CSE 473/573: Computer Vision and Image Processing
Final Project Status Report

Name: Biplab Kumar Das Person#: 50170315

<u>Project Title</u>

Disparity for Stereo Vision – Block Matching and Dynamic Programming

Date of Submission- 12/16/2016

2. LITERATURE REVIEW

2.1) Review of technologies

Block Matching – Technologies used

1) Depth perception using the stereo image pairs
2) Rectification of the image pairs
3) Epipolar scan line
4) Disparity map generation from the stereo image pairs
5) Sum of squared differences and sum of absolute differences
6) Block size selection
7) Image padding
8) Mean square error calculation

Dynamic Programming –

1) Matching cost computation
2) Disparity Optimization
3) Sum of squared differences same as block matching

2.2) Technologies used and their explanation

Block matching

1) SSD – sum of squared differences between 2 different block sizes to find the minimum value (minimum value approximates the best match). Block comparison using SSD (sum of squared differences).
2) Search Range – how much the search should move – to left or to right.
3) Selecting the size of the image block to compare and calculate the SSD in the other image.
4) Image padding – Padding the image with appropriate values so that it is generalized for various values of the block size
5) Disparity map – Usually have the column difference values between the same pixel value in one image to the same pixel value in the other image.
6) Rectification- Since the image is rectified hence we need to carry out the search only along the rows instead of the columns(Epipolar scan line).

Dynamic Programming

1) Maximum Likelihood cost estimation: This is the estimation of local cost of matching two points. We employ two techniques using dynamic programming for the estimation:
   a. In [1] the best disparity match was obtained using the Maximum Likelihood, Minimum Horizontal plus Vertical displacement (MLMH+V) algorithm. We use the same algorithm to find the disparity in our dataset.
   b. Acceptable results were achieved Maximum Likelihood, Minimum Horizontal displacement (MLMH) algorithm. We employ MLMH on our dataset also.
2) Cohesivity Constraints: Since the dynamic programming approach tends to give multiple paths to the optimal solution, we employ cohesivity constrains such as smoothness in such cases.

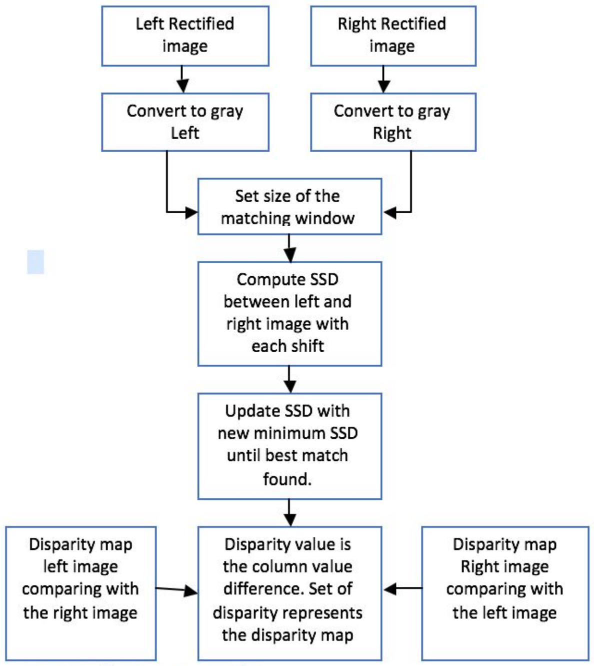
3. INTRODUCTION

3.a)

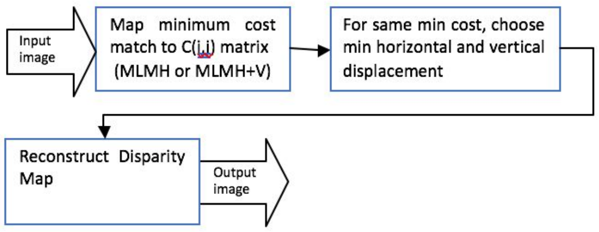
Computer vision is an important field of research wherein the techniques attempt to model the visual environment in using various mathematical model. Stereo vision is an area within the field that addresses the problem of reconstruction of 3D coordinate of points for depth estimation. In a stereo system two cameras are placed horizontally and the images acquired by the system are then processed for recovering visual depth information. The disparity map represents pixels that are horizontally shifted between the left and the right image. In this project we present two methods of mapping disparity in stereo system images.

