

Digital Halftoning

- Many image rendering technologies only have binary output. For example, printers can either “fire a dot” or not.
- Halftoning is a method for creating the illusion of continuous tone output with a binary device.
- Effective digital halftoning can substantially improve the quality of rendered images at minimal cost.

Thresholding

- Assume that the image falls in the range of 0 to 255.
- Apply a space varying threshold, $T(i, j)$.

$$b(i, j) = \begin{cases} 255 & \text{if } X(i, j) > T(i, j) \\ 0 & \text{otherwise} \end{cases} .$$

- What is $X(i, j)$?
- Lightness
 - Larger \Rightarrow lighter
 - Used for display
- Absorptance
 - Larger \Rightarrow darker
 - Used for printing
- $X(i, j)$ will generally be in units of absorptance.

Constant Threshold

- Assume that the image falls in the range of 0 to 255.
- 255 \Rightarrow *Black* and 0 \Rightarrow *White*
- The minimum squared error quantizer is a simple threshold

$$b(i, j) = \begin{cases} 255 & \text{if } X(i, j) > T \\ 0 & \text{otherwise} \end{cases} .$$

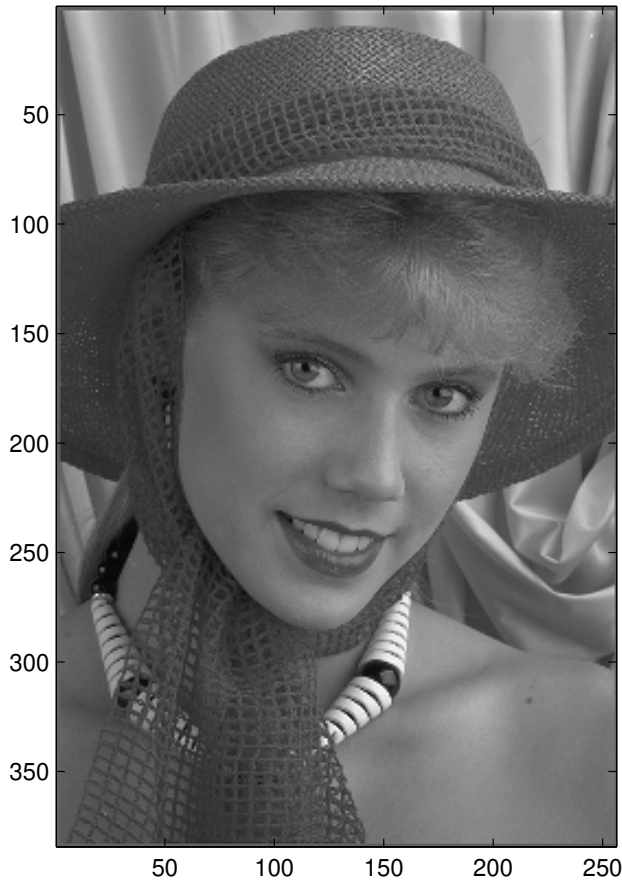
where $T = 127$.

- This produces a poor quality rendering of a continuous tone image.

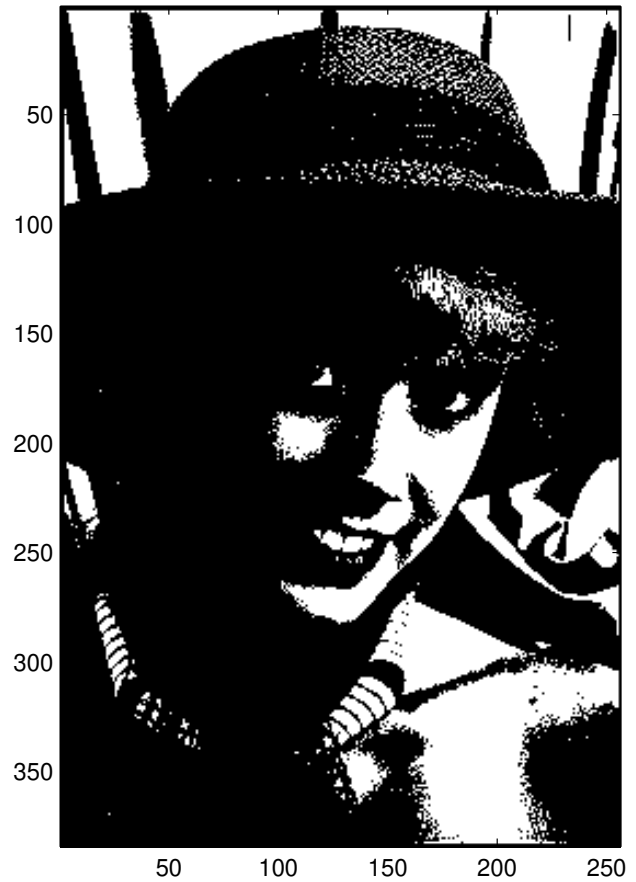
The Minimum Squared Error Solution

- Threshold each pixel
 - Pixel > 127 Fire ink
 - Pixel ≤ 127 do nothing

Original Image



Thresholded Image

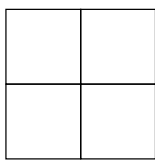


Ordered Dither

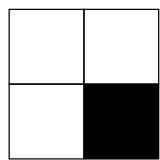
- For a constant gray level patch, turn the pixel “on” in a specified order.
- This creates the perception of continuous variations of gray.
- An $N \times N$ index matrix specifies what order to use.

