Contents

Name: Fung King Shun (56659723)

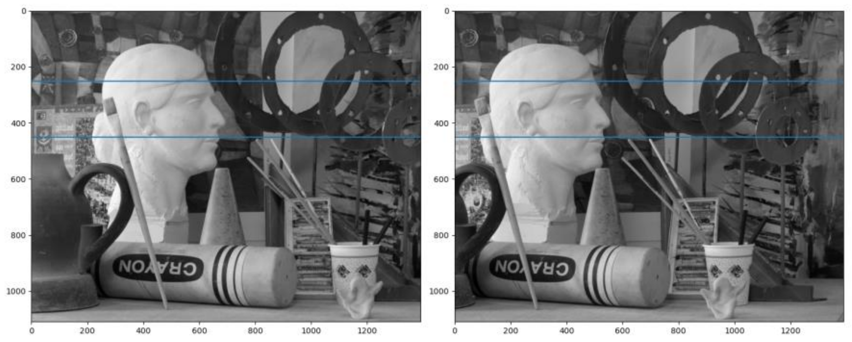
In this report, I will explain what method is used to solve the disparity maps problem and how to implement the solution in coding. Here I choose to use OpenCV SGBM algorithm and downscaled views with post-filtering (Weighted Least Squares filter) for disparity maps.

In addition, since it is needed to detect 3 images containing "Art, Dolls, Reindeer". In order to get the best PSNR value, each of the 3 images uses different parameters to extract the optimized disparity maps.

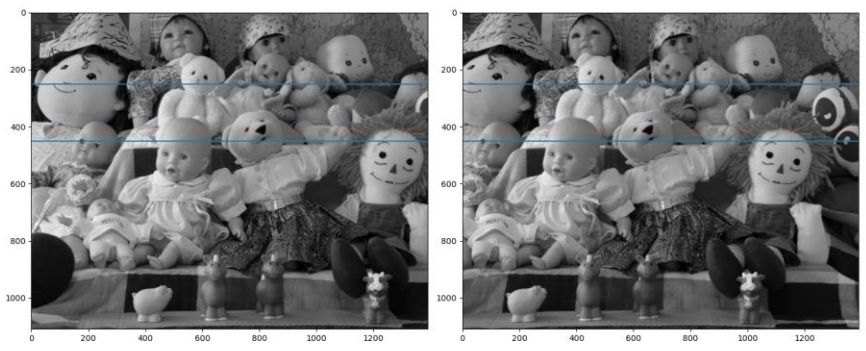
The following are the details steps:

Step 1: Rectification of stereo image

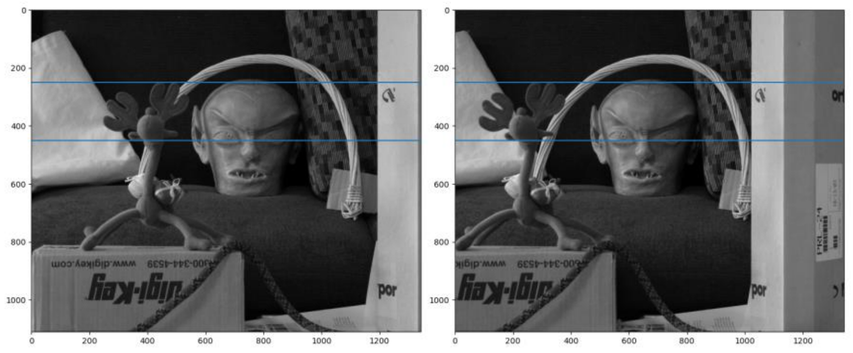
For common disparity maps, the first step is to rectify the stereo images to simplify the problem of finding matching points between the left image and the right image. But for our task, the given data images are already rectified. To check if it has been rectified, I had written a Python to check the result. You can run (.\PSNR_Assignment2\PSNR_Python\preprocess.py). Here are 3 images showing the result:



Rectified image of "Art"



Rectified image of “Dolls”



Rectified image of “Reindeer”

As you can see, all of the corresponding left and right images are purely horizontal by drawing a line.

Step 2: Grayscale images conversion

Before applying the SGBM algorithm, I first transfer the color image to grayscale. You can run (.\PSNR_Assignment2\PSNR_Python\opencv.py). Because grayscale images contain only one channel of information, which makes it easier to calculate the disparity between corresponding pixels in the two images. Additionally, grayscale images require fewer computational resources compared to color images, making them more efficient for processing large amounts of data.

