Obtenção do mapa de disparidade em imagens capturadas através de software próprio

William Takeshi Omoto
Universidade Tecnológica Federal do Paraná
Av. Monteiro Lobato, Km 04
Ponta Grossa, Paraná - Brasil
williamomoto@gmail.com

Cauê Felchar
Universidade Tecnológica Federal do Paraná
Av. Monteiro Lobato, Km 04
Ponta Grossa, Paraná - Brasil
caue.fcr@gmail.com

Erikson F. Morais
Universidade Tecnológica Federal do Paraná
Av. Monteiro Lobato, Km 04
Ponta Grossa, Paraná - Brasil
emorais@utfpr.edu.br

ABSTRACT

This paper describes the importance of obtaining the depth of images in the area of computer vision. From images captured through a stereo pair, calculations can be performed to estimate the depth of objects in the image. For this, an essential step is the estimation of disparity, which is the change of position of objects, given a pair of images captured from different perspectives. However, for disparity to be estimated, two basic problems must be solved. The first one is called correspondence, which is the process of finding the corresponding pixels of the first image in the second one. The second problem, known as Reconstruction, is the depth perception obtained by computing the position difference of corresponding objects in the two images. A solution to the first problem is presented in this developing study, through the window approach.

RESUMO

Este artigo descreve a importância de obter a profundidade de imagens na área de visão computacional. A partir de imagens capturadas através de um par estéreo, cálculos podem ser realizados para estimar a profundidade de objetos na imagem. Para tal, um passo essencial é a estimação da disparidade, que é a mudança de posição de objetos, dado um par de imagens capturadas de perspectivas diferentes. No entanto, para a disparidade ser estimada, dois problemas básicos devem ser resolvidos. O primeiro é com relação a correspondência, que é o processo de encontrar os pixels correspondentes da primeira imagem, na segunda. O segundo problema, conhecido como Reconstrução, é a percepção de profundidade obtida ao computar a diferença de posição dos objetos correspondentes nas duas imagens. Uma solução para o primeiro problema é apresentada neste estudo em desenvolvimento, através da abordagem por janela.

Palavras-chave

Mapa de disparidade; Par Estéreo; Calibração de Câmeras.

1. INTRODUÇÃO

Sistemas de visão computacional são desenvolvidos para obter dados através de imagens. Um dos dados mais importantes para áreas como a robótica, realidade virtual e realidade aumentada, são dados de profundidade [1, 5].

Apesar de uma imagem conter muita informação, não é possível inferir dados sobre a profundidade de objetos somente através dela [2]. A profundidade de um objeto pode ser estimada quando se possui duas ou mais imagens de uma mesma cena, porém com perspectivas diferentes. Através da mudança de posição do objeto entre as imagens, é possível estimar a distância do objeto até a câmera, utilizando cálculos de triangulação. Grande parte dos animais possuem ao menos um par de olhos para estimar distâncias, e assim conseguir se deslocar, evitando obstáculos [2, 3].

O cérebro humano, porém, não obtém a profundidade somente de uma maneira, ele estima a distância de objetos através de diversas formas diferentes, e escolhe a que melhor se encaixa naquele momento. Computadores também podem estimar a profundidade de diversas maneiras, como profundidade por foco, defoco, perspectiva, e a mais conhecida e utilizada atualmente, a estereoscopia [4].

A estereoscopia, em termos simples, é como seres humanos obtém a percepção de profundidade. Através de duas imagens obtidas ao mesmo tempo de perspectivas diferentes, é possível inferir dados sobre a distância de objetos [8]. O processo de obtenção da profundidade através da estereoscopia consiste em quatro passos básicos: Obter as imagens, calibrar a câmera e tratar as imagens, obter o mapa de disparidade, e obter o mapa de profundidade [2, 3].

