Comparative Analysis of Stereo Matching Methods for Disparity Map Reconstruction

2022

1 Image rectification

To reconstruct disparity maps, for each stereo image pair, the two images should be rectified so that all image motion is purely horizontal. Image rectification is not required in this project because the two images are already rectified for each stereo image pair. My code includes the image rectification function in case that new stereo image pair inputs are not rectified. Specifically, the image rectification function includes the following steps using OpenCV functions:

- (1) Feature extraction with scale-invariant feature transform (SIFT) (Lowe, 2004): I use the *xfeatures2d.SIFT-create()* to create a SIFT object to detect keypoints and descriptors in both left and right images.
- (2) Feature matching with brute force matcher: I use the brute force matching method *BF-Matcher()* applied on the descriptors found in the left and right images.
- (3) Match filtering using the ratio test: I use the ratio test to filter out ambiguous matches and keep only the most likely matches between the left and right images.
- (4) Fundamental matrix estimation: I use *findFundamentalMat()* to estimate the fundamental matrix using the matched keypoints between the left and right images.
- (5) Image rectification (Fusiello et al., 2000): I use *stereoRectifyUncalibrated()* to compute the uncalibrated rectification transformation for the left and right images. Then I use *warpPerspective()* to apply the perspective transformations to both images.
- (6) As this project does not require visualization of the epipolar geometry and it does not require epilines, the *computeCorrespondEpilines()* function that is used to compute the epilines corresponding to points in the left and right images is not included.

2 Method 1. Window-based Matching

I implement the window-based matching (block matching) method (Scharstein and Szeliski, 2002) to compute the disparity maps. Window-based matching is a method used to find the matching points between the left and right images in stereo vision tasks. Window-based methods compare a small window of pixels in the left image with corresponding windows in the right image, and compute the

disparity based on the similarity between these windows. Specifically, the window-based method can be described as follows:

- (1) The algorithm first selects a small square window around a point in the left image. The size of the window can be variant, such as $4*4$, $9*9$, or $16*16$ pixels, depending on the image inputs and the stereo vision tasks.
- (2) The algorithm searches for a similar window in the right image. It usually searches along the same row, because it is assumed that the image pair has been rectified, and the matching point is on the same horizontal line.
- (3) The algorithm compares the windows and find the best match using different methods based on a metric. In the project, I use the sum of absolute differences (SAD). Other candidate metrics include the sum of squared differences (SSD) and normalized cross-correlation (NCC).
- (4) The algorithm repeats this process for many points in the left image, creating a map of how far the matching points are between the left and right images. This map is the disparity map.

There are some potential improvements of my implementation that could be tested in experiments:

- Window size: The window size is the size of the window of pixels that is compared between the left and right images to compute the disparity value. I use a window size of 8 in the matching algorithm. The window size of 8 might be too small, leading to more noise and less accurate disparity values. Large values could potentially improve the accuracy of the disparity map, but it will increase the computation time.
- Maximum disparity: The maximum disparity in the maximum number of pixels that can separate the corresponding pixels in the left and right images. I use a maximum disparity value of 256 in the matching algorithm. This value which may be too small in some images or stereo vision tasks. The actual disparity value in the scene is not known. If it is much larger than this value, then the algorithm will not be able to capture them. Increasing the maximum disparity value can cover a wider range of possible disparities.
- Matching cost function: I use the sum of absolute differences (SAD) as the matching cost function. This SAD cost function might not work well for all types of stereo image pairs. Other candidate matching cost functions, such as the sum of squared differences (SSD) or normalized cross-correlation (NCC) can be used.
- Parallel processing: My window-based matching method costs about one hour to process one stereo image pair which suggests its inefficiency in image processing and incapability for real-time applications. Parallel processing can be used to speed up the window-based matching by dividing the workload and processing it in parallel using multiple processing units. It can significantly reduce computation time and is useful in real-time applications.

3 Method 2. Stereo Semi-Global Block Matching (SGBM) in OpenCV

Window-based matching method is easy to implement, but my window-based matching method costs about one hour to process one stereo image pair. This suggests its inefficiency in image processing

and incapability for real-time applications. In terms of disparity map generation quality, window-based matching methods can be sensitive to noises and occlusions. Advanced algorithms can be used and compared with my window-based matching method.

In the code, I use Stereo Semi-Global Block Matching (SGBM) (Hirschmuller, 2005, 2007) (*StereoSGBM* in OpenCV) as the second method to compute disparity maps. SGBM with an appropriate set of parameters can obtain better results. It leads to higher quality of the reconstructed disparity maps and higher values of peak signal-to-noise ratio (PSNR). SGBM is a more advanced stereo matching algorithm that aims to compute disparity maps from stereo images. This method considers not only the matching cost in the current area, but also the overall consistency of the entire disparity map. The process involves combining matching costs from different directions and choosing the most appropriate disparity based on the combined costs.

