EE 569 HOMEWORK 2

GUNJAN JHAWAR
gunjanjh@usc.edu

Contents

1.Problem 1

1.1Abstract and Motivation

Geometrical image modification is one of the important steps in image manipulation apart from image restoration and enhancement. They are extensively used in image morphing, computer graphics and panorama stitching. Operations like scaling, translation and rotation help to produce desired result. They change the position of pixels without changing their color to create special effects, register two images taken of the same image at different time and orientation and to morph one image to another. [1]. Sometimes, images consist distortion in a shape and is transformed to another shape. It can be achieved using geometrical warping concept. When multiple images of a scene are taken and is wants to represent as a single image creating panorama effect, it is done using homographic transformation and image stitching concept.

In this problem, we implement image morphing to transform square image to disc shaped image. The warped image makes sure that the pixels lie on boundaries of square lie on the boundary of the circle. The center of the original image also maps to the center of the warped image and it is one to one mapping.

Then in the second part, we use homographic transformation and image stitching techniques to create panorama consisting of multiple images. We estimate a homographic transformation matrix which is used to find mapping between the input and output images. The algorithm uses the parameters of the camera and uses four common points representing the same object in both images.

1.2 Approach and Procedure

There are two parts in this section. The methodology applied for geometrical warping is described in Section A and Homographic transformation and image stitching to create panorama image is described in Section B.

1.2.1 Geometrical Warping

It is easier to use translation, rotation and scaling on cartesian coordinates as compared to image coordinates. So firstly, we convert image coordinates to cartesian coordinates and then do computation.
To convert an image cartesian we use the following matrix:

$$\begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0.5 \\ -1 & 0 & h - 0.5 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} r \\ c \\ 1 \end{bmatrix}$$

$$Equation\ 1$$

In the above equation, the input image row coordinate 'r' and column coordinate 'c' are multiplied by the conversion matrix to get 'x' the cartesian coordinate x and 'y' the cartesian coordinate y. The 'h' refers to the image height. By taking the inverse of the conversion matrix, we can convert the cartesian coordinate back to image coordinate.

We the translate and scale the image to prevent from getting negative coordinates in the image. To do so we subtract the x-coordinate from (image width/2) and y-coordinate from (image height/2). After that, to normalize it, we divide the x coordinate by (image width/2) and y coordinate by (image height/2).

To convert a square image to a disc image we firstly ensure that our coordinates are a part of the disc by ensuring that $(x^2 + y^2) > 1$, if not, we continue and assign them to black and they represent the background value.
If they are a part of the disc, we convert them from disc coordinate to square coordinate to assign them to the output image. For converting disc coordinate to square coordinate, we use the following equation:

$$x = \frac{1}{2}\sqrt{2 + u^2 - v^2 + 2\sqrt{2}u} - \frac{1}{2}\sqrt{2 + u^2 - v^2 - 2\sqrt{2}u}$$

$$y = \frac{1}{2}\sqrt{2 - u^2 + v^2 + 2\sqrt{2}v} - \frac{1}{2}\sqrt{2 - u^2 + v^2 - 2\sqrt{2}v}$$

After, converting them to square coordinate, it is time to make them in range of [0,511] so x and y are multiplied by (image width/2) and (image height/2) respectively and added by (image width/2) and (image height/2) respectively.

Then cartesian coordinate is converted back to image coordinate to save it as image. This is done using equation 1. Now, the pixel values corresponding to these image coordinates in the original image, we take their pixel value and place them in the output image's row and column. Since, the image coordinates can be floating point number, bilinear interpolation technique is used to the determine the value of the corresponding pixel.

Similarly, using the inverse of the above approach disc image is converted to square image.

Firstly, the image coordinates are converted to cartesian coordinates. We the translate and scale the image to prevent from getting negative coordinates in the image. To do so we subtract the x-coordinate from (image width/2) and y-coordinate from (image height/2). After that, to normalize it, we divide the x coordinate by (image width/2) and y coordinate by (image height/2).

To convert disc image to a square image we firstly ensure that our coordinates are a part of the disc by ensuring that $(x^2+ y^2)$ >1, if not, we continue.
If they are a part of the disc, we convert them from disc coordinate to square coordinate to assign them to the output image. For converting square coordinate to disc coordinate, we use the following equation:

$$u = x\sqrt{1 - y^2/2}$$

$$v = y\sqrt{1 - x^2/2}$$

After, converting them to square coordinate, it is time to make them in range of [0,511] so x and y are multiplied by (image width/2) and (image height/2) respectively and added by (image width/2) and (image height/2) respectively.

Then cartesian coordinate is converted back to image coordinate to save it as image. This is done using equation 1. Now, the pixel values corresponding to these image coordinates in the original image, we take their pixel value and place them in the output image's row and column. Since, the image coordinates can be floating point number, bilinear interpolation technique is used to the determine the value of the corresponding pixel.

1.2.2 Homographic Transformation and Image Stitching

In this problem, we use two images at one time taken from different camera viewpoints. We need to integrate them into one single image. We achieve this by taking common points from both the images. In our problem, we select 4 control points from each image which represent the same object in both images so that both these points can be overlapped after changing of the images into the same orientation as the other image.

We find a relation between the orientation of both the images by taking 'H', homography matrix, which takes care of the intrinsic and extrinsic parameters of the camera. The homography matrix is computed using the formula:

$$\begin{bmatrix} h1 \\ h2 \\ h3 \\ h4 \\ h5 \\ h6 \\ h7 \\ h8 \end{bmatrix} = \begin{bmatrix} X1 \\ Y1 \\ X2 \\ Y2 \\ X3 \\ Y3 \\ X4 \\ Y4 \end{bmatrix} \times pinv \begin{bmatrix} x1 & y1 & 1 & 0 & 0 & 0 & -x1X1 & -y1Y1 \\ 0 & 0 & 0 & x1 & y1 & 1 & -x1Y1 & -y1Y1 \\ . & . & . & . & . & . & . & . \\ . & . & . & . & . & . & . & . \\ . & . & . & . & . & . & . & . \\ . & . & . & . & . & . & . & . \\ x4 & y4 & 1 & 0 & 0 & 0 & -x4X4 & -y4Y4 \\ 0 & 0 & 0 & x4 & y4 & 1 & -x4Y4 & -y4Y4 \end{bmatrix}$$

The H matrix values are determined by multiplying the second image coordinates X_i, Y_i with the pseudo inverse of the reference image (in our case middle image) x_i, y_i The H matrix was calculated using least square's method by keeping h9 as 1. This constraint was used to solve 9 variables using 8 equations. The H matrix was estimated on MATLAB.

The H matrix was found out to be:

$$\text{H} = \begin{bmatrix} 2.3556 & -0.0341 & -618.1912 \\ 0.7335 & 2.1141 & -266.0111 \\ 0.0030 & 0.0001 & 1 \end{bmatrix}$$

After estimating the H matrix, we loop through the second image to find its pixel location corresponding to the first reference image. We find its using:

$$\begin{bmatrix} x' \\ y' \\ w \end{bmatrix} = H \times \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} ; \; X = \frac{x'}{w} \; and \; Y = \frac{y'}{w}$$

$$Equation\ 2$$

Where X, Y are the corresponding Cartesian coordinates of the second image with respect to reference image.

and the maximum of value among the red, green and blue channel is found and similarly is minimum value found. There average value is the new value assigned to the vector for that index.

We find the minimum value of X coordinate and Y coordinate and subtract it from the all the other x and y values. We do this to translate the image from negative axis to origin to positive quadrant.

Using the new translated image coordinates, we find the size of the new image using (Maximum image coordinate – Minimum image coordinate +1) for both X and Y axis. So now, we know the size of the new image.

Now, at this point we have the black image of the new image and we need to place pixel values in the new image by finding out what pixel location does it correspond to in the original image. For that, we convert the image coordinate to cartesian coordinate for the new image. Then multiply inverse H matrix with the coordinates. Translate the image with the same minimum X and minimum Y values that we used in previous steps. Then we use Equation 2 to find x and y value. For the image coordinates we extract their

pixel value from the original image. Place this pixel value to the corresponding row and column of the new image.

1.3Experimental Results

A.

