Stereo Vision

ECE 661 Computer Vision

Name – Gowri Kurthkoti Sridhara Rao

Introduction

"Stereo matching" is the task of estimating a 3D model of a scene from two or more images. The task requires finding matching pixels in the two images and converting the 2D positions of these matches into 3D depths.

The basic idea is the following: a camera takes picture of the same scene, in two different positions, and we wish to recover the depth at each pixel. Depth is the missing ingredient, and the goal of computer vision here is to recover the missing depth information of the scene when and where the image was acquired. Depth is crucial for navigating in the world. It is the key to unlocking the images and using them for 3d reasoning. It allows us to understand shape of world, and what is in the image.

In this project, disparity map using two different images has to be computed. The amount of horizontal distance between the object in Image L and image R (the disparity d) is inversely proportional to the distance z from the observer. The basic idea of finding the disparity of a pixel is looking for the same pixel in the other image along the scan line. There are different methods in which the match can be found.

In this project, two different distance metrics are used, first being SSD (Sum of square difference) and the other one being NCC (Normalized cross-correlation).

SSD(Sum of squared differences): The SSD measure is sum of squared difference of pixel values in two patches. This matching cost is measured at a proposed disparity. If A,B are patches to compare, separated by disparity d, then SSD is defined as:

SSD(A,B)=∑i,j(Aij−Bij)2

Where A and B are of same shape. SSD is measured for the entire scanline and the lowest SSD value is considered a match.

NCC(Normalized Cross-correlation): In normalized cross correlation, the blocks used for calculating the distance is first normalized using the below equation. These normalized blocks are then cross correlated. All the pixels in the scanline are normalized and cross correlated. The value which is high is considered the match pixel and the distance between the matched pixel and current pixel location is given as the disparity value.

$$\hat{f} = \frac{f - \bar{f}}{\sqrt{\sum (f - \bar{f})^2}} \qquad \hat{g} = \frac{g - \bar{g}}{\sqrt{\sum (g - \bar{g})^2}}$$

$$\mathrm{NCC(f,g)} = C_{fg}(\hat{f} \cdot \hat{g}) = \sum_{[i,j] \in R} \hat{f}(i,j)\hat{g}(i,j)$$

Steps followed for the stereo matching algorithm:

1. Left and right images are read.
2. A window and a search range are initially selected.
3. A block of size specified by window is taken from the left image and it is considered window
4. A block of same size is taken in the right image
5. SSD or NCC method is used to identify the disparity
6. Right image block is moved along the same row or until the search range limit is met
7. For each row, the disparity is identified, lowest value for SSD and highest value in NCC
8. The value is returned, and it is stored in the disparity map
9. The disparity map is displayed

Code

Main

```
%% Disparity map using SSD
window = 30; %For street 5
disp_range = 45; %For street 20
tic;
SSD_Disp = ssd(left,right,window,disp_range);
toc;
%%
SSD_Disp = abs(SSD_Disp);
range = [0 50];
figure()
imshow(SSD_Disp,range);title("Using SSD");


%% Disparity map using NCC
window = 4;
disp_range = 9;
tic;
NCC_Disp = ncc(left,right,window,disp_range);
toc;
%%
NCC_Disp = abs(NCC_Disp);
range = [-5 10];
figure()
imshow(NCC_Disp,range);title("Using NCC");
```

SSD function

```
function [I_disp] = ssd(left,right,window,maxDisp)
window = window;
disp_range = maxDisp;

row = size(left,1);
col = size(left,2);
%
for m =1:row
    row_min= max(1, m- window);
    row_max= min(row, m+ window);

    for n =1:col
          col_min= max(1,n-window);
          col_max= min(col,n+window);
          %% setting the pixel search limit

          pix_min= max(-disp_range, 1 - col_min);
          pix_max= min(disp_range, col - col_max);

          template = left(row_min:row_max ,col_min:col_max);

          block_count= pix_max - pix_min + 1;
          block_diff= zeros(block_count, 1);

          for i = pix_min : pix_max
              block= right(row_min:row_max ,(col_min +i ):(col_max+i));
              index= i-pix_min+1;
              block_diff(index,1)= sumsqr(template - block); %SSD

          end
          [B,I]= sort(block_diff);
          match_index= I(1,1); %Picking the minimum
          disparity= match_index+pix_min-1;
          I_disp(m, n) = disparity;
    end
end

end
```