Step 3: Stereo SGBM algorithm

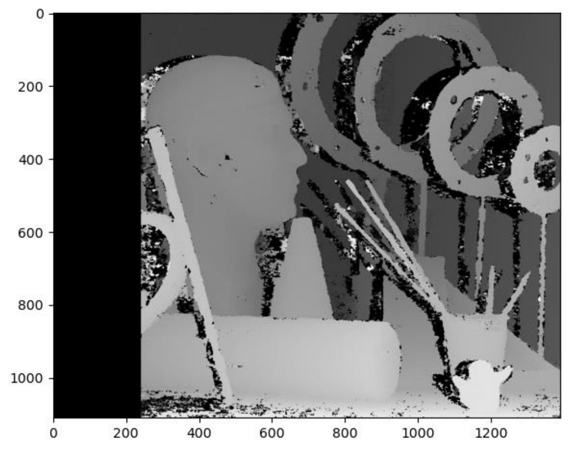
Stereo SGBM (Semi-Global Block Matching) algorithm is a computer vision algorithm used for stereo matching. It is a variant of the block-matching algorithm that uses a cost function to find the disparity between two images taken from different viewpoints. The algorithm works by dividing the image into small blocks and comparing the corresponding pixels in each block. It then calculates the disparity between each pair of pixels and assigns a cost to each disparity value. The algorithm then finds the disparity value with the lowest cost for each pixel, which represents the best match between the two images.

In OpenCV, I used the following parameters for the StereoSGBM object:

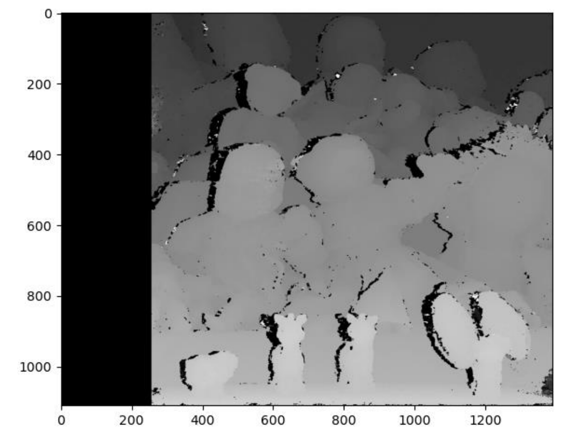
minDisparity	Minimum possible disparity value.
numDisparities	Maximum disparity minus minimum disparity.
blockSize	Matched block size.
uniquenessRatio	Margin in percentage by which the best (minimum) computed cost function value should "win" the second best value to consider the found match correct.
speckleWindowSize	Maximum size of smooth disparity regions to consider their noise speckles and invalidate.
speckleRange	Maximum disparity variation within each connected component.
disp12MaxDiff	Maximum allowed difference (in integer pixel units) in the left-right disparity check.
P1	The first parameter controls the disparity smoothness.
P2	The second parameter controls the

	disparity smoothness.

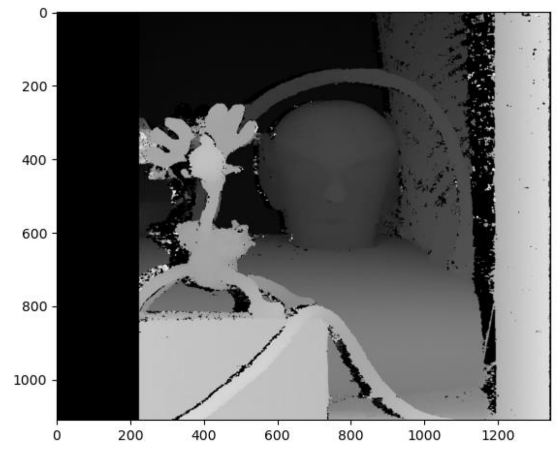
Below is the output image:



SGBM image of “Art”



SGBM image of “Dolls”

