EE569 Digital Image Processing

Spring 2018

Assignment 2

Name: Thiyagarajan Ramanathan

USC ID: 4973341255

Issue Date: 02/04/2017 Due Date: 03/03/2018

Problem 1: Geometric Image Modification (30%)

In this problem, you will need to apply geometric modification and spatial warping techniques to do some interesting image processing tricks. Please note that during these operations, you may need to solve some linear equations to get the matrix parameters.

(a) Geometrical Warping (Basic: 15%)

Design and implement a spatial warping technique that transforms an input square image into an output image of a disk-shaped image. An example is given in Figure 1.

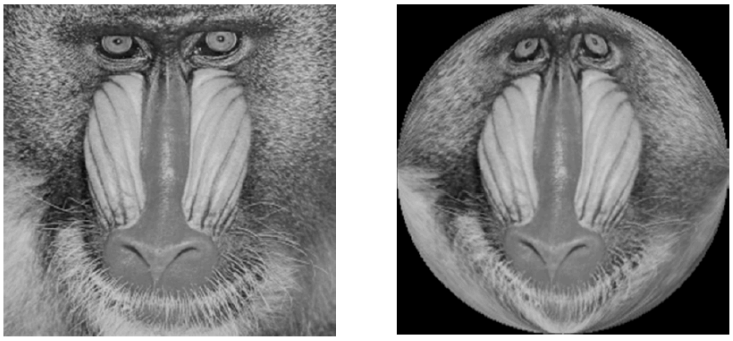


Figure 1: Warp the original image to a disk-shaped image

The wrapped image should satisfy the following three requirements:

- Pixels that lie on boundaries of the square should still lie on the boundaries of the circle.
- The center of original images should be mapped to the center of warped images.
- The mapping should be reversible, i.e. it is a one-to-one mapping.

Apply the same developed spatial warping algorithm to *puppy*, *tiger* and *panda* images in Figure 2. Please first describe your approach as clearly as possible and show the resulting images.

Next, apply the reverse spatial warping to each warped image to recover its original image. Compare the recovered square image with the original square image. Is there any difference between two images? If any, explain sources of distortion in detail.

Figure 2: The puppy, tiger and panda images

(b) Homographic Transformation and Image Stitching (Advanced: 15%)

One can use homographic transformation and image stitching techniques to create panorama that consisting of multiple images. One example (Taken from matlab examples [1]) is shown in Figure. 3. The left image were taken with an uncalibrated smart phone camera by sweeping the camera from left to right along the horizon to capture all parts of the building. The right panorama is the desired output by stitching transformed images.

This example involves five images to composite. However, the basic principle is to process in terms of consecutive pair of images. It could be achieved by following these steps:

- Select control points from both images. You are allowed to do it manually for this step.
- Apply homographic transformation to find a homography mapping (described below).
- Wrap one image onto the other using the estimated transformation.
- Create a new image big enough to hold the panorama and composite the wrapped image into it. You can composite by simply averaging the pixel values where the two images overlap.

The homographic transformation procedure is stated below. Images of points in a plane, from two different camera viewpoints, under perspective projection (pin hole camera models) are related by a homography:

$$P_2 = HP_1$$

where H is a 3x3 homographic transformation matrix, P_1 and P_2 denote the corresponding image points in homogeneous coordinates before and after the transform, respectively. Specifically, we have

$$\begin{bmatrix} x_2' \\ y_2' \\ w_2' \end{bmatrix} = \begin{bmatrix} H_{11} & H_{12} & H_{13} \\ H_{21} & H_{22} & H_{23} \\ H_{31} & H_{32} & H_{33} \end{bmatrix} \begin{bmatrix} x_1 \\ y_1 \\ 1 \end{bmatrix} \quad and \quad \begin{bmatrix} x_2 \\ y_2 \end{bmatrix} = \begin{bmatrix} \frac{x_2'}{w_2'} \\ \frac{y_2'}{w_2'} \end{bmatrix}$$

To estimate matrix H, you can proceed with the following steps:

- Fix $H_{33} = 1$ so that there are only 8 parameters to be determined.

- Select four point pairs in two images to build eight linear equations.
- Solve the equations to get the 8 parameters of matrix H.
- After you determine matrix H, you can project all points from one image to another by following the backward mapping procedure and applying the interpolation technique.

Implement above homographic transformation and stitching techniques to composite the *room* images in Figure 4. Show the results and make discussion on the following questions.

1) How many control points were used? What if we use more than four control points?
2) How did you select control points? Clearly specify your observations and strategies.

Implement above homographic transformation and stitching techniques to composite the *room* images in Figure 4. Show the results and make discussion on the following questions.

1) How many control points were used? What if we use more than four control points?
2) How did you select control points? Clearly specify your observations and strategies.

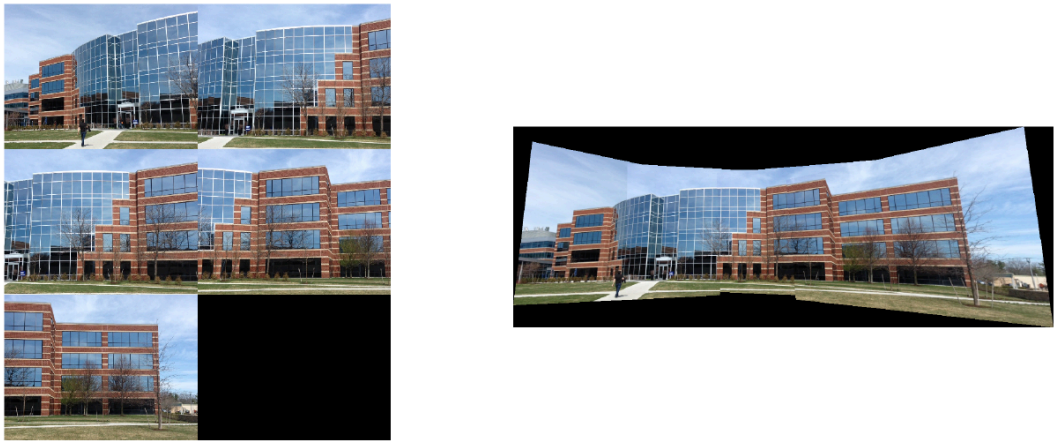


Figure 3: An example of homographic transformation and image stitching

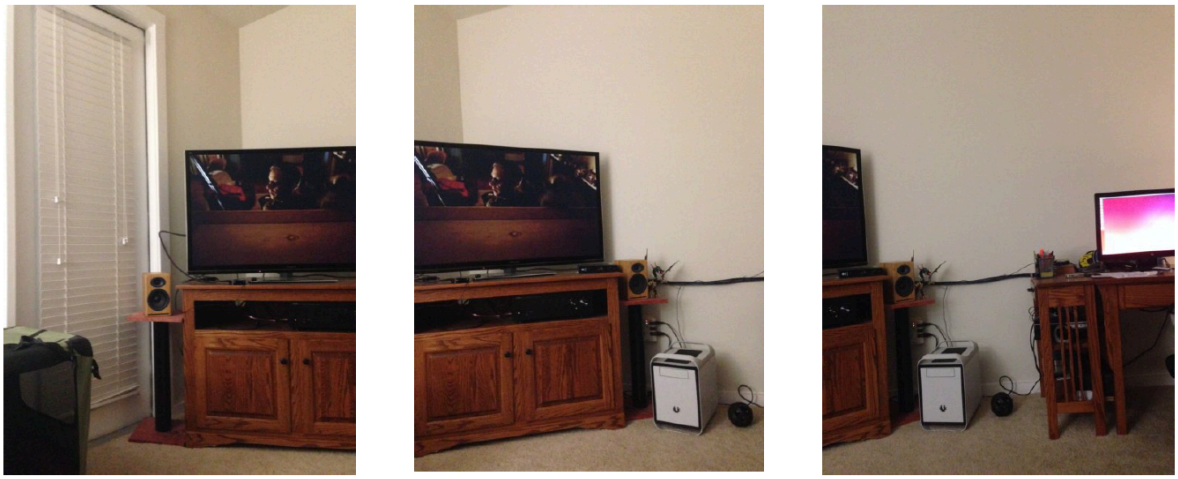


Figure 4: The room images (left, middle and right).

(1.a) **Geometrical Image Modification**

Abstract and Motivation:

Geometrical Manipulation of images is a very important tool in computer graphics. This problem requires us to warp an image from a square to circle and the obtained circular image must be applied the inverse transformation to get the original square image. Geometrical Image manipulations are the basic steps for homographic transformations and all the other image transformations. Geometrical Manipulations of image are very much required for Image morphing and these techniques have been used in the film industry for a long period of time. This is a very important tool in computer vision as different aspects of an image have to be considered to obtain useful information from the image. In this problem, the square image will be converted to a circular image without loss of information. And the reverse also will be performed to understand the mapping from square to circle and vice versa.

Approach and Procedures:

There are many methods for converting a square image to a circular image. The main question is, along what axis is the image to be condensed. For example, the image may be condensed along the radial axis, elliptical axis etc. The elliptical grid mapping method is chosen, as the image looks better when compared to the radial axis condensation. The very first step in geometrical image manipulation, is to convert the pixel locations to Cartesian coordinates. It is better to work on the Cartesian space and then convert it back to the pixel locations. After the Cartesian coordinates are obtained, the transformations can then be applied. The transformation of the elliptical axis method is given below.

Disc to square mapping:

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

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

Square to disc mapping:

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

The x, y coordinates are the Cartesian coordinates of the original square image and u, v of the transformed circular image. There are two methods for mapping the coordinates from the square to the circular image. The first method is forward mapping, that is scanning the pixel of the original square image, finding its Cartesian coordinates and mapping that pixel to the location of the transformed coordinates u and v. The other method is the backward mapping, in the which the pixels having coordinates u and v from the circular image are scanned and the pixel value at the corresponding location x and y of the original square image is loaded in that pixel. This is how the backward mapping works. Either of the 2 set of transformation equations given above can be used. The forward mapping has been used to transform the image from a square to circle and the backward mapping has been used to transform the image from circle to square. After the pixel values are mapped, the other important thing to be implement is the bilinear interpolation.

As we can visualize, each row of the square image is condensed to a the first row of the circular image which is almost half the length of first square image's row. Hence bilinear interpolation has to be used to fit all the pixels in the designated length. This process has to be applied to all the rows and columns to obtain the entire circular image. These are the steps needed to perform a geometrical operation on an image.