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

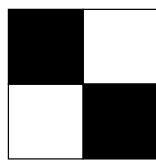
- Pixels are turned on in the following order.



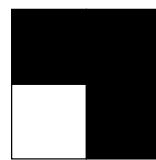
0



1



2



3



4

Implementation of Ordered Dither via Thresholding

- The index matrix can be converted to a “threshold matrix” or “screen” using the following operation.

$$T(i, j) = 255 \frac{I(i, j) + 0.5}{N^2}$$

- The $N \times N$ matrix can then be “tiled” over the image using periodic replication.

$$T(i \bmod N, j \bmod N)$$

- The ordered dither algorithm is then applied via thresholding.

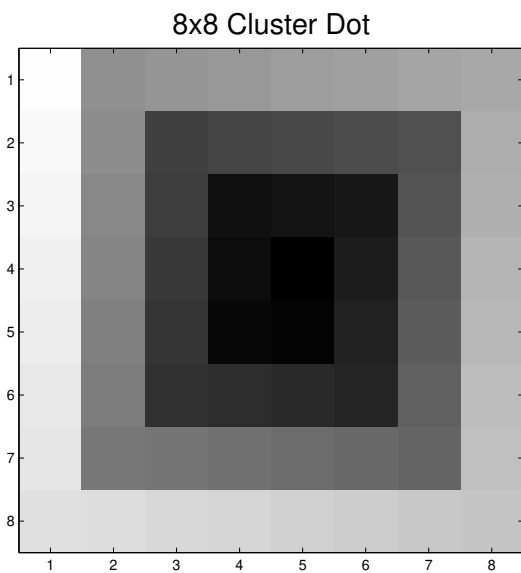
$$b(i, j) = \begin{cases} 255 & \text{if } X(i, j) > T(i \bmod N, j \bmod N) \\ 0 & \text{otherwise} \end{cases} .$$

Clustered Dot Screens

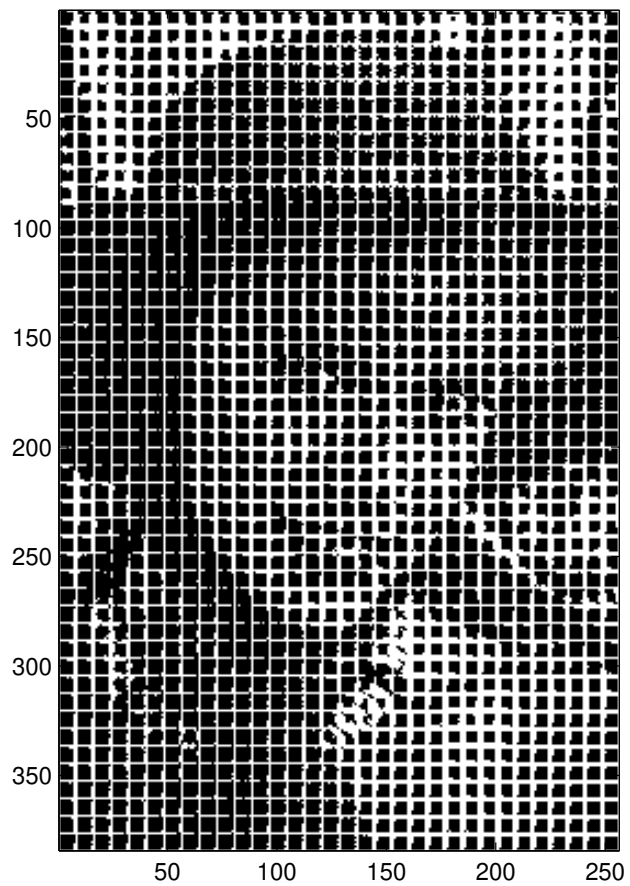
- Definition: If the consecutive thresholds are located in spatial proximity, then this is called a “clustered dot screen.
- Example for 8×8 matrix:

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

Example: 8×8 Clustered Dot Screening

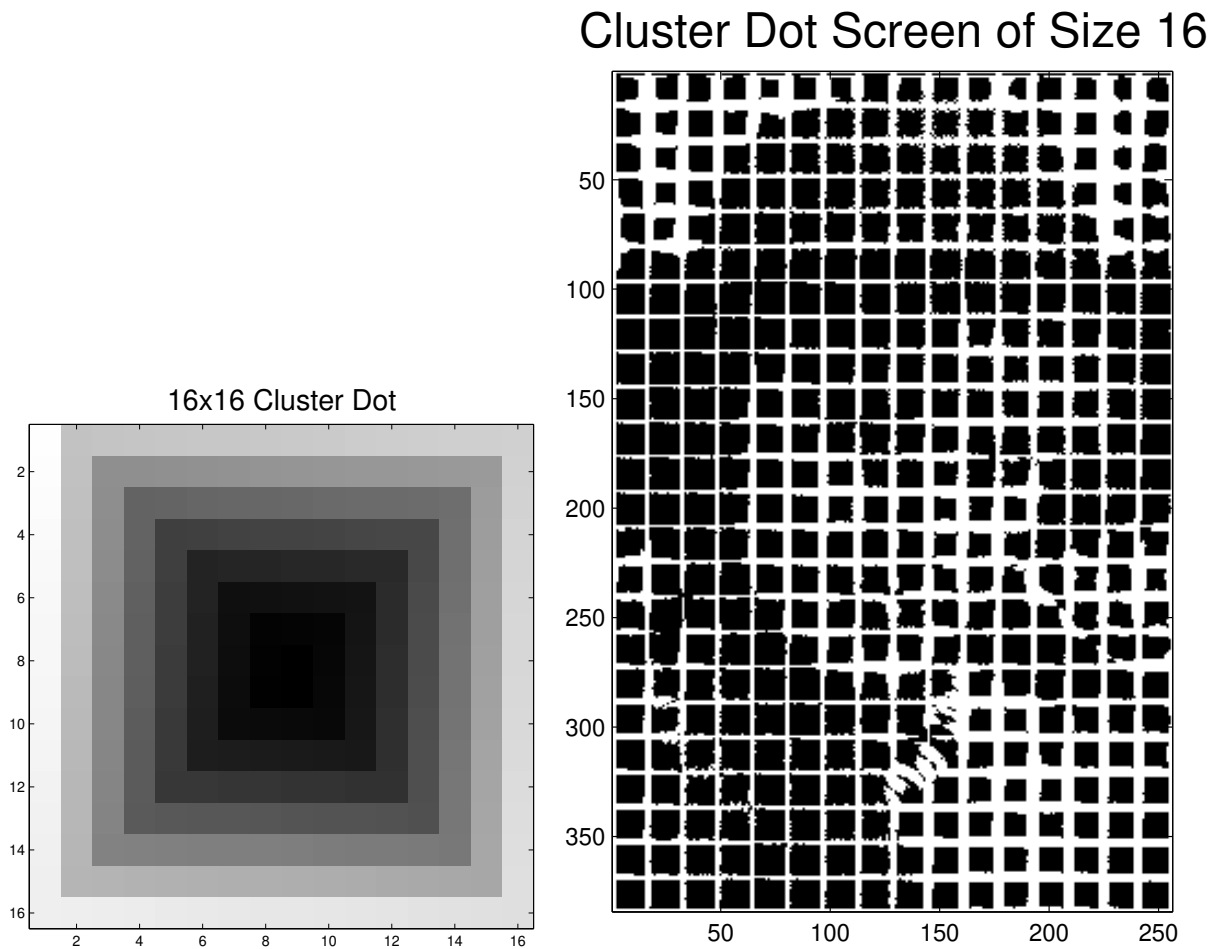


Cluster Dot Screen of Size 8



- Only supports 65 gray levels.

Example: 16×16 Clustered Dot Screening



- Support a full 257 gray levels, but has half the resolution.

Properties of Clustered Dot Screens

- Requires a trade-off between number of gray levels and resolution.
- Relatively visible texture
- Relatively poor detail rendition
- Uniform texture across entire gray scale.
- Robust performance with non-ideal output devices
 - Non-additive spot overlap
 - Spot-to-spot variability
 - Noise

Dispersed Dot Screens

- Bayer's optimum index Matrix (1973) can be defined recursively.

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

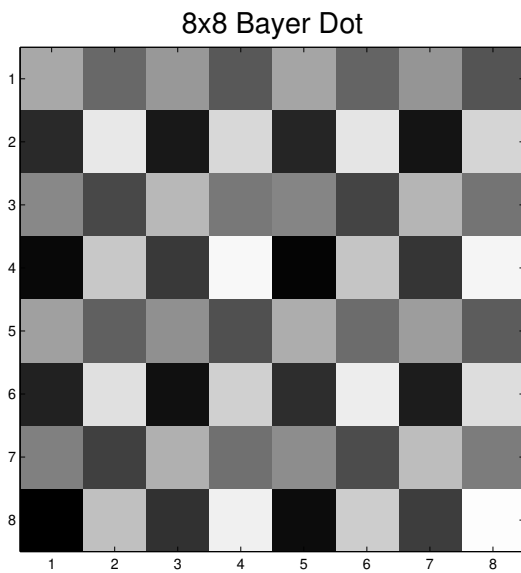
$$I_{2n} = \begin{bmatrix} 4 * I_n + 1 & 4 * I_n + 2 \\ 4 * I_n + 3 & 4 * I_n \end{bmatrix}$$

