

000

001

Projeto Demonstrativo 3

002

003 Pedro Henrique Luz de Araujo
004 pedrohluzaraujo@gmail.com005 Rafael Barbosa de Sousa
006 rafael.1940.b@gmail.com007 Departamento de Ciéncia da
008 Comptutaçáo
009 Universidade de Brasília
010 Campus Darcy Ribeiro, Asa Norte
011 Brasília-DF, CEP 70910-900, Brazil,

012

013

014

015

016

017

018

019

020

021

Abstract

O presente Projeto Demonstrativo visa a implementar um programa capaz de estimar a profundidade de uma cena dadas imagens obtidas por duas câmeras. Para tanto, calcula-se um mapa de disparidade da cena por meio da minimização da soma das diferenças absolutas (SAD) dos pixeis correspondentes nas duas imagens em caso de câmeras paralelas. No caso de câmeras não paralelas, as imagens são retificadas e então é minimizada a SAD. A partir da visão estéreo, obtivemos uma matriz de transformação dos pontos da imagem para o mundo, de modo que foi possível calcular dimensões reais a partir de imagens.¹

022

1 Introdução

023

Um dos ramos da Visão Computacional é a visão estéreo, que se ocupa da reconstrução espacial da cena a partir de duas visões diferentes [1]. Isto é, dado um conjunto de correspondência em imagens $x_i \leftrightarrow x'_i$, obtidos de um conjunto de pontos tridimensionais desconhecidos X_i , a recuperação espacial da cena visa a achar as matrizes \mathbf{P} e \mathbf{P}' tais que

024

$$x_i = \mathbf{P}X_i \text{ e } x'_i = \mathbf{P}'X_i \text{ para todo } i. \quad (1)$$

025

No caso de visões obtidas de câmeras idênticas e perfeitamente paralelas (figura 1), basta saber a distância entre os centros das câmeras, denominada *baseline* (b), e a distância focal das câmeras (f), como evidenciam as equações abaixo:

026

$$X = \frac{b(x_l + x_r)}{2(x_l - x_r)}, Y = \frac{b(y_l + y_r)}{2(x_l - x_r)}, Z = \frac{bf}{(x_l - x_r)}. \quad (2)$$

027

A medida $x_l - x_r$ é denominada disparidade e indica a diferença entre a localização de um mesmo ponto nas duas imagens e é inversamente proporcional à profundidade do ponto. Logo, para estimar a profundidade dos pontos da cena é preciso calcular a disparidade de todos os pontos.

028

No caso em questão, de câmeras paralelas, as coordenadas dos pontos correspondentes possuem o mesmo valor no eixo y . Assim, para encontrar a correspondência de um ponto na outra imagem basta procurar o pixel mais semelhante na mesma fileira do ponto original.

029

© 2018. The copyright of this document resides with its authors.

030

It may be distributed unchanged freely in print or electronic forms.

031

¹Pedro contribuiu com a elaboração do código principal do projeto e com a escrita da introdução, metodologia e resultados. Rafael contribuiu com estruturação do código e do relatório e com a escrita da introdução e metodologia.

046
047
048
049
050
051
052
053
054
055
056
057
058
059
060
061
062
063
064
065
066
067
068
069
070
071
072
073
074
075
076
077
078
079
080
081
082
083
084
085
086
087
088
089
090
091

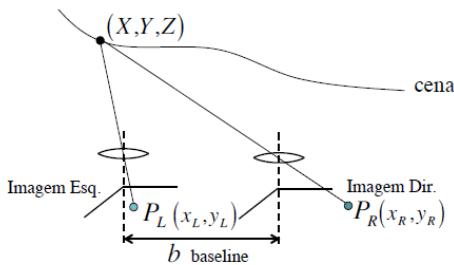


Figure 1: Disparidade e profundidade em câmeras paralelas. Imagem adaptada dos slides de D. A. Forsyth.

Uma medida de semelhança comumente usada é a diferença absoluta, que corresponde à norma l_1 da diferença entre o valor dos pixéis. Por exemplo, no caso de um pixel com três canais:

$$AD = |\mathbf{p} - \mathbf{p}'|_1 = |x - x'| + |y - y'| + |z - z'|. \quad (3)$$

Por outro lado, busca pixel a pixel pode acarretar numa disparidade muito “barulhenta” (*noisy*) ou irregular. Para suavizar (*smooth*) a disparidade, pode-se encontrar correspondência entre janelas de pixéis - quanto maior a janela, maior a suavização da disparidade. Nesse caso, busca-se minimizar a soma das diferenças absolutas (SAD):

$$SAD = \sum_i |\mathbf{p}_i - \mathbf{p}'_i|. \quad (4)$$

Em caso de visões obtidas de câmeras não paralelas, os pontos correspondentes não possuem necessariamente o mesmo valor em y . Torna-se necessário, portanto, diminuir o número de possibilidades dentre os pixéis que podem ser selecionados como correspondentes. Se soubermos os parâmetros extrínsecos e intrínsecos de cada câmera, podemos utilizar as matrizes de rotação e translação de cada uma delas para retificar as imagens, isto é, fazer os pixéis correspondentes terem uma das coordenadas iguais, como demonstram as imagens 2 e 3.