Experimental Results:

Figure 1. Puppy spatial warping

Figure 2. Puppy Inverse Spatial Mapping

Figure 3. Tiger Spatial Warping

Figure 4. Tiger Inverse Spatial Warping

Figure 5. Panda Spatial Warping

Figure 6. Panda Inverse Spatial Warping

Discussion:

The circular images obtained by applying the spatial warping for the three images namely puppy, tiger and panda are given in the figures 1, 3 and 5. The square images obtained using the inverse spatial warping are given in figures 2, 4 and 6. As we can see, the circular images obtained contain all the information from the original square image. The elliptical mapping is the best option of square to circle transformation for the shown images. Forward mapping was applied to obtain the circular warped image of the square image. When the radial axis warping is applied, the orientation of the images changes when the transformation is applied. For example, when the radial axis warping was applied for the Panda image, the legs of the Panda were out of orientation when compared to the original Panda image. And hence, the elliptical grid mapping was the best method of transformation from square to circular.

For the inverse transform from circular to square, the backward mapping technique was employed with the same transformation used for the forward mapping. When we see the results of the inverse spatial warping shown in the experimental results, we can observe that there is some information loss in the corners of the image when compared to the original square image. We can also observe that the image is distorted when compared to the original image.

Even though the mapping is not one-to-one, some pixels are lost when the inverse transformation is applied. While mapping back from the circle to the square, some of the pixels are blackened and hence the image is distorted. When it comes to the corners and the boundaries, the pixels are missing. Even after the application of bilinear interpolation, some of the boundary pixels are black due to the reason that the boundary pixels of the circular image are lost when they are mapped to the square image. The major reason for distortion is that, all the pixels of the circular image are not mapped to a unique location in the inverse spatial warping image. Thus, the images obtained by the inverse spatial transformation and the original square images are not the same even though the mapping is one-to-one.

(1.b) **Homographic Transformation and Image Stitching**

Abstract and Motivation:

Homographic transformations have been widely used for many applications in computer vision. It is mainly used in making Panorama images. Homographic transformations are otherwise called as projective affine transformations. This can be considered a very sensitive transformation. These are mainly used to study visual perspectives. Using Homographic transformations, many new information from the images are obtained. These transformations are specifically needed for Augmented Reality and Virtual Reality applications. The VR headsets make use of these homographic transformations for every frame. Homographic transformations can change our view on a particular image. For example, a short road can be made a look as a very long road with the help of these homographic projections. These are very much used in the film industry and for image processing purposes. Homographic projections and stitching go hand in hand. The effect is better when the homographic transformation is applied to a particular section of the image instead of the entire image. After that particular section has been subjected to homographic transformation, it then has to be connected with the original image, and this is called as Image stitching. Thus, homographic transformation and image stitching can be considered as a single technique for many applications. The purpose of this problem, is to apply homographic transformation and stitching to the 3 images of the room, and get a single panoramic image.

Approach and Procedures:

To apply the homographic transformation, the transformation matrix has to be obtained. This is the first step in Homographic transformation. The matrix is a 3x3 matrix and is given below.

$$\begin{bmatrix} x_2' \\ y_2' \\ w_2' \end{bmatrix} = \begin{bmatrix} H_{11} & H_{12} & H_{13} \\ H_{21} & H_{22} & H_{23} \\ H_{31} & H_{32} & H_{33} \end{bmatrix} \begin{bmatrix} x_1 \\ y_1 \\ 1 \end{bmatrix} \quad and \quad \begin{bmatrix} x_2 \\ y_2 \end{bmatrix} = \begin{bmatrix} \frac{x_2'}{w_2'} \\ \frac{y_2'}{w_2'} \end{bmatrix}$$

The points on the left side give us the locations where the points on the right side are being mapped to. The element H33 is always equal to 1. This matrix can be found by choosing 4 control points in the 2 images that need to be stitched. The control points are points that are present in both the images and they need to be accurately mapped. When the control points are not properly chosen, the transformation may not work as expected. The middle image has to be placed in a bigger image having the area to fit the transformed left and the right image before the control points are chosen. Now we have to choose the control points that are common in the left image and the bigger middle image. After choosing the four control points, the eight equations have to be solved to find the 8 missing parameters of the H matrix.

For stitching the images, the best option is to apply backward homographic transformation mapping. For this, we need the H inverse matrix. The middle bigger image has to be scanned and each pixel's corresponding location in the original left image has to be found and the if that particular point is there in both the images, the average of the pixel values are to be taken. This process is repeated for all the other points in the middle image. For stitching the right and this middle bigger image, the same process has to be repeated and different control points are to be taken and a new H inverse matrix has to be obtained. Using the matrix, the right image can also be stitched with the middle image. Another important step is that, bilinear interpolation has to be applied when finding each pixel, as the pixel coordinate in the left or the right image may not be present. This is how a Panorama image is obtained.

Experimental Results:

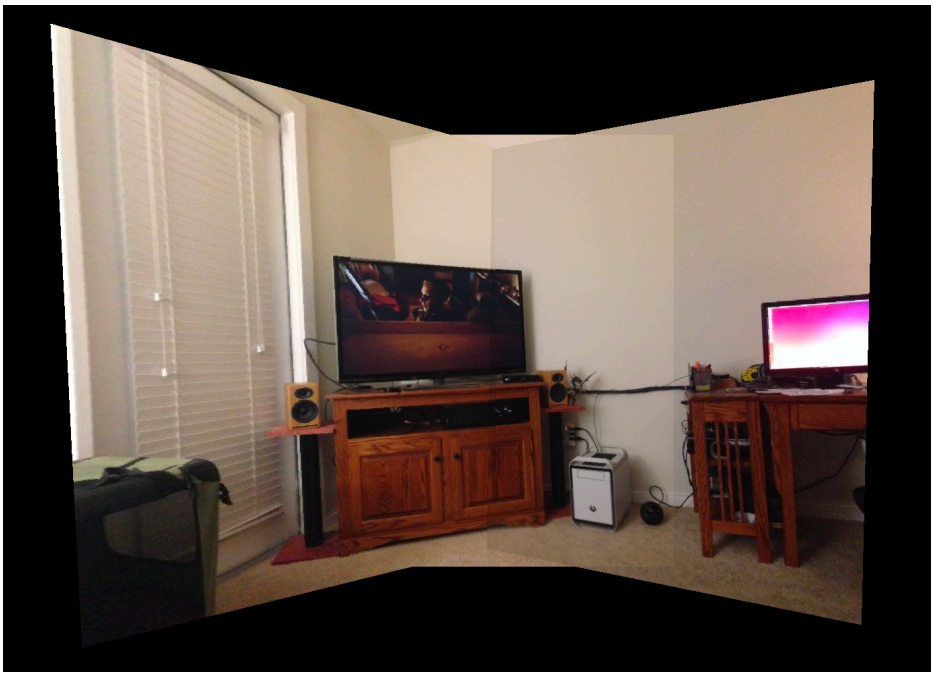


Figure 7. Stitched Panorama Image (4 control points)

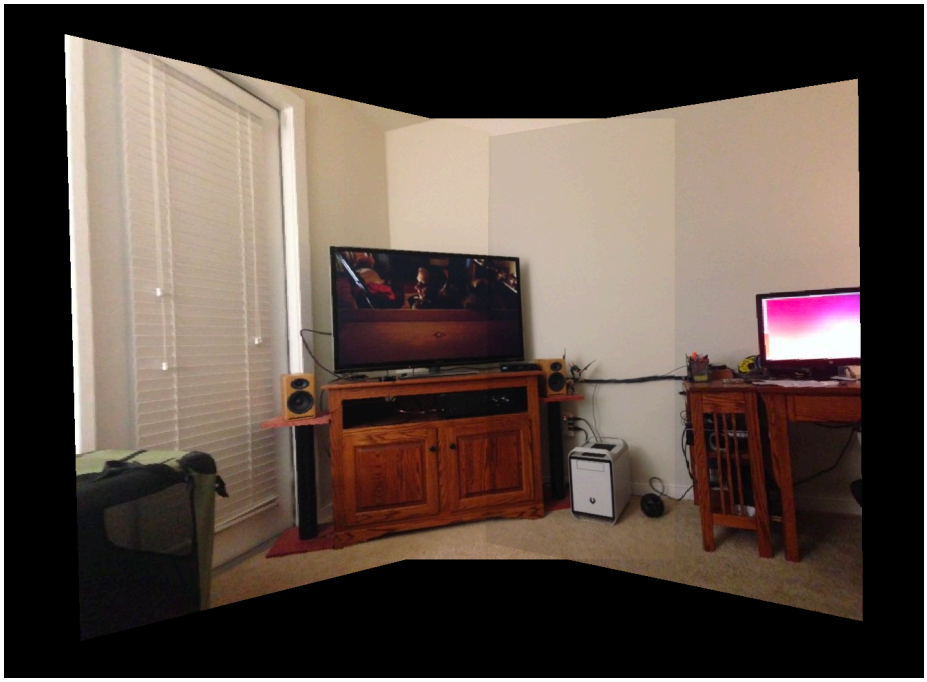


Figure 8. Stitched Panorama Image (8 control points)

Discussion:

The Panoramic Image was constructed by applying homographic projection to the left and the right images and stitching them with the middle image. The control points have to be chosen in such a way that if those points are correctly mapped to the destination pixels, then the other points will be mapped to their destinations without any error. The four control points chosen for the left and the middle image were, the intersection of the wall and the television, two corners of the cupboard door and another point on the television screen which are common in both the images. After the control points are chosen, the inverse H matrix has to be found. Since we are doing the backward mapping, we need the inverse matrix to the highest precision. Since this involves an inverse operation, the matrix is very sensitive. The third row of the matrix, decides the orientation of the image that is being homographically transformed.

After the left image is stitched with the middle image, the same process of choosing the control points is performed for this stitched image and the right image. This gives a new H inverse matrix, and thus the final stitched image is obtained. For pixels which are present in both the images, the average of the pixel values has to be taken for a perfect stitch.