Figure 1 Warping to Disc of Puppy, Panda and Tiger

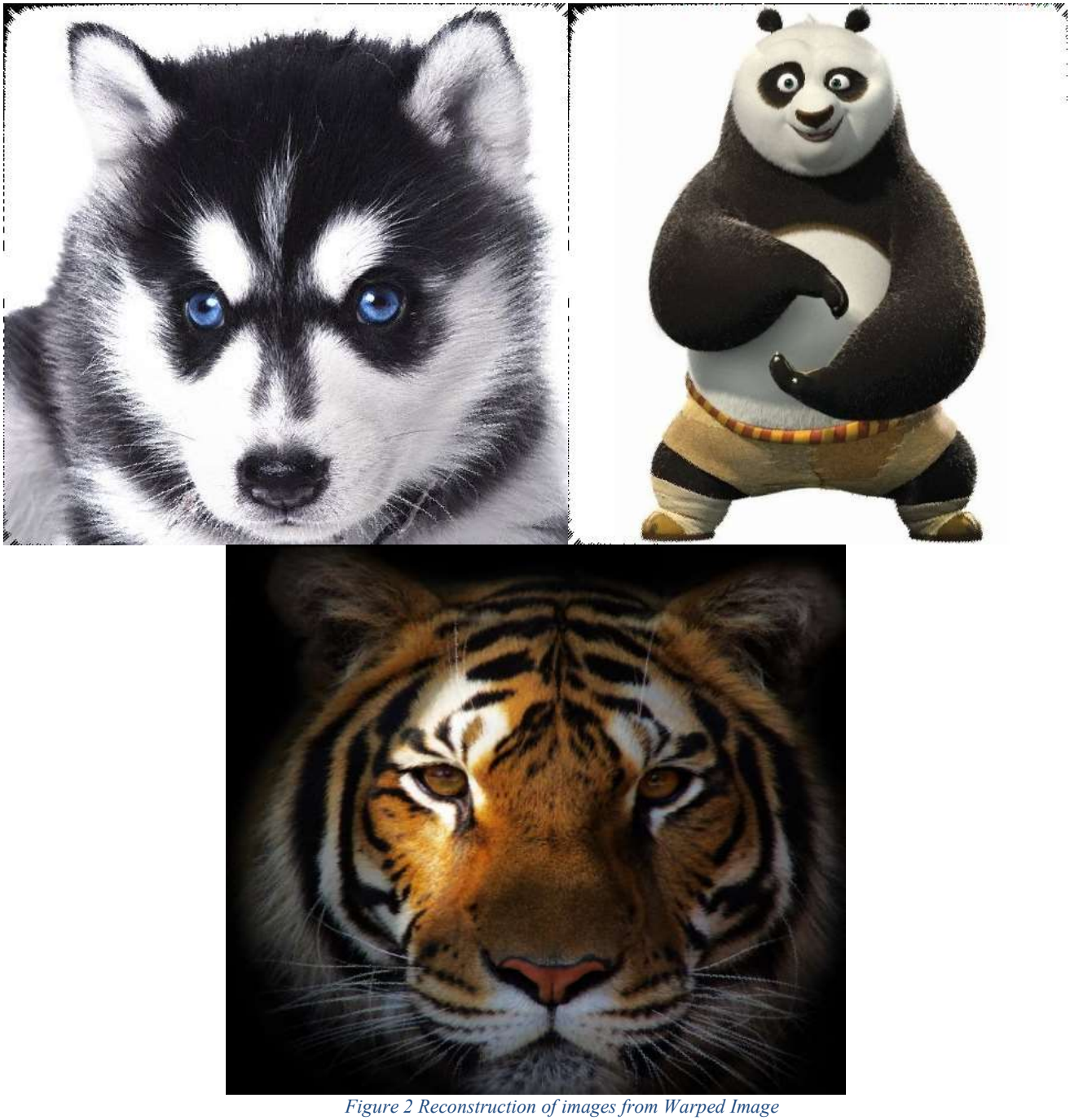

Figure 2 Reconstruction of images from Warped Image

B.