3. a. Flowchart

Basic Block Matching


Left Rectified image
Right Rectified image
Convert to gray Left
Convert to gray Right
Set size of the matching window
Compute SSD between left and right image with each shift
Update SSD with new minimum SSD until best match found.
Disparity map left image comparing with the right image
Disparity value is the column value difference. Set of disparity represents the disparity map
Disparity map Right image comparing with the left image

Dynamic Programming


Input image
Map minimum cost match to C(i,j) matrix (MLMH or MLMH+V)
For same min cost, choose min horizontal and vertical displacement
Reconstruct Disparity Map
Output image

Overview

Stereo vision block matching using sum of squared differences.

Stereo vision Block matching- A pair of rectified stereo images are given and we denote them as left and right stereo image. Our task is to create a disparity map that is basically the depth information which is computed by the distance between the location of a certain pixel in one image to the location of the pixel in the other image (we take the difference of their columns to get the position).

Stereo vision Dynamic Programming matching using sum of squared differences.

The second method we implemented was based on dynamic programming. In disparity mapping, the number of calculations required increases with an increasing number of pixels per image, causing the problem to be computationally complex. Even though the epipolar constraints of the stereo system reduces the problem to 1D searching, the problem still remains computationally difficult [1]. While DP is executed for each (row) independently, the underlying assumption is that of an ordering constraint between neighboring pixels of the same row.

3.b) Our Approach

Block Matching

Most important concept in block matching is using the sum of squared differences in finding the best match possible among the 2 images.

Algorithmic steps -

1) Converting the image into grayscale.
2) Selecting a block size for block matching
3) To compute SSD we select a block size of 3X3 initially from one image and same size of image pixels in the other image. And next we compute their pixel wise differences and add them to get a single value and save them to an array.
4) We do step 3 for some range defined by the highest value of the ground truth image. And after that we select the minimum value and it's position out of the array having the SSD values. Minimum the value better is the similarity between the pixels.
5) After finding the best approximation or the best similarity that is having the minimum value, we put the difference between the location of the pixel in one image to the location of the best match pixel location in the other image (That is basically their column differences since they are rectified images.)

6) We do the above steps for the whole image. While doing so we encounter the boundary conditions so in that case we have padded the image according to the block size selected so that for any block size the code can find the depth estimation.

Dynamic Programming

Numerous methods of implementation for stereo vision disparity mapping have been established in the past few years. Broadly speaking, disparity map algorithms can be classified into two categories - local or global approaches [2]. In the local approach, also known as window based approach, disparity is computed at any pixel depending on the intensity values within a predefined window. Such method considers only the local information and therefore have low computational complexity. Global approach treats disparity mapping as a problem of minimizing an objective global energy function. Such approach usually produces good results but are computationally expensive and are impractical for real time systems.

In this project for dynamic programming, we implement the work in [3] which measures the maximum likelihood as minimizing the cost of occlusion. We calculate the maximum likelihood of two point on the epipolar lines z_1 and z_2 as [2]:

Eq 1

$$\wedge(Z_{i1,i1}) = \left(\frac{1-P_D}{\Phi}\right)^{d_{i1,i2}} [P_D p(X_k) \times P_D p(X_k)]^{1-d_{i1,i2}}$$

P(z|X) is the probability of the likelihood of z at point X.

P_D is the probability of detecting X

$Z_{i1,i2}$ are the measurements along the epipolar lines *i* for camera 1 and camera 2.