NCC function

```
function [I_disp] = ncc(left,right,window,maxDisp)

%% NCC
% Normalizing each patch
% Correlation
% Pick the Maximum
window = window;
disp_range = maxDisp;


row = size(left,1);
col = size(left,2);
```

```
%
for m =1:row
    row_min= max(1, m- window);
    row_max= min(row, m+ window);

    for n =1:col
          col_min= max(1,n-window);
          col_max= min(col,n+window);
          %% setting the pixel search limit

          pix_min= max(-disp_range, 1 - col_min);
          pix_max= min(disp_range, col - col_max);

          template = left(row_min:row_max ,col_min:col_max);
          template = normOwn(double(template)); %Normalizing

          block_count= pix_max - pix_min + 1;
          block_diff= zeros(block_count, 1);

          for i = pix_min : pix_max
              block= right(row_min:row_max ,(col_min +i ):(col_max+i));
              block = normOwn(double(block));%Normalizing
              index= i-pix_min+1;
              out = corrilate(template, block);
              block_diff(index,1)= out; %Cross correlation

          end
          [B,I]= sort(block_diff);
          match_index= I(size(I,1),1); %Picking the minimum
          disparity= match_index+pix_min-1;
          I_disp(m, n) = disparity;
    end
end
end
```

<u>Normalization Function</u>

```
function [outMat] = normOwn(inMat)
f_bar = mean(inMat);
temp = (inMat - f_bar).^2;
deno = sqrt(sum(sum(temp)));
outMat = (inMat - f_bar)./deno;
end
```

Results and Discussion

The above code was executed for two different set of given images, first one is an image which does not have too many details has a background and an object in front. Second is the set of images which has a lot of details and there is an evident change between the two images.

First image requires higher window size and search range to find the disparity map since it does not have very high details. Second set of images give better results with lower window size and lower search range as it has higher details.