Uma vez retificadas as imagens, basta aplicar o mesmo algoritmo usado para o caso de câmeras paralelas.



Figure 2: Imagens antes da retificação.



Figure 3: Imagens após retificação.

092 2 Metodologia

093

094 2.1 Ferramentas

095

096 Usamos a biblioteca OpenCV [1], versão 3.3.0, com implementações eficientes de algorit-
097 mos de correspondência estéreo e de reprojeções de imagens. Para realizar operações sobre
098 matrizes e vetores, utilizamos a biblioteca de computação numérica em Python, NumPy.
099 Usamos ainda, como linguagem, Python 3.5.2, e o gerenciador de bibliotecas Anaconda 3.

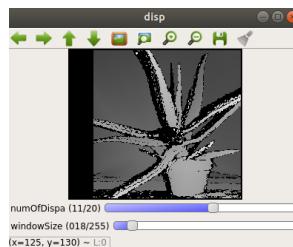
100

101 2.2 Requisito 1

102 O objetivo do requisito 1 é obter os mapas de disparidade e profundidade de imagens obtidas
103 de câmeras paralelas, de modo que os pontos correspondentes nas duas imagens possuem o
104 mesmo valor da coordenada y .

105 Dadas duas imagens pelo usuário, da esquerda e da direita, empregamos a classe *Stere-
106 oBM* da biblioteca OpenCV para obter a disparidade para cada ponto das imagens. De posse
107 da matriz de disparidades, é possível obter as coordenadas do ponto na cena espacial por
108 meio das equações 2. A distância focal das câmeras e o baseline são dados e valem 3640
109 pixels e 160 mm, respectivamente. O tamanho da janela e o número máximo de dispari-
110 dade para a minimização da SAD podem ser escolhidos livremente pelo usuário por meio de
111 interface gráfica, como mostra a imagem 4.

112



113

114 Figure 4: Interface gráfica de ajuste de parâmetros. Máximo de disparidade ajustado em 176
115 (16 × 11, vez que o OpenCV só permite múltiplos de 16 na disparidade) e tamanho da janela
116 ajustado em 19 (18 + 1, pois o OpenCV só permite janelas de tamanho ímpar).

117

118

119

120

121

122 A partir dos valores de disparidade e de profundidade obtidos, normalizamos os valores e
123 construímos uma imagem para o mapa de disparidade e outra para o mapa de profundidade.
124 A normalização foi feita da seguinte forma:

125

126 Para o mapa de disparidade, ajustamos os valores de forma que a maior disparidade seja
127 255 e a menor seja 0. Como a disparidade é inversamente proporcional à profundidade, isso
128 significa que quanto mais próximo o ponto mais claro ele será; quanto mais distante, mais
129 escuro.

130

131

132

133

134

135

136

137

Para o mapa de profundidade, a normalização é semelhante. Os valores são ajustados de
modo que profundidades infinitas ou onde não foi possível calculá-la possuam o valor 255
e a menor profundidade seja 0. Dessa forma, o mapa de profundidade é quase um negativo
do mapa de disparidade: quanto mais próximo o ponto, mais perto de preto ele será; quanto
mais distante, mais perto de branco.

2.3 Requisito 2

O requisito 2, assim como o requisito 1, tem como objetivo a construção de mapa de disparidade e profundidade para uma cena. Entretanto, as câmeras não mais são paralelas, de modo que torna-se necessário retificar as imagens.

Sabendo os parâmetros intrínsecos e extrínsecos de cada câmera, procedemos a retificação das imagens da seguinte maneira. Sejam \mathbf{A}_1 , \mathbf{A}_2 , \mathbf{R}_1 , \mathbf{R}_2 , \mathbf{t}_1 e \mathbf{t}_2 as matrizes de intrínsecos e de rotação e os vetores de translação da primeira e segunda câmera, respectivamente. Podemos obter a matriz de rotação \mathbf{R} e o vetor de translação \mathbf{t} entre a primeira e a segunda câmera da seguinte maneira:

$$\mathbf{R} = \mathbf{R}_2 \mathbf{R}_1^t \quad (5a)$$

$$\mathbf{t} = \mathbf{t}_2 - \mathbf{R}\mathbf{t}_1. \quad (5b)$$

De posse da rotação e da translação entre as câmeras, podemos rotacionar e transladar as imagens de forma que os planos das imagens se tornem paralelos e os pontos correspondentes possuam o mesmo valor da coordenada y . As funções *stereoRectify*, *initUndistortMap* e *remap*, do OpenCV, fazem exatamente isso. A função *stereoRectify* nos retorna, ainda, uma matriz \mathbf{Q} que transforma disparidade em profundidade [1],

$$\mathbf{Q} = \begin{bmatrix} 1 & 0 & 0 & -c_x \\ 0 & 1 & 0 & -c_y \\ 0 & 0 & 0 & f \\ 0 & 0 & \frac{-1}{T_x} & \frac{c_x - c'_x}{T_x} \end{bmatrix}, \quad (6)$$

em que c_x e c_y são as coordenadas do ponto principal da primeira câmera, c'_x é a coordenada x do ponto principal da segunda câmera, f é a distância focal e T_x é a *baseline*.

Após retificarmos as imagens, aplicamos a mesma técnica do requisito 1 para obter os mapas de disparidade e profundidade da cena. Entretanto, em vez de usar as equações 2 para encontrar a profundidade, usamos a matriz \mathbf{Q} :

$$\begin{bmatrix} X \\ Y \\ Z \\ W \end{bmatrix} = \mathbf{Q} \begin{bmatrix} x \\ y \\ d \\ 1 \end{bmatrix} \quad (7a)$$

$$\mathbf{p} = \begin{bmatrix} \frac{X}{W} \\ \frac{Y}{W} \\ \frac{Z}{W} \\ \frac{1}{W} \end{bmatrix}, \quad (7b)$$

em que d é a disparidade do ponto e \mathbf{p} são as suas coordenadas no espaço.

2.4 Requisito 3

Para o requisito 3 deseja-se obter as dimensões da menor caixa capaz de guardar a poltrona onde está posicionado o boneco Morpheus (figuras 5 e 6).

O requisito 2 já fornece uma técnica capaz de obter as coordenadas no mundo real a partir dos pontos na imagem. Portanto, para o requisito 3, desenvolvemos uma aplicação em que o usuário clica em dois pontos e o sistema fornece a distância (norma da diferença entre

184 os pontos) entre os dois pontos em milímetros no mundo real. Assim, medimos três vetores:
185 de largura, de altura e de profundidade. Medimos cada uma dessas distâncias três vezes e
186 calculamos a média e o desvio padrão entre elas, descartando *outliers* evidentes.



199 Figure 5: Imagem capturada Figure 6: Imagem capturada
200 pela câmera 1. pela câmera 2.

3 Resultados

3.1 Requisito 1

207 As figuras 7 e 8 são as duas capturas da imagem de uma planta. As figuras 9 e 10 são,
208 respectivamente, o mapa de disparidade e profundidade da cena, obtidos com uma janela de
209 tamanho 15 e um número máximo de disparidade igual a 176.



216 Figure 7: Planta
217 pela câmera da es-
218 querra.

219 Figure 8: Planta
220 pela câmera da di-
221 reita.

222 Figure 9: Mapa de
223 disparidade.

224 Figure 10: Mapa de
225 profundidade.

226 As figuras 11 e 12, por outro lado, são as duas capturas da imagem de uma boneca. As
227 figuras 13 e 14 são, respectivamente, o mapa de disparidade e profundidade da cena, obtidos
228 com uma janela de tamanho 23 e um número máximo de disparidade igual a 144.

3.2 Requisito 2

229 A figura 15 mostra as imagens de Morpheus após retificação. Já as figuras 16 e 17 mostram
230 o mapa de disparidade e profundidade obtidos por uma janela de tamanho 15 e com número
231 máximo de disparidade igual a 256.



Figure 11: Bebê pela câmera da esquerda.



Figure 12: Bebê pela câmera da direita.



Figure 13: Mapa de disparidade.



Figure 14: Mapa de profundidade.



Figure 15: Imagens de Morpheus após retificação. As linhas horizontais indicam que os pontos correspondentes possuem o mesmo valor da coordenada y.

3.3 Requisito 3

As figuras 18, 19 e 20 mostram, respectivamente, onde medimos a altura, largura e profundidade da poltrona. A tabela 1 exibe os valores obtidos, sendo que o erro foi obtido pelo desvio padrão de cada medida.

Table 1: Medidas da poltrona.

