Multimedia Course

Multimedia Course

By

Dr. Mai Kamal El Den

Lecturer, Computer Science Department,

Faculty of Computers and Artificial Intelligence,
Benha University

Multimedia Course
CSx58

Teams Link

https://teams.microsoft.com/l/team/19%3aY1IWWnAYjMbfpfphN770t_sAvrDZz01ElBVMa-oy2Dw1%40thread.tacv2/conversations?groupId=3677812 3-7bfe-4c77-86a1-fb733e69007c&tenantId=ff4a48d6-4b5e-4fd3-8266-7eafc3e6e23e

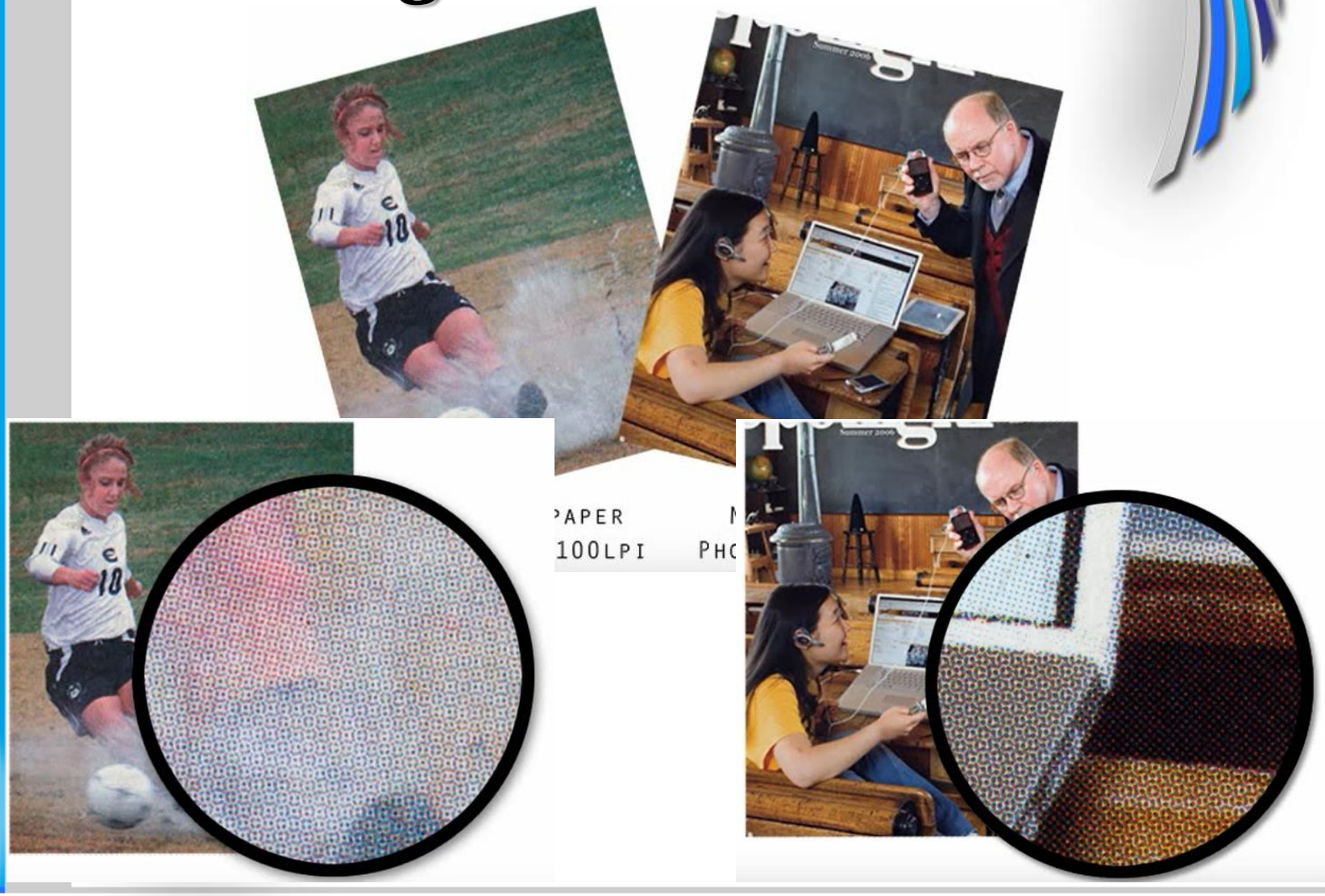
Multimedia Course

Lecture 4

Multimedia Data Representations

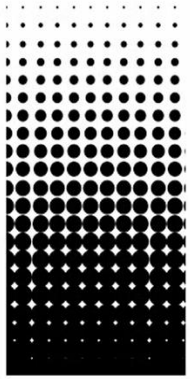
Image Dithering

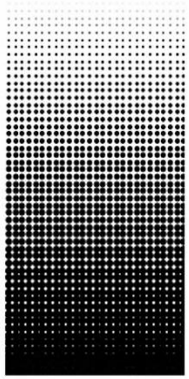
Dithering



Dithering

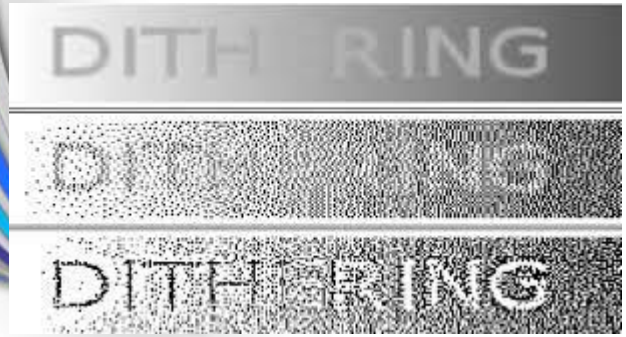
- **Dither** used to randomize quantization error, preventing large-scale patterns such as color banding in images.
- Dither is routinely used in processing of both digital audio and video data, and is often one of the last stages of mastering audio to a CD.





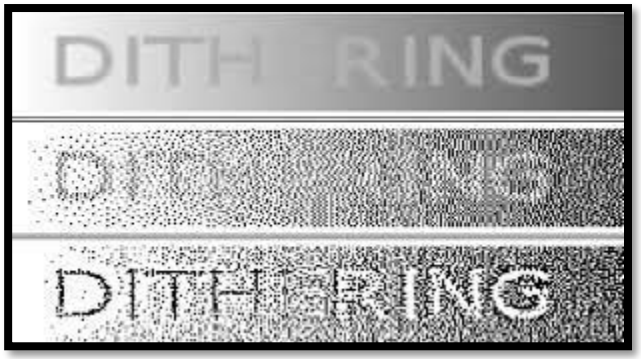
Dithering

- Dithering is the process of reducing the errors that come from image quantization techniques.
- Image dithering is an image processing operation used to create the illusion of color depth in images with a limited color palette.
- The common use of dither is converting a grayscale image to black and white.



Dithering

- **Dithering** is a process that can help in the reduction of the size without losing the quality and information of the data and helps in minimizing the quantization error.
- **Dithering** image is a technique used in graphics to create the illusion of color depth.
- An image's color depth is reduced to 8 bits per pixel or less.



Dithering

- **Dithering methods** create the appearance of more subtle shades by mixing in pixels of different colors.
  - ➢ This is similar to the way newspaper pictures produce the appearance of shades of gray, even though the actual colors are only black and white.

Dithering Types

- *kinds of dithering processes include:*
  1) Threshold Dithering.
  2) Random Dithering.
  3) Error Diffusion.
     - Floyd-Steinberg Dithering
  4) Ordered Dithering.
  5) Pattern Dithering.
  6) Halftone Dithering.

Threshold Dithering

- For every pixel: **If** the intensity < 0.5, replace with black, **else** replace with white.
  - 0.5 is the threshold .
  - The value of threshold is (256/2=128).
  - 128 is the threshold .