Image set 1

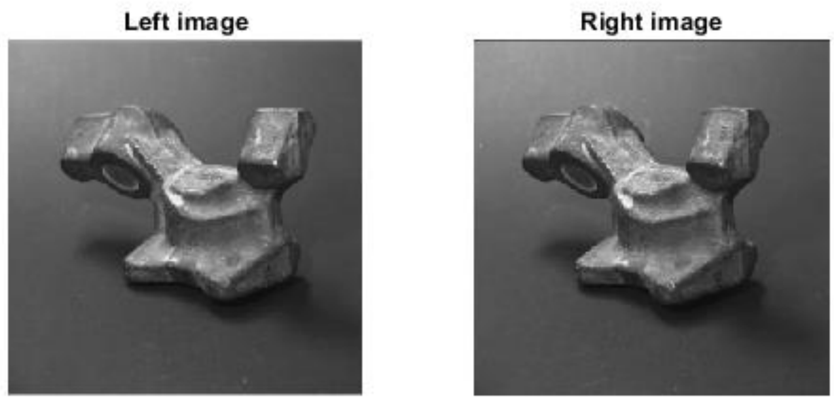


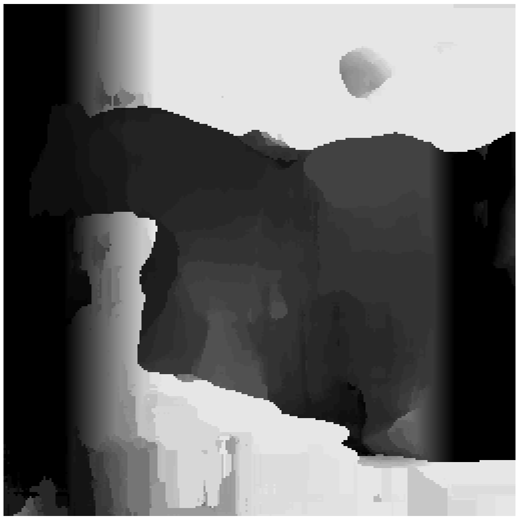
Image set 2

Image 1 using SSD

- Window = 30; Search range = 45
  - Elapsed time is 77.154630 seconds.
- Window = 21; Search range = 48
  - Elapsed time is 58.033090 seconds.
- Window = 19; Search range = 64
  - Elapsed time is 65.559365 seconds.
- Window = 15; Search range = 30
  - Elapsed time is 26.851099 seconds.
- Window = 10; Search range = 25

    - Elapsed time is 18.469434 seconds.
- Window = 5; Search range = 15
    - Elapsed time is 7.328006 seconds.

Using SSD, window = 30, range =45

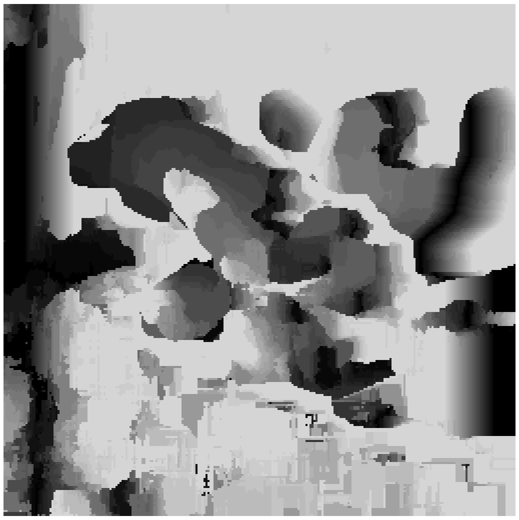


Using SSD, window = 21, range =48

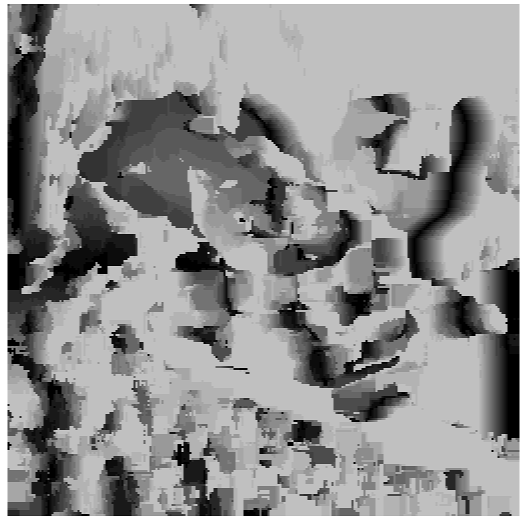
Using SSD, window = 19, range =64

Using SSD, window = 15, range =30

Using SSD, window = 10, range =25



Using SSD, window = 5, range =15



From the above results, it can be inferred that as the window size and the disparity range were reduced, the time elapsed also reduced but even the quality of disparity matching is reduced. The last image is very distorted.