For this problem, four control points were taken. And hence for solving the system of equations, we have eight unknowns and eight equations. If we choose more control points, it will be difficult to solve the system of equations as the inverse does not exist. In this case, the pseudo inverse is obtained. From the image in figure 8 we can observe that this image is better than the image obtained with 4 control points. From the edges of the television and the corner of the wall, we can observe that the image stitched with 8 control points is better when compared to the other image. Hence, if we use more control points, the transformation will make sure that the control points are definitely matched. Hence, by intuition we can say that, if the control points increase, then more points in the image will be stitched properly. Thus, increasing the control points improves the quality of stitching. For perfect stitching, more control points are needed, but the complexity of finding the transformation increases.

The control points have to be chosen in such a way that they are present in both the images, and by strategy that, if you match those 4 points, the entire image has to be mapped

exactly to an extent. Points that satisfy both the above-mentioned conditions can be chosen as control points. Considering the left and the middle images, the television boundaries and the cupboard door boundaries play a very important role in the image stitching. If these points are properly mapped between the two images, the entire image gets mapped as these points define the common edges in both the images. Thus, the strategy for choosing the control points can be summarized as follows,

- The points have to be present in both the images that are going to be stitched
- They need to be points which define the points that must be matched in both the images to a very high accuracy, i.e., the points which define edges that are very look prominent in both the images.

If these conditions are matched, the control points will give us the best stitched image.

Problem 2: Digital Half-toning (30+10%)

3(a) Dithering (Basic: 20%)

Implement the following four methods to convert *colorchecker* to half-toned images. There are 256 gray levels for pixels in *colorchecker* image. In the following discussion, *F(i,j)* and *G(i,j)* denote the pixel of the input and the output images at position (*i,j*), respectively. **Compare the results obtained by these algorithms in your report.**

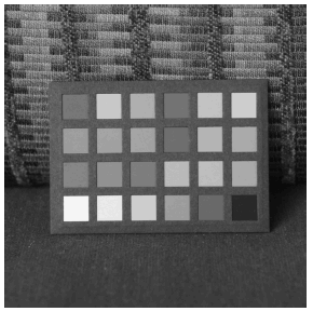


Figure 5: The colorchecker image

1. *Fixed thresholding*

Choose one value, T, as the threshold to divide the 256 levels into two ranges. An intuitive choice of T would be 127. For each pixel, map it to 0 if it is smaller than T, otherwise, map it to 255, i.e.

$$G(i,j) = \begin{cases} 0 & \text{if } 0 \le F(i,j) < T \\ 255 & \text{if } T \le F(i,j) < 256 \end{cases}$$

2. *Random thresholding*

In order to break the monotones in the result from fixed thresholding, we may use a 'random' threshold. The algorithm can be described as:

- For each pixel, generate a random number in the range 0 ~ 255, so called *rand(i,j)*
- Compare the pixel value with *rand(i,j)*. If it is greater, then map it to 255; otherwise, map it to 0, i.e.

$$G(i,j) = \begin{cases} 0 & \text{if } rand(i,j) \le F(i,j) \\ 255 & \text{if } rand(i,j) > F(i,j) \end{cases}$$

The built-in rand function in C/C++/Matlab generate numbers in uniform distribution.

3. *Dithering Matrix*

Dithering parameters are specified by an index matrix. The values in an index matrix indicate how likely a dot will be turned on. For example, an index matrix is given by

$$I_2(i,j) = \begin{bmatrix} 1 & 2 \\ 3 & 0 \end{bmatrix}$$

where 0 indicates the pixel most likely to be turned on, and 3 is the least likely one. This index matrix is a special case of a family of dithering matrices first introduced by Bayer [2]. The Bayer index matrices are defined recursively using the formula:

$$I_{2n}(i,j) = \begin{bmatrix} 4 * I_n(x,y) + 1 & 4 * I_n(x,y) + 2 \\ 4 * I_n(x,y) + 3 & 4 * I_n(x,y) \end{bmatrix}$$

The index matrix can then be transformed into a threshold matrix T for an input gray-level image with normalized pixel values (*i.e.* with its dynamic range between 0 and 1) by the following formula:

$$T(x,y) = \frac{I(x,y) + 0.5}{N^2}$$

where N^2 denotes the number of pixels in the matrix. Since the image is usually much larger than the threshold matrix, the matrix is repeated periodically across the full image. This is done by using the following formula:

$$G(i,j) = \begin{cases} 1 & \text{if } F(i,j) > T(i \bmod N, j \bmod N) \\ 0 & otherwise \end{cases}$$

where *G(i, j)* is the normalized output binary image.

Answer the following questions:

1. Please create $I_2(i,j)$, $I_4(i,j)$ and $I_8(i,j)$ thresholding matrices and apply them to halftone the *colorchecker* image.
2. If a screen can only display FOUR intensity levels, design a method to generate a display-ready *Colorchecker* image. Show your best result in gray-scale with four gray-levels (0, 85, 170, 255) and explain your design idea and detailed algorithm.

3(b) Error Diffusion (Basic: 10%)

Convert the 8-bit *colorchecker* image to a half-toned one using the error diffusion method. Show the outputs of the following three variations, and discuss these obtained results. Compare these results with dithering matrix. Which method do you prefer? Why?

1. Floyd-Steinberg's error diffusion with the serpentine scanning, where the error diffusion matrix is:

$$\frac{1}{16}\begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 7 \\ 3 & 5 & 1 \end{bmatrix}$$

2. Error diffusion proposed by Jarvis, Judice, and Ninke (JJN), where the error diffusion matrix is:

$$\frac{1}{48}\begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 7 & 5 \\ 3 & 5 & 7 & 5 & 3 \\ 1 & 3 & 5 & 3 & 1 \end{bmatrix}$$

3. Error diffusion proposed by Stucki, where the error diffusion matrix is:

$$\frac{1}{42}\begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 8 & 4 \\ 2 & 4 & 8 & 4 & 2 \\ 1 & 2 & 4 & 2 & 1 \end{bmatrix}$$

Describe your own idea to get better results. There is no need to implement it if you do not have time. However, please explain why your proposed method will lead to better results.

3(c) Color Halftoning with Error Diffusion (Bonus 10%)

Figure 6: The flower image

1) Separable Error Diffusion

One simple idea to achieve color halftoning is to separate an image into CMY three channels and apply the Floyd-Steinberg error diffusion algorithm to quantize each channel separately. Then, you will have one of the following 8 colors, which correspond to the 8 vertices of the CMY cube at each pixel:

W = (0,0,0), Y = (0,0,1), C = (0,1,0), M = (1,0,0),
G = (0,1,1), R = (1,0,1), B = (1,1,0), K = (1,1,1)

Note that (W, K), (Y, B), (C, R), (M, G) are complementary color pairs. Please show and discuss the result of the half-toned color *flower* image. What is the main shortcoming of this approach?

2) MBVQ-based Error diffusion

Shakad et al. [3] proposed a new error diffusion method to overcome the shortcoming of the separable error diffusion method. They partition the CMY color space into six Minimum Brightness Variation Quadrants (MBVQ) as shown in Fig. 2 of [3]. Then, the MBVQ-based error diffusion can be conducted as follows. Given the CMY value and the quantization error at a pixel located in (x, y). They are denoted by RGB(x, y) and e(x, y), respectively.

- Determine its MBVQ quadrant based on RGB (x, y);
- Find the vertex V of the MBVQ tetrahedron to closest CMY (x, y) + e (x, y) and output the value of V at location (x, y);
- Compute the new quantization error using CMY (x, y) + e (x, y) - V
- Distribute the quantized error to the future pixel error buckets e using a standard error diffusion process (e.g. the FS error diffusion).

You may refer to [3] for more details.

Implement this algorithm and apply it to the *flower* image in Fig. 6. Compare the output with that obtained in (1). Discuss the difference between these two methods.

(2.a) **Dithering**

Abstract and Motivation:

Dithering is one of the many methods available for image half-toning. Dithering was very important in the past when the display devices could display only very less colors. Dithering is the method of quantizing the colors in an image. For example, if an image has

256 colors, reducing it to 4 or 8 colors is called as dithering. But now, even million color displays are very cheap. Even then, dithering is very important for printing images and for artistic creativities in web design etc., Considering a gray scale image, if that image is directly printed on a paper it looks very bad because of fact that only the pure black pixels (value of 255) will be printed on the paper, as the printer can print dots only which are completely black. To avoid this problem, the image has to be converted to an image which has only color values 0 and 255. To achieve this, the image has to be dithered. By dithering, an image which has the color range from 0-255 can be converted to an image with color values 0 and 255 but most of the information will be preserved. In this problem, different dithering techniques will be applied and their performances will be compared.

Approach and Procedure:

There are various procedures for applying dithering to images. Considering the gray scale images, there are different methods to convert the color values from 0-255 to either 0 or 255. The first method is the fixed threshold image in which the image is converted to 0 if the color value is less than a threshold and to 1 if the color value is greater than the threshold. The formula for fixed thresholding is given below.

$$\bar{G}(i,j) = \begin{cases} 0 & \text{if } 0 \le F(i,j) < T \\ 255 & \text{if } T \le F(i,j) < 256 \end{cases}$$

The second method is random thresholding, in which the threshold varies for every pixel. A random number is generated for every pixel, and if the pixel value is lesser than random value generated, the pixel value is updated to 0 and to 1 if the pixel value is greater than the random number generated. The formula for random thresholding is given below.

$$G(i,j) = \begin{cases} 0 & \text{if } rand(i,j) \le F(i,j) \\ 255 & \text{if } rand(i,j) > F(i,j) \end{cases}$$

The third method is using the dithering matrix. Instead of a fixed or random threshold like the previous methods, in this method a window is considered, and the color values are compared with the values in the matrix. The matrices are called as Bayer matrices and they were introduced by Bayer [1]. The basic 2x2 matrix is given by,

$$I_2(i,j) = \begin{bmatrix} 1 & 2 \\ 3 & 0 \end{bmatrix}$$

The matrices for other bigger windows are obtained by,

$$I_{2n}(i,j) = \begin{bmatrix} 4 * I_n(x,y) + 1 & 4 * I_n(x,y) + 2 \\ 4 * I_n(x,y) + 3 & 4 * I_n(x,y) \end{bmatrix}$$

As the window increases, the above mentioned recursive formula is used to generate the Bayer matrices of different sizes. After the matrices are generated, the normalized output image is obtained with the following formula:

$$G(i,j) = \begin{cases} 1 & \text{if } F(i,j) > T(i \bmod N, j \bmod N) \\ 0 & otherwise \end{cases}$$

where,

$$T(x,y) = \frac{I(x,y) + 0.5}{N^2}$$

N is the window size. And T(x,y) is the threshold matrix based on the matrix I(x,y). So, the image is divided into a number of sub-blocks, each block having a size of the Bayer matrix that is going to be used. Considering the 2x2 window, pixels in each block are compared with the 2x2 Bayer matrix. Each pixel of the sub-block is compared with the corresponding element in the Bayer matrix and they are assigned a value of 255 if they are greater and a value of 0 if they are lesser than the matrix element. The same process is

applied for other bigger windows as well. These are the 3 methods of dithering and their results are discussed below.