Threshold Dithering-Example

20	150	80	120
150	90	180	64
50	176	16	200
235	128	190	70

?	?	?	?
?	?	?	?
?	?	?	?
?	?	?	?

Threshold=128

0	150	80	120
150	90	180	64
50	176	16	200
235	128	190	70

0	1	80	120
150	90	180	64
50	176	16	200
235	128	190	70

0	1	0	0
1	0	1	0
0	1	0	1
1	1	1	0

0	1	0	120
150	90	180	64
50	176	16	200
235	128	190	70

0	1	0	0
1	0	1	0
50	176	16	200
235	128	190	70

Random Dithering

- **Random dithering:** can be considered as the process of converting a grayscale image into a black and white or monochrome image.

Original image

Random dithering

Random Dithering-Example

- **Random dithering:** generates a random number between 1 and 256:
  - ✓ **If** it is greater than the image value at that point, plot the point white, otherwise, plot it black.

✓**Example**:

20	150	80	120
150	90	180	64
50	176	16	200
235	128	190	70

?	?	?	?
?	?	?	?
?	?	?	?
?	?	?	?

Generated number is 17

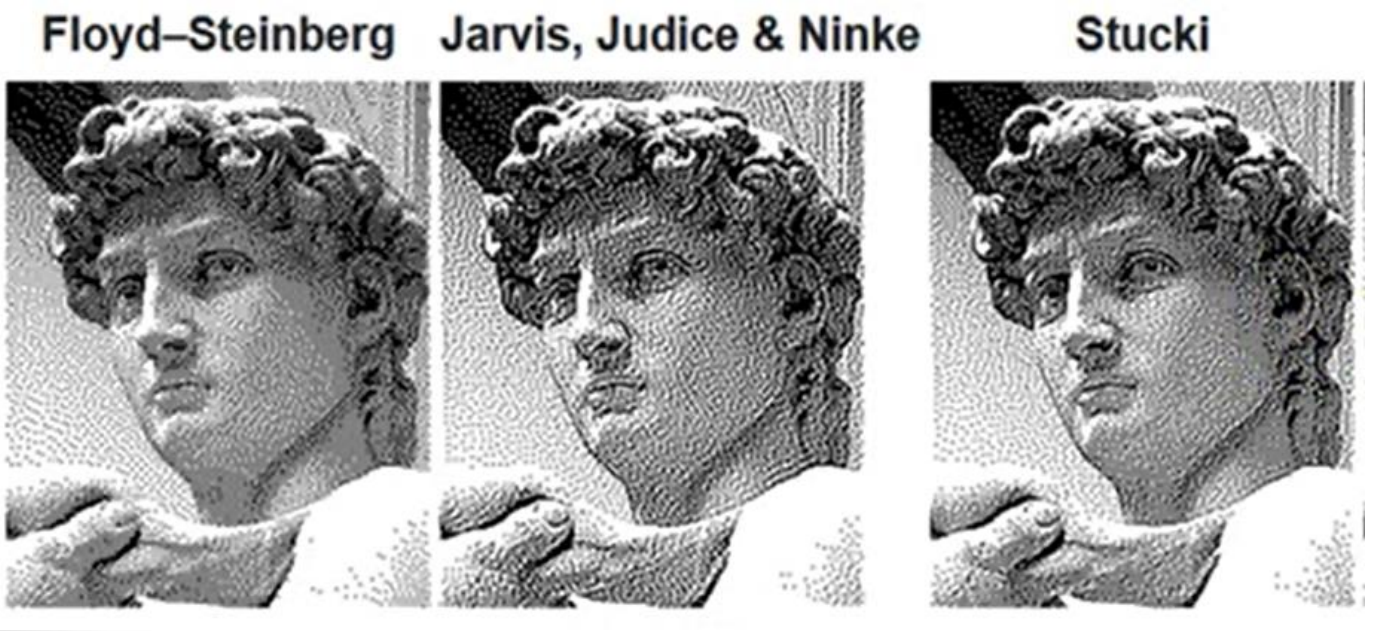
0	0	80	120
150	90	180	64
50	176	16	200
235	128	190	70