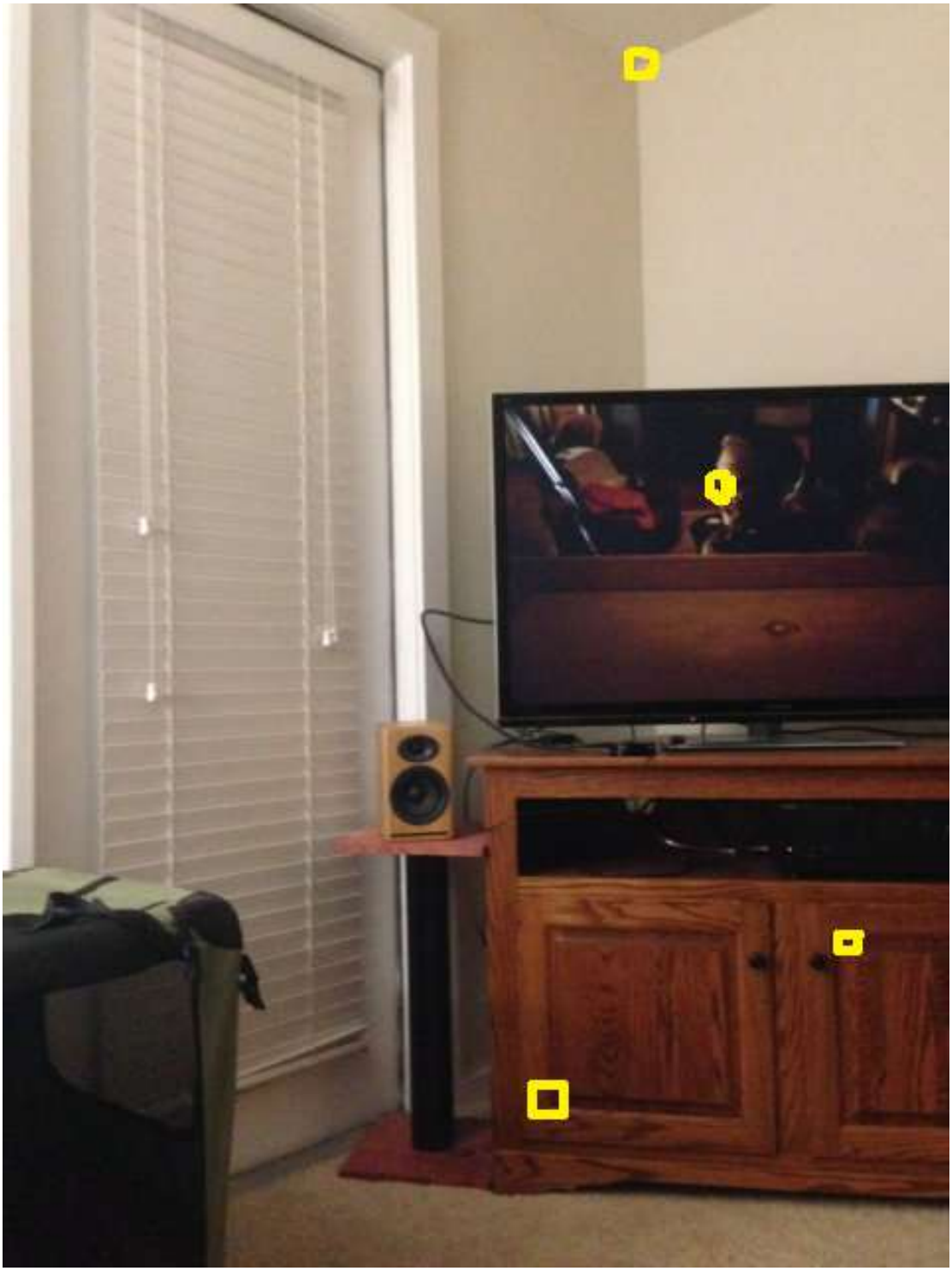

Figure 3Control points in left.raw

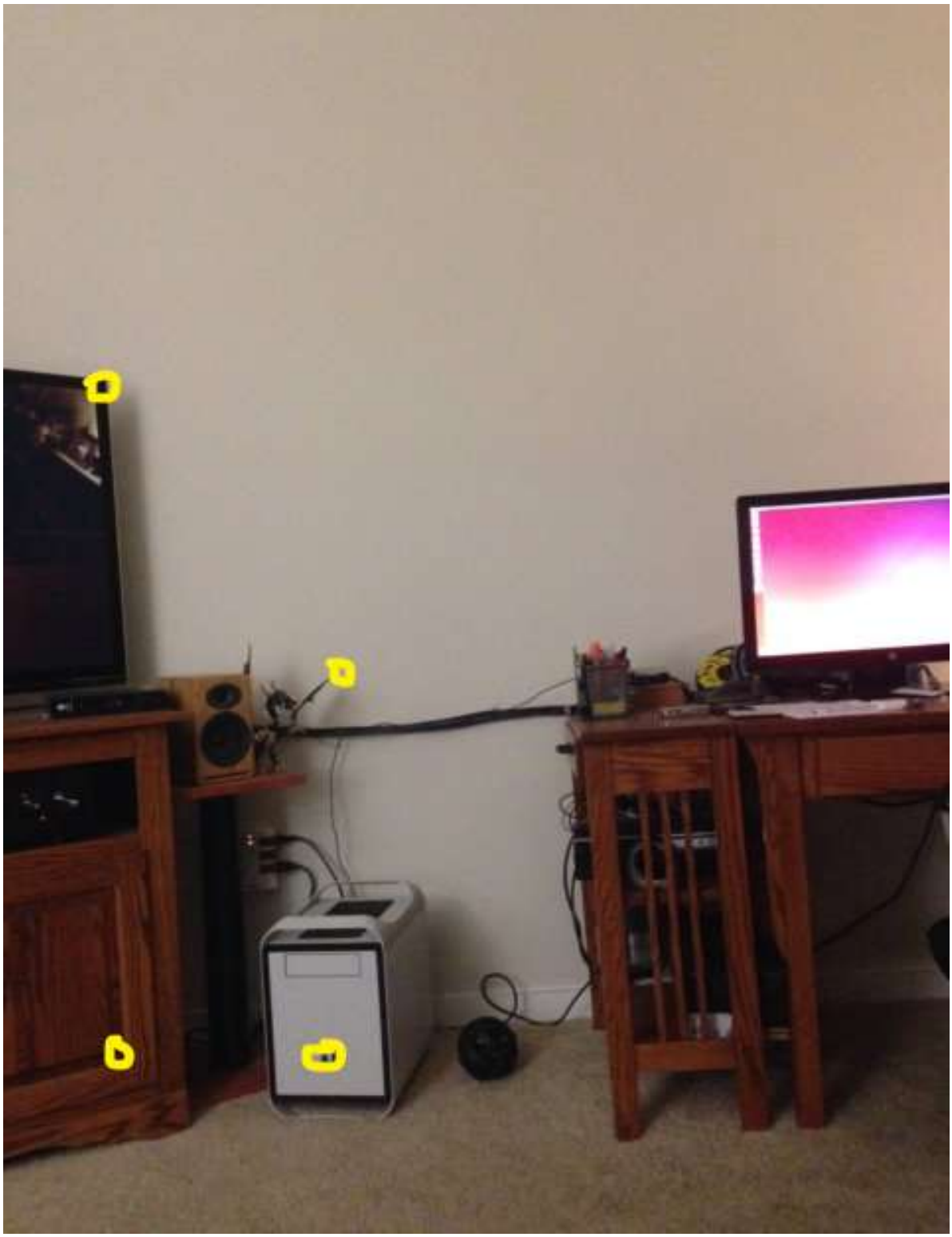

Figure 4 Control Points in right.raw

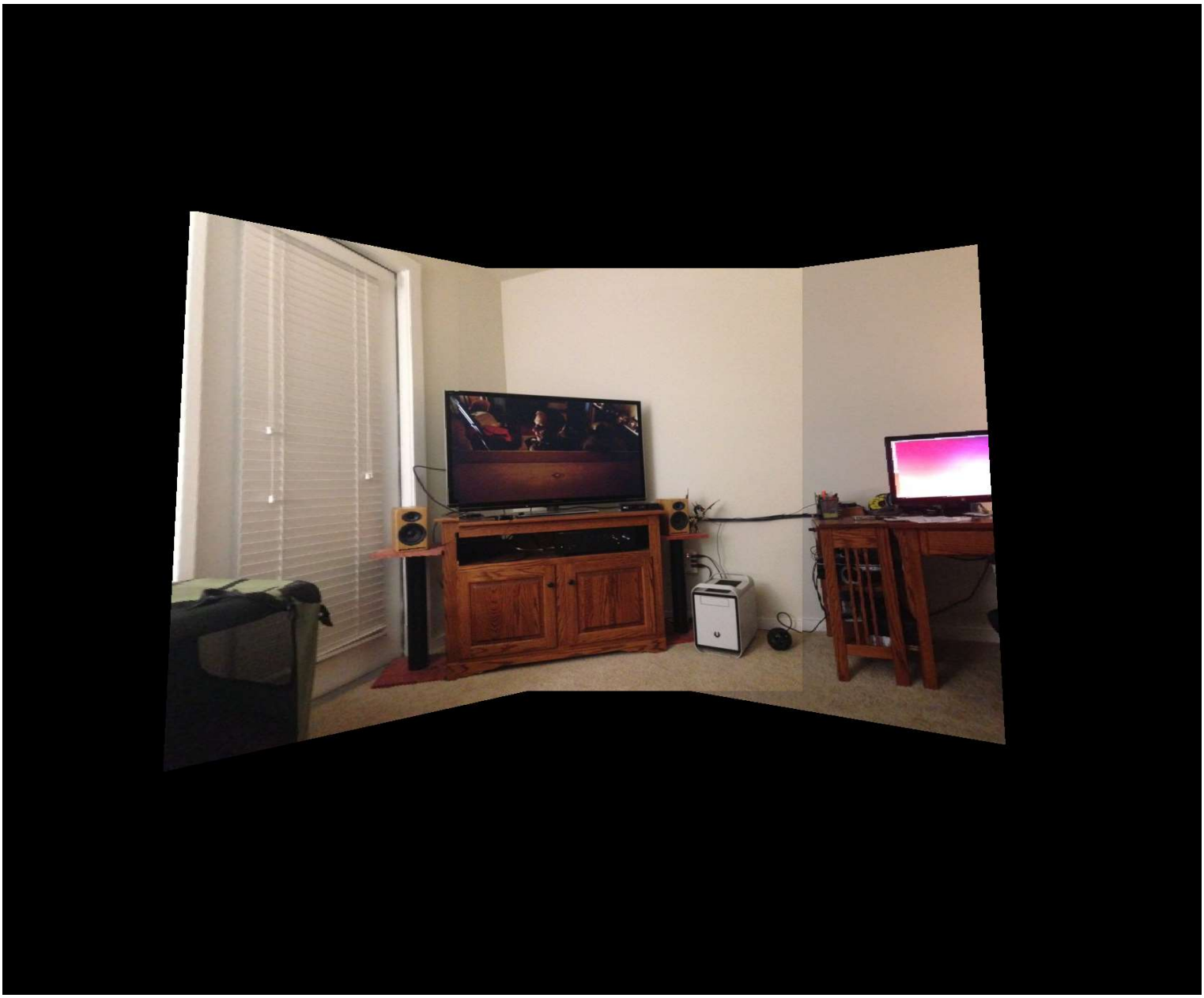

Figure 5 Panorama Image

1.4 Discussion

A.

- The mapping used in warping algorithm is one to one. When we compare the original image to the reconstructed image we see that the shape of the output image is distorted as a lot of original pixels

are lost due to under sampling the original image. It is advisable to perform resampling using correct filtering method to avoid artifacts. Practically, low pass filter is used to resample.

- We also use global transformation which uses a single mapping function on the whole image. While, the image might have local distortions that differ at different location. In that case, local transformation might be a better option to get different mapping function to different parts of the image.

B.

- We used 4 control points to find the homography matrix. We can more than four points but that may or may not improve the accuracy. Small difference cannot always be visually seen but they make enormous difference in the result. Like, when there are three control points almost line up in target and reference images they cause mathematical instability. But, this problem does not occur when we choose four points.

- The method that can be implemented to select the control points can be starting with four points which are spaced as widely as possible and adding one additional point. Then try the transformation if it improves or get worse. Stop adding points there is no visual improvement. Also, tried using control points nearer to one another but the transformation did not overlap properly with the middle image, thus the homography matrix was not up to the mark.

2.Problem 2

2.1Abstract and Motivation

Dithering is a thresholding technique where a color or grayscale image is converted into a cluster pixel of binary pixels. It is applied form of noise used to randomize quantization error, preventing large scale patterns [2].

Printing an image with high color information on a limited color screen can cause problems. The original image may contain details of an object that would be replaced with blobs of uniform color if dithering is not performed. Dithering is performed to get the best quality in the machines which support less color.

Halftoning is a technique to simulate continuous tone imagery using dots which vary in size and spacing. Error diffusion is a halftoning technique in which the quantization residual which is generated when

converting multi-level image into a binary image is distributed to neighboring pixels that have been not been processed [3].

2.2Approach and Procedures

There are three parts in this section. The methodology applied for dithering using Fixed Thresholding, random thresholding and dithering matrix is described in Section 2.2.1 in A.B and C respectively. The method of error diffusion using Floyd-Steinberg Error diffusion with serpentine scanning, Jarivs Judice and Ninke and Stucki is described in Section 2.2.2 in A, B and C respectively. The method of color halftoning with error diffusion using separable error diffusion and MBVQ based error diffusion is described in Section 2.2.3 in A and B respectively.