4 Method 3. Pyramid Stereo Matching Network (PSMNet)

I use pre-trained Pyramid Stereo Matching Network (PSMNet) (Chang and Chen, 2018) as the deep learning-based stereo matching method to generate the disparity maps. This approach uses a special module called a "spatial pyramid pooling module" and "stacked dilated convolutions" to capture image features. The design of this network enables it to deal with images with different levels of complexity and provide accurate depth estimates. PSMNet has demonstrated state-of-the-art performance on popular benchmarks such as the Scene Flow and KITTI datasets.

Specifically, the PSMNet model in the code is pre-trained on the Scene Flow dataset (Mayer et al., 2016). This is a dataset that was created for training and testing stereo matching and optical flow algorithms. The dataset includes about 40,000 samples, each with a pair of stereo images, disparity maps, and optical flow fields. These samples were generated using a virtual 3D environment, which makes it possible to produce highly accurate ground-truth labels. The Scene Flow dataset is widely used as a benchmark to evaluate and compare the performance of different stereo matching and optical flow methods.

Because the pre-trained PSMNet model requires much GPU memory usage in inference and I only have limited GPU memory resources, I have to resize the stereo image pairs to reduce the memory usage. Specifically, for the three stereo image pairs, both the stereo image pairs and their corresponding ground-truth disparity map are reduced to 70% of their original sizes.

5 Reconstructed Disparity Map

Disparity maps are the pixel-wise difference in the horizontal positions of corresponding points in the left and right images. Figure 1-3 compares the reconstructed disparity map generated by three stereo matching methods with the ground-truth disparity map for the three stereo image pairs: (1) Art, (2) Dolls, and (3) Reindeer.

In summary, PSMNet obtains the most visually similar disparity map to the ground-truth map, followed by the OpenCV Stereo SGBM method. The reconstructed disparity map of my window-based matching suffers many clutters and noises. As mentioned, the window-based matching methods are generally easy to implement, but they can be sensitive to noise and occlusions. Its performance might be improved by adjusting the two parameters, window size and maximum disparity.

6 Peak Signal-to-noise Ratio (PSNR)

The peak signal-to-noise ratio (PSNR) is a measure of the amount of noise in a signal compared to the maximum power of that signal. It is often used to evaluate how accurately an image or video can be reconstructed after lossy compression. The PSNR provides an estimate of how well the reconstructed signal matches the original signal, as perceived by humans. Table 1 compares PSNR values using the reconstructed disparity map against the ground-truth disparity map in three stereo matching methods for the three stereo image pairs: (1) Art, (2) Dolls, and (3) Reindeer.

In summary, OpenCV SGBM and PSMNet have comparable performance in terms of the PSNR values, while my implemention of window-based matching method has lower PSNR values. When PSMNet method is used, both stereo image pairs and their corresponding ground-truth disparity map are resized to 70%, and the resize might influence its PSNR values.

Table 1: Peak-SNR values for different stereo matching methods

Method	Art	Dolls	Reindeer
My Implementation	12.41	12.15	10.84
OpenCV SGBM	12.67	16.15	15.58
PSMNet (resized 70%)	15.79	15.26	15.95

References

Chang, J.-R. and Chen, Y.-S. (2018). Pyramid stereo matching network. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5410–5418.

Fusiello, A., Trucco, E., and Verri, A. (2000). A compact algorithm for rectification of stereo pairs. *Machine Vision and Applications*, 12:16–22.

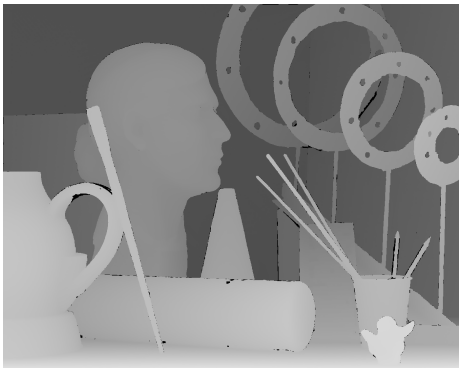
Hirschmuller, H. (2005). Accurate and efficient stereo processing by semi-global matching and mutual information. In *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*, volume 2, pages 807–814. IEEE.

Hirschmuller, H. (2007). Stereo processing by semiglobal matching and mutual information. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 30(2):328–341.

Lowe, D. G. (2004). Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision*, 60:91–110.

Mayer, N., Ilg, E., Hausser, P., Fischer, P., Cremers, D., Dosovitskiy, A., and Brox, T. (2016). A large dataset to train convolutional networks for disparity, optical flow, and scene flow estimation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4040–4048.

Scharstein, D. and Szeliski, R. (2002). A taxonomy and evaluation of dense two-frame stereo correspondence algorithms. *International Journal of Computer Vision*, 47:7–42.