If ɣ denotes all the pairings of the measurements, Γ all feasible partitions and L(ɣ) the likelihood of a partition, then L(ɣ) is defined as

Eq 2

$$L(\text{ɣ}) = p(\text{ɣ}) = \prod_{Z_{i1,i2} \in \text{ɣ}} \wedge(X)$$

Putting equation (i) in equation (ii) we get the total cost function that needs to be minimized.

Software and other methodologies

Software package – matlab and canopy.

This algorithm requires to specify how much the search should be carried out and in what direction. So for the range we basically can find the range by going through the ground truth and for the direction we need to check the input images so that the correct direction where the search should be carried out can be specified. Another experiment is the block size. Different block size will get different disparity maps. Methodologies tried- Sum of squared differences, Block size variation, dynamic Programming, cost matrix construction

3.c) Outcome and deviations

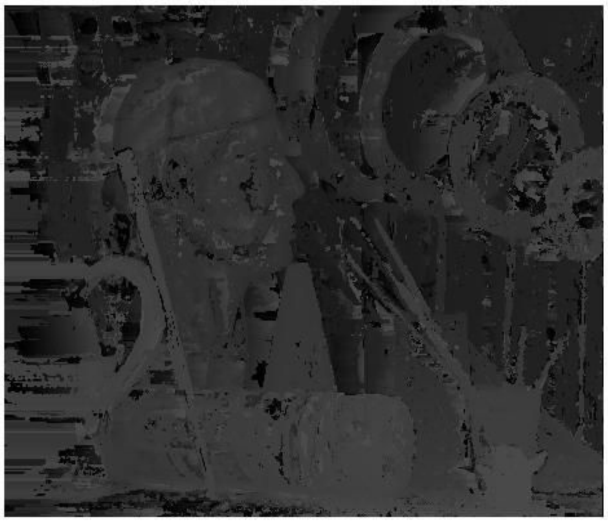
3.c.1) Presentation of the project outcome

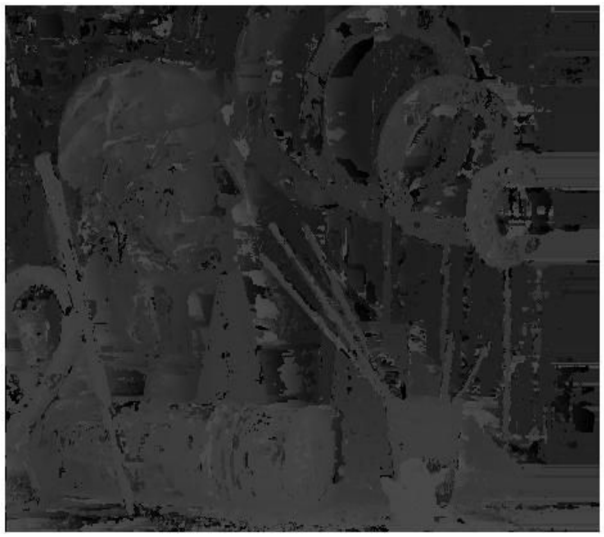
Block matching – 3X3 LEFT AND RIGHT DISPARITY MAPS