- Examples

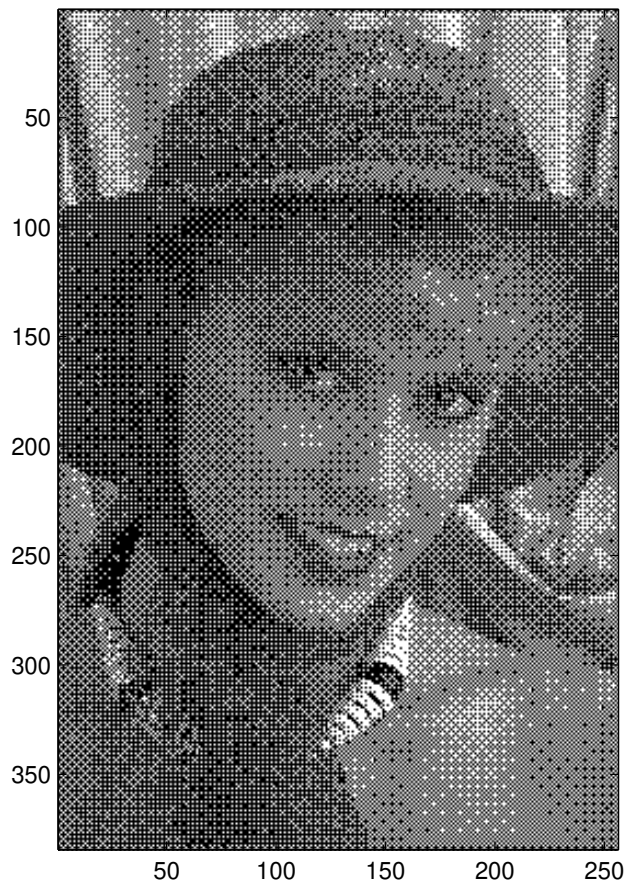
<table style="border-collapse: collapse;"> <tr><td style="padding: 2px 10px;">1</td><td style="padding: 2px 10px;">2</td></tr> <tr><td style="padding: 2px 10px;">3</td><td style="padding: 2px 10px;">0</td></tr> </table>	1	2	3	0		<table style="border-collapse: collapse;"> <tr><td style="padding: 2px 10px;">5</td><td style="padding: 2px 10px;">9</td><td style="padding: 2px 10px;">6</td><td style="padding: 2px 10px;">10</td></tr> <tr><td style="padding: 2px 10px;">13</td><td style="padding: 2px 10px;">1</td><td style="padding: 2px 10px;">14</td><td style="padding: 2px 10px;">2</td></tr> <tr><td style="padding: 2px 10px;">7</td><td style="padding: 2px 10px;">11</td><td style="padding: 2px 10px;">4</td><td style="padding: 2px 10px;">8</td></tr> <tr><td style="padding: 2px 10px;">15</td><td style="padding: 2px 10px;">3</td><td style="padding: 2px 10px;">12</td><td style="padding: 2px 10px;">0</td></tr> </table>	5	9	6	10	13	1	14	2	7	11	4	8	15	3	12	0		<table style="border-collapse: collapse;"> <tr><td style="padding: 2px 10px;">21</td><td style="padding: 2px 10px;">37</td><td style="padding: 2px 10px;">25</td><td style="padding: 2px 10px;">41</td><td style="padding: 2px 10px;">22</td><td style="padding: 2px 10px;">38</td><td style="padding: 2px 10px;">26</td><td style="padding: 2px 10px;">42</td></tr> <tr><td style="padding: 2px 10px;">53</td><td style="padding: 2px 10px;">5</td><td style="padding: 2px 10px;">57</td><td style="padding: 2px 10px;">9</td><td style="padding: 2px 10px;">54</td><td style="padding: 2px 10px;">6</td><td style="padding: 2px 10px;">58</td><td style="padding: 2px 10px;">10</td></tr> <tr><td style="padding: 2px 10px;">29</td><td style="padding: 2px 10px;">45</td><td style="padding: 2px 10px;">17</td><td style="padding: 2px 10px;">33</td><td style="padding: 2px 10px;">30</td><td style="padding: 2px 10px;">46</td><td style="padding: 2px 10px;">18</td><td style="padding: 2px 10px;">34</td></tr> <tr><td style="padding: 2px 10px;">61</td><td style="padding: 2px 10px;">13</td><td style="padding: 2px 10px;">49</td><td style="padding: 2px 10px;">1</td><td style="padding: 2px 10px;">62</td><td style="padding: 2px 10px;">14</td><td style="padding: 2px 10px;">50</td><td style="padding: 2px 10px;">2</td></tr> <tr><td style="padding: 2px 10px;">23</td><td style="padding: 2px 10px;">39</td><td style="padding: 2px 10px;">27</td><td style="padding: 2px 10px;">43</td><td style="padding: 2px 10px;">20</td><td style="padding: 2px 10px;">36</td><td style="padding: 2px 10px;">24</td><td style="padding: 2px 10px;">40</td></tr> <tr><td style="padding: 2px 10px;">55</td><td style="padding: 2px 10px;">7</td><td style="padding: 2px 10px;">59</td><td style="padding: 2px 10px;">11</td><td style="padding: 2px 10px;">52</td><td style="padding: 2px 10px;">4</td><td style="padding: 2px 10px;">56</td><td style="padding: 2px 10px;">8</td></tr> <tr><td style="padding: 2px 10px;">31</td><td style="padding: 2px 10px;">47</td><td style="padding: 2px 10px;">19</td><td style="padding: 2px 10px;">35</td><td style="padding: 2px 10px;">28</td><td style="padding: 2px 10px;">44</td><td style="padding: 2px 10px;">16</td><td style="padding: 2px 10px;">32</td></tr> <tr><td style="padding: 2px 10px;">63</td><td style="padding: 2px 10px;">15</td><td style="padding: 2px 10px;">51</td><td style="padding: 2px 10px;">3</td><td style="padding: 2px 10px;">60</td><td style="padding: 2px 10px;">12</td><td style="padding: 2px 10px;">48</td><td style="padding: 2px 10px;">0</td></tr> </table>	21	37	25	41	22	38	26	42	53	5	57	9	54	6	58	10	29	45	17	33	30	46	18	34	61	13	49	1	62	14	50	2	23	39	27	43	20	36	24	40	55	7	59	11	52	4	56	8	31	47	19	35	28	44	16	32	63	15	51	3	60	12	48	0
1	2																																																																																							
3	0																																																																																							
5	9	6	10																																																																																					
13	1	14	2																																																																																					
7	11	4	8																																																																																					
15	3	12	0																																																																																					
21	37	25	41	22	38	26	42																																																																																	
53	5	57	9	54	6	58	10																																																																																	
29	45	17	33	30	46	18	34																																																																																	
61	13	49	1	62	14	50	2																																																																																	
23	39	27	43	20	36	24	40																																																																																	
55	7	59	11	52	4	56	8																																																																																	
31	47	19	35	28	44	16	32																																																																																	
63	15	51	3	60	12	48	0																																																																																	
2×2		4×4		8×8																																																																																				