Generated number is 200

1	1	80	120
150	90	180	64
50	176	16	200
235	128	190	70

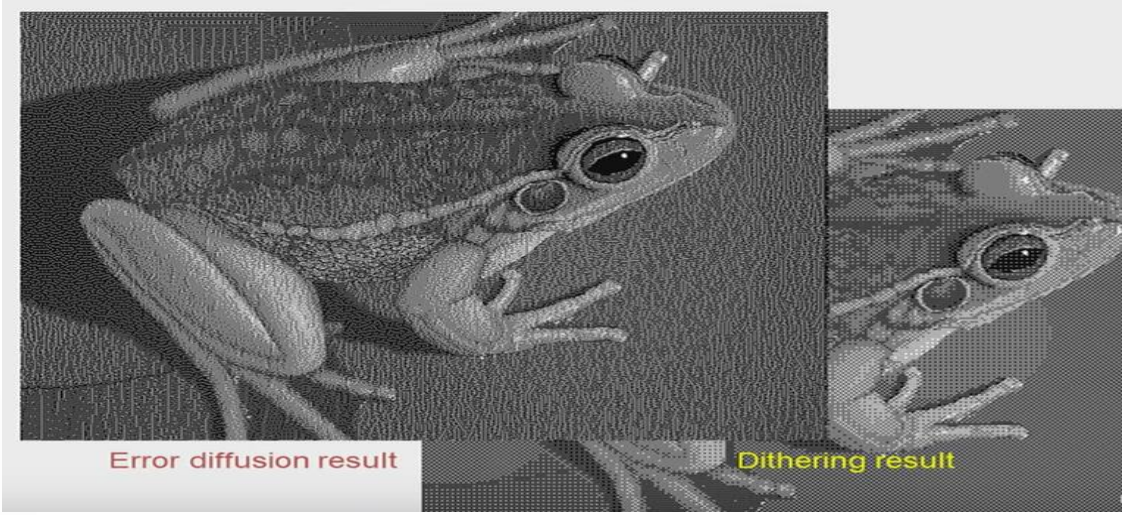
Error Diffusion

- **Floyd-Steinberg Dithering** technique generates the best results of any classical error diffusion methods.



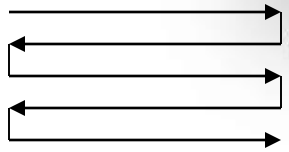
Floyd-Steinberg Dithering

- **Floyd-Steinberg Dithering** is similar to ordered dithering. The process works by choosing different patterns from the image but these patterns are relatively few repeated patterns from the colors of the image. It can be used with both gray-scale and monochromic images and also colored images.

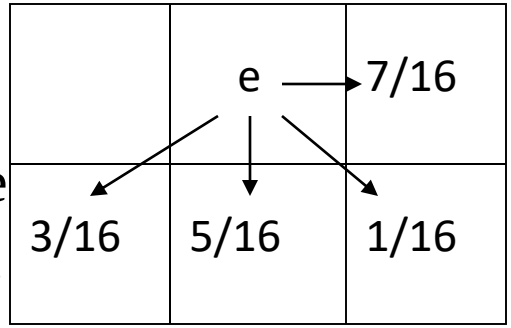


Floyd-Steinberg Dithering

1) Start at one corner and work through the image pixel by pixel
   - Usually scan top to bottom in a zig-zag



2) Threshold each pixel
3) Compute the error at that pixel: The difference between what should be there and what you did put there
   - If you made the pixel 0, *e* = original; if you made it 1, *e* = original-1
4) Propagate error to neighbors by adding some proportion of the error to each unprocessed neighbor
   - A *mask* tells you how to distribute the error.