Experimental Results:

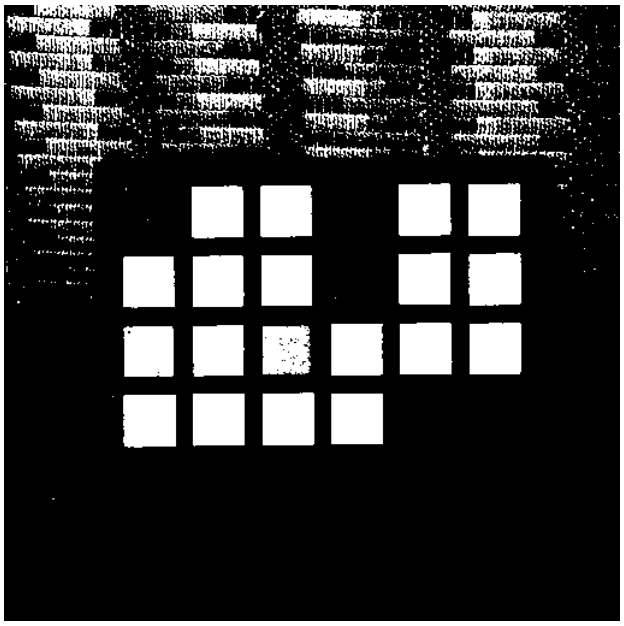


Figure 9. Fixed Thresholding Dithering

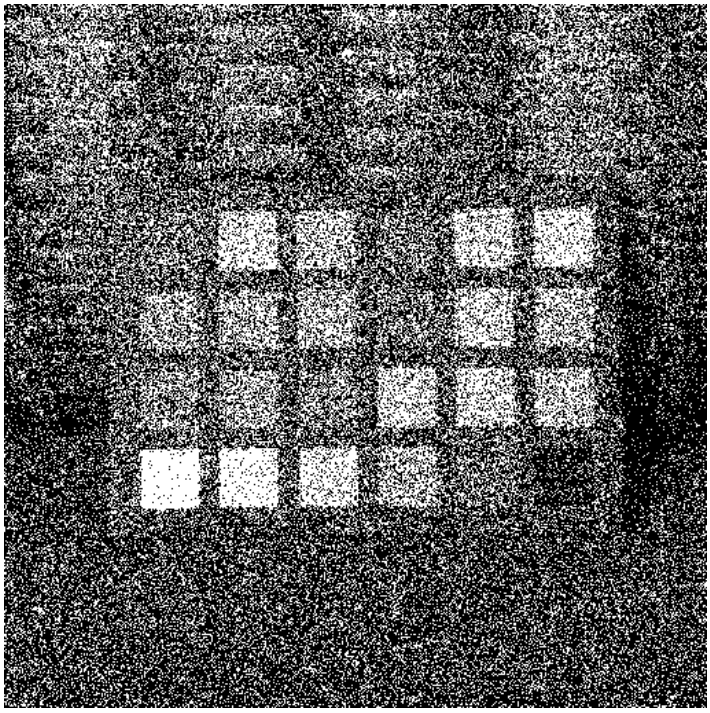


Figure 10. Random Thresholding Dithering

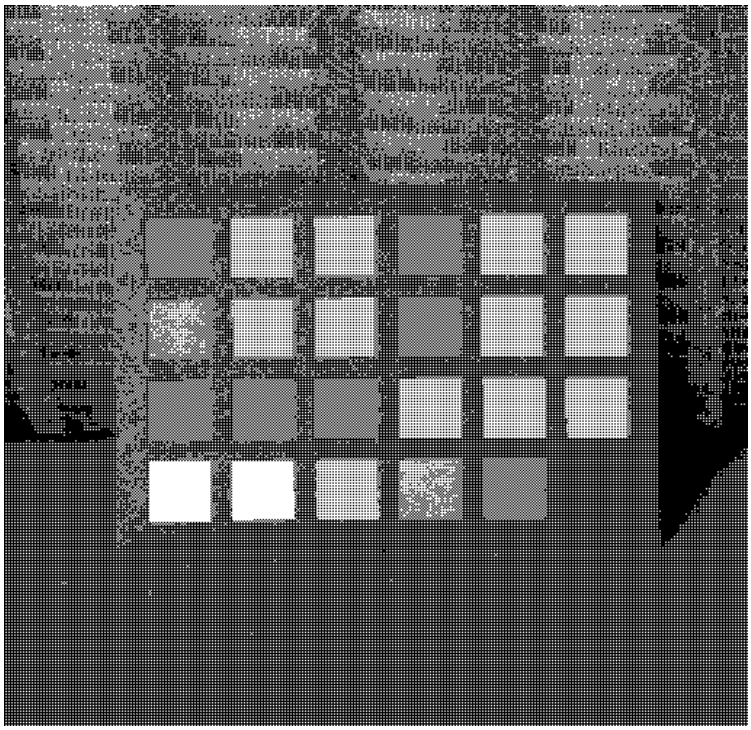


Figure 11. 2x2 Dithering matrix

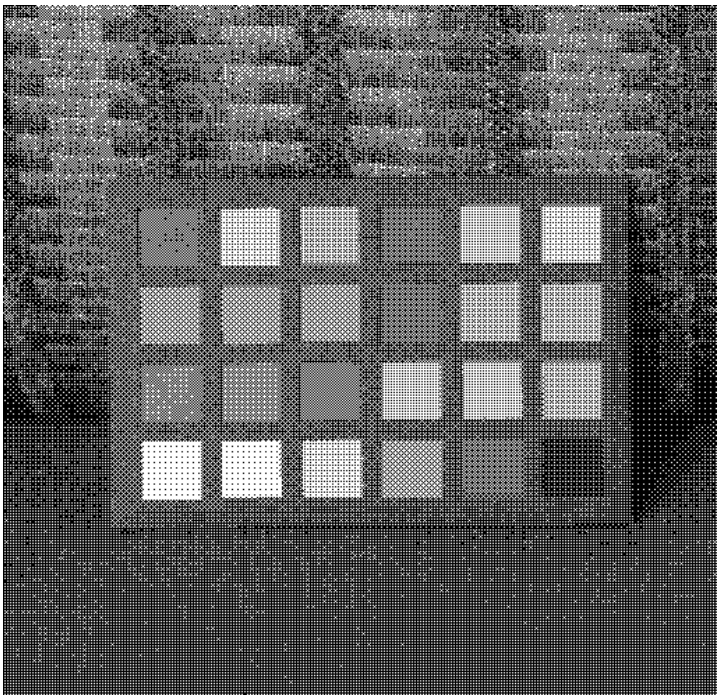


Figure 12. 4x4 Dithering Matrix

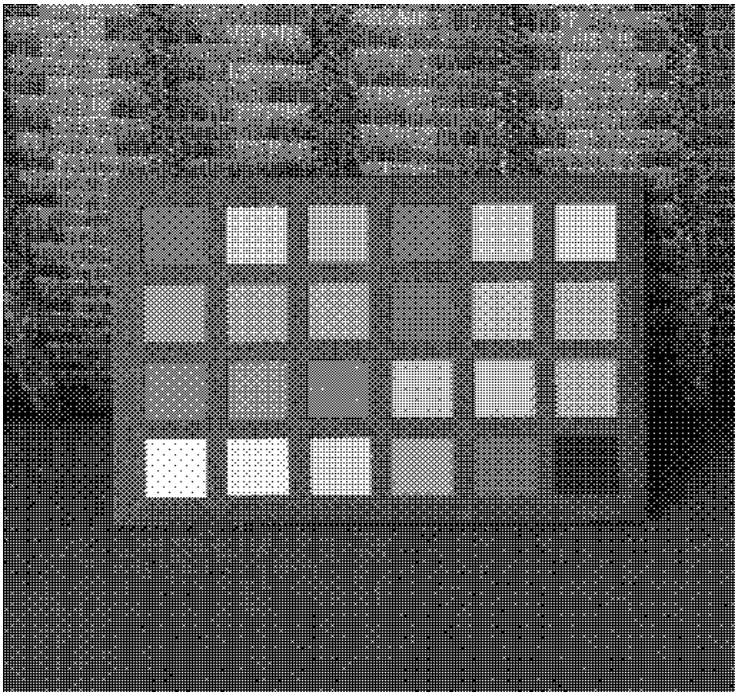


Figure 13. 8x8 Dithering Matrix

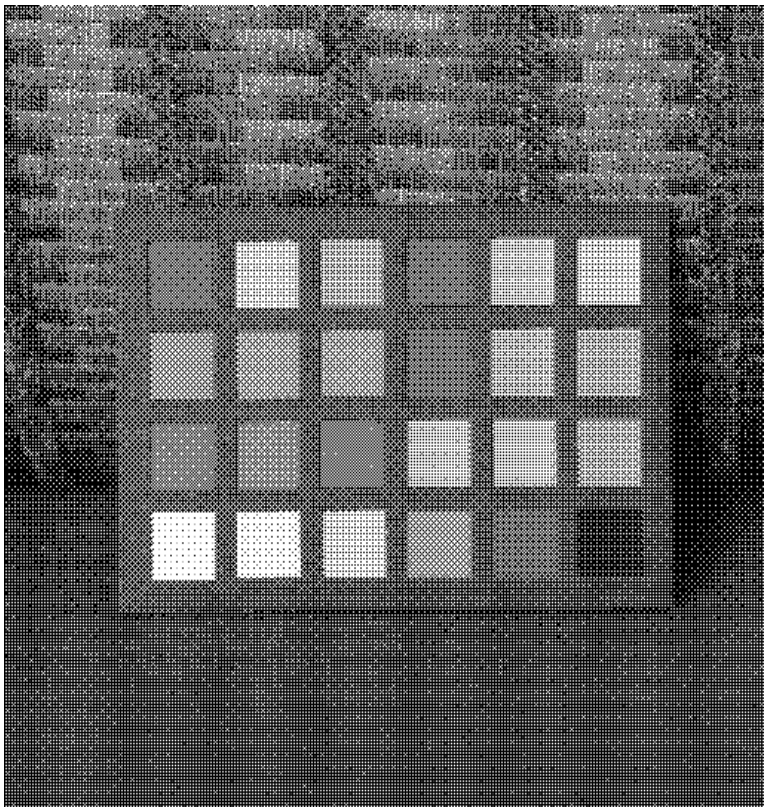


Figure 14. Dithering with 4 gray levels

Discussion:

The figures 9 and 10 are obtained using the fixed and random thresholding correspondingly. As we can see, a lot of information from the image is lost when these thresholding methods are used. To improve the image, the dithering matrix method is used. The images shown in figures 11, 12 and 13 are obtained for windows 2x2, 4x4 and 8x8. As we can observe from the quality of the images, the image dithered with the Bayer 8x8 matrix is better when compared to the other two windows. The image obtained with the 2x2 Bayer matrix is very poor because the sub-blocks are very small and hence, it considers only the surrounding local region only.