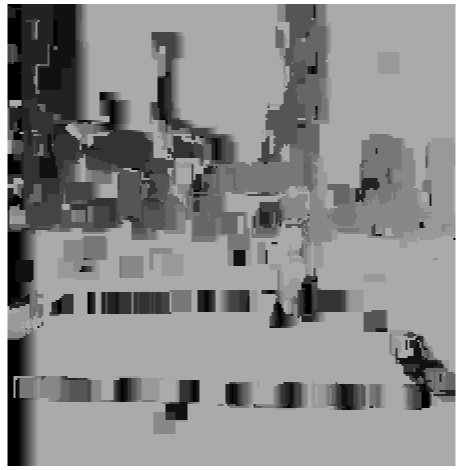
Image set 2 using SSD

- Window = 5; Search range = 5
    - Elapsed time is 2.269490 seconds.
- Window = 5; Search range = 10
    - Elapsed time is 4.701842 seconds.
- Window = 5; Search range = 20
    - Elapsed time is 11.387171 seconds.
- Window = 7; Search range = 20
    - Elapsed time is 9.663164 seconds.
- Window = 15; Search range = 30
    - Elapsed time is 22.313932 seconds.
- Window = 21; Search range = 40
    - Elapsed time is 35.128920 seconds.

From the results of image set 2, when the window size and search range is very low, all the details in the picture could not be captured. At window size of 5 and range of 20, the disparity map captured almost all details and matched them. When the size was further increased, most of the details were lost and the image got blurred.

Using SSD, window = 5, range =5

Using SSD, window = 5, range =10



Using SSD, window = 5, range =20

Using SSD, window = 7, range =20

Using SSD, window = 15, range =30

Using SSD, window = 21, range =40

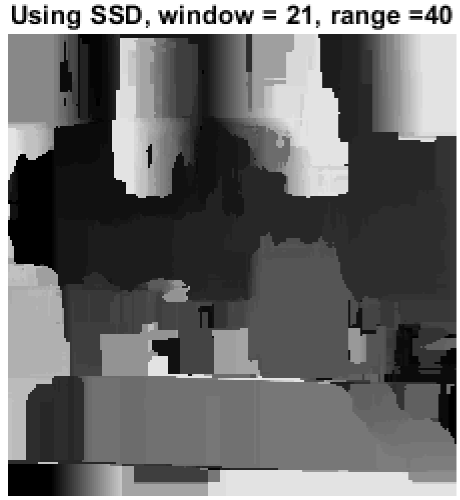


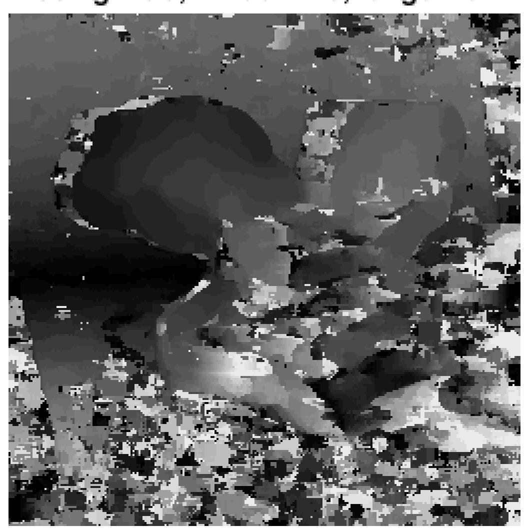
Image set 1 using NCC

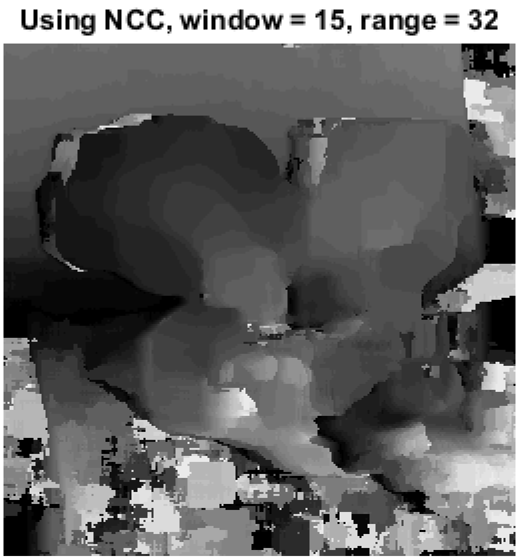
- Window = 11; Search range = 25
  - Elapsed time is 24.546122 seconds.
- Window = 5; Search range = 20
  - Elapsed time is 19.846177 seconds.
- Window = 15; Search range = 25
  - Elapsed time is 42.908523 seconds.
- Window = 11; Search range = 32
  - Elapsed time is 42.566406 seconds.
- Window = 5; Search range = 32
  - Elapsed time is 31.267162 seconds.
- Window = 21; Search range = 40
  - Elapsed time is 63.451289 seconds.