5) Easiest to work with floating point image
   - Convert all pixels to 0-1 floating point

Floyd-Steinberg dithering

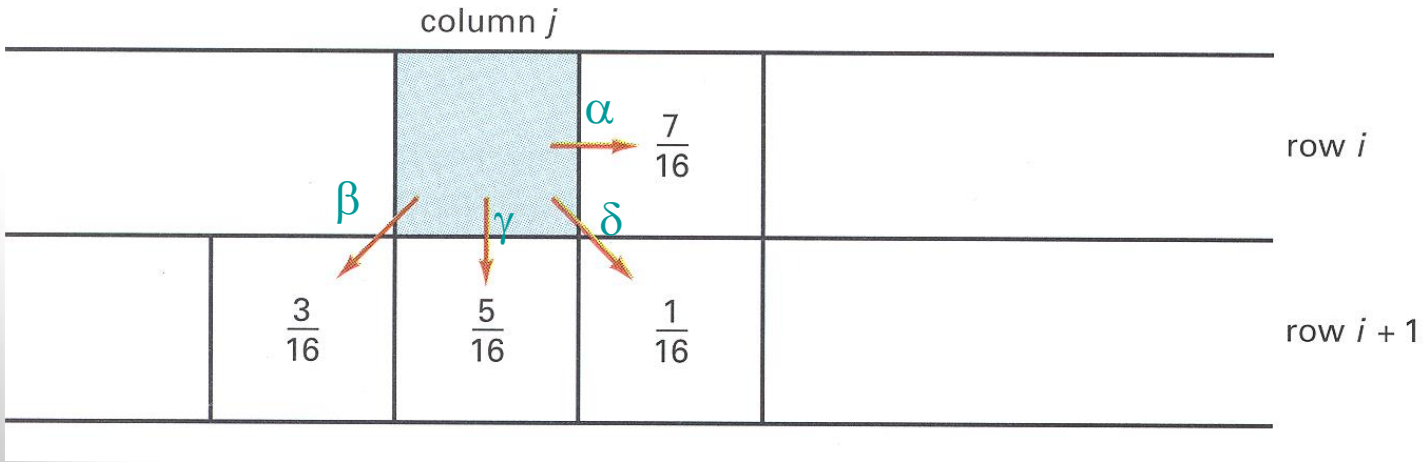
6) Any "rounding" errors are distributed to other pixels
   - Specifically to the pixels below and to the right
     - α=7/16 of the error to the pixel to the right
     - β =3/16 of the error to the pixel to the lower left
     - γ= 5/16 of the error to the pixel below
     - δ =1/16 of the error to the pixel to the lower right

- **Error = oldpixel – newpixel after threshold.**
- Assume the 1 in the middle gets "rounded" to 0

$$\begin{bmatrix} 0.00 & 0.00 & 0.00 \\ 0.00 & 1.00 & 0.00 \\ 0.00 & 0.00 & 0.00 \end{bmatrix} \rightarrow \begin{bmatrix} 0.00 & 0.00 & 0.00 \\ 0.00 & 0.00 & 0.44 \\ 0.19 & 0.31 & 0.06 \end{bmatrix}$$

Error Diffusion Dither

- Spread quantization error over neighbor pixels
  - Error dispersed to pixels right and below



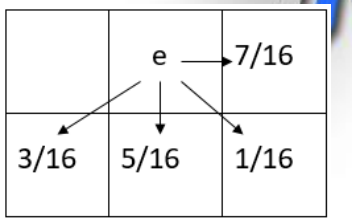
$$\alpha + \beta + \gamma + \delta = 1.0$$

Floyd-Steinberg Dithering-Example

Threshold=128

20	150	80	120
150	90	180	64
50	176	16	200
235	128	190	70

?	?	?	?
?	?	?	?
?	?	?	?
?	?	?	?



1) Compute new value pixel

   If oldpixel >= 128 -> newpixel = 255

   If oldpixel < 128 -> newpixel = 0

2) Compute error: error = 20-0=20

3) Propagation error to neighbor:

   a) 150 + α * error = 150 + (7/16) * 20 =158.75 =**159**

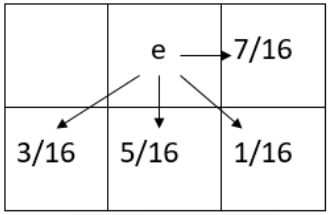
   b) 90 + (1/16)*20=91.25= **91**

   c) 150 + (5/16)*20=156.24= **156**

Floyd-Steinberg Dithering-Example

20	150	80	120
150	90	180	64
50	176	16	200
235	128	190	70

?	?	?	?
?	?	?	?
?	?	?	?
?	?	?	?



