Trabalho 1: Medições e rastreamento em vídeo estéreo

Myllena de Almeida Prado
myllena.almeidap@gmail.com
Roberta Costa Silva
robertacs42@gmail.com

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

Abstract

A visão estéreo na Visão Computacional é utilizada para a compreensão do mundo 3D a partir de duas imagens 2D, sendo possível estimar a profundidade e a distância real entre os elementos das imagens. Neste projeto, a partir da calibração das câmeras para obtenção dos parâmetros instrísecos, seguida da estimação da pose das câmeras através dos parâmetros extrínsecos, foram calculados os mapas de disparidade e profundidade das imagens que pertencem a dois vídeos adquiridos por um par de câmeras com relativa distância entre si. O objetivo final é o rastremento das coordenadas reais da trajetória de um objeto em movimento nos vídeos.

1

1 Introdução

A reconstrução 3D de imagens consiste na recuperação da percepção da dimensão espacial através de duas imagens. Na visão estéreo, médodo da visão computacional para entender o mundo 3D baseado entendimento da profundidade realizado pelos olhos, essas imagens são capturadas geralmente com um pequeno deslocamento horizontal, de modo a apresentarem diferentes pontos de vista de uma mesma cena [].

Entendendo a relação entre as duas imagens capturadas como a imagem pelo ponto de vista do olho esquerdo e a outra pelo olho direito, Trucco, no livro *Introductory techniques for 3-D computer vision*, define que a visão estéreo consiste na resolução de dois problemas: a correspondência entre os pontos das imagens, ou seja, a localização dos pontos que aparecem da imagem da esquerda na imagem da direta, e a reconstrução, que a partir do cálculo da disparidade deve interpretar dimensão de profundidade.

No Trabalho 1, analisamos 2 vídeos contendo correspodências gravados por câmeras distintas para rastrear pontos de 2D em 3D. A etapas para esse objetivo são: 1. Calibrar

[1]Todas as atividades de desenvolvimento foram realizadas em conjunto, assim como a escrita do relatório. Na implementação do código, ambas participaram no entendimento do problema e também na análise dos resultados. Para a escrita do relatório, os treschos escritos por uma foram revisados pela outra. Nenhuma atividade foi feita por completo e exclusivamente por uma pessoa. Agradecemos ao colega Alexandre, pela dica dos métodos da opencv para retificação das imagens e do tracker, ao colega Matheus Leal, pela dica de utilização do método que não utiliza calibração utilizado no requisito 3 e ao colega Welton, que apontou a necessidade de trocar as câmeras nos parâmetros dos métodos da opencv.

dos parâmetros relativos às câmeras; 2. Estimar a pose das câmeras a partir dos parâmetros relativos à orientação e posição da câmera em relação ao mundo; 3. Estimar a profundidade dos componentes do vídeo; 4. Rastrear pontos dos blocos de brinquedo empilhados em movimento.

2 Calibração dos intrínsecos

O primeiro requisito foi determinar os parâmetros intrínsecos das duas câmeras que foram utilizadas na obtenção de imagens do projeto. Esses parâmetros são os valores da distância focal da câmera e do ponto principal (origem das coordenadas no plano da imagem), definidos por Hartley e Zisserman no livro *Multiple View Geometry in Computer Vision* [2]. Os autores determinam a matriz de projeção dos pontos do mundo real em uma imagem de uma câmera pinhole como sendo $P = K[R|t]$. Nessa equação estão os parâmetros a serem determinados no requisito representados por K como a matriz de intrínsecos, e R e t são os extrinsecos.

2.1 Metodologia

No livro *Introductory techniques for 3-D computer vision* [3] é demostrado que para se determinar os parâmetros da matriz de projeção de uma câmera é necessário ter pontos conhecidos de uma mesma imagem no mundo 3D e na imagem 2D, pois a calibração se dá pela resolução da seguinte equação: $x = PX$, onde x são as coordenadas dos pontos na imagem, X as coordenadas dos pontos no mundo real e P é a matriz de projeção.