Porém, obter as imagens, e calibrar a câmera, não são processos triviais. As duas câmeras utilizadas devem estar em um *setup* fixo e conhecido para que não ocorra divergências entre as imagens coletadas. Além disso, caso o plano de visão de uma das câmeras sofrer mudanças, a outra câmera deve ser alterada da mesma forma.

A disparidade é qualquer mudança entre a posição de objetos no par de imagens, e é inversamente proporcional à distância do objeto até a lente da câmera [1]. Portanto a distância pode ser calculada obtendo a disparidade de um par estéreo, e a disparidade pode ser estimada através da correspondência dos objetos nas imagens.

Segundo Trucco et al [3], um sistema estéreo deve resolver dois problemas principais. O primeiro é a correspondência, que é o processo de encontrar os pixels da primeira imagem na segunda imagem. O segundo é conhecido como Reconstrução, que é a percepção de profundidade obtida através da diferença de posição dos objetos correspondentes nas duas imagens.

O problema da correspondência, pode ainda ser dividido em duas abordagens. A primeira, é a abordagem por Intensidade, na

qual o objeto encontrado na primeira imagem será procurado na segunda imagem inteira. A segunda, é a abordagem por Janela, na qual é determinada uma janela com tamanho fixo, e a correspondência do objeto na primeira imagem será procurado somente dentro dessa janela, seguindo a linha epipolar [1]. No primeiro caso, é mais provável que se encontre a correspondência de todos os pixels, visto que a procura é realizada na imagem inteira, porém a busca se torna dispendiosa devido ao grande número de cálculos que devem ser realizados. No segundo caso, a janela limita o número de cálculos a serem realizados, porém uma janela muito pequena pode não englobar a correspondência correta.

Este trabalho em desenvolvimento, propõe a construção de um *software* para o controle das câmeras estéreo, e a estimação da disparidade através da correspondência por Janela, visto que o mesmo realiza menos cálculos, e com uma janela de tamanho adequado, o resultado final da disparidade pode ser equivalente à correspondência por Intensidade.

2. DESENVOLVIMENTO DO SOFTWARE

A coleta de imagens e posicionamento de câmeras para fins científicos pode ser feita de várias maneiras, uma delas seria utilizando a própria interface das câmeras, se a mesma permitir que isso seja feito de uma maneira eficiente, ou podemos utilizar um middleware, que teria o papel de fazer a configuração e captura de imagens. Criar um middleware para tal tarefa seria contra-produtivo caso o objetivo seja utilizar poucas câmeras e obter poucas imagens, mas como se fez necessário garantir que as imagens sejam síncronas, e por se fazer necessário obter uma grande quantidade destas imagens a automação do processo de captura e controle das câmeras começam a fazer sentido.

Este trabalho utiliza um middleware para captura, o CtrlCam[6], desenvolvido especialmente para abstrair a interface de controle de câmeras, e obtenção da mesma, além da biblioteca opencv[7]. Com essas partes, é possível posicionar as câmeras adequadamente e capturar imagens de duas câmeras simultaneamente. Um resultado deste programa é demonstrado na Figura 3.

As câmeras utilizadas neste projeto foram do modelo IP607W, da marca Smarteye, com as configurações de brilho em 8, contraste em 4, frequência 60Hz e resolução 640x480, escolhidos empiricamente como as que produziam a melhor qualidade de imagens [8].

3. METODOLOGIA

Essa seção apresenta os passos e ferramentas utilizadas para o desenvolvimento deste trabalho. O primeiro passo é a obtenção das imagens através do *middleware* desenvolvido, e calibração das câmeras utilizando a *toolbox* de calibração estéreo do Matlab. O segundo é o cálculo da disparidade utilizando a biblioteca de visão computacional OpenCV. O último passo é a análise dos resultados para otimizar o algoritmo desenvolvido.

4. CALIBRAÇÃO DA CÂMERA

Existem métodos da geometria epipolar, que permitem a estimação da disparidade sem a calibração da câmera, como desenvolvido por Kukelova [9]. Porém, para a reconstrução da cena tridimensionalmente, é necessário conhecer os parâmetros intrínsecos e extrínsecos da câmera no mundo real [2].