0	150	80	120
150	90	180	64
50	176	16	200
235	128	190	70

0	159	80	120
156	91	180	64
50	176	16	200
235	128	190	70

0	255	38	120
138	61	174	64
50	176	16	200
235	128	190	70

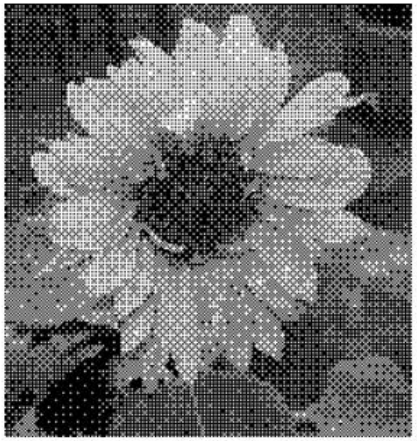
0	255	0	137
138	68	189	86
50	176	16	200
235	128	190	70

0	255	0	255
255	0	255	0
0	255	0	255
225	0	177	32

Ordered Dithering

- **Ordered Dithering:** This type of dithering process is also used for the conversion of colored images into monochrome. But this process works by choosing a different pattern from the image depending on the color presented in the working area of the image.

Original image



Ordered Dithering

Ordered Dithering

- Break the image into small blocks
- Define a *threshold matrix*
  - Use a different threshold for each pixel of the block
  - Compare each pixel to its own threshold
- The thresholds can be clustered, which looks like newsprint
- The thresholds can be "random" which looks better

- **Example**

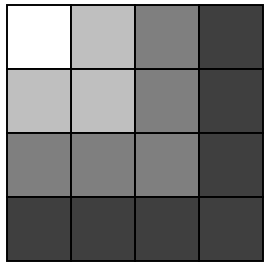
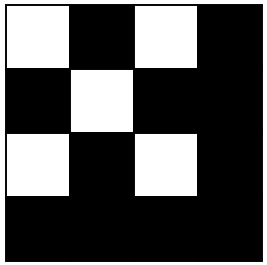


Image block

$$\begin{bmatrix} 1 & 0.75 & 0.5 & 0.25 \\ 0.75 & 0.75 & 0.5 & 0.25 \\ 0.5 & 0.5 & 0.5 & 0.25 \\ 0.25 & 0.25 & 0.25 & 0.25 \end{bmatrix}$$

Threshold matrix

$$\frac{1}{16}\begin{bmatrix} 2 & 16 & 3 & 13 \\ 10 & 6 & 11 & 7 \\ 4 & 14 & 1 & 15 \\ 12 & 8 & 9 & 5 \end{bmatrix}$$



Result

$$\begin{bmatrix} 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ 1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}$$

Ordered Dithering

- Threshold maps come in various sizes, which is typically a power of two:

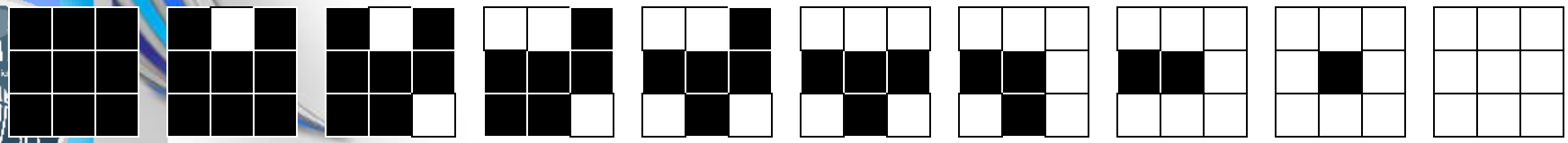
$$\mathbf{M} = \frac{1}{4} \times \begin{bmatrix} 0 & 2 \\ 3 & 1 \end{bmatrix}$$