Na resolução dessa equação são necessários pelo menos 11 pontos correspondentes no mundo real e na imagem de calibração. Assim foram utilizadas várias imagens fornecidas com 48 pontos em cada uma como demostrado na figura 1. Para identificar os pontos na imagem foi utilizado o método *findChessboardCorners* da openCV, que retorna os pontos dos cantos dos retângulos como na imagem 1.

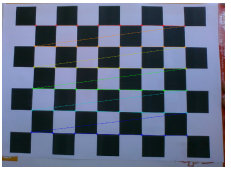

Figure 1: Pontos identificados no tabuleiro.

Após determinado o x e o X da equação, foi utilizado o método *calibrateCamera* da opencv para obter os parâmetros intrísecos. Esse método executa três etapas: primeiro estima os parâmetros intrisecos iniciais, depois estima a posição da câmera usando o método *solvePnP*, que calcula a pose de um objeto, os coeficientes de distorção, a projeção da imagem e a matriz da câmera. Por último o *solvePnP* executa um algoritmo de otimização para minimizar erros. O *calibrateCamera* retorna a matriz de intrínsecos e outros parâmetros como a matriz de extrínsecos.

2.2 Resultado

Ao executar a metodologia acima os resultados para as matrizes dos instrínsecos $K1$ e $K2$ das câmeras foram:

$$K1 = \begin{pmatrix} 1348 & 0 & 945 \\ 0 & 1351 & 534 \\ 0 & 0 & 1 \end{pmatrix} K2 = \begin{pmatrix} 953 & 0 & 627 \\ 0 & 954 & 344 \\ 0 & 0 & 1 \end{pmatrix}$$

As matrizes geradas foram satisfatórias um vez que seguem o formato da matriz intrínseca K da matriz de projeção, com valores para a distância focal no eixo x e y e para as coordenadas do ponto central no eixo x e y. Os valores entre as matrizes foram distintos por se tratarem de duas câmeras distintas e pela qualidade da imagem de calibração ser diferente entre elas, pois as imagens da câmera 1 estavam mais nítidas.

3 Estimativa da pose das câmeras

Para estimar a pose das câmeras no mundo real, referente ao Requisito 2, utilizamos a compreensão dos parâmetros extrínsecos de uma câmera.

3.1 Metodologia

Segundo Trucco [5], os parâmetros extrínsecos, através de uma transformação, definem a localização e orientação da câmera, *camera reference frame*, em relação a pontos conhecidos do mundo, *world reference frame*. A transformação citada pelo autor utiliza um vetor de translação com 3 elementos, t, que descreve as posições relativas das origens de dois frames de referência, e uma matriz 3x3 de rotação R, que fornece os eixos correspondentes entre dois frames.

Após capturarmos dois frames dos vídeos de entrada em que os 4 blocos de brinquedo aparecem, t e R das câmeras 1 e 2 foram obtidos com o método *solvePnP* do opencv. A matriz de instrísecos e o coeficiente de distorção da respectiva câmera são entradas do método. Outras duas entradas são o conjunto de coordenadas (X,Y,Z) das Figuras 2 e 3, sendo A a origem, e os pontos (x,y) correspondente nos *frames*, que são capturados na implementação através de clique duplo sobre a imagem, sendo a origem o ponto no canto superior direito.

