CSE 573: Computer Vision and Image Processing

FINAL PROJECT REPORT

Disparity for Stereo Vision – Block Matching and Dynamic Programming

GROUP NUMBER 9

ANVITA UPADHYAY	ISHA GUPTA
50147506	50208184

Date of Submission: 16th December 2016

INTRODUCTION

In this project, we will be using the given rectified stereo images to create a disparity map using two different methods.

Using rectified images has given us the advantage of searching for similarity only in 1-Dimension

In the first method we will be using basic block matching to compute the distance in pixels between the location of a feature in one image and its location in the other image.

In the second technique we will estimate the disparity map using maximum likelihood stereo algorithm using dynamic programming according to the provided reference paper

Following techniques shall be used for implementation:

- **Block matching technique:** It involves using 3x3 and 9x9 blocks to calculate sum of squared differences (SSD) between intensities of the corresponding pixels in the two images. Those blocks with the minimum SSD shall be selected which measure the similarity between the two image patches. An offset is then calculated from the image chosen as reference and that is the disparity value.

- **Search Range and Direction:** We need to specify how far and in which direction (either left or right or both) do we move the blocks in the second image after taking a reference patch from the first image. This will be done by visual inspection and testing many images in order to get a generic range and direction. Also, we will be searching only horizontally as the images are rectified.

- **Dynamic Programming:** We will create the disparity maps using dynamic programming for implementing the Maximum Likelihood Stereo Algorithm and proceed with the MSE.

- **Mean Squared Error:** Mean squared error will be calculated for both the methods by comparing the generated disparity maps with the provided ground truth disparity map as reference

APPROACH

BLOCK MATCHING TECHNIQUE:

Initially the code was developed for calculating disparity between left image with respect to right image and tested with a 3x3 block.
Based on the MSE values obtained using the ground truth, different values of shiftRange were tested to obtain a low MSE

Once an optimum disparity map was attained, the code was then tested for Right (view2) with respect to Left image (view1)

Eventually, the code was tested with a 9x9 block and testing was performed and further updates were made to shiftRange to get a low MSE

DYNAMIC PROGRAMMING TECHNIQUE:

Code was developed using the pseudo code provided in the reference paper.
Further reading of the paper and online reading material on stereo imaging and maximum likelihood gave a better understanding of the theory involved with the pseudocode.

The maximum likelihood stereo algorithm assumes that any two corresponding features are normally distributed about their true value. This leads to a local matching cost that is the weighted squared error between the features.

The code was tested for different values of Occlusion starting from the value 20.
The following MSE was obtained for different values of occlusion and the optimum was chosen.

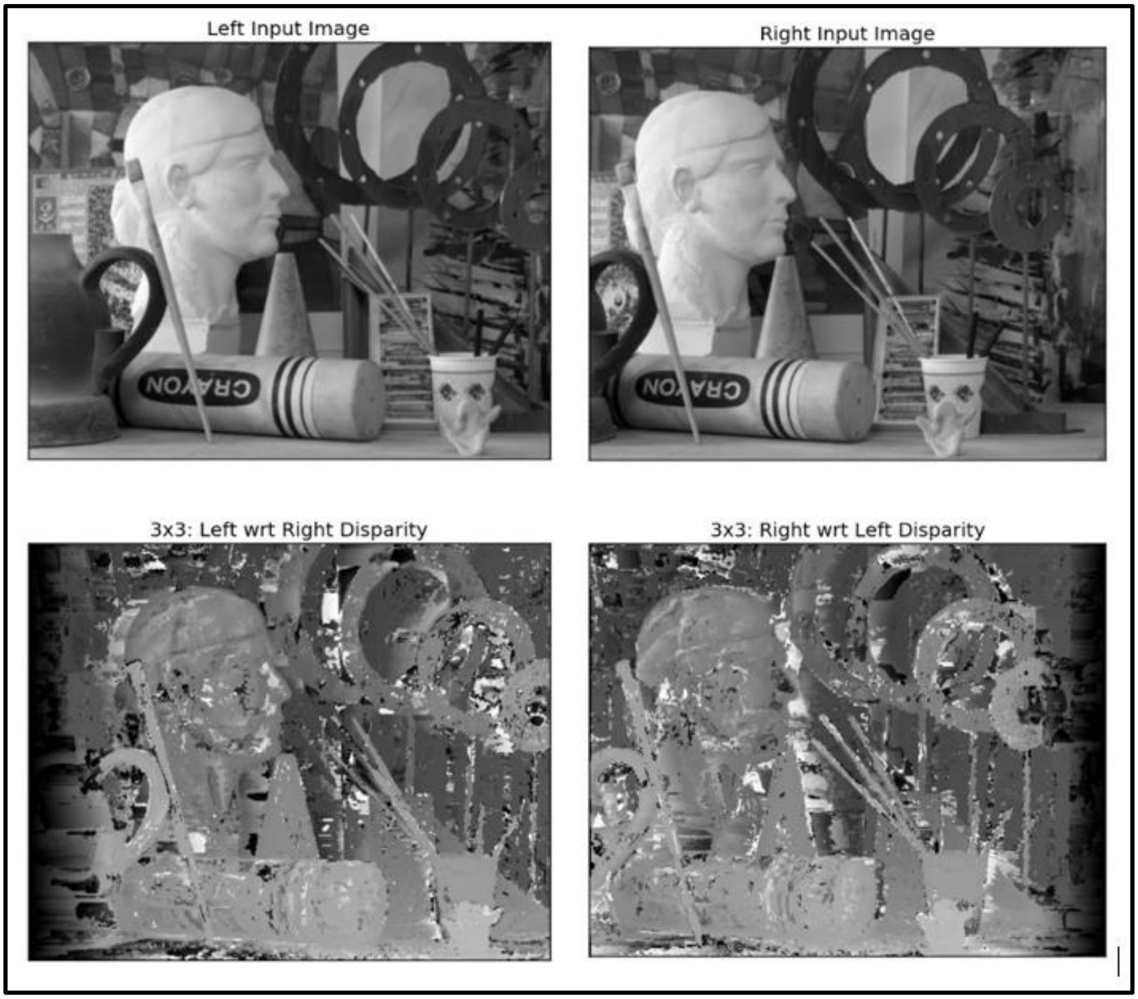
Occlusion Value	Left Image wrt Right	Right Image wrt Image
20	62.81	68.26
10	41.66	46.05
12	46.94	51.38
15	54.72	59.79
6	36.32	41.55
5	59.98	63.19
3	406.35	399.85
1	1183.21	1229.94