Block size 3

mstmp_right – 242.6453

mstmp_left – 192.8107



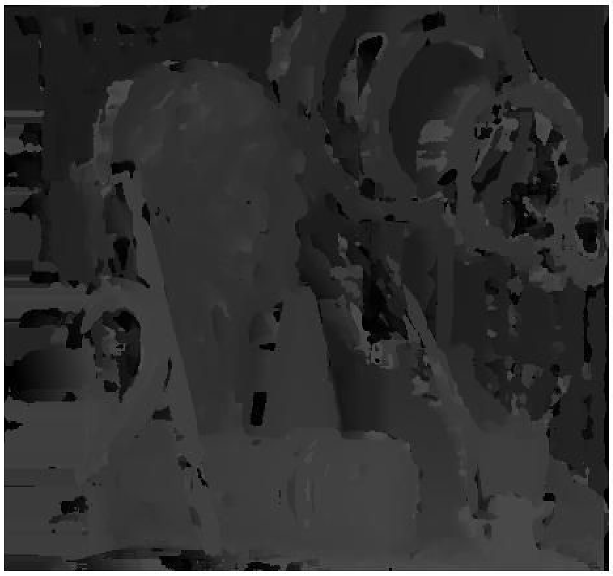
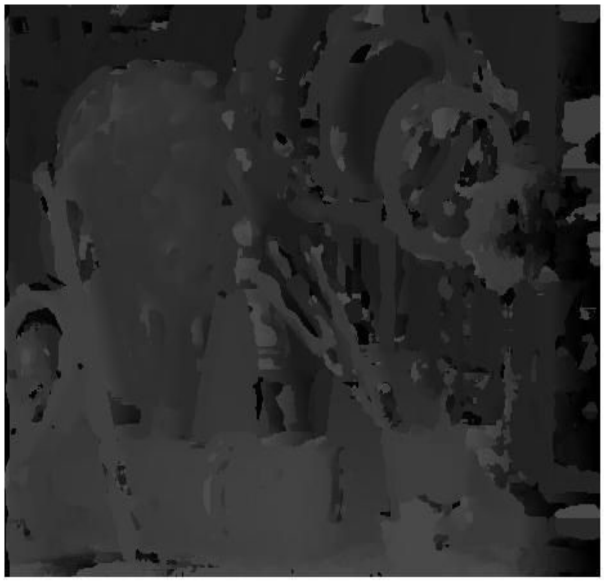