2.2.1 Dithering

Fixed Thresholding also known as average dithering, each pixel value is compared against a fixed value. This may be the simplest dithering algorithm but it results in immense loss of detail and contouring.

Random dithering was the first attempt to overcome the drawbacks of fixed thresholding. Each pixel value is compared against a random threshold resulting in staticky image. Although this method does not generate patterned artifacts, the noise swamps the detail of the image [4].

Ordered dithering dithers using dither matrix. For every pixel in the image the value of the pattern at the corresponding location is used as a threshold. Neighboring pixels do not affect each other making suitable for animations. Different patterns can generate completely different dithering effects. Though simple to implement, this dithering algorithm is not easily changed to work with free form arbitrary palettes. The Bayer index matrices that are square matrices with the length of sides as power as 2. We implemented 2×2, 4×4 and 8×8 Bayer index matrices individually and compare the resulting dithered image.

Bayer index matrixes were calculated using recursion formula. The Bayer matrices were obtained:

$$I2 = \begin{bmatrix} 0 & 2 \\ 3 & 1 \end{bmatrix} \qquad I4 = \begin{bmatrix} 0 & 8 & 2 & 10 \\ 12 & 4 & 14 & 6 \\ 3 & 11 & 1 & 9 \\ 15 & 7 & 13 & 5 \end{bmatrix} \qquad I8 = \begin{bmatrix} 0 & 32 & 8 & 40 & 2 & 34 & 10 & 42 \\ 48 & 16 & 56 & 24 & 50 & 18 & 58 & 26 \\ 12 & 44 & 4 & 36 & 14 & 46 & 6 & 38 \\ 60 & 28 & 52 & 20 & 62 & 30 & 54 & 22 \\ 3 & 35 & 11 & 43 & 1 & 33 & 9 & 41 \\ 51 & 19 & 59 & 27 & 49 & 17 & 57 & 25 \\ 15 & 47 & 7 & 39 & 13 & 45 & 5 & 37 \\ 63 & 31 & 55 & 23 & 61 & 29 & 53 & 21 \end{bmatrix}$$

We implement the below formula on every row and column:

$$T(x,y) = (I(x,y) + 0.5\ )/N^2$$

where T is the threshold matrix value and N^2 denotes the number of pixels in the matrix. Then we normalize every pixel value by dividing it by 255. We check whether this pixel is greater than $T(modN, jmodN)$ then we assign it 255 otherwise we assign it 0.

In order to quantize the image to 4 quantization levels, we use the same concept as the above but for thresholding we use the following constraints:

1. $0 \leq pixelValue \leq \frac{T(row\%size, column\%size)}{2}\ \ then\ assign\ 0$
2. $\frac{T(row\%size, column\%size)}{2} < pixelValue \leq T(row\%size, column\%size)\ then\ assign\ 85$
3. $T(row\%size, column\%size) < pixelValue \leq 0.5 + T(row\%size, column\%size) then\ assign\ 170$
4. $0.5 + T(row\%size, column\%size) < pixelValue \leq 1\ then\ assign\ 255$

2.2.2 Error Diffusion

Error diffusion dithering is a feedback process that diffuses the quantization error to neighboring pixels. The quantization error in above dithering methods results in blobs of uniform color if they are not treated properly. To avoid such losses, we diffuse the error in neighboring pixels, thus the process is called error diffusion.