As it can be seen in the values above, the lowest MSEs were obtained with the occlusion value of **10** which was chosen as the final value

OUTCOME AND DEVIATIONS

BLOCK MATCHING TECHNIQUE:

3x3 BLOCK OUTPUT:



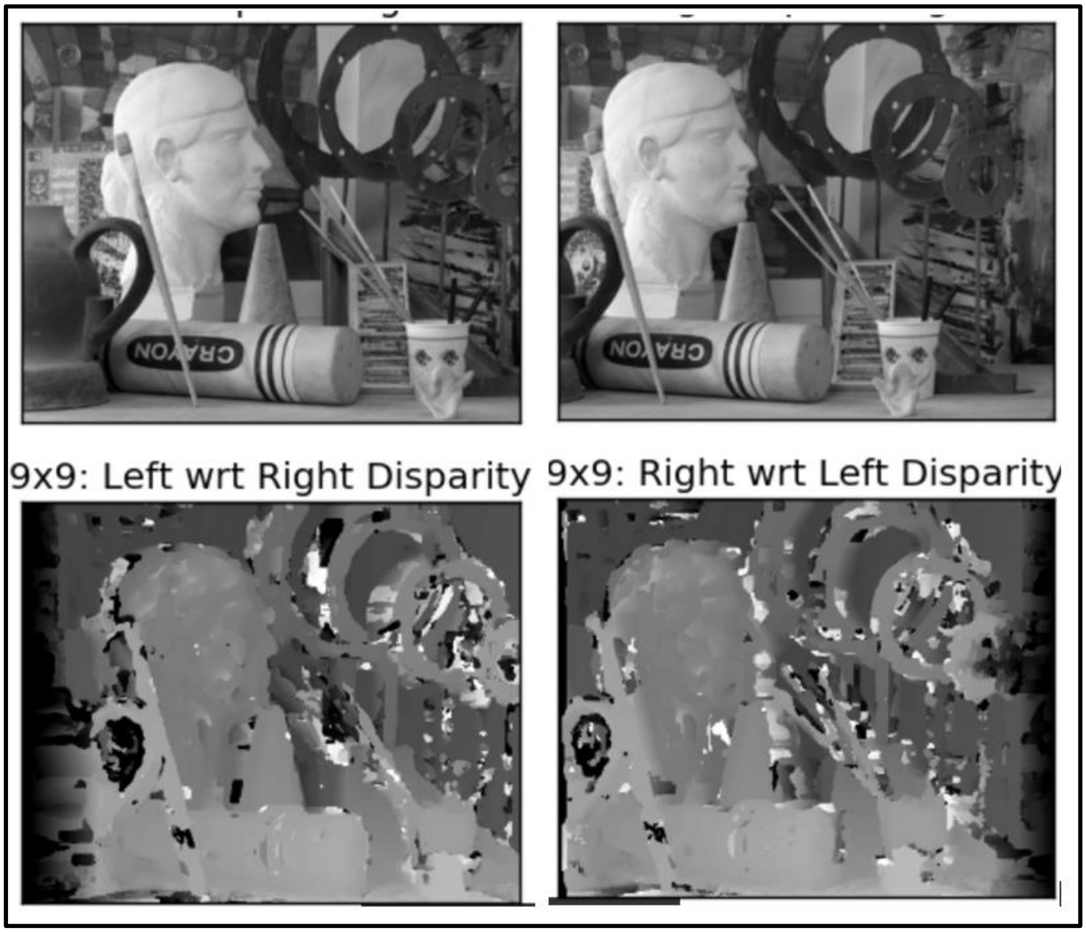
1. DISPARITY MAP: LEFT IMAGE with respect to RIGHT IMAGE