In the case of the other bigger windows, the block size increases and hence it relates more to the global relation in that image, instead of considering only the local surrounding pixels. Hence, the bigger Bayer matrices give better results when compared to the smaller dithering matrices.

The final objective is to quantize the image with 4 colors. There are many ways in which this can be done. The idea followed here is that, the dithered images obtained with Bayer matrices 4x4 and 8x8 are taken. Depending the pixel values obtained from both the images, the 4 gray levels are allotted. If the both the pixels have value 1, then the output pixel will also have 1. If image obtained with 4x4 window has 1 and the image obtained with the 8x8 windows has 0, the image output value is 85. If image obtained with 4x4 window has 0 and the image obtained with the 8x8 windows has 1, the image output value is 170 and if both the pixels are 0, then the output pixel is also 0. Using this method, 4 quantization colors are obtained. This is one of the methods using which an image can be quantized to 4 gray levels. This is shown in the image in figure 14.

(2.b) **Error Diffusion**

Abstract and Motivation:

The motivation for this problem is same as the motivation for dithering. The objective is to convert an image having colors from 0-255 to either 0 or 255, i.e., to quantize them. Dithering is very abstract and the results are not very appealing. Hence, the other method for quantizing an image is Error diffusion. In dithering, the pixel values are either 0 or 255 and we ignore the error that is being added to each pixel to change it to 0 or 255 from its original value. This method of Error diffusion considers the error each and every time the pixel value is converted to 0 or 255 and diffuses them to the surrounding pixels, hence the name error diffusion. Both diffusion and dithering give us half-toned images, but the quality greatly differs.

Approach and Procedure:

The main idea in the diffusion technique is that the error that is added to each pixel to make it 0 or 255 is stored and diffused among the surrounding pixels with different weighted component of the error added to each of them. There are different methods for diffusing the error among the neighboring pixels.

The first method called as Floyd-Steinberg's method, diffuses the error among 4 neighbors. The weight matrix is for this method is given below.

$$\frac{1}{16}\begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 7 \\ 3 & 5 & 1 \end{bmatrix}$$

The error is divided by 16 and then multiplied by the corresponding weights and then added to the corresponding neighboring pixels. The error is signed, that is it is positive when a pixel having value less than the threshold is made 0 and negative when the pixel having value higher than the threshold is made 255. Another important step to be noted is, Serpentine scanning must be followed when the error is diffused. Instead of accumulating the error in the same order right to left, when the serpentine scanning method is used, the error is diffused in an alternating fashion between right to left and left to right. In this way, the error is not accumulated towards one side alone.

The second method and third are very similar to the first method, just that the diffusion matrices are different. The diffusion matrix proposed by Jarvis, Judice and Ninke is given below.

$$\frac{1}{48}\begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 7 & 5 \\ 3 & 5 & 7 & 5 & 3 \\ 1 & 3 & 5 & 3 & 1 \end{bmatrix}$$

The diffusion matrix proposed by Stucki is given below.

$$\frac{1}{42}\begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 8 & 4 \\ 2 & 4 & 8 & 4 & 2 \\ 1 & 2 & 4 & 2 & 1 \end{bmatrix}$$

The algorithm for all the methods is the same. Only the weights with which the error is diffused among the neighboring pixel changes.

Experimental Results:

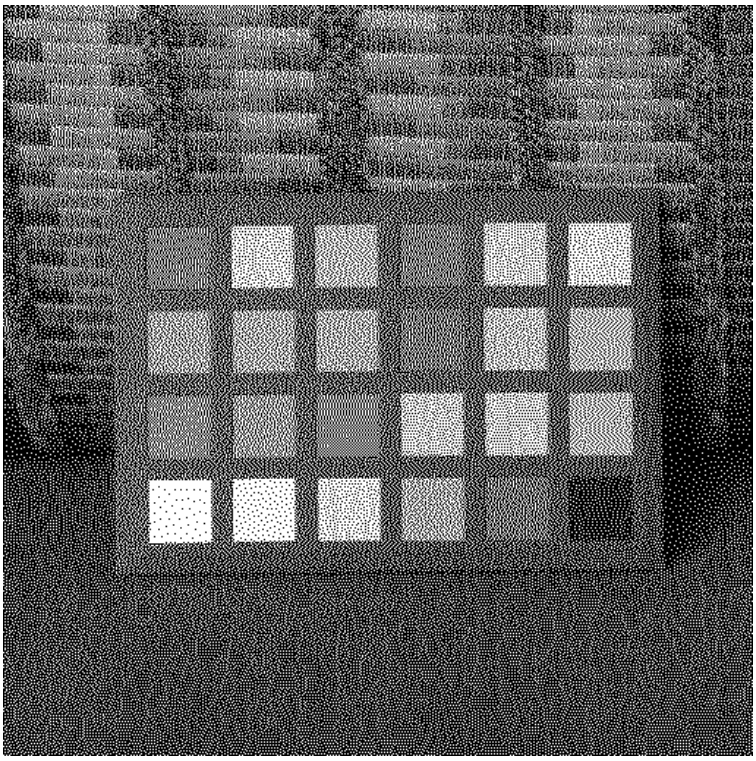


Figure 15. Floyd Steinberg Error Diffusion

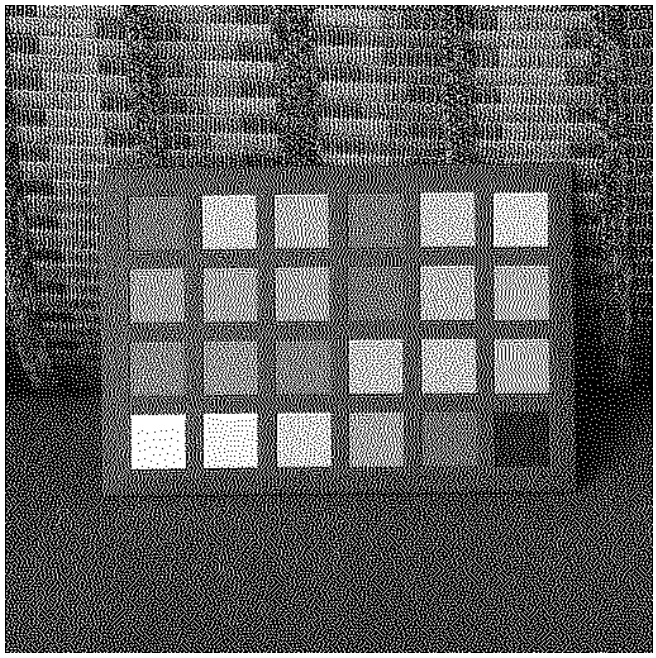


Figure 16. JJN Error diffusion

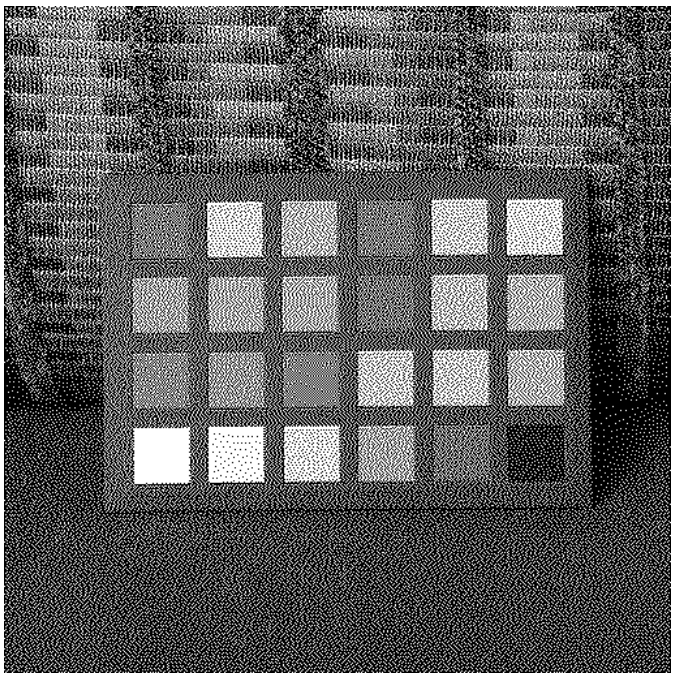


Figure 17. Stucki Error diffusion

Discussion:

The images given in figure 15, 16 and 17 are obtained using the error diffusion methods specified earlier. The difference between these three methods is only the error weighting matrix. When these images are compared with the dithering images, the image quality is much better. This is observed visually as well. The images obtained with diffusion is better because, each and every time a pixel is made to be 0 or 255, the error in that conversion is stored. This is not the case in dithering. In dithering, the pixels are converted, but the error is just not processed. In diffusion, after the error is obtained, the error is distributed among the neighboring pixels and hence the surrounding pixels now have an increased or decreased value.