To implement the Floyd Steinberg's error diffusion, we firstly extend our image. We compare the pixel value against a threshold which we assigned as 127 and if it is less than the threshold, we asssign it 0 otherwise 255. We find the quantization error by subtracting old pixel value by new pixel value. Then for every pixels, we use the error diffusion matrix is used taking in consideration the quantization error and affecting the future pixel according to the error diffusion matrix.

We then loop through the pixel values in serpentine manner. This is done by scanning in every alternate row. Firstly, for even rows and then for odd rows. For odd rows, the scanning is done from right to left while for even rows it is done from left to right.

2.2.3 Separable Error Diffusion

Error diffusing each of three color planes independently is separable error diffusion. It is based on certain characterstics of human color perception.In order to carry it out, we firstly convert red channel to cyan, green to magenta and blue to yellow. After that, we perform floyd error diffusion on each channel. Then,

we saving each color channel we convert it back to red, green and blue channel. Then save the output image.

2.3Experimental Results

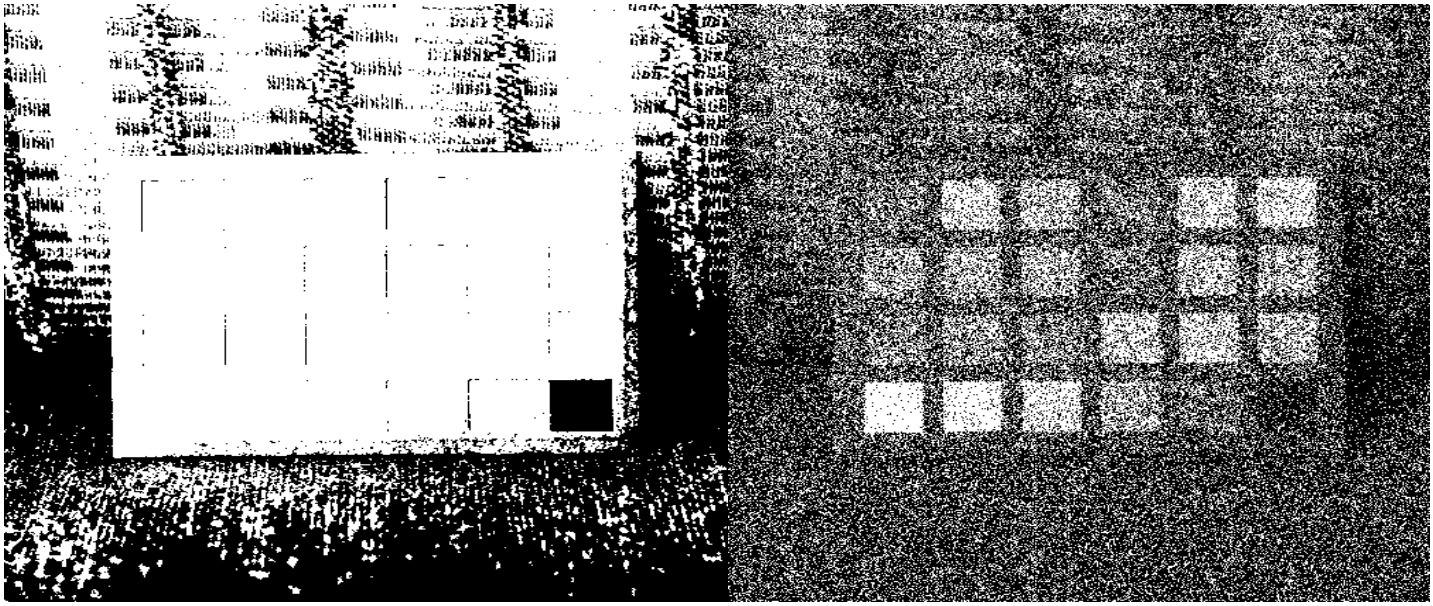

Figure 6 Fixed Thresholding and Random Threshold

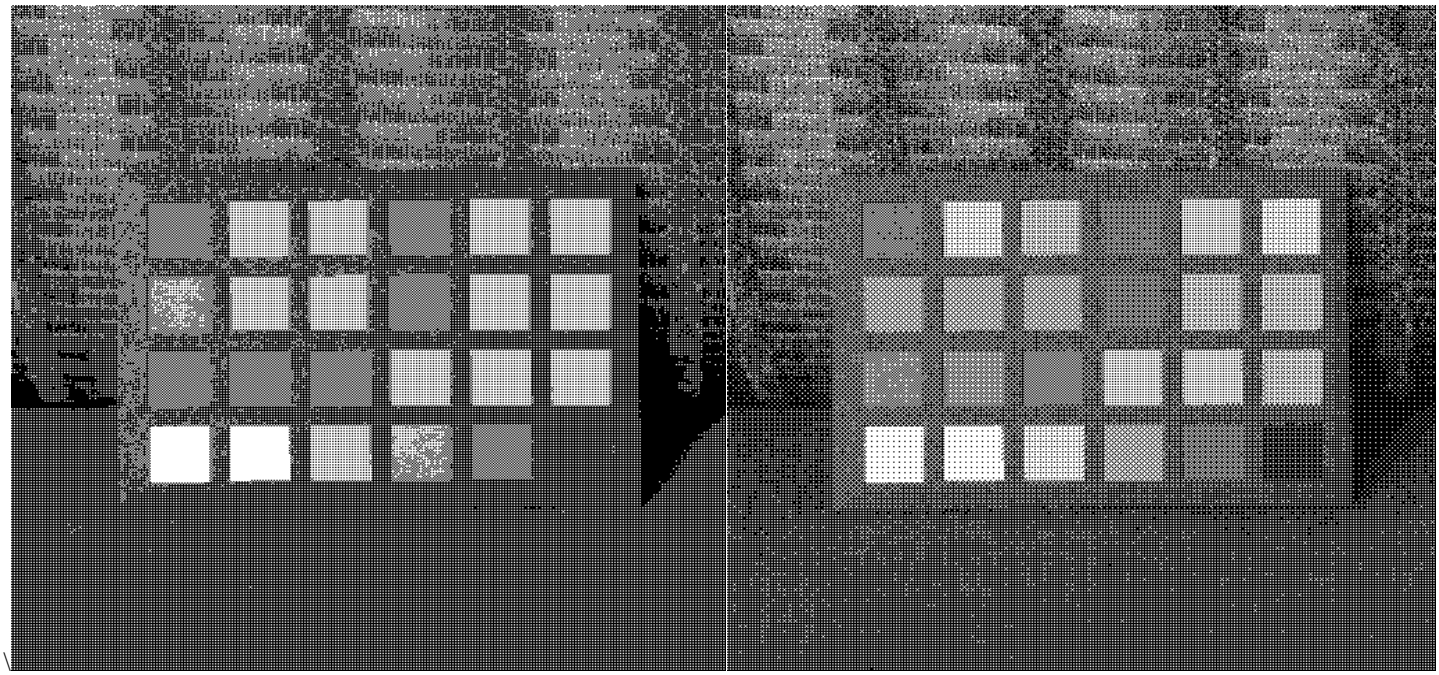

Figure 7 Halftoned image using I2 and I4 Bayer matrix

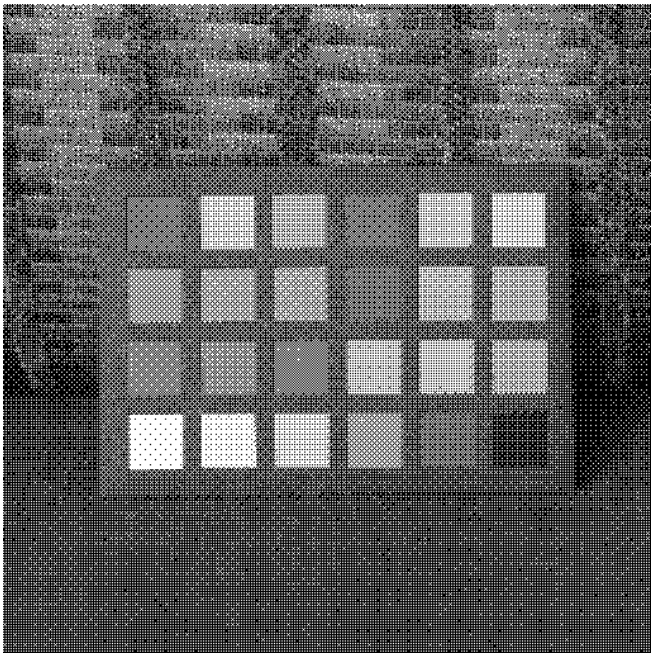

Figure 8 Halftoned image using I8 Bayer matrix

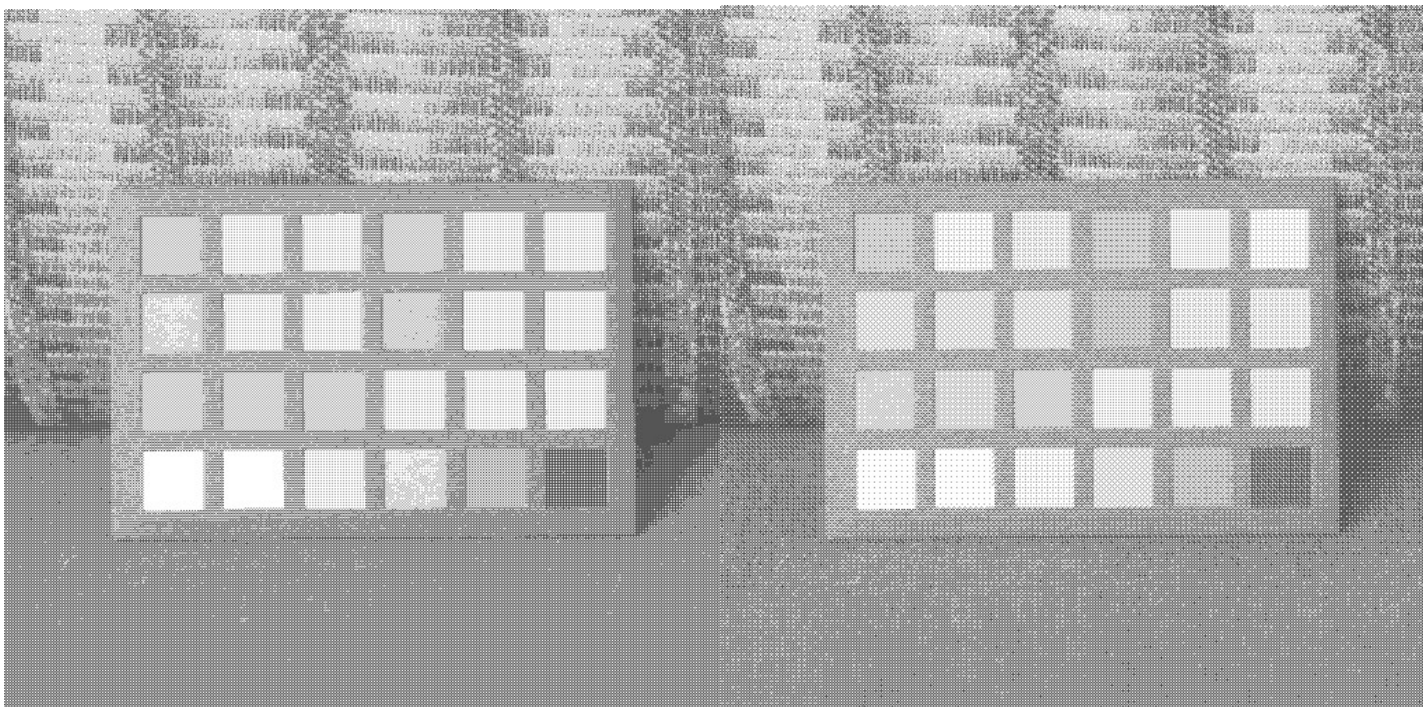

Figure 9 Halftoned Gray Level Image using I2 and I4 respectively

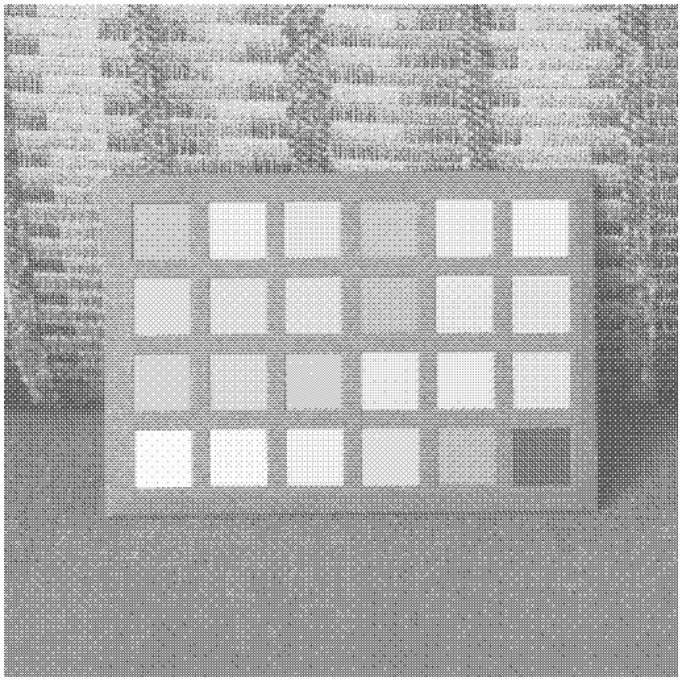

Figure 10 Halftoned Gray Level Image using I8

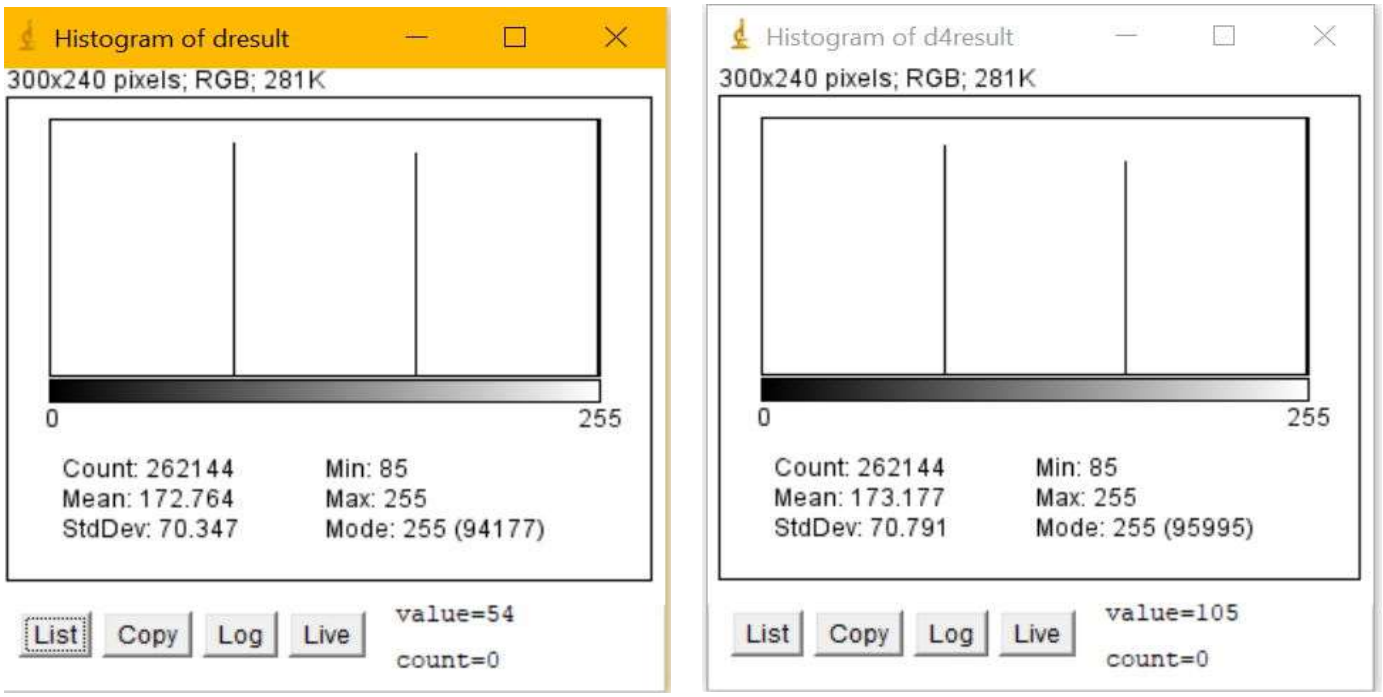

Figure 11Histogram representing 4 intensity levels when I2, I4 is used respectively