   Mean Square Error for Left view with 3x3 block = 430

2. DISPARITY MAP: RIGHT IMAGE with respect to LEFT IMAGE

   Mean Square Error for Right view with 3x3 block = 338

9x9 BLOCK OUTPUT:



1. DISPARITY MAP: LEFT IMAGE with respect to RIGHT IMAGE

   Mean Square Error for Left view with 9x9 block = 452

2. DISPARITY MAP: RIGHT IMAGE with respect to LEFT IMAGE

   Mean Square Error for Right view with 9x9 block = 326

DYNAMIC PROGRAMMING:

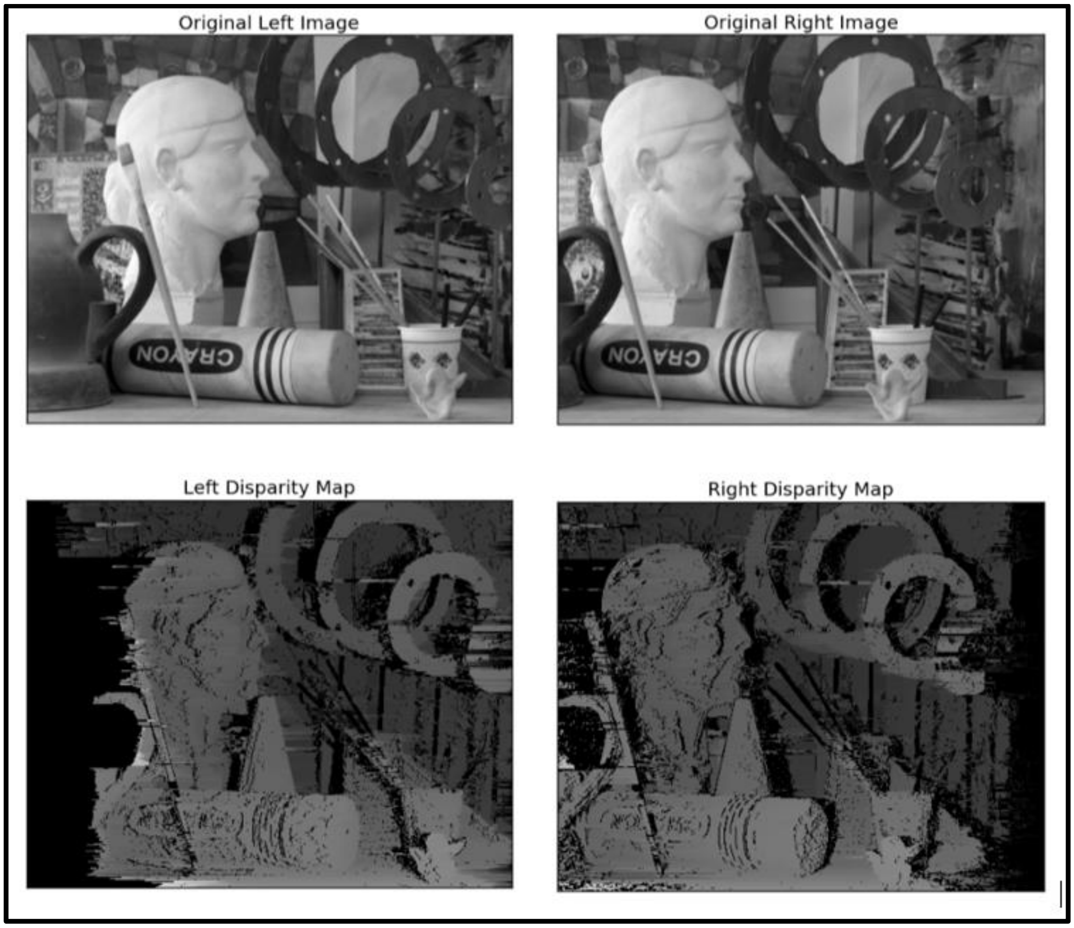


Figure 1: MSE for Left and Right Image

```
Mean Square Error of Left Image :  36.326822719
Mean Square Error of Right Image :  41.5531609363
```