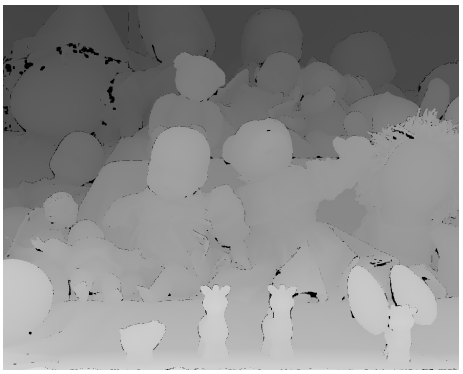
(a) Ground-truth

(b) My window-based matching

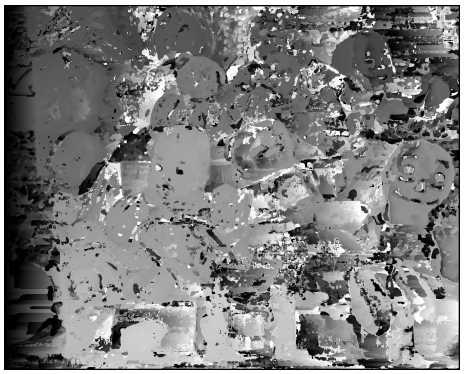
(c) OpenCV Stereo SGBM

(d) Pyramid Stereo Matching Network

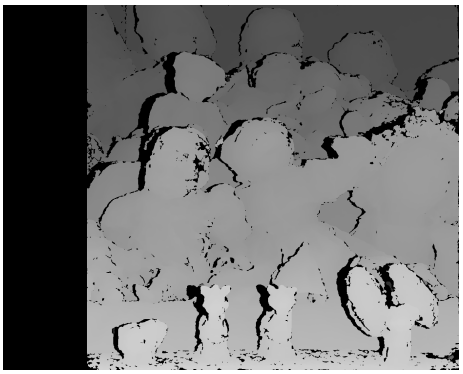
Figure 1: Reconstructed disparity map vs. ground-truth disparity map, stereo image pair "Art".



(a) Ground-truth



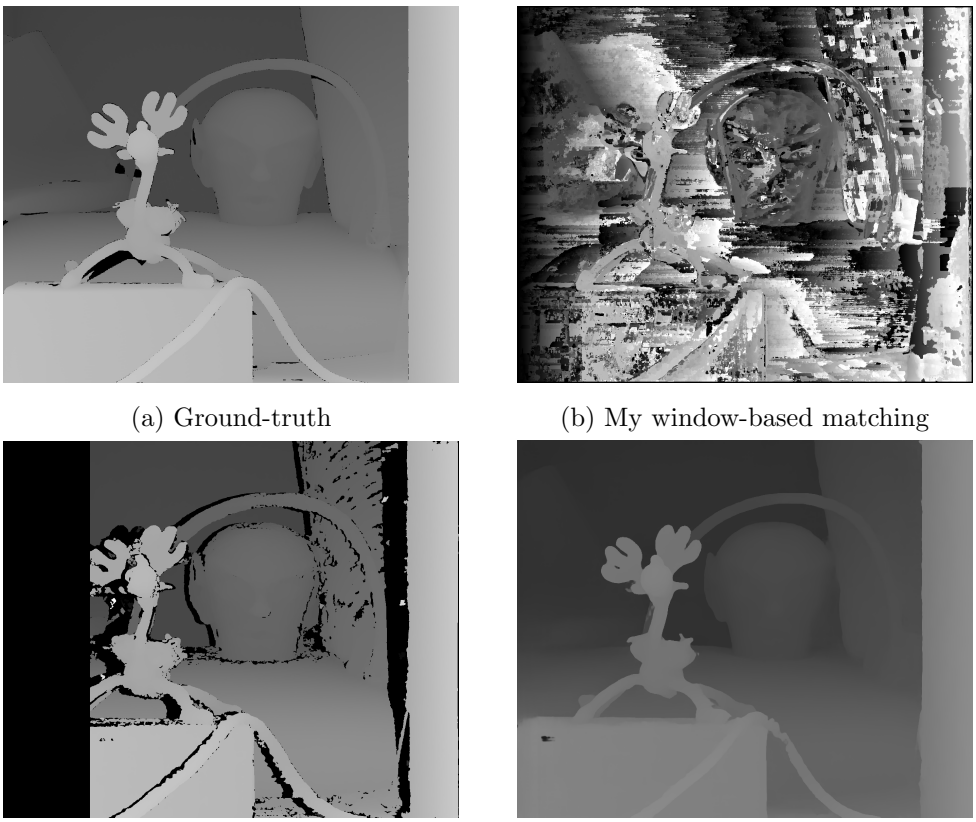
(b) My window-based matching



(c) OpenCV Stereo SGBM

(d) Pyramid Stereo Matching Network

Figure 2: Reconstructed disparity map vs. ground-truth disparity map, stereo image pair "Dolls".



(a) Ground-truth

(b) My window-based matching

(c) OpenCV Stereo SGBM

(d) Pyramid Stereo Matching Network

Figure 3: Reconstructed disparity map vs. ground-truth disparity map, stereo image pair “Reindeer”.