

000

001 

# Projeto Demonstrativo 3

002

003  
004 Raphael Soares 14/0160299  
005 raphael.soares.1996@gmail.com006  
007 Departamento de Ciéncia da  
008 Computaçao  
Universidade de Brasília  
Campus Darcy Ribeiro, Asa Norte  
Brasília-DF, CEP 70910-900, Brazil,

009

010

011

**Abstract**

012

013 Todos nós estamos familiares com a capacidade de imageamento estéreo que nossos  
014 olhos nos fornecem. Em qual grau podemos simular essa capacidade em sistemas com-  
015 putacionais? Os computadores realizaram essa tarefa achando correspondências entre  
016 pontos que são vistos pelas duas câmeras. Com essa correspondência e com a distância  
017 de separação entre as duas câmeras conhecida é possível calcular a localização tridi-  
018 mensional dos pontos. Esse segundo projeto tem como objetivo principal explorar e  
019 desenvolver algoritmos para extração de mapas de profundidade a partir de pares estéreo  
020 de imagens. Esses mapas foram obtidos a partir do mapa de disparidade, que contém  
021 informação de disparidade dos pontos correspondentes vistos pelas duas câmeras. Para  
022 as imagens obtidas de câmeras que não estavam alinhadas em paralelo foi necessário  
023 retificá-las antes. Além disso, medidas de um objeto de uma imagem foram estimadas  
024 calculando a distância da localização tridimensional dos pontos, assim como no projeto  
025 demonstrativo 2.

026

027

## 1 Introdução

028

029 Nós achamos pontos correspondentes em nossos olhos esquerdos e direitos e usamos isso  
030 para trabalhar o quanto longe algum objeto está de nós. Com apenas um olho nós temos  
031 algumas pistas monoculares que podemos usar para estimar profundidade, entretanto o ver-  
032 dadeiro “3D”, a verdadeira percepção de profundidade só existe quando temos dois olhos.  
033 Com um único olho é possível obter apenas deduções, como saber a distância de um objeto  
034 observando o tamanho dele em dois instantes de tempo diferentes. Os computadores real-  
035 izam essa tarefa de imageamento estéreo dos nossos olhos achando correspondências entre  
036 pontos que são vistos por duas câmeras. Para computadores, apesar da busca de pontos cor-  
037 respondentes ser computacionalmente cara, é usado o conhecimento de geometria do sistema  
038 para limitar a busca o máximo possível [1]. Na prática, o imageamento estéreo feito nesse  
projeto envolveu 3 passos, já que as imagens usadas foram obtidas a partir de duas câmeras:

039

- 040
1. Ajuste dos ângulos e das distâncias entre as câmeras, que é conhecido como retifi-  
041 cação. A saída desse passo são as imagens retificadas e alinhadas por linha<sup>1</sup>.

042

043 © 2018. The copyright of this document resides with its authors.

044

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

046

047 <sup>1</sup>A principal informação que o computador precisa para fazer imageamento estéreo é saber onde estão nossas  
câmeras. Note que no caso dos nossos olhos, o cérebro já sabe onde estão nossos olhos e eles já estão “alinhados  
por linha”, ou seja, mesma coordenada y.

2. Busca das mesmas características na visão das duas câmeras (que também poderiam estar orientadas verticalmente, mudando assim as disparidades), um processo conhecido como correspondência. A saída desse passo é o mapa de disparidade, onde as disparidades são as diferenças no eixo  $x$  nos planos da imagens das mesmas características vistas na câmera da esquerda e da direita:  $x_l - x_r$ . 046  
047  
048  
049  
050
3. Sabendo o arranjo geométrico das câmeras, é possível transformar o mapa de disparidade em profundidade, usando triangulação. Esse passo é chamado de reprojeção e a saída é o mapa de profundidade. 051  
052  
053  
054

Normalmente, seria necessário um passo adicional para remover as distorções radiais e tangenciais da lente antes da retificação. Entretanto, as imagens usadas tanto no requisito 1 quanto no 2 já estão sem distorção. 055  
056  
057

## 2 Metodologia

Nesta seção são apresentados os métodos e procedimentos utilizados em cada um dos requisitos para obter os resultados pedidos. 060  
061  
062  
063  
064

### 2.1 Requisito 1

No Requisito 1 foi necessário fazer a correspondência estéreo (casamento de pontos tridimensionais em visões diferentes da câmera) entre as duas imagens. A título de comparação, dois algoritmos foram utilizados para fazer a correspondência estéreo. Ambos algoritmos de casamento estéreo servem ao mesmo propósito: converter duas imagens, uma esquerda e uma direita, em uma única imagem de profundidade. Esta imagem basicamente irá associar com cada pixel uma distância das câmeras para o objeto que esse pixel representa. 067  
068  
069  
070  
071  
072