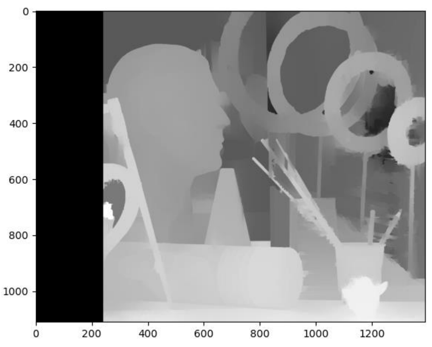


SGBM image of “Reindeer”

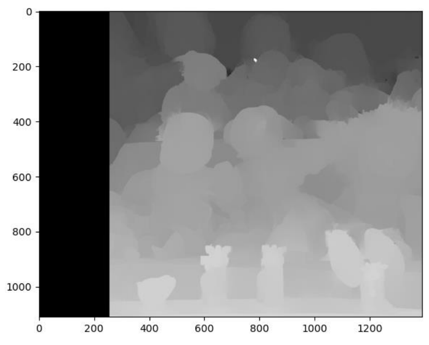
Step 4: Weighted Least Squares filter

Since only applied SGBM for disparity maps, the image is not smooth. So, I use the WLS filter to minimize the sum of the squared errors between the observed data and the predicted values. In WLS, each observation is assigned a weight based on its reliability or importance. The weights are used to adjust the contribution of each observation to the overall error function, with more weight given to more reliable observations. This results in a filter that is better able to handle outliers and noisy data.

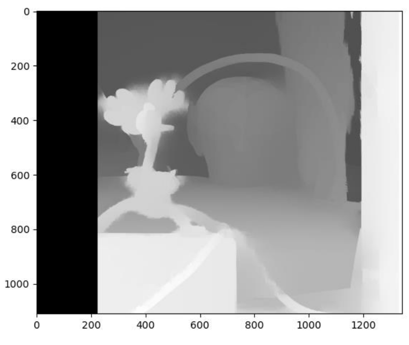
Below is the final output image:



WLS filter image of “Art”



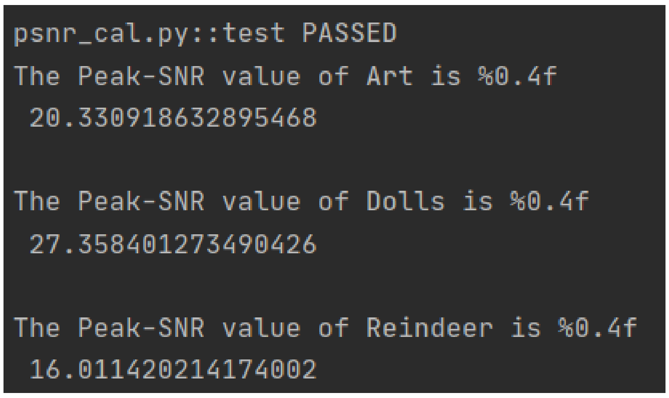
WLS filter image of “Dolls”



WLS filter image of “Reindeer”

Step 5: PSNR calculation

Finally, calculate the PSNR value on each image. You can run (.\PSNR_Assignment2\PSNR_Python\psnr_cal.py). Below shows the result:



```
psnr_cal.py::test PASSED
The Peak-SNR value of Art is %0.4f
 20.330918632895468

The Peak-SNR value of Dolls is %0.4f
 27.358401273490426

The Peak-SNR value of Reindeer is %0.4f
 16.011420214174002
```

Summary and future improvements

The SGBM algorithm is a popular method for generating disparity maps from stereo images. It works by comparing the pixel intensities of corresponding points in the left and right images and calculating the disparity between them. However, SGBM can be computationally expensive and may not produce accurate results in all situations.

To address these issues, downscaled views with post-filtering can be used to improve the accuracy and speed of SGBM. This involves creating smaller versions of the stereo images and applying a Weighted Least Squares filter to smooth out noise and inconsistencies in the disparity map.

Although from the results, this method gets a “good” disparity map. But there is still a gap compared with Ground-truth.

Therefore, we may consider using deep learning techniques to further improve the accuracy of disparity maps. For example, convolutional neural networks (CNNs) can be trained on large datasets of stereo images to learn how to generate more accurate disparity maps. But in this method, we need a large dataset and time for study.

Overall, using SGBM with downscaled views and post-filtering is a useful technique for generating disparity maps, but there is still room for improvement through the use of deep learning methods.