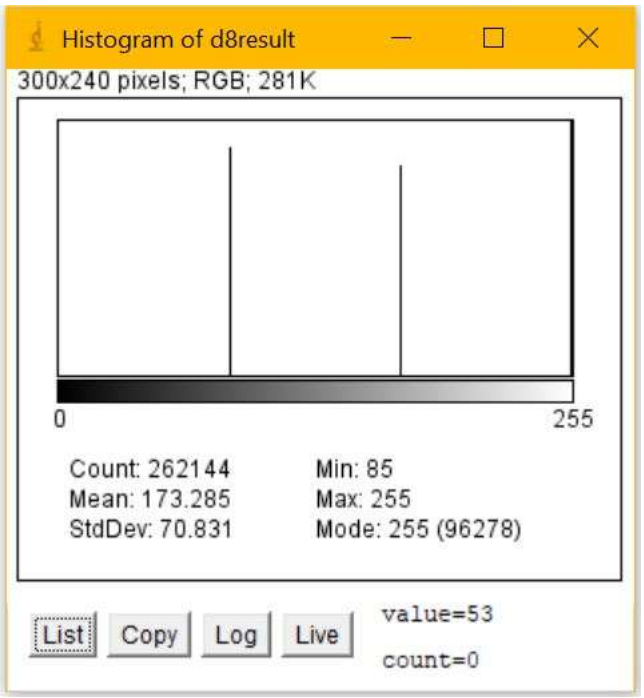

Figure 10Histogram representing 4 intensity levels when I8 is used

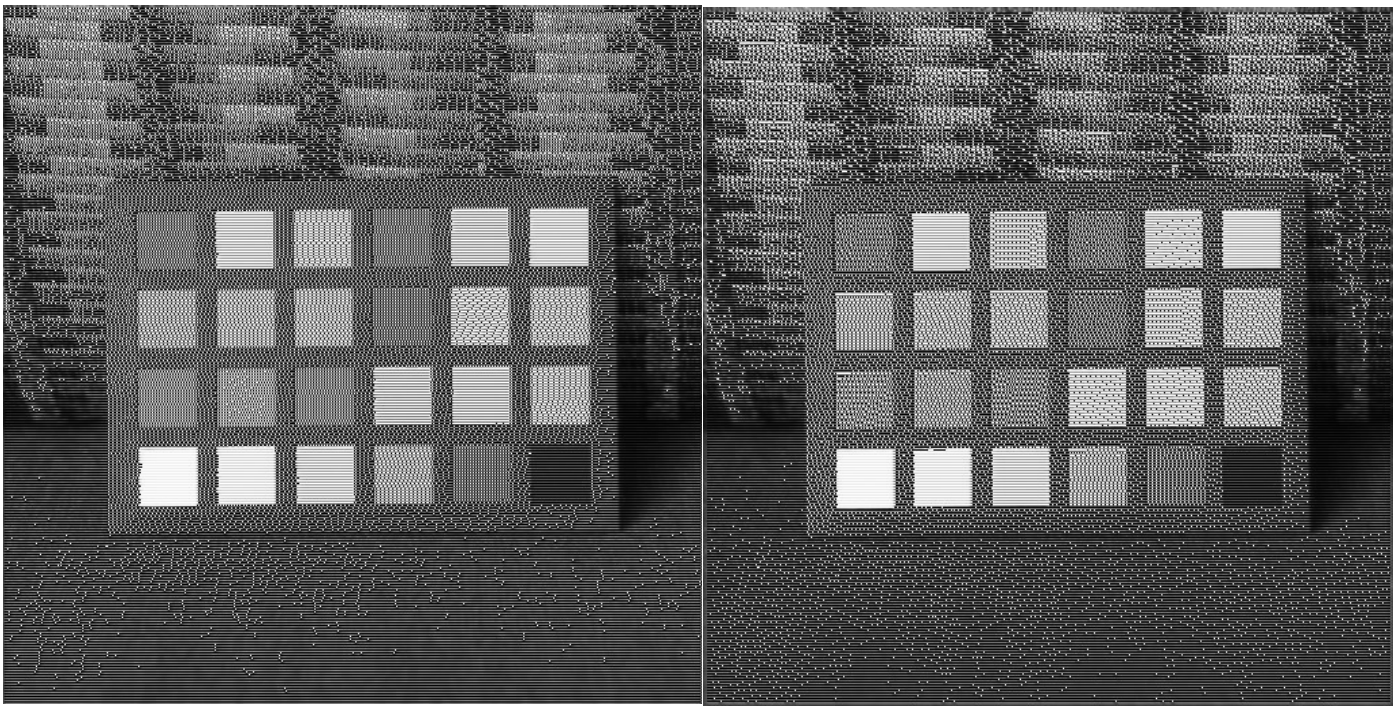

Figure 11Error Diffusion using Floyd and Jarvis diffusion matrix

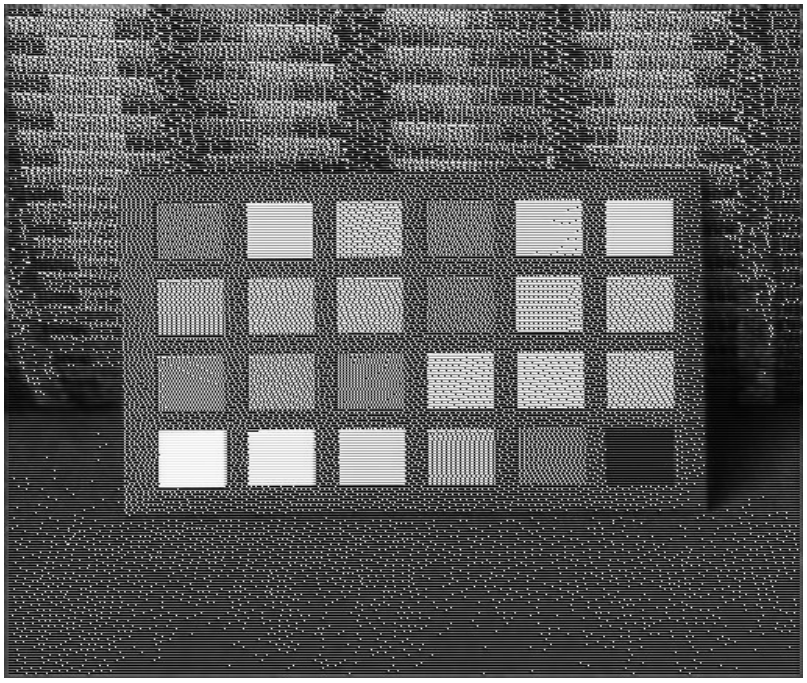

Figure 12 Error Diffusion using Stucki diffusion matrix

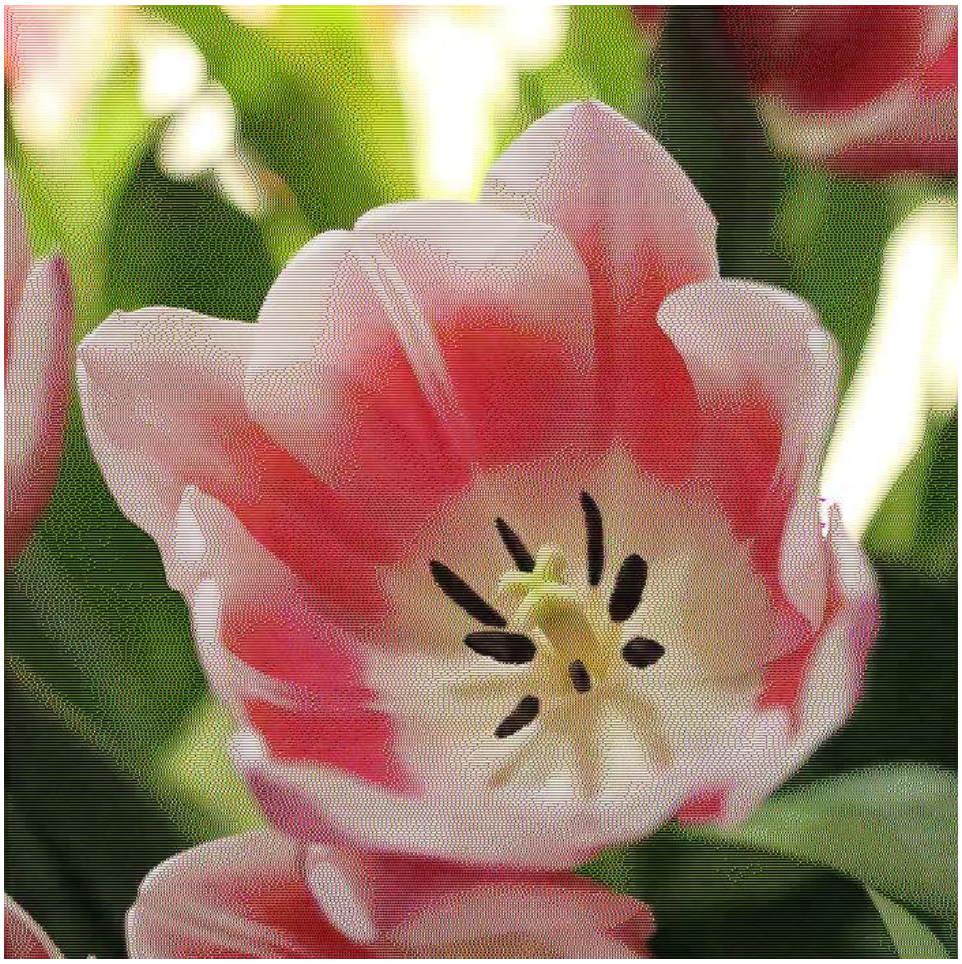

Figure 13 Color Halftoning with Error Diffusion

2.4 Discussion

- We can observe that Bayer matrix of different sizes distribute the quantization error. It produces cross hatch pattern artifacts when closely observed. This is one of the drawbacks of ordered dithering. Apart from this drawback, its main advantage is its speed and ease of implementation. Additionally, it does not introduce any spatial distortion which is a big problem with dithering techniques.

- When the image is quantized to 4 levels, it looks more enhanced and the corner information is lost.

- Error diffusion gives better result as it disperses the quantization error over a large area. Floyd- Steinberg dithering only diffuses the error to neighboring pixels. Thus, resulting in very fine grained dithering. In case of Jarvis, Judoce and Ninke dithering diffuses the error to pixels one step further away.The dithering is coarser but has fewer visual artifacts. It is slower than Floyd Steinberg dithering because it distributes the error among 23 nearby pixels instead of 4 like in Floyd. Stucki dithering also does the same as Jarvis but is slightly faster. Its output tends to be clean and sharp. Thus, Stucki is more preferred out of the other day.

- The main problem faced by dithering algorithm is artifacts such as band of speckles in areas with uniform values that forms grainly images. In order to reduce that, a method that propogates only a part

of quantization error to its neighbors. It would preserve the detail well but might not work very well with low and high intensity values. I also suggest to use more error diffusion matrices instead of just one.

- Separable error diffusion's output is of higher quality as compared to separable channel diffusion in RGB. It does not requires no additional memory and entails a reasonable increase in run time.

3.Problem 3

3.1Abstract and Motivation

Morphological image processing is a collection of non linear operations which relates to the shape or morphology of features in an image. Morphological operations rely on relative ordering of pixel values and not on their numerical values[5]. Morphological techniques probe an image with a small shape or template called a structuring element. The structuring element is positioned at all possible locations in the image and it is compared with the corresponding neighborhood of pixels. Some operations tesst whether the element fits within the nighborhood while others test whether it hits or intersects the neighborhood.

In this problem, we will implement shrinking to find the number of stars, count the different stars sizes ans their frequency. Morphological operations are extensively used in shape classifcation. To extract the shape information of objects in an image, a method for representing shape is required. We use thinning and skeletonizing to represent the shape of jigsaw puzzle.

In the problem, we make an attemplt to use geometrical operations like rotation and flipping to find the unique number of jigsaw pieces which are not identical to each other. Also, find the total number of pieces given in the board game. For the count, a method called connected component analysis is used which will be explained further.

3.2Approach and Procedures

There are four parts in this section. The methodology implementing morphological operation such as shrinking is applied in part A, thinning in part B, skeletonizing in part C and counting game in part D. applied for dithering using Fixed Thresholding, random thresholding and dithering matrix is described.