- Yields finer amplitude quantization over larger area.
- Retains good detail rendition within smaller area.

Example: 8×8 Bayer Dot Screening

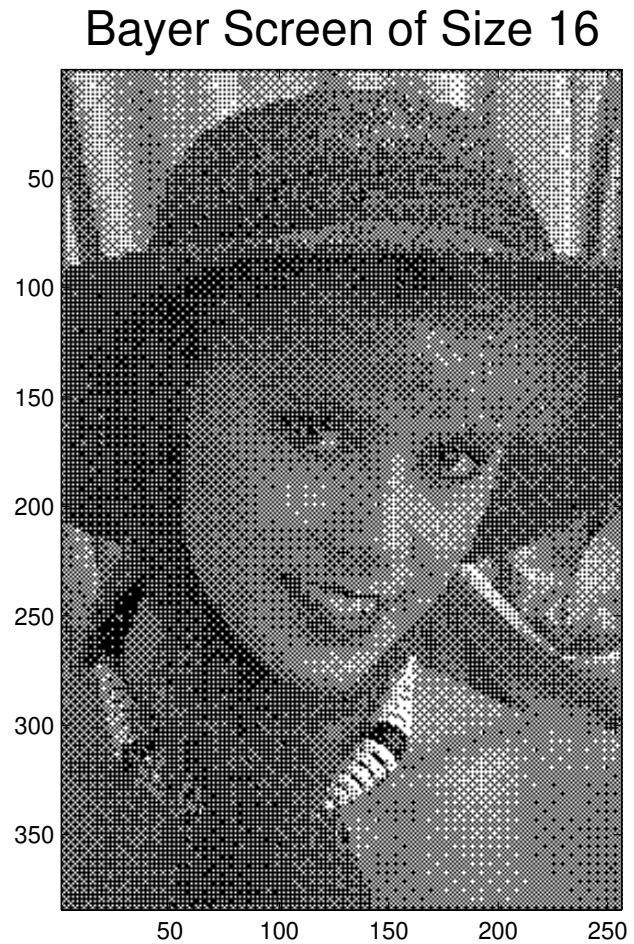
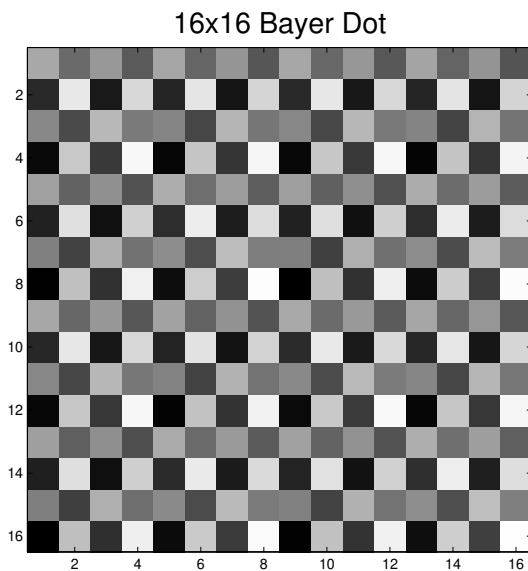