O primeiro, denominado *block matching (BM)* é um algoritmo rápido e efetivo que é similar ao desenvolvido por Kurt Konolige [7]. Ele funciona usando pequenas janelas de “somas de diferenças absolutas” (SAD) para encontrar pontos correspondentes entre as imagens estéreo retificadas da esquerda e da direita. Este algoritmo encontra somente pontos com alta correspondência entre as duas imagens (alta textura). Assim, em uma cena altamente texturizada todos os pixels vão ter profundidade computada. Em uma cena com pouca textura, como um corredor, poucos pontos devem registrar profundidade. 073  
074  
075  
076  
077  
078  
079

O segundo é conhecido como *semi-global matching (SGBM) algorithm*. SGBM, uma variação do SGM introduzido em [8], difere do BM em dois aspectos. O primeiro é que o casamento é feito em nível de subpixel usando a métrica Birchfield-Tomasi [9]. A segunda diferença é que o SGBM tenta impor uma limitação global de suavidade, na informação de profundidade computada. Esses dois métodos são complementares, no sentido que o BM é mais rápido, mas não fornece a confiança e acurácia do SGBM. 080  
081  
082  
083  
084  
085

Ambos os algoritmos são implementados pela OpenCV [9] e são melhor detalhados e explicados em [10]. A saída destes algoritmos é o mapa de disparidade. O mapa de profundidade é calculado a partir desse mapa de disparidade, usando os valores de  $b$  e  $f$  fornecidos. A dimensão da janela  $W$ , utilizada para a realização da correspondência, foi 9. Esse bloco  $W$  é necessário tanto para o BM quanto para o SGBM. 086  
087  
088  
089

Para normalizar a imagem em tons de cinza com valores de intensidade no intervalo (Min, Max) para valores de intensidade no intervalo (newMin, newMax) foi usada a seguinte 090  
091

092 fórmula:

$$I_N = (I - \text{Min}) \frac{\text{newMax} - \text{newMin}}{\text{Max} - \text{Min}} + \text{newMin}$$

095 Essa fórmula foi utilizada para que o intervalo dinâmico dos valores dos pixels não fosse  
096 perdido. Aqui  $I$  representa a antiga imagem e  $I_N$  a nova.

097

### 098 2.1.1 Block Matching

099

100 O algoritmo estéreo BM implementando na OpenCV é uma versão modificada do que se  
101 tornou uma das técnicas canônicas para computação estéreo. O mecanismo básico é retificar  
102 e alinhar as imagens de tal forma que as comparações precisem ser feitas apenas em linhas  
103 individuais, e então ter um algoritmo que procura linhas nas duas imagens para grupos corre-  
104 spondentes de pixels. O resultado é um algoritmo confiável que é vastamente usado. Existem  
105 três estágios para o algoritmo estéreo bm, que funciona em pares de imagens retificadas e  
106 sem distorção:

107

- 108 1. Pré-filtragem para normalizar o brilho da imagem e realçar a textura.
- 109 2. Busca por correspondência ao longo das linhas horizontais epipolares usando uma  
110 janela SAD.
- 111 3. Pós-filtragem para eliminar correspondências ruins de casamentos.

112

### 113 2.1.2 Semi-global block matching

114

115 O algoritmo SGM, que deriva o SGBM<sup>2</sup> implementado pela OpenCV, possui várias no-  
116 vas ideias, mas um custo computacional bem maior que o BM. As mais importantes no-  
117 vas ideias introduzidas no SGM são o uso de informação mútua como uma medida su-  
118 perior de correspondência local, e o reforço de restrições consistentes ao longo de outras  
119 direções além da linha (epipolar) horizontal (na implementação foi utilizado o valor *Stere-  
120 oSGBM::MODE\_SGBM* que denota a versão com cinco direções do algoritmo). De um  
121 modo geral, os efeitos dessas adições são fornecer uma maior robustez para iluminação e  
122 outras variações entre as imagens da esquerda e da direita, e ajudar a eliminar erros im-  
123 pondo restrições geométricas mais fortes através da imagem. O ponto principal do algoritmo  
124 é como atribuir um custo para cada pixel para todos as possíveis disparidades. Essencial-  
125 mente, isso é análogo ao que é feito no *block matching*, mas há alguns novos passos. O  
126 primeiro novo passo é que é usado as métricas de Birchfield-Tomasi para comparar pixels,  
127 em vez de usar soma das diferenças absolutas. O segundo novo passo é que é usado uma  
128 importante suposição para a continuidade de disparidade: pixels vizinhos provavelmente tem  
129 a mesma ou disparidade similar. Ao mesmo tempo é usado um bloco de tamanho menor. In-  
130 clusivo, no BM, janelas grandes tendem a ser um problema perto de discontinuidades (borda  
de algum elemento da imagem).