A calibração de imagens estéreo calcula a relação geométrica entre as matrizes de rotação e translação nas duas câmeras no espaço tridimensional [10]. Para efetuar a calibração das câmeras, foi utilizado a *toolbox* de calibração estéreo do Matlab.

A utilização da *toolbox* é simples, bastando apenas indicar quais são as imagens referentes as duas câmeras. Como pode ser visualizado na Figura 1, o próprio software calcula a disposição das imagens no plano tridimensional, e os representa em um gráfico tridimensional.

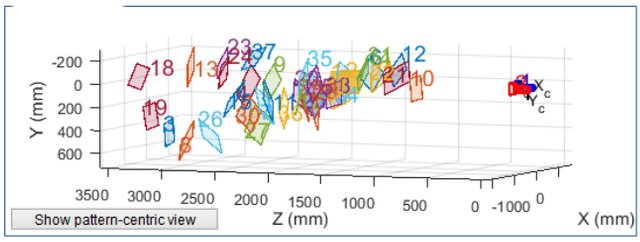

Figura 1: Disposição das imagens de teste no plano

Além da representação das imagens no plano tridimensional, a *toolbox* também calcula os erros de reprojeção em cada imagem, e os representa em um gráfico de barras, como pode ser visualizado na Figura 2. Assim, é possível descobrir os *outliers*, bem como a média do erro da reprojeção.

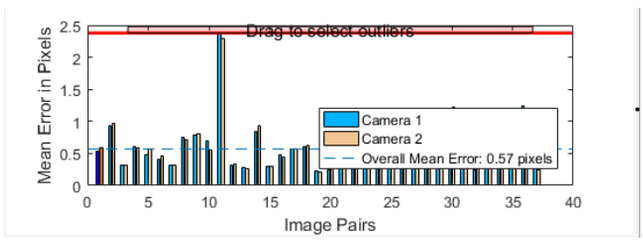

Figura 2: Erros de reprojeção

4.1 Correção das imagens

Com as câmeras calibradas, é possível realizar o processo de retificação e eliminação da distorção radial das imagens. Esses dois processos eliminam erros que ocorrem durante a captura das imagens, como pode ser visualizado na Figura 3 (a), linhas que deveriam ser retas na imagem, aparecem curvas, o que não ocorre na Figura 3 (b), onde foi realizado o processo de retificação e eliminação da distorção radial.

(a) (b)

Figura 3: Correção das imagens

5. OBTENÇÃO DO MAPA DE DISPARIDADE

Para o cálculo do mapa de disparidade, foi utilizada a linguagem de programação C++, juntamente com a biblioteca de visão computacional OpenCV na versão 3.1, pois versões anteriores não possuem um suporte satisfatório a operações estéreo, necessárias para a realização deste trabalho. A IDE utilizada é o Netbeans na versão 8.1.

5.1 Conversão de Cores

Após especificadas as imagens à serem analisadas, ambas são convertidas para tons de cinza utilizando uma função disponível na biblioteca OpenCV.

Mapas de disparidade calculados com base nas três matrizes, geralmente obtém resultados melhores, porém exigem muito mais processamento computacional. Ao converter a imagem para tons de cinza, os cálculos são limitados a somente uma matriz, obtendo resultados bons, sem comprometer o desempenho [9].

5.2 Obtendo o Mapa de Disparidade

Para a obtenção do mapa de disparidade, é preciso resolver os dois problemas básicos, citados na seção 1. Para isso o OpenCV disponibiliza algumas funções que ajudam nesses problemas. As principais funções utilizadas serão detalhadas abaixo.

5.2.1 StereoBM::Create()

Função utilizada para estabelecer a correspondência entre as imagens analisadas. Recebe como parâmetros dois inteiros. O primeiro define alcance máximo da busca pela disparidade. O algoritmo buscará a melhor disparidade desde 0, que é a disparidade padrão, até o número informado.