Bayer Screen of Size 8



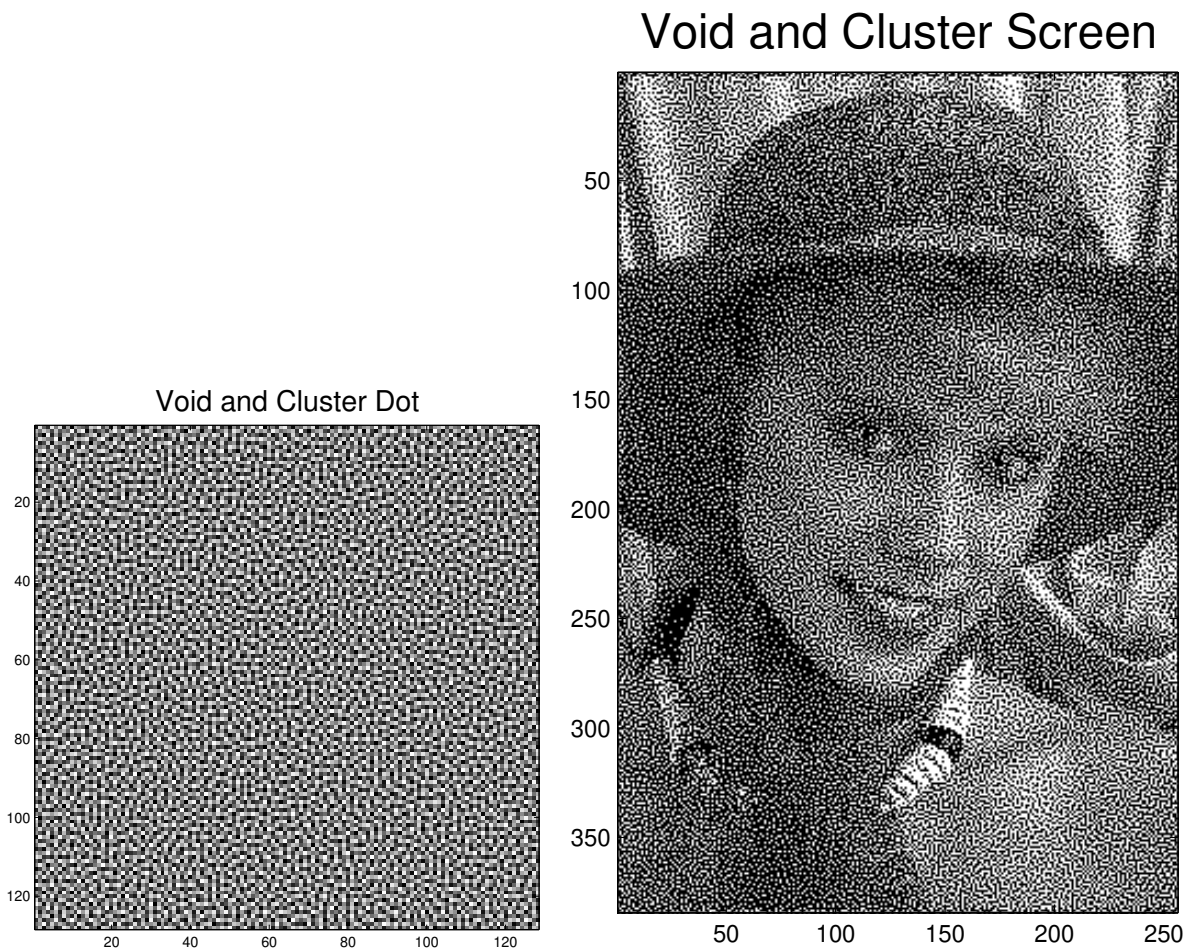
- Again, only 65 gray levels.

Example: 16×16 Bayer Dot Screening



- Doesn't look much different than the 8×8 case.
- No trade-off between resolution and number of gray levels.

Example: 128×128 Void and Cluster Screen (1989)



- Substantially improved quality over Bayer screen.

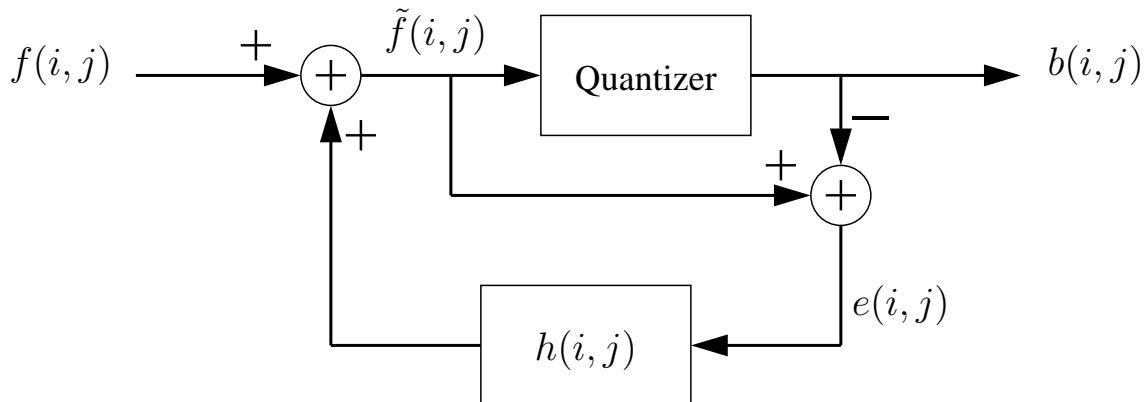
Properties of Dispersed Dot Screens

- Eliminate the trade-off between number of gray levels and resolution.
- Within any region containing K dots, the K thresholds should be distributed as uniformly as possible.
- Textures used to represent individual gray levels have low visibility.
- Improved detail rendition.
- Transitions between textures corresponding to different gray levels may be more visible.
- Not robust to non-ideal output devices
 - Requires stable formation of isolated single dots.