131

## 132 2.2 Requisito 2

133

134 É mais fácil calcular a disparidade estéreo quando os dois planos da imagem se alinham  
135 exatamente. Como mostra a **Figura 1**, nesse requisito foi feita a estéreo retificação que

136

<sup>2</sup>No SGBM também é usado um bloco, entretanto esse bloco de tamanho  $W$  configura o tamanho da região em  
137 torno de cada pixel onde a métrica do “sinal da diferença absoluta” será computada.

consiste em reprojetar os planos das imagens das duas câmeras de tal forma que eles residam no mesmo plano, com as linhas das imagens perfeitamente alinhadas em uma configuração frontal paralela.

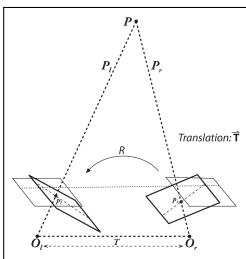


Figure 1: O objetivo é matematicamente alinhar as duas câmeras em um plano de visão, de forma que as imagens fiquem exatamente alinhadas e a busca por pixels fique mais restrita.

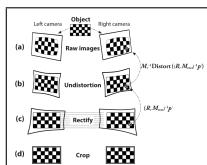
Para um ponto tridimensional  $\vec{P}$  nas coordenadas do objeto, nós podemos separá-lo usando a calibração de uma única câmera para as duas câmeras. Assim, para por  $\vec{P}$  nas coordenadas da câmera para cada câmera fazemos:  $\vec{P}_l = R_l \cdot (\vec{P}) + \vec{T}_l$  e  $\vec{P}_r = R_r \cdot (\vec{P}) + \vec{T}_r$ . As duas vistas deste ponto  $\vec{P}$  (obtidas das duas câmeras) são relacionadas por  $\vec{P}_l = R^T \cdot (\vec{P}_r - \vec{T})$ , onde  $R$  e  $\vec{T}$  são, respectivamente, a matriz de rotação e o vetor de translação entre as câmeras. Usando essas três equações é possível obter a rotação  $R$  e a translação  $\vec{T}$  que foi usada para a retificação das imagens do Morpheus:

$$R = R_r \cdot R_l^T \text{ e } \vec{T} = \vec{T}_r - R \cdot \vec{T}_l.$$

Aplicando o método para alinhar os planos da imagens descrito acima, que é conhecido como o algoritmo de Bouget apresentado em [8], é obtida a configuração estéreo necessária. Os novos centros da imagens e as novas bordas são então escolhidas para as imagens rotacionadas, de forma a maximizar a área de visualização sobreposta. No contexto da biblioteca OpenCV, o algoritmo de Bouget é implementado pela função `cv::stereoRectify()` [9]. Para esta função é fornecido as matrizes das câmeras, o tamanho da imagem,  $R$  e  $\vec{T}$  calculados anteriormente. Os parâmetros de retorno são  $R_l$ ,  $R_r$  (as rotações para a retificação dos planos esquerdo e direito da imagem),  $P_l$  e  $P_r$  (matrizes 3x4 de projeção) e, por fim, a matriz de reprojeção  $Q$ .

O processo de retificação é então feito pela função `cv::initUndistortRectifyMap()`, usando as saídas  $R_l$ ,  $R_r$ ,  $P_l$  e  $P_r$ . Esta função é chamada duas vezes, uma para a imagem da esquerda e uma para a da direita. O processo de retificação é ilustrado na Figura 2. Como mostrado na equação da figura, o processo de retificação funciona de (c) para (a) em um processo conhecido como mapeamento reverso. Para cada pixel inteiro na imagem retificada (c), é encontrado as suas coordenadas na imagem sem distorção (b) e eles são usados para procurar as verdadeiras coordenadas (ponto flutuante) na imagem de origem (a). O valor da coordenada do pixel em ponto flutuante é então interpolado com os pixels vizinhos inteiros na imagem original, e este valor é usado para preencher a posição dos pixels inteiros retificados na imagem de destino (c).

Depois da retificação é usada a função `remap` para finalizar o processo de retificação e permitir a busca por elementos ao longo da mesma linha (mesma coordenada y) nas duas imagens. Os mapas de disparidade foram obtidos assim como no 2.1, usando o SGBM.

