Projeto Demonstrativo 2

Raphael Soares 14/0160299
raphael.soares.1996@gmail.com

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

Abstract

Este segundo projeto tem como objetivo principal a avaliação dos aspectos envolvidos em calibração de câmeras. Para isso, foi desenvolvida uma "régua visual" que tenta estimar a altura ou largura de um objeto através apenas da sua imagem capturada pela câmera. Ou seja, através da medida em pixels de um objeto o programa é capaz de estimar a medida real, em metros, deste objeto. Para atingir esse objetivo foi necessário transformar coordenadas em pixels da imagem em coordenadas tridimensionais do mundo real, usando a matriz de parâmetros intrínsecos da câmera obtida através da sua calibração.

1 Introdução

A visão começa com a detecção de luz do mundo. Esta luz começa com raios emanando de alguma origem, que viajam pelo espaço até atingir algum objeto. Quando esta luz atinge algum objeto, muito dela é absorvida, e o que não é absorvido nós percebemos como a cor do objeto. A luz refletida que encontra o caminho até o nosso olho (ou nossa câmera) é colecionada na nossa retina/imager (ou nosso "filme/aparelho"). A geometria desse arranjo (particularmente da viagem dos raios do objeto através da lente em nossos olhos ou câmera e para a retina ou filme) é de grande importância para a prática de visão computacional. Um modelo simples, porém útil de como isso acontece é o modelo da câmera pinhole apresentado em [1]. Infelizmente, um pinhole real não é uma boa forma de obter imagens porque ele não obtém luz suficiente para exposições rápidas e isso é um dos motivos de nossas câmeras e olhos usarem lentes, porque dessa forma podemos obter mais luz do que estaria disponível em um ponto.

Este projeto tem como objetivo usar calibração de câmeras para corrigir (matematicamente) desvios principais que o uso de lentes nos traz, as distorções. Existem dois tipos principais de distorções de lentes: radial, que é resultado da forma das lentes; e a tangencial, que surge do processo de montagem da câmera. Na teoria é possível definir lentes que não introduzem distorções, porém na prática nenhuma lente é perfeita. A calibração de câmeras é importante também, para relacionar as medidas da câmera com as medidas no mundo real tridimensional, isso é importante não só porque as cenas são tridimensionais, mas porque elas também são espaços físicos com unidades físicas. Portanto, a relação entre a unidade natural da câmera (pixels) e as unidades do mundo físico (metros, por exemplo) é um componente crítico para reconstruir uma cena tridimensional ou construir uma régua visual.

© 2018. The copyright of this document resides with its authors.
It may be distributed unchanged freely in print or electronic forms.

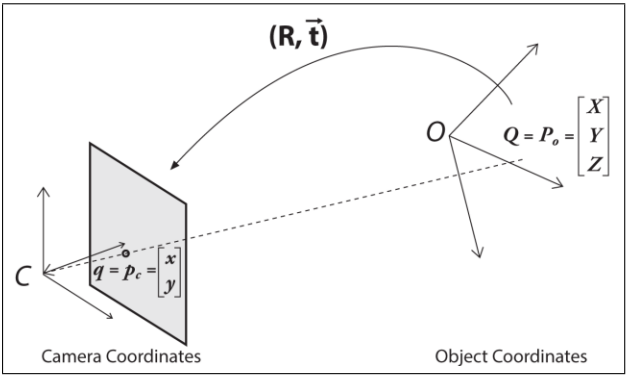

Figure 1: Convertendo do sistema de coordenadas do objeto para o sistema de coordenadas da câmera: o ponto P no objeto é visto como o ponto p no plano da imagem; nós relacionamos o ponto p com o ponto P aplicando uma matriz de rotação R e um vetor de translação t em P

2 Metodologia

Nesta seção são apresentados os métodos e procedimentos utilizados em cada um dos requisitos para obter os resultados pedidos.

2.1 Requisito 1

2.2 Requisito 2

O OpenCV [2] fornece vários algoritmos para nos ajudar a computar os parâmetros intrínsecos e o vetor de distorção. A calibração é feita via *cv::calibrateCamera()*, que já fornece os vetores de translação, rotação, o vetor de distorção e a matriz com os parâmetros intrínsecos da câmera. Para cada imagem que a câmera captura de um objeto particular, nós podemos descrever a pose do objeto relativo ao sistema de coordenadas da câmera em termos da rotação e da translação. Em geral, uma rotação pode ser descrita em termos da multiplicação de um de um vetor de coordenadas por uma matriz quadrada de tamanho apropriado. Basicamente uma rotação é equivalente a introduzir uma nova descrição da localização de um ponto em um sistema de coordenadas diferente. O vetor de translação é como nós representamos um deslocamento de um sistema de coordenadas para outro cuja origem é deslocada para outra localização. Ou seja, o vetor de translação é apenas o deslocamento da origem do primeiro sistema de coordenadas para o segundo. Assim, para mudar de um sistema de coordenadas centrado em um objeto para um centrado na câmera, o vetor de translação apropriado é: $\overrightarrow{T} = origin_{object} - origin_{camera}$. Assim, é possível notar pela **Figura** 1 que um ponto das coordenadas do objeto (ou mundo) $\overrightarrow{P_o}$ tem coordenadas $\overrightarrow{P_c}$ nas coordenadas do frame da câmera: $\overrightarrow{P_c} = R \cdot (\overrightarrow{P_o} - \overrightarrow{T})$.