Error Diffusion

- Error Diffusion
 - Quantizes each pixel using a neighborhood operation, rather than a simple pointwise operation.
 - Moves through image in raster order, quantizing the result, and “pushing” the error forward.
 - Can produce better quality images than is possible with screens.

Filter View of Error Diffusion



- Equations are

$$b(i, j) = \begin{cases} 255 & \text{if } \tilde{f}(i, j) > T \\ 0 & \text{otherwise} \end{cases}$$

$$e(i, j) = \tilde{f}(i, j) - b(i, j)$$

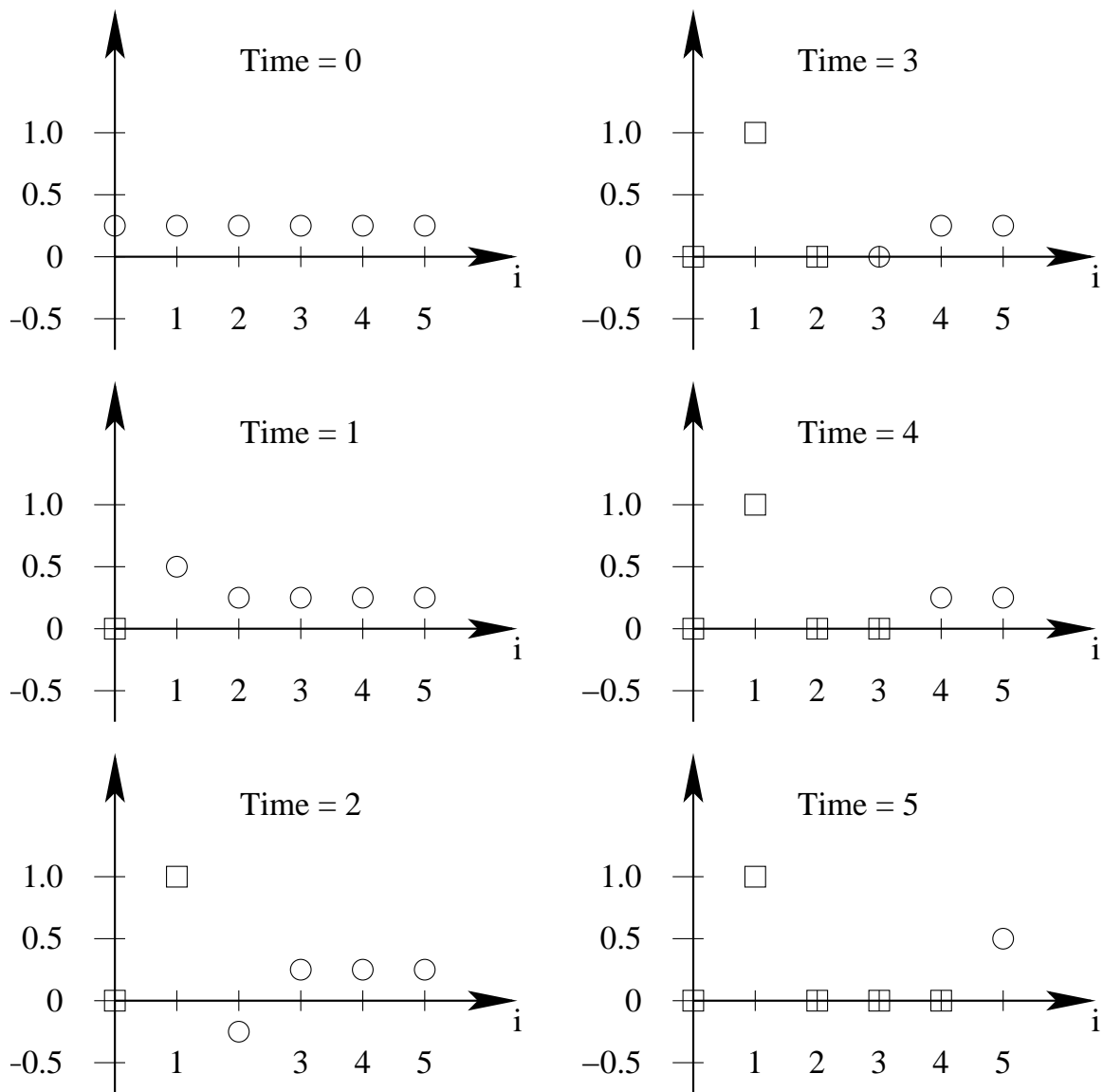
$$\tilde{f}(i, j) = f(i, j) + \sum_{k, l \in S} h(k, l) e(i - k, j - l)$$

- Parameters

- Threshold is typically $T = 127$.
- $h(k, l)$ are typically chosen to be positive and sum to 1

1-D Error Diffusion Example

- $\tilde{f}(i) \Rightarrow$ circles
- $b(i) \Rightarrow$ boxes



Two Views of Error Diffusion

- Two mathematically equivalent views of error diffusion
 - Pulling errors forward
 - Pushing errors ahead
- Pulling errors forward
 - More similar to common view of IIR filter
 - Has advantages for analysis
- Pushing errors ahead
 - Original view of error diffusion
 - Can be more easily extended to important cases when weights area time/space varying

