**Task 3**

**Task:** Implement a MATLAB function that estimates disparity from the Middlebury stereo images with different block radii using stereo matching algorithm SAD (sum of absolute difference). Measure the quality of the reconstructed map with the bad pixels -function you implemented at different radii. Make a single plot that contains a graph for each stereo pair depicting the quality metric as a function of block radius.

**Answer:** Please find the attached code.

**Analysis:** Which block size is the best for each image? Is there a correlation between the type of content and the best block size you observed? Attach the best estimate of each image to the report. Take a look at the output parameter cost from the depth estimation function. What kind of problem areas could you identify based on that information? On the other hand, what problem areas cannot be determined from it?

**Answer:**

Method of Sum of Absolute difference**:**To measure similarity between reference block and the block checked, sum of absolute difference (SAD) was used. We can summarize this approach as following equation.

*diff(y,x)d=∑(-W≤j≤W) ∑\_(-W≤i≤W) |I\_left (y+j,x+i)-I\_right (y+j,x+i+d)|, for d=[0,d\_max]*

In that formula (y,x) refers pixel position of the image. Ileft and Iright shows reference (left) and right image. W refers the half of the block size. As you can see the summing space is from –W to +W. So we need to calculate 2*W+1 rows and cols, which means the total of block size is 2*W+1. We need to calculate this sum of absolute value for every single pixel and for every d value in range of [0, dmax]. As a result we can assign disparirt value of pixel located (y,x) by following equation.

*disparity(y,x)=argmin┬d⁡(diff(y,x)\_d)*

Which means, we need to select optimum d value which provide us minimum diff value which calculated for location (y,x) along the 2\*W+1 block size. To run that method, we provide a function named blockmatching, you can test it with following command,

disp=blockmatching(leftI,rightI,blockSize,maxd);

In that command, you should give left image, right image, blocksize and maximum disparity as parameter, then you will get calculated disparity map

From the different block radii tried, it was observed that the smaller size was more suitable for the SAD algorithm if the disparity was minimum but the cost was maximum, given the algorithm is ideal.

The bad pixel count and the computational time for smaller block radii was less. The time taken to find the estimated disparity was higher for the SAD method compared to the Belief Propagation method. The matching aka the quality metric also varies such that the recent algorithm is better.

For example, in the case of

Data [Teddy] –

0.6min for 3\*3 matrix

0.65min for 5\*5 matrix

0.72min for 7\*7 matrix

0.82min for 9\*9 matrix

Similarly, for Data [Reindeer]

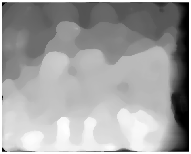
0.57min for 3\*3 matrix

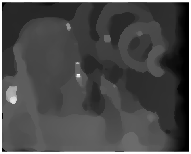
0.62min for 5\*5matrix

0.69min for 7\*7 matrix

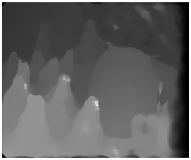
0.78min for 9\*9 matrix

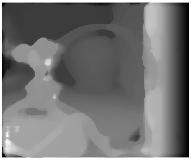
Estimates found:

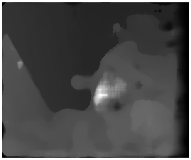




The best estimate was found for dataset – DOLLS 3\*3- As Sum of Absolute difference is both window-based approach and pixel value-based approach: chancing of having less bad pixel is high for this set. I Believe this is due to the large number of objects compared to other datasets and also the range of pixel values. Also, for the dataset-art, the reverse was found true i.e , the bad pixel count is higher for 3\*3 matrix

.

Problem areas: The most problematic side of simple block matching is it is lack of provide a spatial consistency which means the output of simple matching algorithm has big variation. That is why we can see too many artifacts on the result. The main reason of that problem is simple block matching just tries to find best match blocks but not take into account the neighbor pixels disparity. Also, since it does not take the vertical neighborhood into account it fails significant amount of pixel.

The area-based approach has other problems which can’t been determined includes:

* Assumes constant depth within the window – this implicit assumption is violated at – Depth discontinuities – Slanted/non-planar surfaces
* Repetitive textures
* Uniform areas
* Thin structures (window larger than the structure
* Problems with Discontinuities in Detail