Block matching – 9X9 LEFT AND RIGHT DISPARITY MAPS

Block size 9

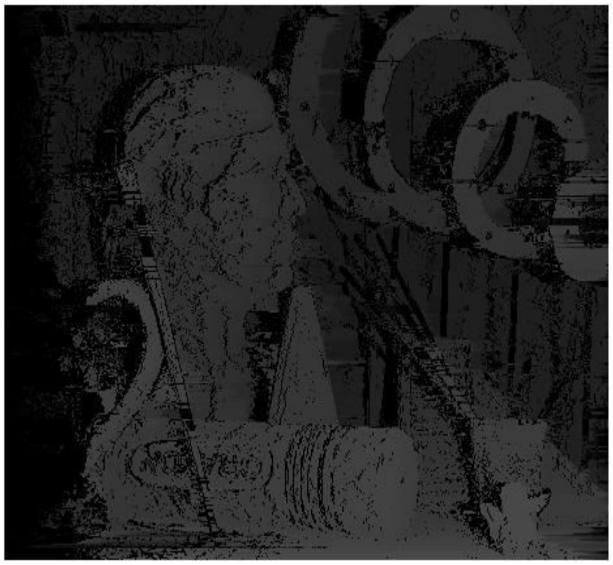
mstmp_right – 215.931,

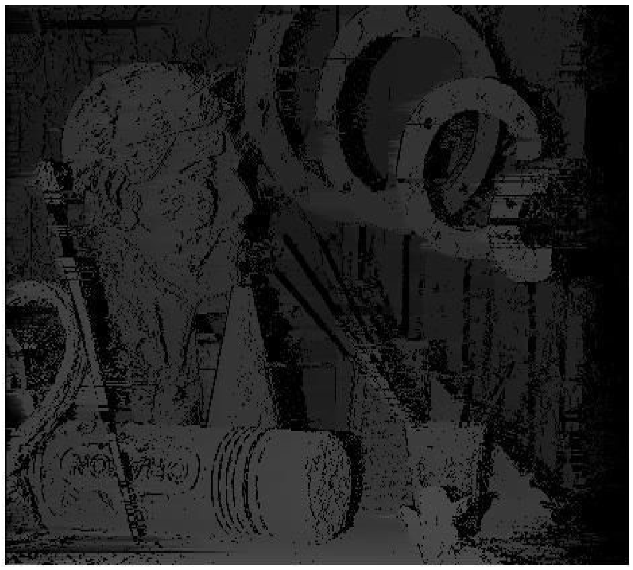
mstmp_left – 210.1133

Dynamic programming

mstmp_right – 215.931,
mstmp_left – 210.1133





3.c.2) Discussion of the project outcome

Block matching
As we experimented with different block sizes we found that larger the block sizes better are the results and with less noise but at a higher cost(increase computation time).
With smaller block size(3) we get lot of noise and but computation time is less.
Reasons- Since we are taking a larger block size we are taking lot of pixels for finding the best match hence there is increase computation time.

Dynamic Programming

3.c.3) Lessons learned from algorithm development.

Block matching-

Better approximation while finding the best match will lead to better results that is shown with larger block size.

Better search range. For example we are searching only in the one direction with fixed range. If we can adaptively approximate that range, then we might have a good result.

Careful when converting the types of the images while making the computation.

Dynamic Programming-

3.d) Presentation of the software and program development

3.d.1) Presentation of the software and program development

Block matching

Software packages used –Matlab and Canopy

Disparity map calculation using SSD for block sizes 3X3 and 9X9 and it can be generalized to any size. And we calculate the disparity maps for both left and right image. Left disparity map – Keep left image constant and find the best match in right image buy carrying out the search in the same. Similarly calculating the disparity map for right image by keeping right image constant and carrying out the search in the left image and finding the best match.