Altura(mm)	Largura(mm)	Profundidade(mm)
184 ± 2	198 ± 3	106 ± 4

4 Discussão e Conclusões

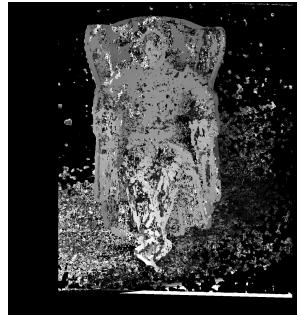
Em relação às reconstruções das cenas da planta e da boneca, é possível constatar que a técnica trouxe bons resultados, sendo possível identificar a forma da planta e da boneca sobre a caixa, com um certo grau de detalhamento. Destacamos que o mapa de profundidade é praticamente um negativo do mapa de disparidade, o que vai ao encontro do esperado teoricamente, uma vez que a profundidade é inversamente proporcional à disparidade.

Quanto à reconstrução da cena que envolve um boneco do personagem Morpheus, inicialmente notamos que a retificação foi satisfatória, já que os pontos correspondentes se encontram dentro da mesma linha horizontal. Os mapas de profundidade e disparidade, igualmente, são razoáveis, sendo que o boneco sobre a poltrona se apresentam mais próximos que o fundo e os contornos são perceptíveis, inclusive em relação à pose de Morpheus.

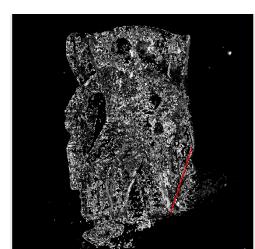
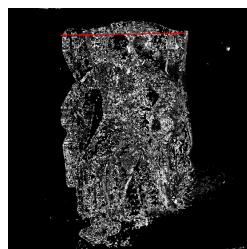
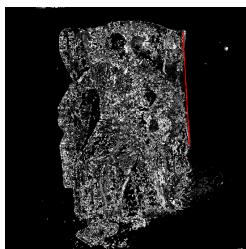
230
231
232
233
234
235
236
237
238
239
240
241
242
243
244
245
246
247
248
249

250
251
252
253
254
255
256
257
258
259
260
261
262
263

264
265
266
267
268
269
270
271
272
273
274
275



276
277
278
279
280
281
282
283
284
285
286 Figure 16: Mapa de dispari- Figure 17: Mapa de profun-
287 dade.
288



299
300
301
302
303
304
305
306
307
308
309
310 Figure 18: Medida da altura.

311
312
313
314
315
316
317
318
319 Figure 19: Medida de largura

320
321 Figure 20: Medida de profundidade.

Tanto no caso do requisito 1, quanto no do requisito 2, percebemos que quanto maior o tamanho da janela de busca de pixels correspondentes, maior a suavização do mapa de disparidade. Como o valor ideal de tal parâmetro variava para cada imagem, optamos por deixar o usuário escolhê-lo em tempo real.

Os valores encontrados para as dimensões de uma caixa capaz de guardar a poltrona também são razoáveis. Todos estão na ordem dos milímetros, o que parece ser compatível com o tamanho de um boneco, embora não tenhamos os valores verdadeiros deste. Por outro lado, no momento do cálculo das medidas era comum a presença de claros *outliers*, com medidas na ordem dos metros. Isso pode ocorrer por decorrência de um clique do mouse em um região onde não foi possível calcular a disparidade, que no momento de conversão para profundidade acaba assumindo um valor muito grande.

O principal problema evidenciado pelos resultados é a dificuldade do algoritmo em encontrar correspondência de pontos em regiões uniformes, como as paredes presentes nas imagens das plantas e do bebê e o fundo das imagens do Morpheus. No primeiro caso, tal dificuldade acarreta a má identificação da profundidade da parede: os pontos da parede devem ser os pontos com maior profundidade, mas não são identificados assim. No caso do Morpheus, os pontos do fundo sequer tem sua disparidade identificada.

Uma possível solução para isso seria o uso combinado com outras técnicas para a estimativa de profundidade de regiões homogêneas, como a segmentação, por exemplo, e o uso de informações de cores.

References

[1] G. Bradski. The OpenCV Library. <i>Dr. Dobb's Journal of Software Tools</i> , 2000.	322
	323
[2] Gary Bradski and Adrian Kaehler. <i>Learning OpenCV: Computer Vision in C++ with the OpenCV Library</i> . O'Reilly Media, Inc., 2nd edition, 2013. ISBN 1449314651, 9781449314651.	324
	325
	326
	327
[3] Richard Hartley and Andrew Zisserman. <i>Multiple View Geometry in Computer Vision</i> . Cambridge University Press, New York, NY, USA, 2 edition, 2003. ISBN 0521540518.	328
	329
	330
	331
	332
	333
	334
	335
	336
	337
	338
	339
	340
	341
	342
	343
	344
	345
	346
	347
	348
	349
	350
	351
	352
	353
	354
	355
	356
	357
	358
	359
	360
	361
	362
	363
	364
	365
	366
	367