For instance, if a region has color ranges in the region between 200 and 220 (gray color), the first pixel will be converted to 255. The error, 200-255 is then allocated to the neighboring pixels using the diffusion matrix. And hence this region will have both black and white pixels, as the pixel values of the neighboring pixels is decreased since the error is negative. In this way, there is a balance between the black and the white pixels. If there are more black pixels and less white pixels, the original gray color can be visualized by the viewer. Whereas, in dithering the entire region is made black and color difference is lost. This is why error diffusion gives better results when compared to the dithering method.

Among the three methods of error diffusion, the JJN method and the Stucki method have larger windows when compared to the Floyd Steinberg method. In JJN and Stucki method, the error is equally distributed to the left and right of the center pixel, but in Floyd Steinberg method, there is more weightage on one side depending on whether that row is being scanned from right to left or from left to right. Floyd Steinberg has a better result when compared to the other two methods, as the grains are less sharp in Floyd Steinberg method. Thus, error diffusion is better when compared to dithering and the reasons have been explained above.

There are many other ways to improve the error diffusion methods. The thresholding values play a very important role. In this case, the threshold value is always at 127. If the pixel value is greater than the threshold, we convert it to 255. This threshold can be changed depending on the nature of the image. For some images, threshold of 200 or 220 will be better and for some other image, threshold values of 40 or 60 will be better.

This all depends on the overall tone of the image. Another way to improve the error diffusion method is to decrease the error that is being diffused to the neighboring pixels. The error values can be reduced by 50% or 75% to get better results. By doing this, we will overcome the problem of color bleeding reduction. Thus, error diffusion can be modified in the following ways:

i) Change the threshold values based on the overall tone of the image. This can be decided on the contrast of the image. If the histogram of the image is more towards the darker regions, the threshold can be low in order to create the variations in the image and if the histogram is towards the brighter side, the threshold can be higher. By doing this, we will not lose the contrast of the original image.

ii) Reducing the error by 50% or 75%, we can control how much the error is going to affect the neighboring pixels. Instead of using the original error value, we can modify it in order to reduce its effect on the neighboring pixels, thereby reducing the loss of information. This also depends on the type of image that is going to be quantized.

Thus, by implementing the above-mentioned new methods, we can improve the results obtained by error diffusion algorithms.

(2.c) **Color Half-toning with Error Diffusion**

Abstract and Motivation:

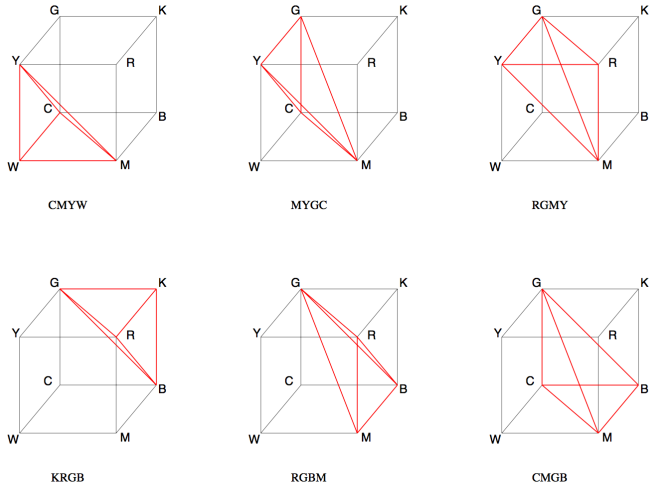
Previously, color displays were very costly and they had very limited color displays such as 64 or 512 having 4 or 8 colors per channel respectively. And hence, it was very important to reduce higher color images to lower color images to make them displayable in the low-cost devices. But nowadays, 64 million color displays are also affordable. Diffusion is still needed to halftone the color image, to make them printer friendly. This is the reason why Error Diffusion for images is still very important. The purpose of this problem is to halftone color images using 2 methods of color image error diffusion.

Approach and Procedures:

For color image half-toning, there are two methods of error diffusion namely Separable error diffusion and Minimum brightness variation quadrants (MBVQ) method. This method was first proposed by Shakad et al., for HP labs [2]. In separable error diffusion, the same concept of error diffusion for gray scale images is used, but that has to be applied for all the three channels separately. That is the only difference between single channel error diffusion and multi-channel error diffusion. Any diffusion matrix among the 3 diffusion matrices can be chosen for diffusing the error.

The separable error diffusion follows the same procedure as the gray scale error diffusion. Each channel is considered separately, and they are rounded off to either 0 or 255 based on their values. The error of conversion is diffused among the neighboring pixels and the same procedure is repeated for all the other three channels. Floyd Steinberg diffusion matrix has been used for distributing the error among the surrounding pixels.

In Minimum Brightness Variation Quadrants (MBVQ) method, each pixel (all three channels) is scanned and it is allotted the color of the cubic vertex, based on the closeness to each of the vertices of the cube. So, the first step in this method is to find out, on which tetrahedron the pixel lies, based on the RGB value. The different tetrahedrons in a cube are given below.



Based on the tetrahedron chosen, the distance (norm) between each of the tetrahedron vertex and the current pixel is figured out. The norm can be any norm. Here, Euclidean distance is used to find the shortest distance vertex. After the shortest distance vertex is found, the current pixel is allocated that value. The error that is added to that pixel to round

each channel of the pixel to either 0 or 255 is noted. The error is then diffused among the neighboring pixels. Any one of the above-mentioned diffusion matrices can be used to distribute the error. The same procedure is repeated for all the pixels in the image. This is the algorithm of MBVQ based color error diffusion method.

Experimental Results:

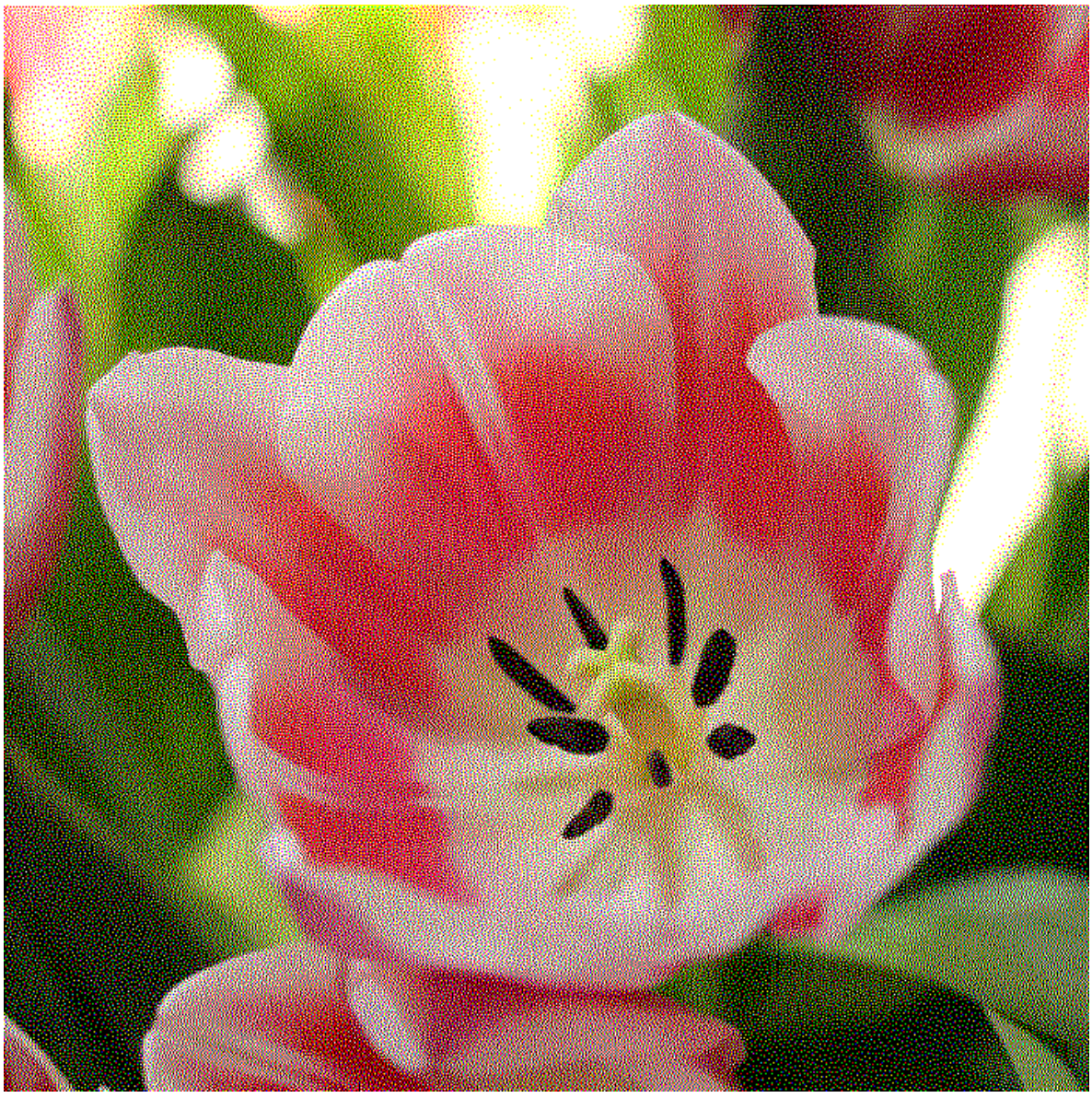


Figure 18. Color diffusion using separable error diffusion

Figure 19. MBVQ based color error diffusion

Discussion:

The image obtained with separable error diffusion is given in figure 17 and the image obtained with the MBVQ error diffusion method is given in figure 19. The MBVQ algorithm is more sophisticated than the ordinary separable error diffusion. From the images, we can observe that the image obtained with the MBVQ method is sharper than

the separable error diffusion method. In the linear separable method, the color is just found by rounding each channel value to either 0 or 255. So, each pixel can be allocated to one of the available 8 colors (2 colors per channel). In the case of the MBVQ method, the closest vertex is chosen by selecting the vertex of the tetrahedron which has the shortest distance with the current pixel. This is the main difference between the separable method and the MBVQ method.

From the images obtained, we can clearly see the decrease in the halftone noise between the 2 images, from separable error diffusion to MBVQ. This can be even more clearly observed when a specific patch is compared. Thus, the separable error diffusion has more half-tone noise when compared to MBVQ based color error diffusion.

Problem 3: Morphological Processing (40%)

In this problem, you will implement three morphological processing operations: shrinking, thinning, and skeletonizing. A pattern table (patterntables.pdf) is attached for your reference. Please show outputs for all following parts in your report and discuss them thoroughly. Please state any assumptions you make in your solution.