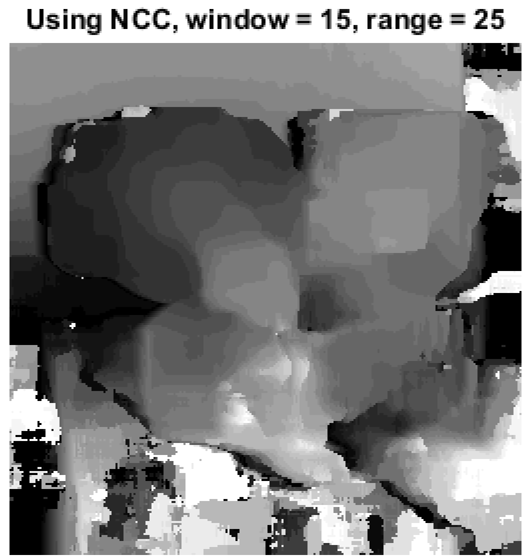
Results of the images with varied window size and search range are presented. The image is distorted and has too many details when the window size is low but all the edges are smoothened when the window size is increased too much. At an optimal value it gives satisfactory results.

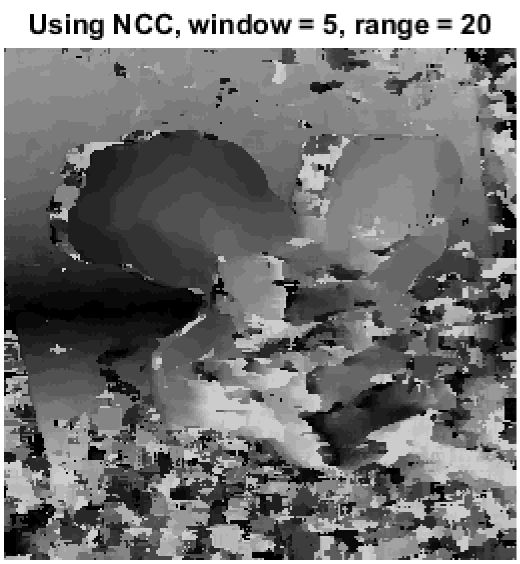
Using NCC, window = 21, range = 40

Using NCC, window = 5, range = 32









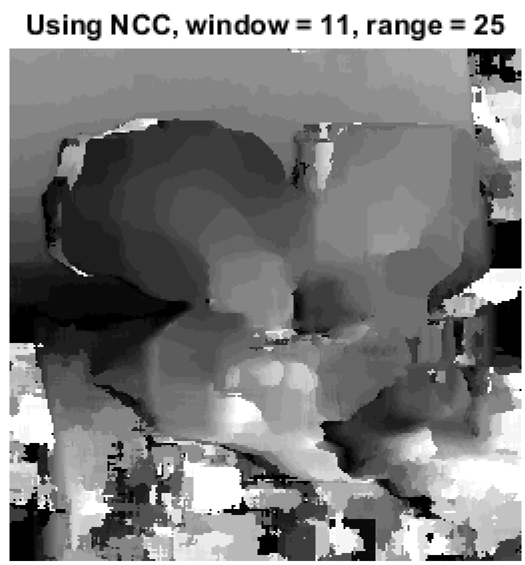
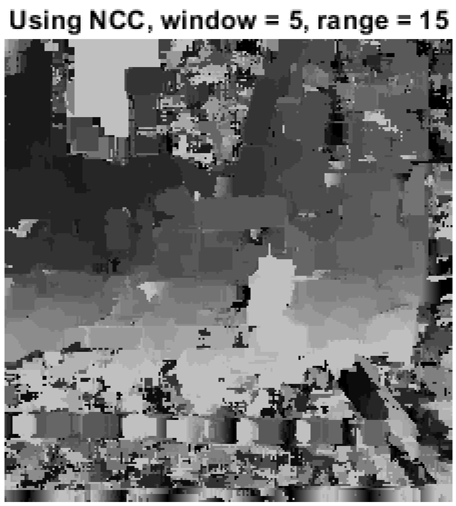