184  
185  
186  
187  
188189 Figure 2: Retificação estéreo para as imagens das câmeras da esquerda e da direita.  
190191  
192 

## 2.3 Requisito 3

193 Pontos em duas dimensões podem ser reprojetados em três dimensões dados as coordenadas  
194 das câmeras e a matriz intrínseca da câmera. A matriz de reprojeção  $Q$  é:  
195

$$\begin{bmatrix} 1 & 0 & 0 & -c_x \\ 0 & 1 & 0 & -c_y \\ 0 & 0 & 0 & f \\ 0 & 0 & \frac{-1}{T_x} & \frac{c_x - c_x'}{T_x} \end{bmatrix}$$

201  
202 Os parâmetros são da imagem da esquerda, exceto por  $c_x'$ , que é o ponto principal na coor-  
203 denada x da imagem da direita. Se os raios principais se cruzam no infinito, então  $c_x' = c_x$ .  
204 Dado um ponto bidimensional homogêneo e sua disparidade associada  $d$ , nós podemos pro-  
205 jetar o ponto em 3D usando a matriz  $Q$ , conforme a função da OpenCV utilizada: *reprojec-*  
206 *tImageTo3D()*.207 Após os pontos terem sido projetados nas coordenadas do mundo real, foi calculado a  
208 norma entre os pontos para obter as medidas pedidas.

209

210 

## 3 Resultados

211

212 Nesta seção são apresentados em forma de figuras e tabelas os resultados da aplicação para  
213 cada um dos requisitos.

214

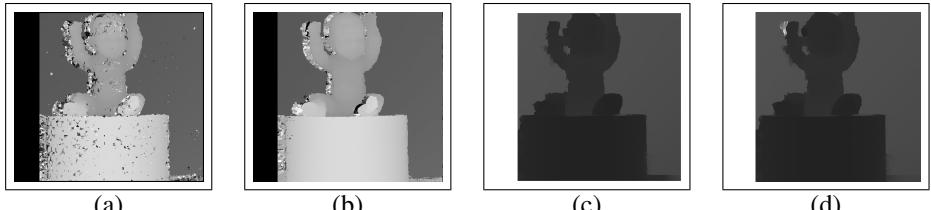
215 

### 3.1 Requisito 1

216

217 As imagens da **Figura 3** e **4** mostram a comparação do algoritmo BM e SGBM para calcular  
218 os mapas de disparidade e profundidade na imagem *baby* e *aloe*.

219

220  
221 Figure 3: (a), (b), (c) e (d) são as imagens de disparidade usando bm e sgbm e as imagens de  
222 profundidade usando bm e sgbm, respectivamente.  
223

224

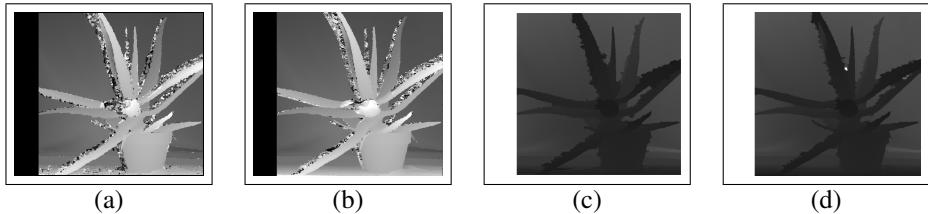


Figure 4: (a), (b), (c) e (d) são as imagens de disparidade usando bm e sgbm e as imagens de profundidade usando bm e sgbm, respectivamente.

Conforme mostra a **Tabela 3.1**, o SGBM teve um tempo de execução mais de 20 vezes superior ao BM, para que a correspondência/casamento entre os pontos apresentada em [2.1](#) fosse realizada, o que já era esperado.

**Table 1: Tabela de comparação BM x SGBM.**

	Baby	Aloe
BM	<b>Correspondência: 0.22s</b> <b>Filtragem: 0.1s</b>	<b>Correspondência: 0.15s</b> <b>Filtragem: 0.09s</b>
SGBM	<b>Correspondência: 3.6s</b> <b>Filtragem: 0.095s</b>	<b>Correspondência: 3.8s</b> <b>Filtragem: 0.096s</b>

### 3.2 Requisito 2

Como pode-se notar pela **Figura 5**, a retificação estéreo explicada em [2.2](#) foi feita com sucesso. Ou seja, as imagens estão alinhadas e os planos de projeção ficaram paralelos de modo que um pixel de um elemento da imagem esquerda está, na teoria, “alinhado” com o mesmo pixel no mesmo elemento da imagem direita. Assim, a busca por elementos correspondentes nas imagens se tornou bem mais restrita.