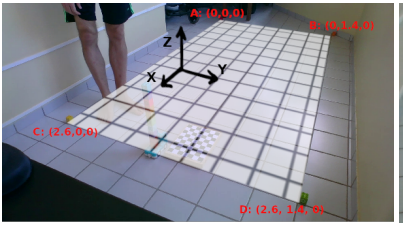

Figure 2: Câmera 1

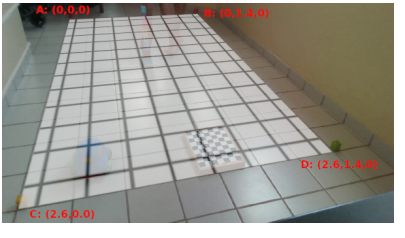

Figure 3: Câmera 2

A partir dos parâmetros extrínsecos t e R, o vetor $C = [Cx, Cy, Cz]$, que representa as coordenadas das câmera, foi calculado atráves da relação entre a matriz de rotaçao transposta e o vetor de translação [4]:

$$t = -R^T C$$

3.2 Resultado

As cooderdenadas calculadas para cada câmera $C1$ e $C2$ foram:

$$C1 = \begin{pmatrix} 3.51 \\ 1.35 \\ 0.99 \end{pmatrix} C2 = \begin{pmatrix} 3.97 \\ 0.23 \\ 1.09 \end{pmatrix}$$

Os valores obtidos são coerentes. Nota-se que os valores da coordenada Y estão no intervalo de 0.0 até 1.4, sendo a câmera 2 próxima a 0.0 e a câmera 1 próxima a 1.4. O valor de X para ambas as câmeras é coerente, sendo um pouco acima de 2.6. Como esperado, os valores da coordenada X para as duas câmeras são próximos, assim como os valores da coordenada Z, característica observável nas imagens.

4 Estimativa do mapa de profundidade relativo à câmera 1

Nessa etapa há dois requisitos: o mapa de disparidade e o mapa de profundidade. No artigo *An Active Multibaseline Stereo System With Real-Time Image Acquisition* [3] disparidade é definida como o deslocamento de um ponto correspondente entre duas imagens, portanto o mapa de disparidade possui o deslocamento de todos os pontos de uma imagem em relação a outra. Já a profundidade na imagem pode ser calculada em função dessa disparidade, da distânica focal e da distância *baseline*, fórmula demostrada também no artigo. Assim ,o requisito 3 foi divido em duas etapas, calcular o mapa de disparidade e posteriormente o mapa de profundidade.

4.1 Metodologia

A disparidade é calculada partir da correspondência entre pontos de duas imagens estéreo de uma mesma cena. Trucco [5] descreve o processo obter essa correspondência de um ponto do mundo real em duas imagens. Uma das alternativas apresentadas é a partir do cálculo da matriz fundamental que permite encontrar linhas epipolares entre as imagens e, após aplicar retificação nas imagens, é possível obter a correspondênica entre os pontos. A biblioteca openCV possui metódos para esse processo que envolve o cálculo da matriz fundamental, da retificação e da disparidade, métodos esses usados para calcular o mapa de disparidade.

Outro método apresentado no livro [5] utiliza uma matriz essencial ao invés da matriz fundamental, porém o método da openCV que determina essa matriz, o *stereoCalibrate*, não funcionou corretamente, assim foi escolhido o primeiro método para o mapa de disparidade.

Para o mapa de profundidade, é demostrado no livro de trucco [5] que a profundidade de um ponto se relaciona com a disparidade através da seguinte equação: $Z = \frac{fT}{d}$, sendo f a distância focal, T a distância *baseline* e d a disparidade. O *baseline* é a distância entre os dois centros das câmeras da visão estéreo e foi calculado através da distância euclidiana entre as coordenadas obtidas no requisito 2. Foi utilizado o método *reprojectImageTo3D* para gerar o mapa de profundidade a partir do mapa de disparidade.

A partir desse requisito foi necessário inverter as câmeras 1 e 2, ou seja, para esses requisitos a câmera 1 passou a ser a da esquerda e a câmera 2 a da direira, a fim de seguir o padrão dos parâmetros dos métodos do openCV. Além disso, foi necessário redimensionar as imagens para a resolução (640, 480).

4.2 Resultado

Para se determinar do mapa da disparidade foi obtida a matriz fundamental utilizando sete pontos correspondentes nas imagens e o método *findFundamentalMat* da openCV. Após isso as imagens foram retificadas através do método *stereoRectifyUncalibrated* e o método *warpPerspective* foi utilizado para gerar as imagens correspondentes 4 e 5.