O segundo define o tamanho do bloco que será utilizado pelo algoritmo para realizar a comparação das imagens. Deve sempre ser um número ímpar, pois deve existir um pixel central. Quanto maior o bloco, mais suave é o resultado, porém mais impreciso. Quanto menor o bloco, mais detalhado é o mapa, porém a chance de ocorrer erros é maior.

(a) (b)

Figura 4: Resultado com janelas de tamanhos diferentes

A Figura 4 (a) demonstra o resultado da estimação da disparidade utilizando uma janela de 7 pixels. A Figura 4 (b) demonstra o resultado da estimação de disparidade utilizando uma janela de 25 pixels. Podemos perceber que a Figura 4 (a) possui apesar de mais ruidosa, possui um detalhamento maior da disparidade. Já Figura 4 (b) apesar de mais suave, apresenta menos detalhes.

Vale ressaltar que no mapa de disparidade quanto maior o valor do pixel, ou seja, quanto mais branco o pixel na imagem, mais próximo ele está da câmera.

5.2.2 Compute()

Função utilizada para computar a disparidade do par estéreo analisado. Recebe como parâmetros o par estéreo e a matriz de saída da disparidade. A matriz de saída é computada em 16-bits, e portanto deve ser normalizada para 8-bits antes de ser exibida.

5.2.3 Normalize()

Função utilizada para normalizar a matriz de saída. Recebe como parâmetros a matriz de entrada, matriz de saída, os valores mínimos e máximos de saída, e o tipo da saída, nesse caso *unsigned integer 8-bits*.

6. Resultados

Como pode ser visualizado na Figura 5, os resultados obtidos com as imagens capturadas não foram satisfatórios. Um dos fatores considerados para esse resultado, é a qualidade ruim do sensor da câmera utilizada, resultando em imagens com baixa resolução e muito ruído.

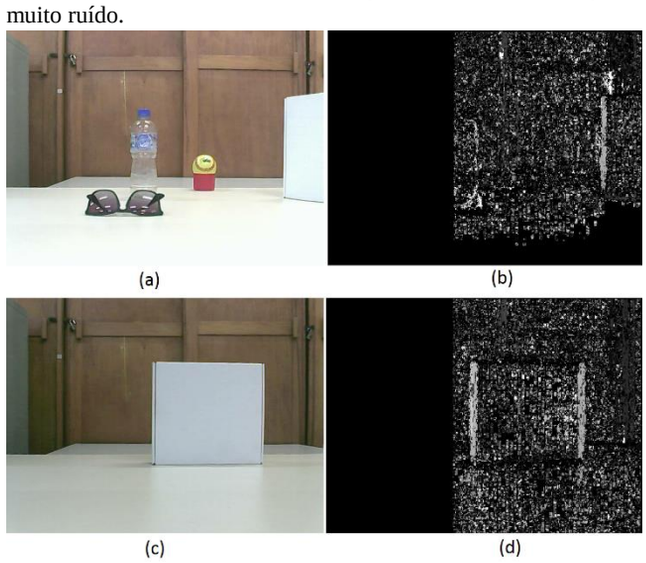

Figura 5: Resultados com as imagens capturadas

Para testar a efetividade do algoritmo desenvolvido, optou-se por utilizar imagens disponibilizadas por Nesic [11], as quais também fazem parte de um Sistema estéreo, e possuem uma qualidade melhor do que as obtidas pelas câmeras próprias.

