CS4186 Homework 2: Stereo Disparity Estimation

Assan KOZHIN
56279462
akozhin2-c@my.cityu.edu.hk

I. Introduction

I intend to start with the StereoSGBM implementation in OpenCV as a baseline in order to address the issue of stereo disparity matching. The Pyramid Stereo Matching Network (PSMNet), a deep learning-based strategy that has demonstrated promising outcomes in stereo matching tests, will also be covered in this report. I'll also experiment with various post-processing techniques to improve the disparity map and boost the PSNR score.

In this particular problem, the two images are rectified, which simplifies the process by reducing the problem to a one-dimensional horizontal search for corresponding pixels in the two images.

Throughout this report, I will present my experiments using the Art images and evaluate the performance of each method based on the average PSNR scores of all three images, including Art, Dolls, and Reindeer.

II. Stereo SGBM

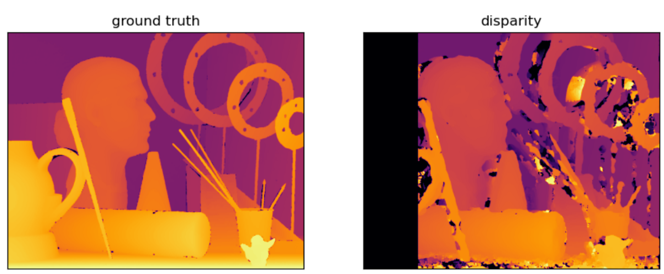

Figure 1: Results of Stereo SGBM from OpenCV. PSNR: 10.94

I utilized the StereoSGBM approach, which calculates a matching cost for each pixel in the left and right images and assigns the disparity value that results in the minimum cost to the pixel in the left image. To generate a dense disparity map, StereoSGBM employs a semi-global approach that aggregates the matching cost over multiple paths.

However, after generating the disparity map using StereoSGBM, I noticed some issues. There were **vertical black bars** on the left side of the map, which occurred in areas where there was no corresponding point in the other image, resulting in an invalid disparity value. Additionally, the **disparity map was noisy** with black spots due to textureless regions where there were no distinctive visual features, making it challenging for StereoSGBM to find corresponding pixels in the left image.

To address these issues, I conducted experiments and set a threshold for intensity values to be classified as "unknown" and handled by post-processing methods. Based on my analysis, intensity values lower than 70 were considered unknown. This approach helped to mitigate the vertical black bars and noise in the disparity map.

III. Post Processing Methods

A. *Inpainting using Navier Stokes Equation*

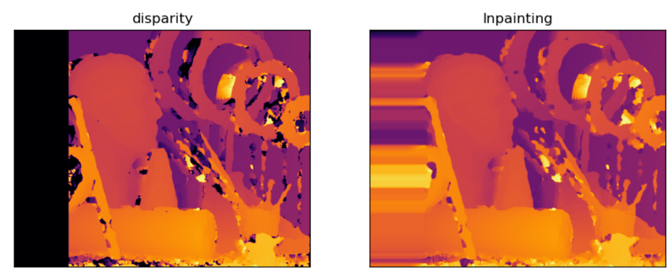

Figure 2: Results before and after applying of Navier Strokes Equation inpainting. PSNR: 17.59

To fix the issue of vertical black bars in images, we can use post-processing techniques like inpainting to fill in the missing areas. One such technique is Navier-Stokes inpainting, which uses fluid flow modeling to fill in the gaps and improve the accuracy and completeness of disparity maps. I tried this method and found that it **increased the PSNR score by 6.65 points**. However, there were some unwanted artifacts that appeared in the image. Specifically, the leftmost column of pixels was pulled to the left side of the image, causing some distortion.

B. *Mean Filter*

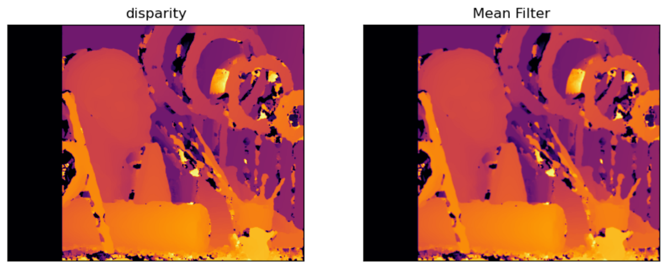

Figure 3: Results before and after applying of Mean Filter. PSNR: 11.13

I used mean filtering to reduce the impact of noisy pixels on the overall disparity map output. This technique works by replacing each pixel's value with the average value of its neighboring pixels, smoothing out the image and effectively removing noise from the disparity map. After applying mean filtering, I found that the PSNR score **improved slightly by about 0.44**.