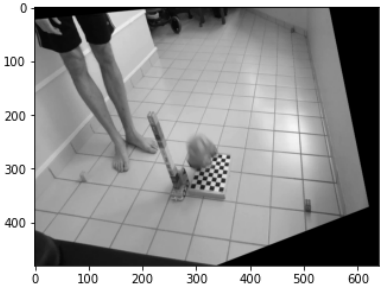

Figure 4: Câmera 1

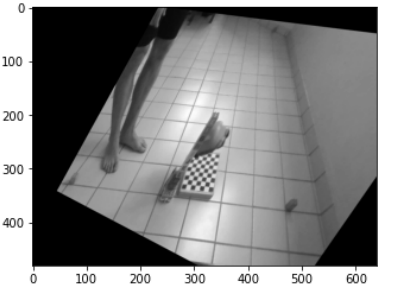

Figure 5: Câmera 2

As imagens 4 e 5 são satisfatórias e é possível perceber que os pontos correspondentes nas imagens estão na mesma linha horizontal como era esperado. Com as imagens retificadas foi calculada a disparidade através da classe *StereoBMcreate* e do *compute*, sendo o segundo um método da classe. A figura 6 demostra o mapa obtido, nele é possível perceber que quando mais próximo da borda inferior mais claro é o mapa, indicando assim uma maior disparidade, e quando mais próximo da borda superior mais cinza, indicando uma menor disparidade. Esse comportamento era esperado visto que pontos mais distantes da câmera apresentam uma dispariade menor.

Com a imagem do mapa de disparidade foi gerado o mapa de profundidade mostrado na figura 7. A imagem possui tons de azul mais claros nas regiões perto da bora inferior e tons mais fortes perto da borda superior, demonstrando assim que quando mais próximo da borda superior mais distante da câmera o ponto está e assim o mapa gerado foi satisfatório.

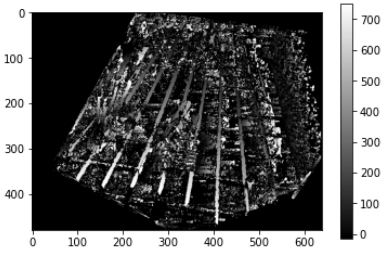

Figure 6: Disparidade

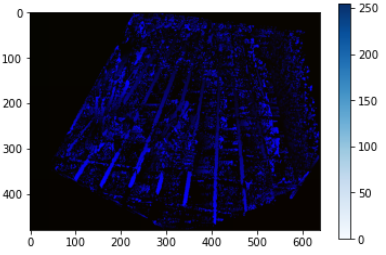

Figure 7: Profundidade

5 Rastreamento de 2 pontos em 3D

No último requisito do projeto foi proposto gerar gráficos das coordenadas da base de uma torre de carrinho que se movimenta em um carrinho de brinquedo em vídeos gravados pelo par de câmeras. O carrinho pode ser visto na imagem 4.

5.1 Metodologia

No livro *Multiple View Geometry in Computer Vision* [2] é demonstrado o metódo de triangulação que a partir de dois pontos correspondentes em duas imagens estéreo é possível estimar as coordenadas 3D desse ponto no mundo real. Nesse método é usado os pontos nas imagens e as matrizes de projeção das câmeras, assim foi utilizado os parâmetros obtidos no requisito 1 e 2 para gerar uma matriz de projeção de cada câmera. Para o cálculo da triangulação foi usado o método *triangulatePoints* da openCV e os pontos nas imagens foram obtidos através do rastreamento da base da torre junta ao carrinho.

Para obter as coordenadas em pixels da trajetória do carrinho, foi utilizado o *tracker* MOSSE, que a partir da seleção com o *mouse* de um objeto no primeiro frame do vídeo realiza o seu rastreamento. Foram utilizados os vídeos com alguns segundos do início cortados em que o carrinho está parado para otimização da execução.