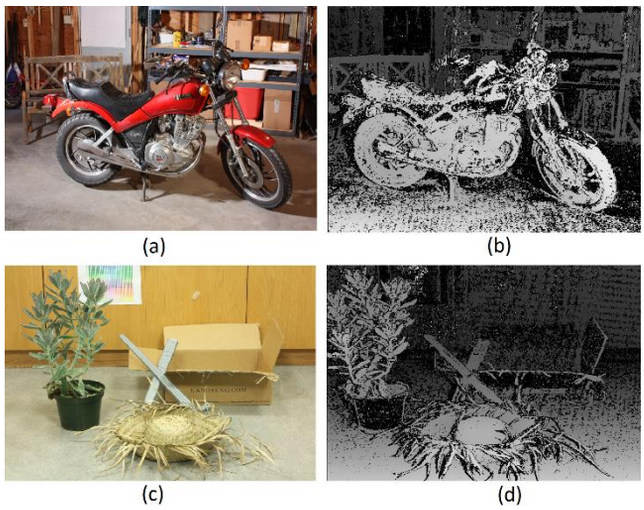

Figura 6: Resultados com imagens de Nesic

Como pode ser visualizado na Figura 6, os resultados obtidos com as imagens de Nesic foram muito melhores, apresentando um mapa de disparidade mais denso.

7. CONCLUSÕES E CONTINUAÇÃO DO TRABALHO

Como pode ser visualizado na Figura 6, apesar dos dois testes utilizarem o mesmo algoritmo com os mesmos parâmetros, os resultados obtidos com as imagens de Nesic foram muito melhores. Uma hipótese para essa diferença dos resultados, é a baixa qualidade das imagens obtidas.

Apesar do resultado obtido utilizando as imagens de Nesic ser melhor do que o obtido com as câmeras próprias, este ainda não é o ideal, portanto o algoritmo ainda deve ser melhorado.

Uma das propostas desse trabalho, é a obtenção do mapa de disparidade utilizando imagens próprias. Portanto um dos problemas a serem resolvidos é a baixa qualidade das imagens, através de câmeras melhores, porém utilizando o mesmo *middleware* aqui desenvolvido

8. REFERÊNCIAS

[1] Acharyya, A. et al. 2016. Depth Estimation From Focus and Disparity. In *IEEE International Conference on Image Processing (ICIP)* (Phoenix, USA, September 25 - 28, 2016).

[2] Forsyth, D., Ponce, J., Computer Vision, A Modern Approach, Pearson, Nova Jersey, 2011.

[3] Trucco, E., Verri, A., Introdutory Techiniques for 3-D Computer Vision, Prentice Hall, Nova Jersei, 1998.

[4] Lakshmanan, R. and Senthilnathan, R. 2016. Depth Map Based Reactive Planning to Aid in Navigation for Visually Challenged. In *IEEE International Conference on Engineering and Technology (ICETECH)* (Coimbatore, India, March 17 - 18, 2016)

[5] Fernandez, C.S. et al. 2015. Stereoscopy and Haptics Human Eye AR App. In *IEEE Games and Entertainment Media Conference (GEM)* (Toronto, Canada, October 14 - 16, 2015)

[6] Felchar, C. 2017. CtrlCam, Acesso em 23/08/2017, Disponível em < https://bitbucket.org/Cauef/ctrlcam>.

[7] OpenCV. 2014. Home, Acesso em 23/08/2017, Disponível em <http://opencv.org/>.

[8] Smart Eye Group. 2014. Home, Acesso em 23/08/2017, Disponível em <http://www.smarteyegroup.com/>.

[9] Kukelova, Z. et al. 2015.Efficient Solution to the Epipolar Geometry for Radially Distorted Cameras. In *IEEE International Conference on Computer Vision (ICCV)* (Santiago, Chile, December 13 - 16, 2015)

[10] Shete, P. Sarode, D. Bose, S. 2014.A Real-Time Stereo Rectification of High Definition Image Stream Using GPU. In *International Conference on Advances in Computing, Communications and Informatics (ICACCI)* (Nova Delhi, India, September 24 - 27, 2014)

[11] Nesic et al. 2014. Stereo datasets with ground truth, Acesso em 23/08/2017, Disponível em <http://vision.middlebury.edu/stereo/data/scenes2014/>.