C. *Combination of Navier-Stokes Equation and Mean Filter*

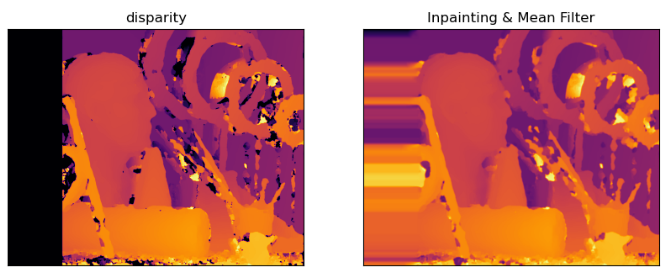

Figure 4: Results before and after applying this post-processing. PSNR: 17.65

By using a combination of Navier-Stokes inpainting and mean filtering, I was able to improve the disparity map output. I first employed inpainting to fill in larger areas of unknown disparities, and then applied mean filtering to eliminate any remaining noise. This combined approach produced a higher PSNR score compared to using the two post-processing techniques separately.

After trying out different approaches, the combined method of using StereoSGBM with Navier-Stokes inpainting and mean filtering yielded the best results. Specifically, this approach achieved an **average PSNR score of 19.76**, which is the highest compared to other techniques tested.

IV. Pyramid Stereo Matching Network

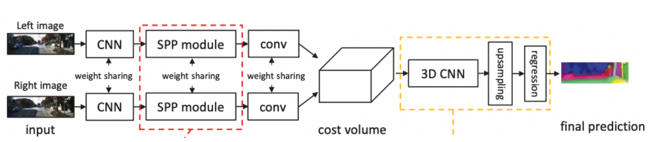

Figure 5: PSMNet architecture.

The PSMNet architecture consists of three main components: the feature extraction network, the cost volume computation, and the disparity computation. The feature extraction network is a convolutional neural network that extracts feature maps from the left and right input images. The cost volume computation involves computing the matching cost between the left and right feature maps at different disparities. This cost volume is then fed into a 3D convolutional network to obtain a set of feature maps which are then passed through a soft-argmin operation to generate the disparity map [1].

The soft-argmin operation is a key feature of PSMNet that allows for sub-pixel accuracy in the disparity predictions. It computes the weighted average of the cost volume along the disparity dimension, where the weights are the softmax of negative matching costs. The resulting disparity map is then upsampled to the original image resolution using a learned interpolation function. This multi-scale approach allows the network to capture both fine and coarse details in the image [1].

The PSMNet's Github repository[2] provides a pre-trained model that was trained on the Scene Flow dataset. To fine-tune the model on the KITTI 2015 dataset, I followed the instructions provided in the repository and finetuned the model. It took me about 250 epochs, but I decided to use checkpoint at 248 since it had worse results on some images. B The training process took around 18 hours to complete on an NVIDIA GTX 970 using batch size of 1.

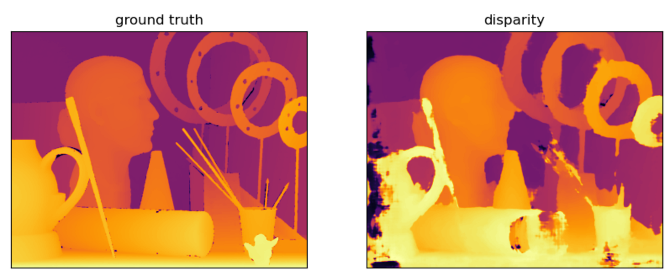

Figure 6: PSMNet results compared with ground Truth. PSNR: 18.81

PSMNet outperforms StereoSGBM in terms of **average PSNR score of 20.05** without any additional post-processing. However, there were still some unknown intensities in the disparity map which could be reduced by fine-tuning the model with more data. Unfortunately, due to time constraints, I did not explore this option. Instead, I experimented with several post-processing techniques to improve the performance and fill in the unknown disparities.

V. Post-processing methods

Please note that images in report were generated using wrong sequences of normalization steps. I fixed that bug in the final submission and final scores indicate improved version even though improvement is very small of 0.0001.