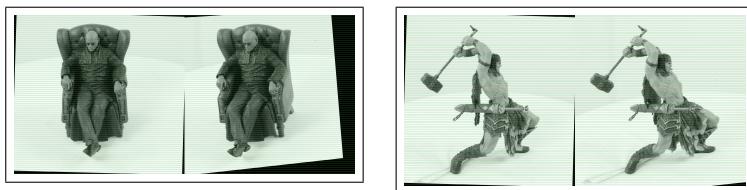


Figure 5: Imagens do morpheus e do warrior retificadas e alinhadas.

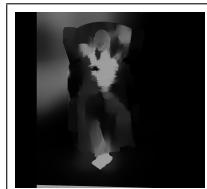
A **Figura 6** apresenta a disparidade e profundidade do Morpheus e do Warrior.

### 3.3 Requisito 3

As dimensões para a caixa (largura, altura e profundidade) obtidas foram, respectivamente: 15.1407 x 21.6495 x 9.45939 cm. Elas foram obtidas primeiro medindo a largura a partir do canto superior esquerdo do sofá, depois medindo a altura partindo do mesmo ponto, e por fim a profundidade foi obtida a partir da diagonal, clicando primeiro no canto inferior direito e depois no canto superior direito.



(a)



(b)



(c)



(d)

Figure 6: (a) e (b) são as imagens de disparidade e profundidade obtidas para o Morpheus, respectivamente. Assim como (c) e (d) são as do Warrior.

## 4 Discussões, Conclusões e Análise de Parâmetros

Pode-se notar pela **Figura 3** e **4** que o SGBM de fato cumpriu com o que prometeu e forneceu maior robustez e acurácia nos resultados, comparado ao mais velho BM, apesar do mapa de profundidade não ter tantas diferenças aparentes. No mapa de profundidade, ao contrário do mapa de disparidade, pixels mais escuros significam uma maior proximidade do elemento da imagem para a câmera.

Conforme dito anteriormente, no BM a correspondência é computada deslizando a janela SAD (soma das diferenças absolutas) nas imagens e no SGBM é usada um bloco com uma métrica de diferença absoluta dos sinais. Para cada característica na imagem da esquerda, nós procuramos a linha correspondente na imagem da direita por um melhor casamento. Depois da retificação, cada linha é uma linha epipolar de modo que a posição correspondente do ponto procurado na imagem da direita deve estar na mesma linha (mesma coordenada y) da imagem da esquerda. Esta posição pode ser encontrada se possui textura suficiente para ser detectável e se não está oculto na visão da câmera da direita. As posições não encontradas observadas nos mapas de disparidade apresentados em **3** são justamente aquelas que não satisfazem estas duas condições. Isso vale, principalmente, para as imagens do morpheus, onde há mais regiões ocultas e uniformes. Nestas regiões uniformes (cenários com pouca textura), poucos pontos registraram profundidade. Enquanto em regiões com maior textura, muitos pixels registraram profundidade.

Além disso, quanto maior o tamanho  $W$  da janela usada tanto no BM quanto no SGBM, menos falsas correspondências deveríamos encontrar. Entretanto, não só o custo computacional pode aumentar com o aumento de  $W$ , mas também há o problema que surge com a suposição implícita que a disparidade é a mesma na área da janela. Perto de descontinuidades de pixels (contornos dos objetos) essa suposição é falsa, e é possível não haver casamento. O resultado será regiões vazias onde não há disparidade perto dos contornos/bordas dos objetos.

## References

- [1] Stan Birchfield and Carlo Tomasi. Depth discontinuities by pixel-to-pixel stereo. *International Journal of Computer Vision*, 35(3):269–293, 1999.
- [2] R. I. Hartley and A. Zisserman. *Multiple View Geometry in Computer Vision*. Cambridge University Press, ISBN 0521540518, 2003.

- [3] Heiko Hirschmuller. Stereo processing by semiglobal matching and mutual information. *IEEE Transactions on pattern analysis and machine intelligence*, 30(2):328–341, 2008. 322  
323
- [4] *The OpenCV Reference Manual*. Itseez, 3.2.0 edition, Dezembro 2016. 324  
325
- [5] Itseez. Open source computer vision library. <https://github.com/itseez/opencv>, 2018. 326  
327  
328
- [6] 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. 329  
330
- [7] Konolige Kurt. Small vision system: Hardware and implementation. In *Robotics research*, pages 203–212. Springer, Japan, 1998. 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