$$\mathbf{M} = \frac{1}{16} \times \begin{bmatrix} 0 & 8 & 2 & 10 \\ 12 & 4 & 14 & 6 \\ 3 & 11 & 1 & 9 \\ 15 & 7 & 13 & 5 \end{bmatrix}$$

$$\mathbf{M} = \frac{1}{64} \times \begin{bmatrix} 0 & 32 & 8 & 40 & 2 & 34 & 10 & 42 \\ 48 & 16 & 56 & 24 & 50 & 18 & 58 & 26 \\ 12 & 44 & 4 & 36 & 14 & 46 & 6 & 38 \\ 60 & 28 & 52 & 20 & 62 & 30 & 54 & 22 \\ 3 & 35 & 11 & 43 & 1 & 33 & 9 & 41 \\ 51 & 19 & 59 & 27 & 49 & 17 & 57 & 25 \\ 15 & 47 & 7 & 39 & 13 & 45 & 5 & 37 \\ 63 & 31 & 55 & 23 & 61 & 29 & 53 & 21 \end{bmatrix}$$

Pattern Dithering

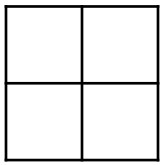
- Compute the intensity of each sub-block and index a pattern. Pattern must be squared.
- Number of patterns = n^2+1
- NOT the same as before
  - Here, each sub-block has one of a fixed number of patterns – pixel is determined only by average intensity of sub-block
  - In ordered dithering, each pixel is checked against the dithering matrix before being turned on
- Used when display resolution is higher than image resolution – not uncommon with printers
  - Use 3x3 output for each input pixel



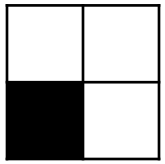
Pattern Dithering-Example

20	150	80	120
150	90	180	64
50	176	16	200
235	128	190	70

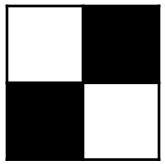
?	?	?	?
?	?	?	?
?	?	?	?
?	?	?	?



0≤1 ≤0.2



0.2≤1 ≤0.4



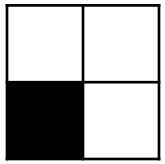
0.4≤1 ≤0.6

0.6≤1 ≤0.8

0.8≤1 ≤1

20	150	80	120
150	90	180	64
50	176	16	200
235	128	190	70

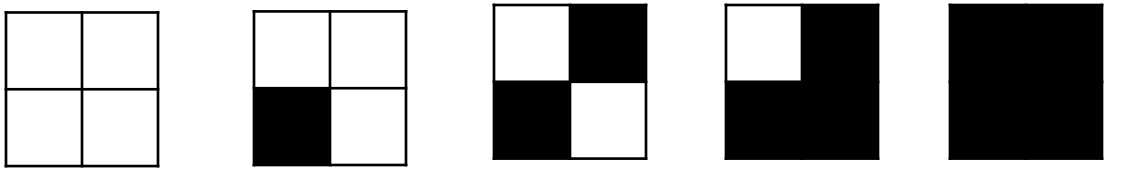
x=(20+150+150+90)/4 = 103
103/255 = 0.4



Pattern Dithering-Example

20	150	80	120
150	90	180	64
50	176	16	200
235	128	190	70

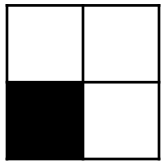
?	?	?	?
?	?	?	?
?	?	?	?
?	?	?	?



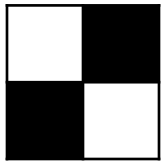
0≤1 ≤0.2 **0.2≤1 ≤0.4** **0.4≤1 ≤0.6** **0.6≤1 ≤0.8** **0.8≤1 ≤1**

20	150	80	120
150	90	180	64
50	176	16	200
235	128	190	70

x=(80+120+180+64)/4 = 111
111/255 = 0.435



x=(50+176+128+235)/4 = 147
111/255 = 0.577



Next Lecture:

LEC.4: GRAPHICS AND IMAGE DATA REPRESENTATIONS

Thank you

Any question?