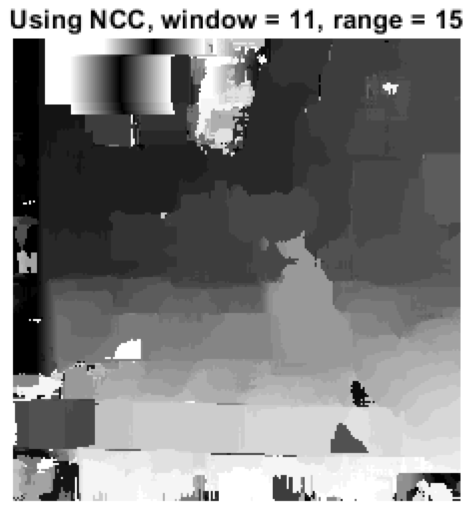
Image set 2 using NCC

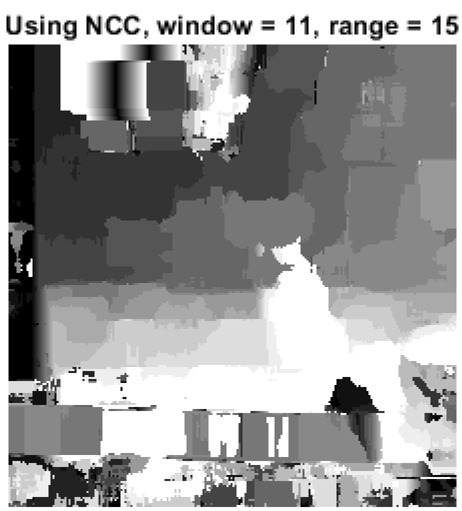
- Window = 4; Search range = 9
  - Elapsed time is 5.852282 seconds.
- Window = 7; Search range = 10
  - Elapsed time is 7.169799 seconds.
- Window = 5; Search range = 20
  - Elapsed time is 15.300560 seconds.
- Window = 5; Search range = 15
  - Elapsed time is 12.985475 seconds.

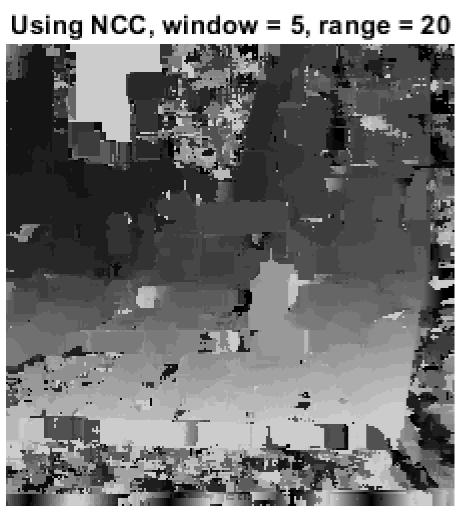
- Window = 11; Search range = 15
  - Elapsed time is 11.295690 seconds.
- Window = 11; Search range = 25
  - Elapsed time is 23.972149 seconds.

The inference from the result images is similar to what was inferred during the SSD process. During NCC, more details are captured as the window is reduced compared to SSD.









Using NCC, window = 7, range = 10

Using NCC, window = 4, range = 9

Conclusion

Disparity map of two sets of images were computed in this project. For the calculations, two metrics, SSD and NCC were used, and their results were compared. The method and the code used can be improved and any feedback that can help the improvement will be very helpful.