Dynamic programming

3.d.2) Discussion of the code development

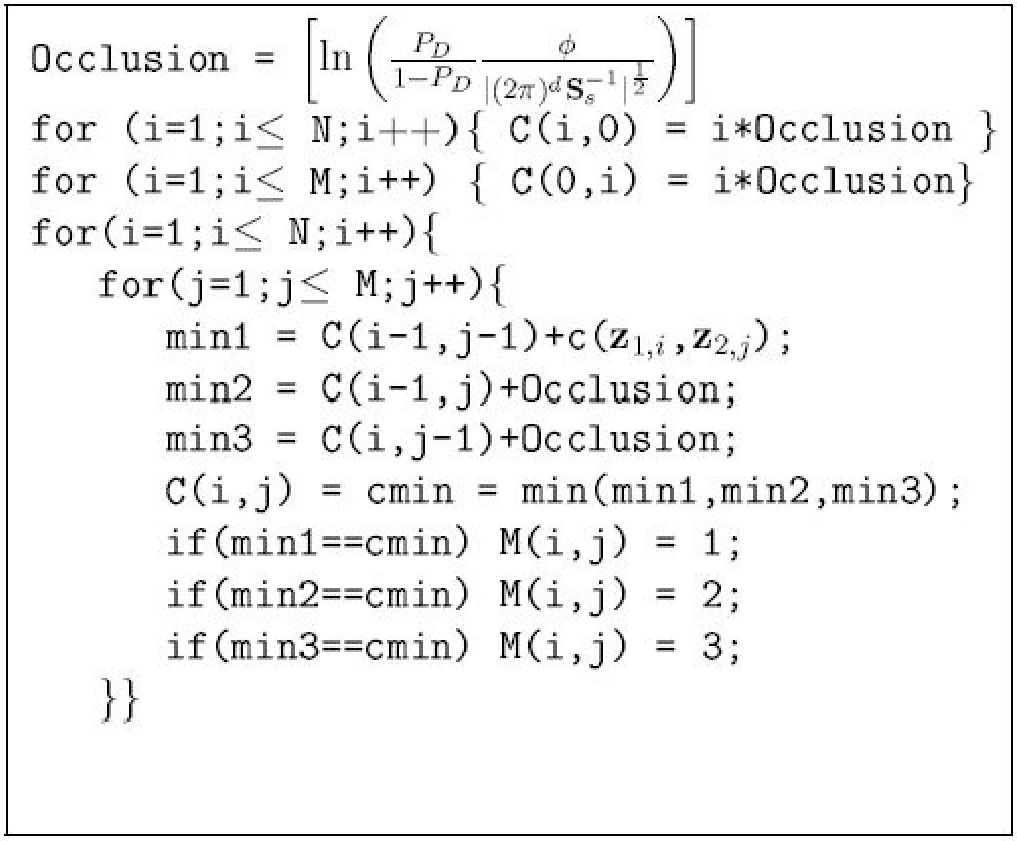
Block matching

Algorithmic steps -

1. Converting the image into grayscale.
2. Selecting a block size for block matching
3. To compute SSD we select a block size of 3X3 initially from one image and same size of image pixels in the other image. And next we compute their pixel wise differences and add them to get a single value and save them to an array.
4. We do step 3 for some range defined by the highest value of the ground truth image. And after that we select the minimum value and it's position out of the array having the SSD values. Minimum the value better is the similarity between the pixels.
5. After finding the best approximation or the best similarity that is having the minimum value, we put the difference between the location of the pixel in one image to the location of the best match pixel location in the other image(That is basically their column differences since they are rectified images.)
6. We do the above steps for the whole image. While doing so we encounter the boundary conditions so in that case we have padded the image according to the block size selected so that for any block size the code can find the depth estimation.

Dynamic Programming-

???



```
Occlusion = [ln($\frac{P_D}{1-P_D}\frac{\phi}{|(2\pi)^d \mathbf{S}_s^{-1}|^{\frac{1}{2}}}$)]
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($\mathbf{z}_{1,i}$,$\mathbf{z}_{2,j}$);
      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;
   }}
```

3.d.3) Lessons learned from the program code development

Block Matching

We started with the simplest case of 1X2 block to simply the code and then we generalized it for any block size. We encountered the boundary conditions for all the corners and the boundary pixels (We overcame by padding). Next we had to design a logic for checking the first pixel let's say. So we needed to find the same pixel in the other image which does not exist. (Here we adaptively implemented the code so that it increases the search size with every time the block moves from one pixel to the next. for example – 1st pass will have 0 shifts, 2nd pass will have 1 shift and so on till the range of groundtruth).

Dynamic Programming

?????

3.e) Summary and Discussion

3.e.1) Summary

Block matching

We carried out the Stereo imaging depth map estimation using block matching method and were successful in generalizing the code for any size(say 3,5 9).

Dynamic Programming

3.e.2) Lessons learned in classroom, HW, Projects, Assignments

Biplab

Various techniques in image processing and how the process can be used for computer vision. Techniques such as edge detection, hough transform, Ransac algorithm (even though the assignment were not given for Ransac and hough transform, it was good to carry out the code for these algorithms). The main lesson I learned from this course is how difficult is to apply these concepts in the real life examples but most importantly it was worth doing the codes to see the results in front of your eyes. In one of the challenging concept to implement for me was Ransac algorithm through random data.

Purbarag

?????

3.f) Bonus part

Referrence-

References:
1. Ingemar J. Cox, Sunita L. Hingorani, Satish B. Rao, “A Maximum Likelihood Stereo Algorithm”, Computer Vision and Image Understanding, pp. 542-567, May 1996.

2. Rostam Affendi Hamzah and Haidi Ibrahim, “Literature Survey on Stereo Vision Disparity Map Algorithms**”,** Journal of Sensors, Volume 2016 (2016).

3. K. R. Pattipati, S. Deb, and Y. Bar-Shalom, “Passive multisensory data association using a new relaxation algorithm”, Multi-target multi-sensor tracking: advanced application, pp. 219-246, Artech House, 1990.

4. A stereo matching algorithm with an adaptive windw: Theory and Experiment by Takeo Kanade, Fellow, IEEE, and Masatoshi Okutomi IEEE TRANSACTIONS ON PATTERN ANAYLSIS AND MACHINE INTELLIGENCE, VOL. 16, NO. 9, SEPTEMBER 1994