DISCUSSION OF THE OUTCOME:

BLOCK MATCHING V/S DYNAMIC PROGRAMMING

Dynamic Programming is a faster approach and gives a lower MSE value when compared with block matching technique which is a rather crude and basic way of calculating disparity.

Dynamic programming, as explained in the reference paper is dependent on global cost estimation as compared to local cost calculation in block matching technique

DEVIATIONS:

The optimization and testing was performed to bring MSE to its lowest value possible, however, we could not achieve a zero MSE when compared with the ground truth images provided with the problem statement.

LESSONS LEARNED:

1. We learnt how to calculate disparity map from images obtained from 2 different cameras from different angles which can be further processed to determine depth of an image.
2. We studied two techniques of disparity calculation and compared results while optimizing both by testing with different parameters
3. We learnt the applications of Bayesian probability and maximum likelihood estimation in image processing

EXPLANATION OF PROGRAM AND SOFTWARE DEVELOPMENT:

1. Tasks have been divided to maintain equal distributions.
2. The two methods of implementation have been distributed among the two team members.
3. Peer testing and code reviews were performed on both the tasks, hence the testing task as a whole is assigned to both the team members.

BLOCK MATCHING TECHNIQUE:

The overall flow of the code is as below:

1. Depending on the block size (3 or 9), the original images are padded with zeros
   No. of Zeros (Rows and Columns) added = block size/2

2. Based on whether the image is left or right, the disparity map is calculated by shifting the iterator right or left respectively

3. The shift value was chosen as 100 after testing with a few other values (20,30,50,70). In other words, each pixel in one image is compared to 100 pixels on the left or right decided in step2

4. For each shift in step 3, the SSD is calculated inbetween the blocks chosen from image1 and image2

5. A global minSSD 2D array is updated for each SSD calculated which helps us keep track of lowest SSDs for the pixels

6. The disparity is calculated from step5 using the offset from 1 image to the other

7. The whole process is repeated for each blocksize and combinations of left and right image (view1.png and view5.png)

LESSONS LEARNED:

1. Extensive testing was performed and the code was improved to get lower MSE for different directions
2. A generalized code was developed for both 3x3 and 9x9 blocks to calculate disparity for left with respect to right and vice versa. It can be further improved and optimized for speed and memory optimization
3. The code runs slowly because of many nested for loops
4. Code review and peer testing was performed for this method which helped to optimize and reduce the MSE further

DYNAMIC PROGRAMMING:

The code was developed using the following pseudo-code provided in the reference paper:

$$\text{Occlusion} = \left[\ln\left(\frac{P_D}{1-P_D}\frac{\phi}{|(2\pi)^d \mathbf{S}_s^{-1}|^{\frac{1}{2}}}\right)\right]$$

```
for (i=1;i≤ N;i++){ C(i,0) = i*Occlusion }
for (i=1;i≤ M;i++) { C(0,i) = i*Occlusion}
for(i=1;i≤ N;i++){
   for(j=1;j≤ M;j++){
      min1 = C(i-1,j-1)+c(z_{1,i},z_{2,j});
      min2 = C(i-1,j)+Occlusion;
      min3 = C(i,j-1)+Occlusion;
      C(i,j) = cmin = min(min1,min2,min3);
      if(min1==cmin) M(i,j) = 1;
      if(min2==cmin) M(i,j) = 2;
      if(min3==cmin) M(i,j) = 3;
   }}
```

LESSONS LEARNED:

1. Extensive testing was performed to achieve a lower MSE starting from value 20
2. The code is much faster as it involves calculating global costs as compared to local costs in block matching
3. Maximum likelihood estimation algorithm was a new and interesting learning and its application for disparity calculation which can be further used for depth estimation in stereo image processing

SUMMARY AND DISCUSSION:

The project involves in detail study of stereo image processing and how left and right images can be used to calculate disparity maps which have further applications in depth estimation and reconstruction of 3D models using 2D images from different angles in the real world.

In conclusion, the software was developed to match the ground truth disparity images as close as possible

The two approaches were compared with respect to speed and MSE

Dynamic programming approach matches more closely to the ground truth as compared to block matching approach due to its inherent nature involving global cost estimation