(a) Shrinking (Basic: 8%)

Please apply the 'shrinking' filter to the squares image (stars.raw). Please implement the filter, and discuss your solution for the following questions:

- Count the total number of stars in the image.
- How many different square sizes are present in the image? What is the frequency of these star sizes? (Hint: Plot the histogram of the square size with respect to frequency.)

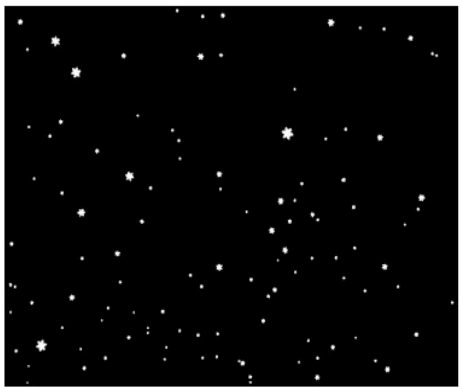


Figure 7: stars.raw

(b) Thinning (Basic: 8%)

Please apply the 'thinning' filter to the jigsaw image (jigsaw_1.raw) and show your result.

Figure 8: jigsaw_1.raw

(c) Skeletonizing (Basic: 8%)

Please apply the 'skeletonizing' filter to the jigsaw image (jigsaw_2.raw) and show your result.

Figure 9: jigsaw_2.raw

(d) Counting game (Advanced: 16%)

Figure 10 (see below) is target image with 16 jigsaw pieces (the background is white). Some of these pieces are obtained from geometrical operations, i.e. rotation, flipping, etc. You need to design an algorithm that uses morphological and logical operations to solve and answer the questions below. You may use any of these operators that you learned in class or ones that you invent, but all of them must be detailed in your report with respect to how they operate and how their results help you solve the problem. In your report, discuss your algorithm, results, and analysis in detail. You should also submit your MATLAB or C/C++ programs along with your source code submission. State any assumptions you make for this problem.

Note: Do NOT use any built-in morphological processing functions. Manual counting is not permitted; your code must automatically do this.

- Count the number of pieces in the board image. You may take pieces in red and blue bounding box are different for this question.
- Find the number of unique pieces in the board image. Pieces in red and blue bounding box are identical because they are rotated version of each other.

Note: Hits will be given during discussion sessions.

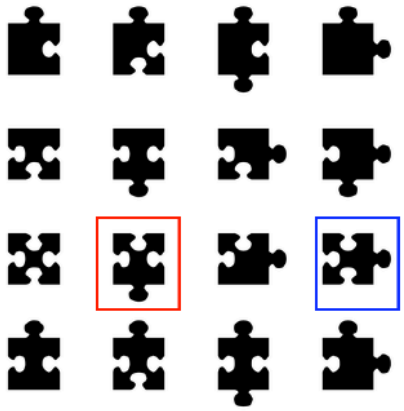


Figure 10: board.raw

(3.a) **Shrinking- Counting Stars**

Abstract and Motivation:

Morphological process is any operation which deals with the shapes in an image. It used to increase the Image shrinking is the process of shrinking an object in image to just one pixel. All connected components in an image reduce to just one pixel. This is very helpful to find the number of objects in an image. This concept can be used for finding the connected components, size of each component, etc., This function of shrinking an object, can be used to find the key points in an image which are needed in Machine learning algorithms and many other advanced techniques. The purpose of this problem is to find the number of stars in the given image and plot a histogram of the star sizes from the image.

Approach and Procedure:

Any morphological processing technique is just a hit ‘n mask filter. There are a specific set of pattern tables for every morphological processing technique. There are basically 2 stages in morphological image processing, stage 1 and stage 2. Stage 1 comprises of conditional pattern mask and stage 2 is unconditional pattern mask. The pattern tables are constructed and they are fixed for one algorithm.

In stage 1, the conditional mask is checked. For this, the bond value for each pixel is calculated. The bond is calculated by multiplying the 4-connected components by 2 and multiplying the 8-connected components by 1. After the bond is calculated, each pixel is

checked with the conditional mask patterns. If there is a match, the corresponding pixel is made 1 in another Hit array. After this is done for all the pixels of the image, the complete hit array is generated. This Hit array is used for checking the unconditional pattern mask. Each pixel value that is 1 in the hit array is compared with the unconditional pattern mask and made 0 if there is no match. This is the shrinking step. If there is a match, the output image pixel is 1. This process is repeated for a number of iterations, and finally the stars are shrunk to a single pixel. The histogram of the star sizes can be found by calculating the number of iterations each star needs to be shrunk to a single pixel.

Experimental Results:

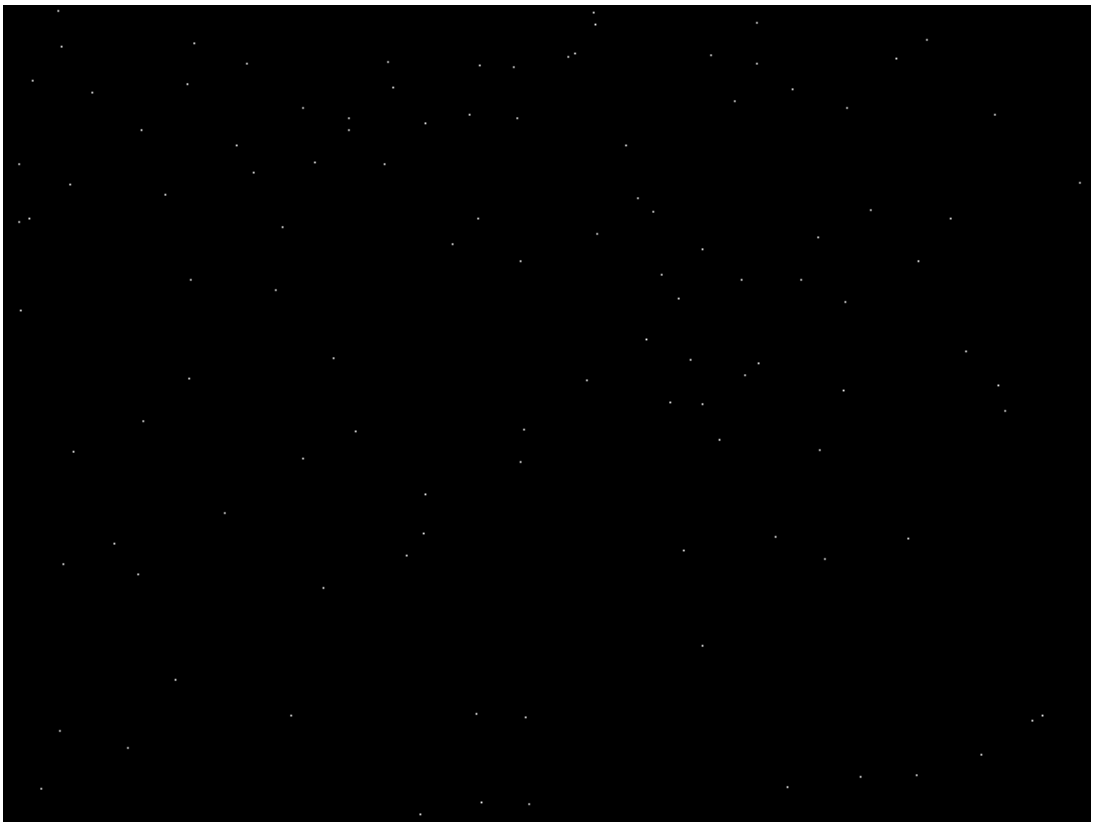


Figure 20. Stars Image after shrinking

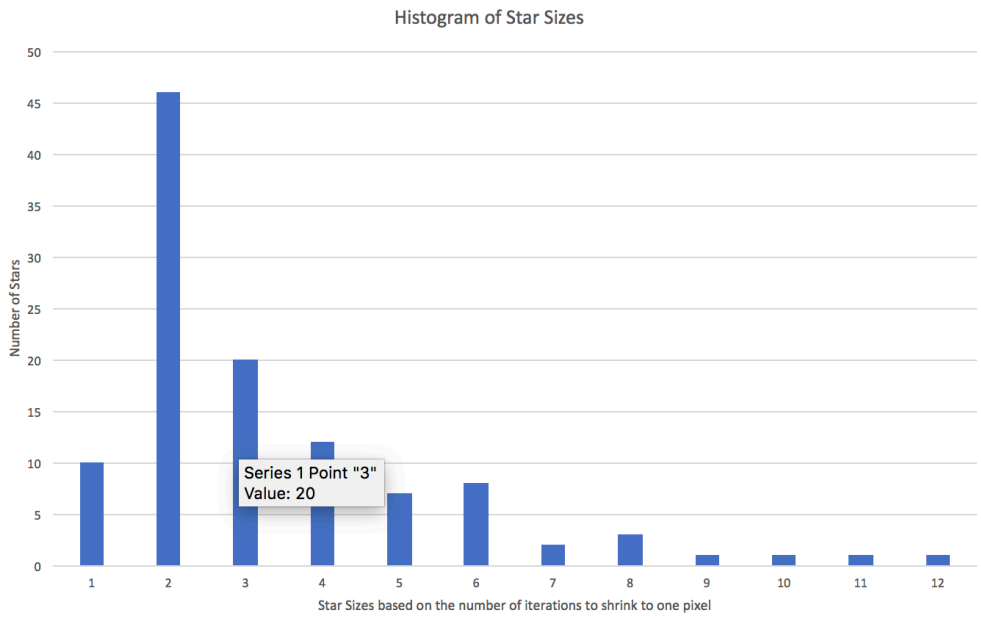


Figure 21. Histogram of star sizes

Discussion:

Binary image thinning was applied to the given stars image after dithering the image. The pattern masks for both the conditional and unconditional masks work only if the image is binary. Hence it is very important to perform dithering. The threshold value of dithering may play a very important role in deciding connected components in an image. For example, if the dithering value is very high, some information will be lost, as gray pixels will be converted to white instead of black and many other problems like these may appear. Hence it is always better to choose the dithering value as the mid-point of the color ranges, in this case 128.

The total number of stars from this image are 112. These were the number of stars that remained after applying the shrinking algorithm for 20 times. Once the final shrinking has been achieved, repeated shrinking will not have any effect on the image as there are one single pixels representing the entire star.