3.2.1 Morphological Processing

A. Shrinking

It is one of the basic operators in mathematical morphology. The basic effect on the operator on a binary image is to erode the boundary regions of foreground pixels. The area of the foreground pixels shrinks in size and holes within those areas become larger. We firstly convert the grayscale image to binary image by using a threshold which we took as 127. If the pixel value is greater than 127, it is assigned pixel value 1 otherwise 0. We then use pattern matching tables for conditioning shrinking. If the pixel value is 0, we don't compare the neighboring pixel values with the mask, we directly assign it 0. If the pixel value is 1, we take the neighboring pixels. Using the neighboring pixel, we create the bond value using:

$$\begin{bmatrix} pixelvalue & 2 * pixelvalue & pixelvalue \\ 2 * pixelvalue & Our\ pixel\ Value & 2 * pixelvalue \\ pixelvalue & 2 * pixelvalue & pixelvalue \end{bmatrix}$$

$$\text{Bond} = \text{Values} + \text{Values}$$

and compare it to the masks in the conditional pattern associated with the bond value. If it is identical to any mask, it is a hit and we assign it 1 otherwise if it is a miss we assign it to 0.

Then, we use the image frame created using conditional mask, check if the pixel value is 0 then we assign it the original pixel value. If it is one, we check if the neighboring pixels match the unconditional mask, if it is a hit we assign the value original image had. If not, we assign it 0 value.

We assign the output pixel value to the initial frame and carry out the same computation for many iterations.

To count the frequency of star size, we calculate the number of white pixel (pixel value =255) we get in an iteration. The number of iterations a star takes to get into a single white pixel denotes its size. For every iteration, we get the number of stars. We subtract the old count with new count because it represents the new star for that iteration.

B. Thinning

We carry out the same procedure as used in shrinking only the mask pattern for conditional and unconditional will change.

C. Skeletonizing

We carry out the same procedure as used in shrinking only the mask pattern for conditional and unconditional will change.

D. Counting Game

To find out the number of jigsaw pieces in the image. We firstly convert the image to binary using a threshold. We consider the threshold value to be 127. Then we invert the image to make the foreground pixel background and vice versa. It is easier to visualize as we are used to normally considering background as 0-pixel value and 255 as foreground.

Then we use connected component analysis to find out the total number of structures in our image. It scans an image and groups its pixels into components based on pixel connectivity that is all pixels which share similar pixel intensity values are grouped together in some way.

The algorithm for carrying out connected component analysis would be:

1. Start scanning an image by moving along a row until a point has pixel intensity 1.
2. If it has pixel intensity 1, examine its four neighbors in the matrix which stores the label values.
3. If all four neighbors are 0, assign it a new label, else
4. If one neighbor has a pixel value, assign it that, else
5. If more than one members has a label, assign it the lowest label and store the equivalence in a table.

After completing the scan, the second pass each label is replaced by the label assigned to its equivalence classes.

To calculate the number of unique piece, we firstly skeletonize the binary image that we form by thresholding the pixel value. Then we skeletonize the image. We get the boundary of the outline of every board piece. We then need to apply the connected component analysis on the skeletonized image. We have the pixel frequency for every label. Then label with unique frequency represents the number of unique board pieces in the board image.

3.3Experimental Results

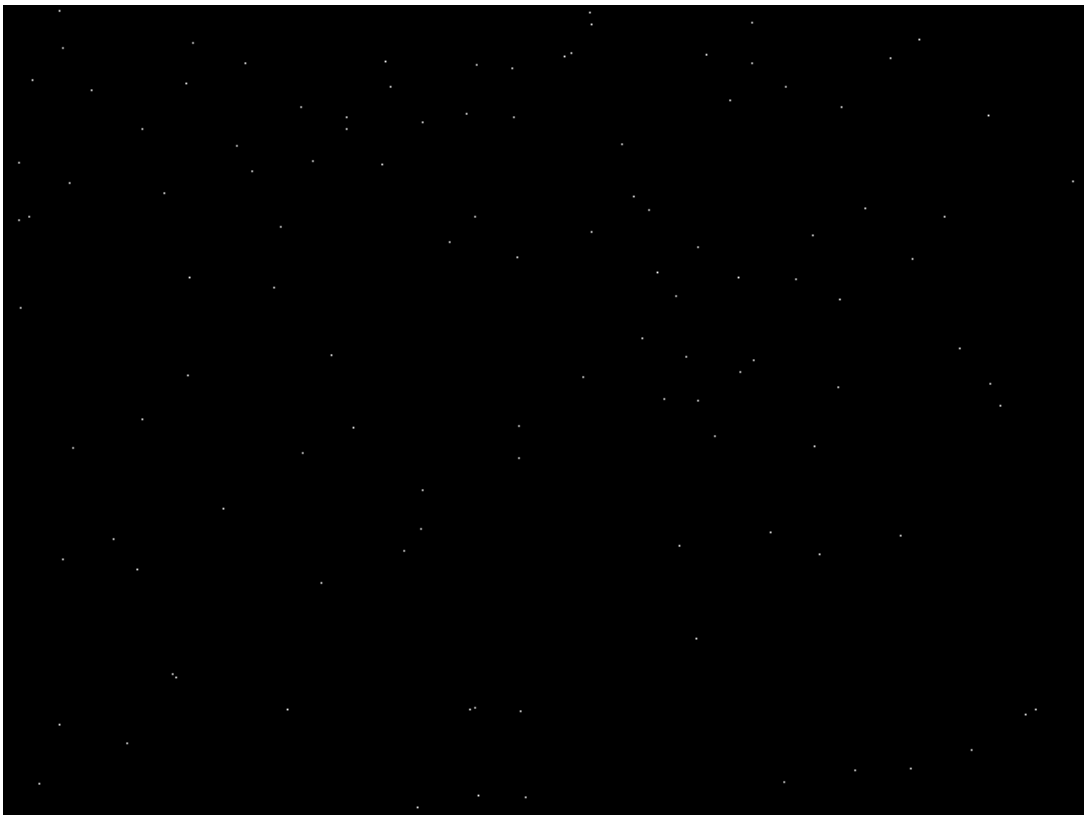

Figure 14 Total number of stars in stars.raw

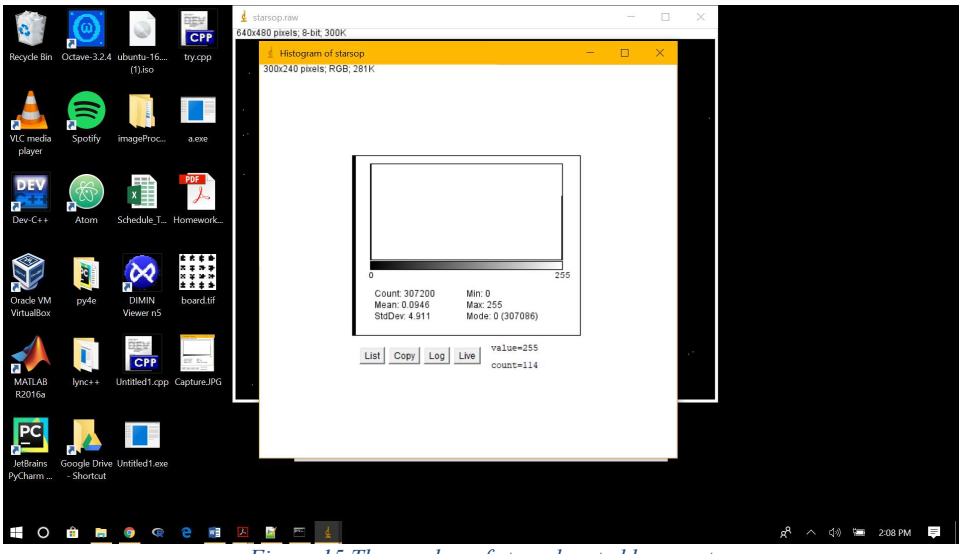

Figure 15 The number of stars denoted by count

```
The number of stars for1is15
The number of stars for2is41
The number of stars for3is17
The number of stars for4is15
The number of stars for5is12
The number of stars for6is6
The number of stars for7is3
The number of stars for8is1
The number of stars for9is2
The number of stars for10is0
The number of stars for11is2
The number of stars for12is0
Total number of stars114
```