No OpenCV nós temos 4 parâmetros associados a matriz intrínseca da câmera, cinco (ou mais) parâmetros de distorção, que consistem de três (ou mais) parâmetros radiais, e dois tangenciais. Os parâmetros intrínsecos controlam a transformação linear de projeção que relacionam o objeto físico com a imagem produzida. Na teoria seria necessário apenas

três pontos de cantos em um padrão conhecido para resolver os nossos 5 parâmetros de distorção. Logo, apenas uma *"screenshot"* do tabuleiro de xadrez seria suficiente. Entretanto, devido ao casamento dos parâmetros intrínsecos com os extrínsecos, apenas uma "screenshot" não é suficiente. Pode-se notar que os parâmetros extrínsecos incluem três parâmetros de translação (T_x, T_y, T_z) e três de rotação dando um total de 6 por imagem do tabuleiro de xadrez. Junto com os 4 parâmetros da matriz dos intrínsecos da câmera (f_x, f_y, c_x, c_y) nós temos um total de 10 que precisamos resolver, em um caso de uma única imagem, e 6 adicionais para cada imagem. Supondo que temos N cantos e K imagens do tabuleiro de xadrez (em posições diferentes) é necessário ter $2 \cdot N \cdot K \geq 6 \cdot K \cdot 4((N-3) \cdot K \geq 2)$, onde $K \geq 1$ para resolver o problema de calibração. Entretanto, conforme apresentado em [3], na prática para resultados de alta qualidades é necessário pelo menos 10 imagens de um xadrez 7 x 8 ou maior. Esta disparidade entre as 2 imagens na teoria, e 10 ou mais requeridas na prática, é resultado de um alto grau de sensibilidade que os parâmetros intrínsecos possuem, mesmo com pouco ruído. Por isso, nesse passo, foram obtidas 25 snapshots para obter a matriz de calibração, vetores de translação, rotação e coeficientes de distorção.

2.3 Requisito 3

O parâmetro *objectPoints* da função *cv::calibrateCamera()* foi definido da seguinte forma no programa: o primeiro canto no tabuleiro de xadrez está no ponto (0,0,0), o próximo no (0,1,0), o próximo (0,2,0), e assim em diante. Dessa forma, a escala do vetor *tvec* de translação da saída da função *cv::calibrateCamera* foi implicitamente alterada. Ou seja, ao definir os pontos dos cantos dos tabuleiros de xadrez desta forma mencionada anteriormente, as distâncias são medidas em "quadrados do tabuleiro". Entretanto, os parâmetros da matriz intrínseca da câmera são sempre reportados em pixels. Assim, ao calcular a norma do vetor de translação foi necessário depois multiplicar pela largura ou altura do quadrado do tabuleiro (na unidade correspondente sas distâncias) para comparar com as distâncias $d_{min}, d_{med}, d_{max}$.

Os parâmetros extrínsecos são os vetores de rotação e translação (*tvec e rvec*). Eles são saídas da função *cv::calibrateCamera()* e são apresentados no terminal durante a execução do programa. Cada vetor possui 3 elementos, correspondentes aos eixos *x, y, z*, e temos n de cada um desses vetores, onde n é o número de capturas do tabuleiro.

3 Resultados

Nas subseções seguintes são apresentados os resultados das implementações efetuadas, na forma de figuras.

3.1 Requisitos 1 e 2

Para estes requisitos é feita a seleção de pixels em imagens de acordo com o pixel clicado pelo mouse, conforme detalhado na Introdução [1].

3.2 Requisitos 3 e 4

Para estes requisitos é feita a seleção de pixels em vídeos de acordo com o pixel clicado pelo mouse, conforme detalhado na Introdução. Para o requisito 3 o vídeo deve estar salvo no

computador, e para o 4 deve ser aberto de uma câmera.

4 Discussões e Conclusões

The camera intrinsic matrix is perhaps the most interesting final result, because it is what allows us to transform from three-dimensional coordinates to the image's two- dimensional coordinates.

References

[1] R. I. Hartley and A. Zisserman. *Multiple View Geometry in Computer Vision*. Cambridge University Press, ISBN 0521540518, 2003.

[2] Itseez. Open source computer vision library. https://github.com/itseez/opencv, 2015.

[3] Adrian Kaehler and Gary Bradski. *Learning OpenCV 3: Computer Vision in C++ with the OpenCV Library*. O'Reilly Media, Inc., ISBN 978-1-4919-3800-3, 2016.