In this image, there are stars of different sizes. To find the histogram of the star sizes, each and every time before the next iteration starts, the pixels surrounding which all the pixels are black are noted. So, when a pixel surrounding which all pixels are black comes in, the current iteration number is loaded as the size of that pixel. After this, the histogram is plotted by measuring the sizes of all the stars and plotting the star sizes along

x-axis and the occurrence of that star size along the y-axis. From the histogram, we can see that there are 10 different star sizes and we can also see how many times these star sizes are repeated.

(3.b, c) **Thinning and Skeletonizing**

Abstract and Motivation:

Thinning and skeletonizing can reveal structural information of the binary image. This is very important to do shape analysis on an image. Thinning and Skeletonizing may be very similar in operation, but there is a difference in the algorithm that is followed. Both operations are very similar, but can help us get different interpretations from the same image. The purpose of this problem is to apply thinning and skeletonizing algorithms to the 2 given jigsaw pieces.

Approach and Procedure:

Similar to shrinking, thinning and skeletonizing also follows the same algorithm. These are also 2 stage filters, containing the conditional and the unconditional mask patterns, built for a hit ‘n mask filter. Before applying the morphological operations, the images have to be converted to binary form by applying dithering. Here also, the dithering threshold plays a very important role. The stage 1 gives the hit array which is processed in stage 2. By performing 2 stages, we effectively consider a bigger window 5x5, instead of considering just the smaller windows 3x3. This helps us achieve a global perspective for a particular pixel and hence helping us to consider the non-adjacent neighboring pixels as well. Also, both the unconditional and the conditional mask patterns differ for both these algorithms. Hit array obtained using stage 1 is processed with the unconditional mask patterns of stage 2. And repeating this for a number of iterations, gives us the final thinned and the skeletonized output images.

Experimental Results:

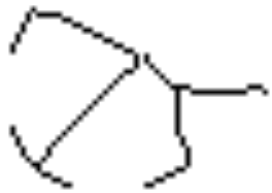


Figure 22. Jigsaw_1 thinned

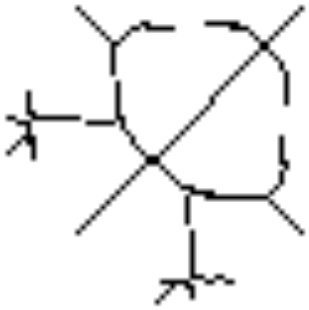


Figure 23. Jigsaw_2 Skeletonized

Discussion:

The images shown figure 22 is the thinned image and the image shown in figure 23 is the skeletonized image. As explained in the abstract, both these morphological operations are very similar but the results are very similar. In thinning, the image is shrunk to its central axis. The image is shrunk completely along its axis. If you consider a human being, thinning reduces it to the central axis of our boy, the spinal cord and all the other important and significant bones. Skeletonizing is different. In skeletonizing, the image is

shrunk to its axis along all directions. Even if there is a small protrusion on the sides, they also still remain in the skeletonized image unlike thinning. If we consider skeletonizing of human body, this operation reduces it to all the bones inside the body giving us the axis in all directions. This is the main difference between skeletonizing and thinning. Both these morphological operations help us get different interpretations from the images. These operations are very important, as they help us get the important features in an image, which is used in the advanced machine learning image classifier algorithms.

(3.d) **Counting Game**

Abstract and Motivation:

The purpose of this problem is to identify the unique pieces in the given jigsaw image. The question has 2 parts. The first part is to calculate the number of jigsaws in the image. The second part, is to calculate the unique jigsaws in the image. This concept will be very helpful to identify the unique objects in an image as well as help us figure out how to choose the critical key points for further applications like machine learning.

Approach and Procedures:

The first part of the question, is to identify the number of jigsaw pieces in the image. This can be easily done by performing the labelled component analysis. The idea of labelled component analysis is to identify the objects in the image and label them with the same label. This can be done in 2 steps:

i) Consider the presence of objects to be 1. Label all the pixels which are not surrounded by any white pixels by a unique number (Label+1). For pixels which already have a label around them, label it with the non-zero minimum label among its neighbors. Do this for all the pixels. After this, all the pixels will have a label on them. Even pixels of the same object can have different labels.

ii) The second step is to update the label with the value of the non-zero minimum of all the pixels surrounding the current pixel. When this process is repeated for a

considerable number of times, all the different objects in the image will have unique labels.

After the connected component analysis, we will have successfully labelled all the pixels with unique labels. The next step is to calculate the unique numbers in the label array to calculate the number of disconnected components in the image. Once this is performed, the number of objects, in this case jigsaws can be easily counted.

The second part of the problem is to identify the unique jigsaws in the image. This is because, some of the jigsaws in the image are just other jigsaws with different rotations or flips. After performing the connected component analysis, we know how many jigsaws are there in the puzzle. To identify the unique jigsaws, the following steps can be performed:

i) Each unique labelled pixel has to be compared with the other jigsaws (objects having different labels), pixel by pixel. If there is a match, then we say that the latter jigsaw is the repetition of the former. The same operation is repeated with the different orientation of the main image. So, we consider the first jigsaw by extracting the label of the first object. When we rotate or flip the image, we also need to make sure that the label is rotated.

ii) The first jigsaw is compared with the jigsaws having the other labels in different orientations. By rotating or flipping the label array, we are able to find the location of that compared pixel in the rotated or the flipped image.

iii) This operation is then performed for all the jigsaws and the jigsaws that are matching are stored separately. Another important thing is that, we need to allow a tolerance factor for the jigsaw matching. Even though 2 pixels are not different, they may not have the same number of pixels in them, mainly because the image is dithered before the application of this process. Hence there can be a tolerance. If the difference between one pixel and the other pixels is less than that particular value, we can say that these jigsaws are not unique. When this process is repeated for all the jigsaws, we can identify the unique jigsaws in the image.

Once the above steps are completed, we can identify the jigsaws which are unique and by counting the number of these jigsaws, we can conclude the number of unique jigsaws in the image.

Experimental Results:

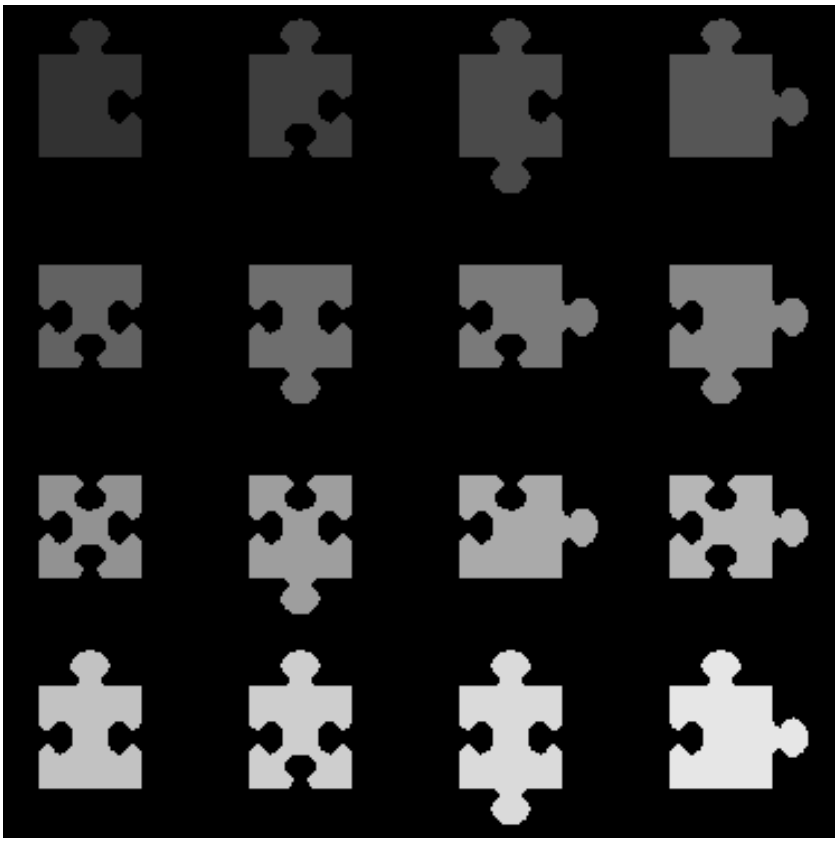


Figure 24. Jigsaws with different labels

```
**********************************
The total number of jigsaws are 16
**********************************
The jigsaws 2 and 7 are the same
The jigsaws 6 and 13 are the same
The jigsaws 7 and 11 are the same
The jigsaws 8 and 16 are the same
The jigsaws 10 and 12 are the same
The jigsaws 10 and 14 are the same
The jigsaws 12 and 14 are the same
**********************************
The number of unique jigsaws are 10
**********************************
```

Figure 24. Output obtained after evaluating the designed algorithm

Discussion:

This logic for finding the number of jigsaws and the unique number of jigsaws has been explained above. Through successful execution of the code the results shown in figure 24 have been obtained. The image shown in figure 23 has been shown to visualize the connected component analysis. The intensity values were changed for different labels. We can clearly observe that each jigsaw has got a different color. This is a concept that can be used in image segmentation. In a binary image, we just check for connected component with just 1s and 0s, but in case of image segmentation we need to check for a particular color value. Using this, we can look for the image regions with that particular color intensity. After we label that region, we can change the color of that specific region to visualize and understand the image better. This is the idea of image segmentation. A summary of the algorithm followed for identifying the unique jigsaws is given below:

i) Label the components using the connected component analysis mentioned above. The number of different labels will give us the number of jigsaw.

ii) Pick one jigsaw out, compare it with the whole image with all individual labels (jigsaws), in all orientations.

iii) If there is a match, reduced the number of unique labelled components. Repeat the process for all the 16 jigsaws.

The above algorithm was followed to identify the number of jigsaws and the number of unique jigsaws. The results have been given in figure 24.

REFERENCES:

[1] B. E. Bayer, “An optimum method for two-level rendition of continuous-tone pictures,” SPIE MILE- STONE SERIES MS, vol. 154, pp. 139–143, 1999

[2] D. Shakad, N. Arad, A. Fitzhugh, I. Sobel, "Color Diffusion: Error-Diffusion for Color Halftones", HP Labs Technical Report, HPL-96-128R1, 1996