Figure 16 Size and frequency of Star along with total number of stars

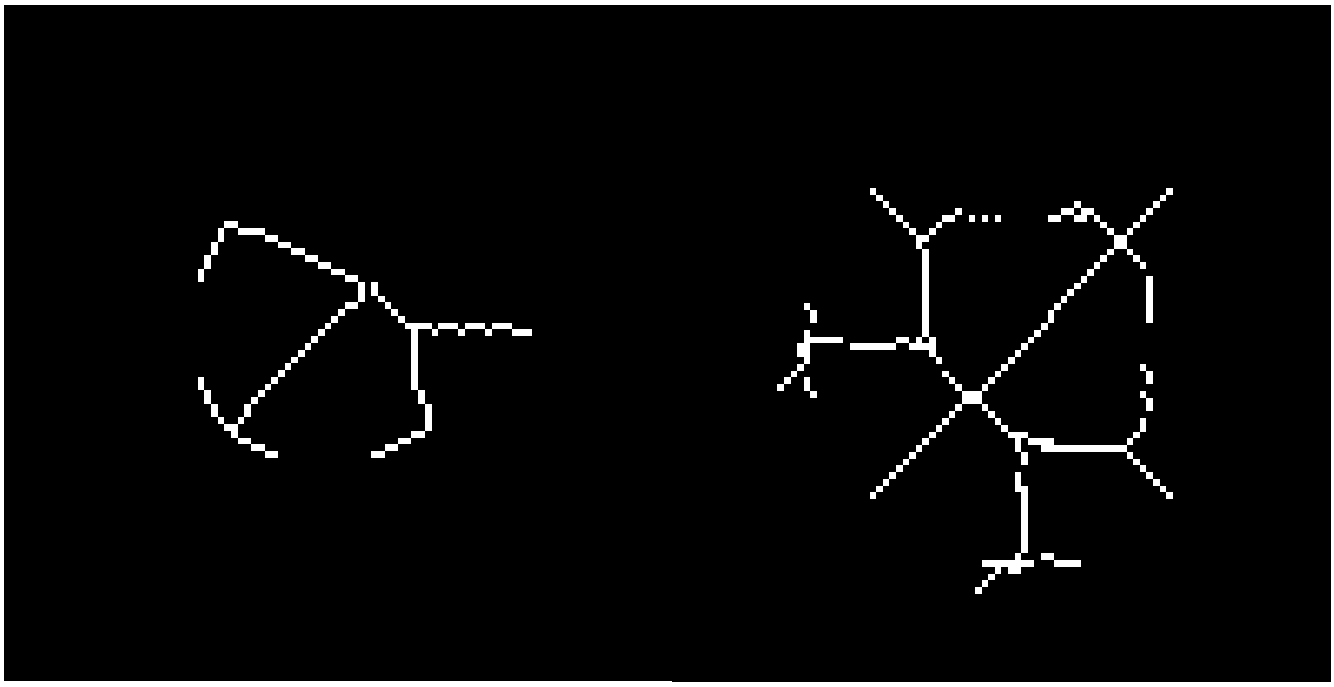

Figure 17 Thinning on jigsaw_1.raw and Skeletonizing on jigsaw_2.raw

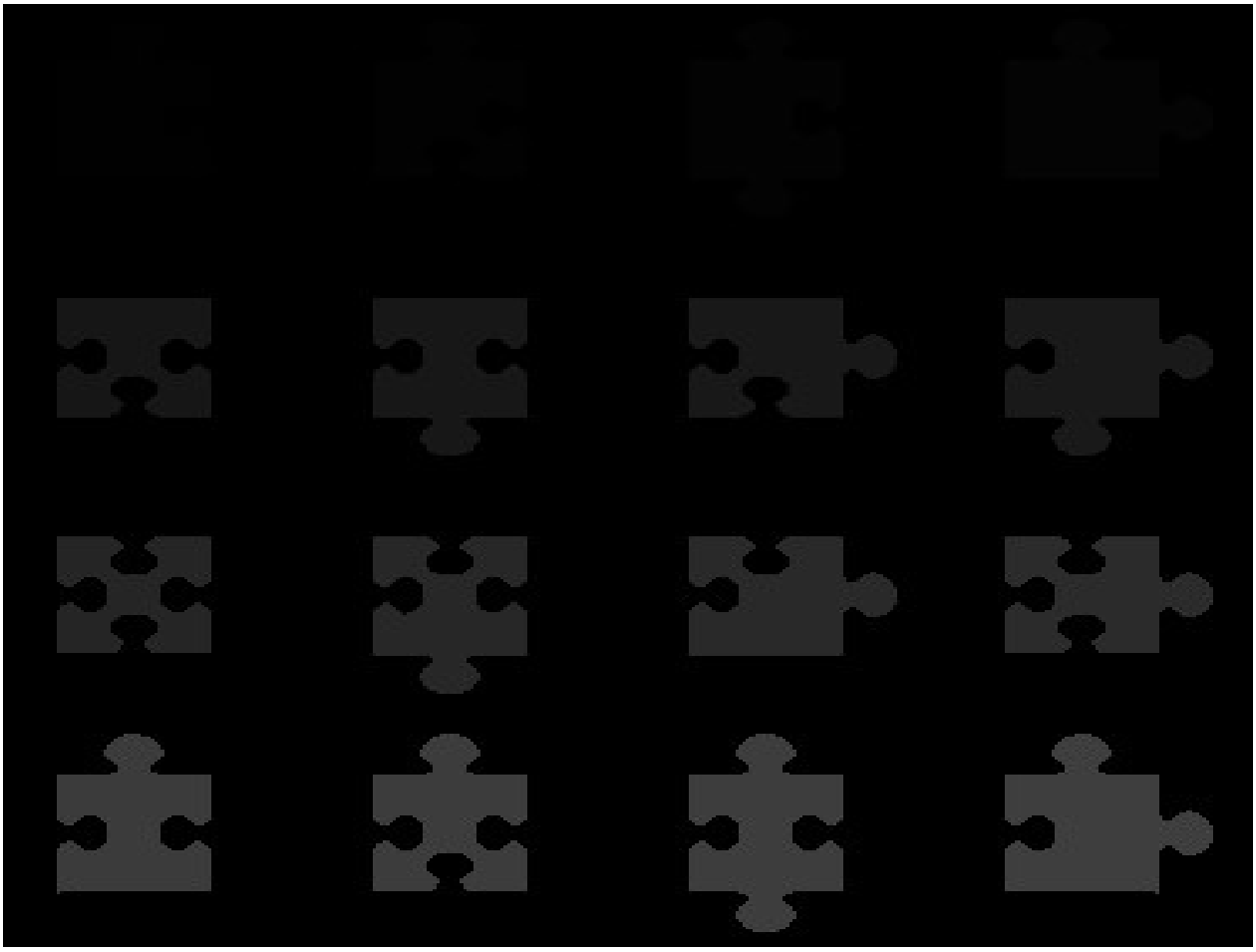

Figure 18 No of board pieces in board.raw

Figure 19 Histogram of different label in figure 11

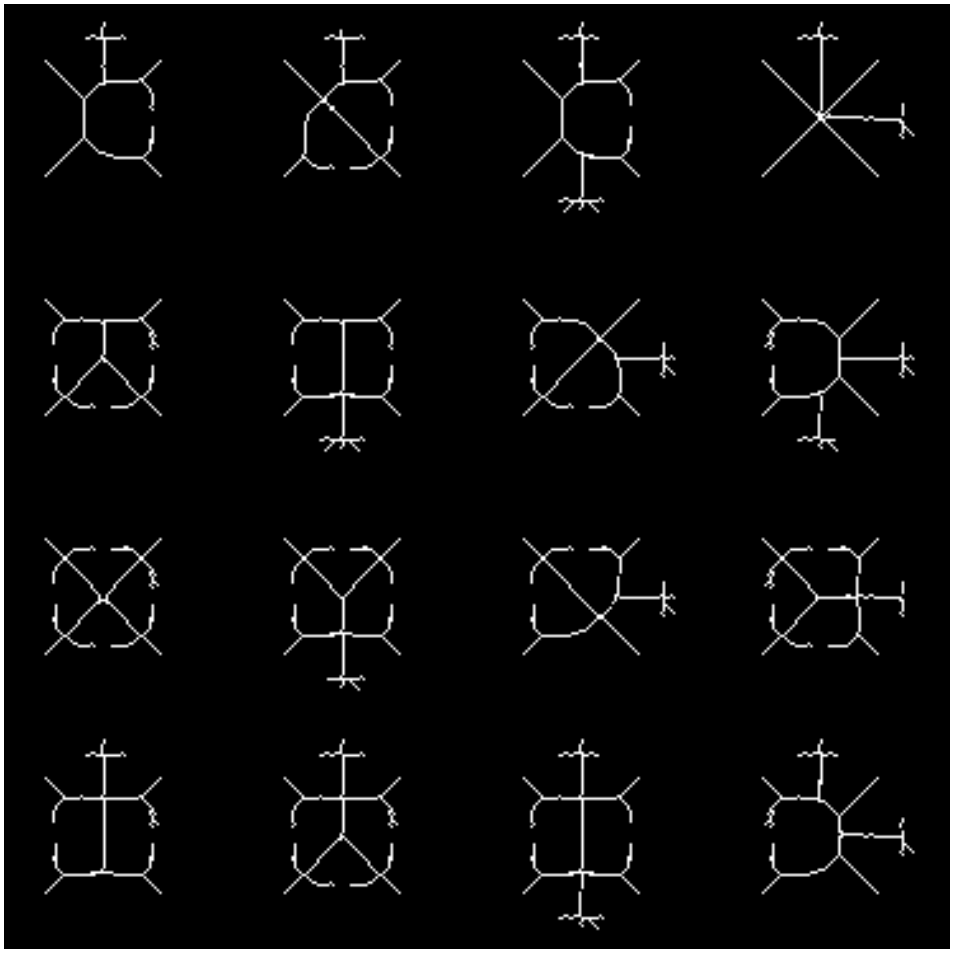

Figure 20 Skeletonized image for unique pieces

3.4 Discussion

- The number of stars can be counted by observing the histogram and checking the value of 255-pixel intensity. It comes out to be 114 as seen in figure 15. The count of star sizes is represented in Figure 16.
- We can also observe that less iterations are required for simpler geometry while more iterations are required for complex geometry.
- Thinning and skeletonizing are applied and result is shown in Figure 17.
- Using the above figures, we can deduce that shrinking reduces objects in binary image to a single point located at the geometric center of the object. This will be helpful to apply this method for finding the number of unique objects in an image.
- Thinning generates minimally connected lines which helps in defining the structure of the object. Skeletonizing is like thinning but it preserves more internal information of the structure than thinning.
- The number of board pieces are 16. We can verify that by checking the histogram of the image that we get after performing connected component analysis. Figure 19 represents the image with the label. As it is values like 1,2,3 etc. It is not visible in the image.

- The number of unique pieces come about to be 10. The skeletonized image as shown in Figure 20 is then used for doing connected component analysis to find the unique pieces.

4 References

[1] http://eeweb.poly.edu/~yao/EL5123/lecture12_ImageWarping.pdf

[2] https://en.wikipedia.org/wiki/Dither

[3] https://en.wikipedia.org/wiki/Error_diffusion

[4] https://brucebcampbell.files.wordpress.com/2013/04/dithering-methods.pdf

[5] https://www.cs.auckland.ac.nz/courses/compsci773s1c/lectures/ImageProcessing-html/topic4.htm