A. Filling in missing values from StereoSGBM

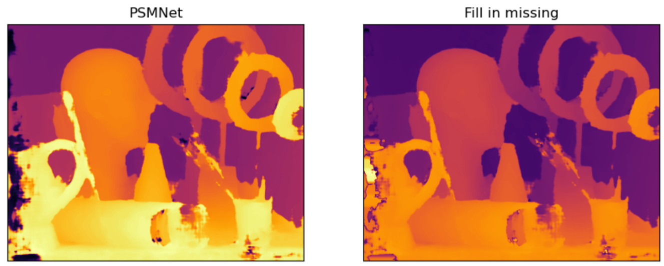

Figure 7: Results before and after filling in values from StereoSGBM. PSNR: 21.01

In this approach, we replaced the unknown intensity values in PSMNet with values from the StereoSGBM implementation, **resulting in a 2.2 point increase in the PSNR score**.

B. Filling in missing values from StereoSGBM + Mean Filter

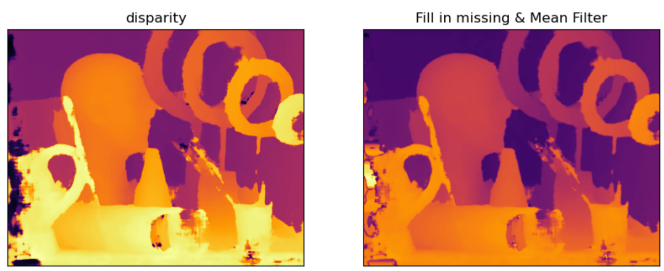

Figure 8: Comparison of using StereoSGBM with Mean filter. PSNR: 21.19.

In this method, we used mean filtering to remove noise from the disparity map produced by method 1. This led to a **slightly improved PSNR score**.

VI. Combination of 2 approaches: Average between PSMNet and StereoSGBM

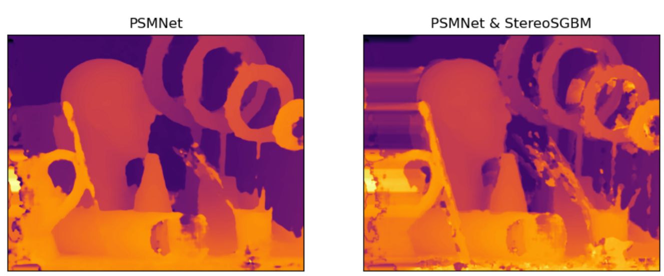

Figure 9: Averaging results from PSMNet and StereoSGBM. PSNR: 20.53

When we averaged the disparity maps generated by StereoSGBM and PSMNet, the performance was worse than using inpainting with mean filter. Additionally, the artifacts present in inpainted areas of StereoSGBM map were also present in this approach.

After experimenting with different post-processing methods, we found that filling in missing values from StereoSGBM + mean filtering produced the best results. The **average PSNR score achieved with this method was 21.65**. Therefore, we will use method 2 as our preferred approach for post-processing the disparity map.

VII. Conclusion

After conducting various experiments, I discovered that the accuracy and completeness of disparity maps in Stereo SGBM could be improved by utilizing post-processing techniques such as mean filtering and inpainting. Combining these methods produced the most optimal results.

However, PSMNet with post-processing was the most effective approach overall, resulting in an **average PSNR score of 21.65**, surpassing Stereo SGBM with post-processing. Despite this, additional fine-tuning and experimentation may be necessary to achieve even better outcomes.

If given more time, I am interested in exploring recent papers such as "End-to-End Learning of Geometry and Context for Deep Stereo Regression," "StereoNet: Guided Hierarchical Refinement for Real-Time Edge-Aware Stereo," and "Adaptive Cost Aggregation Network for Stereo Matching." These papers introduce new techniques for stereo disparity matching, such as learning geometry and context for deep stereo regression and guided hierarchical refinement for real-time edge-aware stereo. I believe that investigating these methods could provide valuable insights into improving stereo matching and disparity estimation.

References

[1] J.-R. C. and Yong-Sheng Chen, "Pyramid stereo matching network," *Corr*, 2018. [Online]. Available: http://arxiv.org/abs/1803.08669

[2] J.-R. Chang, "Psmnet: pyramid stereo matching network," GitHub, 2018. (\url{https://github.com/JiaRenChang/PSMNet})