malesuada eu, pulvinar at, mollis ac, nulla. Curabitur auctor semper nulla. Donec varius orci eget risus. Duis nibh mi, congue eu, accumsan eleifend, sagittis quis, diam. Duis eget orci sit amet orci dignissim rutrum.

References

[1] Ramesh Raskar. *Computational Photography*. Cambridge, MA 02139, USA, 2007. URL http://web.media.mit.edu/raskar/.

Appendices

Matrizes de parâmetros intrínsecos