5.2 Resultado

Os gráficos obtidos da trajetória do carrinho são mostrados nas figuras 8, 9 e 10. O sistema de coordenadas das imagens pode ser visto nas imagens 2 e 3 do requisito 2. A trajetória do eixo x e y foram satisfatórias e demonstram o percurso de ida e volta do carrinho. Já o eixo z deveria possuir um valor mais constante, porém esse mau comportamento era esperado uma vez que na calibração da câmera o eixo z não teve tanto peso quando os outros eixos.

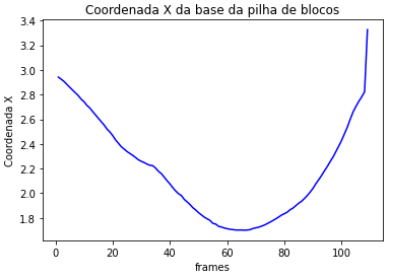

Figure 8: Eixo X

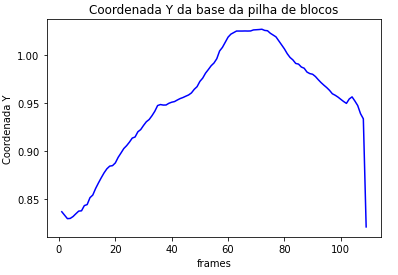

Figure 9: Eixo Y

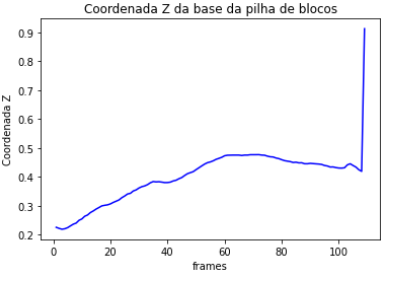

Figure 10: Eixo Z

6 Discussão e Conclusões

Todos os requisitos do projeto foram realizados de forma satisfatória. A calibração da câmera e a estimação de sua pose no mundo real geraram valores dentro de uma faixa esperada. O mapa de disparidade e profundidade apresentaram valores que condizem com a realidade e os gráficos da trajetória do carrinho demonstraram o caminho de ida e volta do seu percurso. Atribuímos o leve erro para os valores reais esperados à diferença de resolução entre os vídeos, que não foram redimensionados para o requisito 4 por limitação do *tracker* utilizado, que não funcionou quando o redimensionamento foi realizado, e à taxa de captura diferente. Essa última afeta na captura de imagens que precisam ser no mesmo instante. Para realização do trabalho e a utilização de todos os métodos aqui citados do openCV seguimos a documentação da biblioteca [1].

Um problema encontrado durante o desenvolvimento foi que na execução do requisito 3 houve a anomalia do código rodar corretamente em apenas um dos computadores que estava sendo desenvolvido o projeto, no outro computador os mapas de profundidade e disapridade não tiveram o mesmo formato, porém neste relatório foram apresentados os resultados do código que executou adequadamente.

References

[1] *OpenCV Open Source Computer Vision 3.4.4*. URL https://docs.opencv.org/3.4.4/index.html.

[2] R. I. Hartley and A. Zisserman. *Multiple View Geometry in Computer Vision*. Cambridge University Press, ISBN: 0521623049, 2000.

[3] Sing Bing Kang, Jon Webb, and C. Zitnick. An active multibaseline stereo system with real-time image acquisition. 11 1999.

[4] Kyle Simek. *Dissecting the Camera Matrix, Part 2: The Extrinsic Matrix*, 2012 (acessado em Outubro 25, 2020). URL http://ksimek.github.io/2012/08/22/extrinsic/.

[5] E. Trucco and A. Verri. *Introductory techniques for 3-D computer vision*. Prentice Hall PTR, Upper Saddle River, 1998.