ED: Pulling Errors Forward

1. For each pixel in the image (in raster order)

(a) Pull error forward

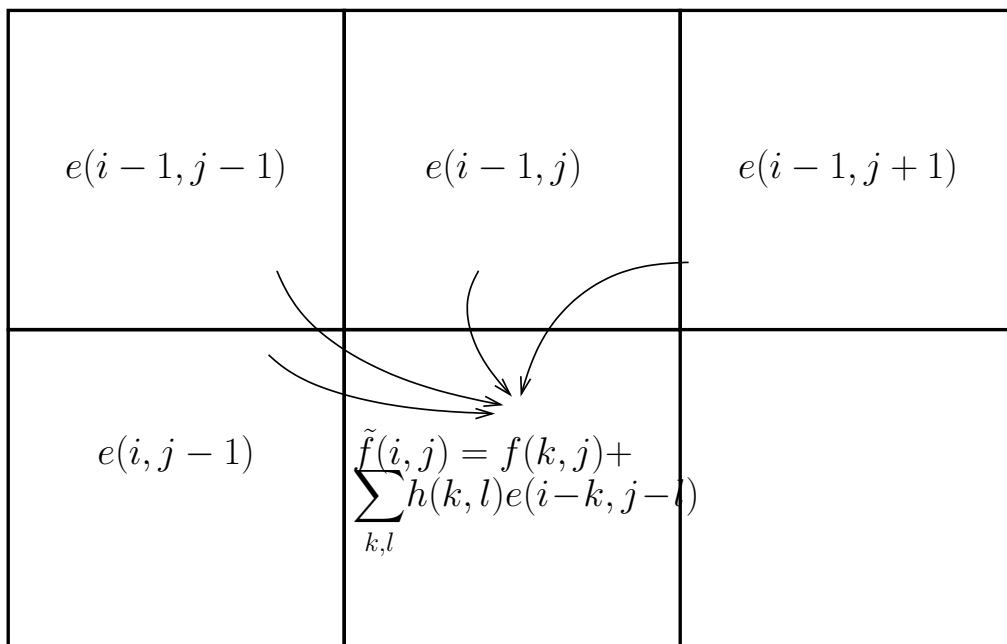
$$\tilde{f}(i, j) = f(i, j) + \sum_{k, l \in S} h(k, l)e(i - k, j - l)$$

(b) Compute binary output

$$b(i, j) = \begin{cases} 255 & \text{if } \tilde{f}(i, j) > T \\ 0 & \text{otherwise} \end{cases}$$

(c) Compute pixel's error

$$e(i, j) = \tilde{f}(i, j) - b(i, j)$$



2. Display binary image $b(i, j)$

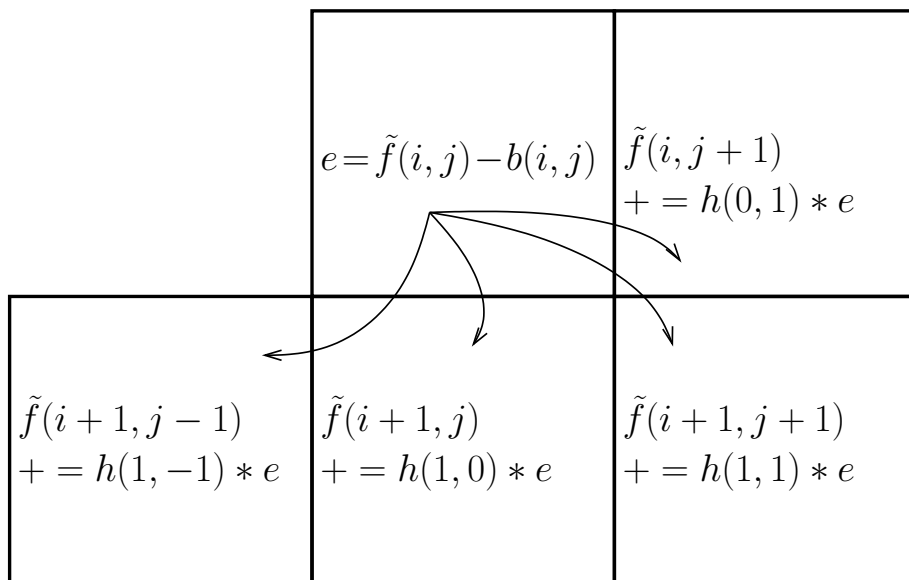
ED: Pushing Errors Ahead

1. Initialize $\tilde{f}(i, j) \leftarrow f(i, j)$
2. For each pixel in the image (in raster order)

(a) Compute

$$b(i, j) = \begin{cases} 255 & \text{if } \tilde{f}(i, j) > T \\ 0 & \text{otherwise} \end{cases}$$

(b) Diffuse error forward using the following scheme



3. Display binary image $b(i, j)$

Commonly Used Error Diffusion Weights

- Floyd and Steinberg (1976)

		$7/16$
$3/16$	$5/16$	$1/16$

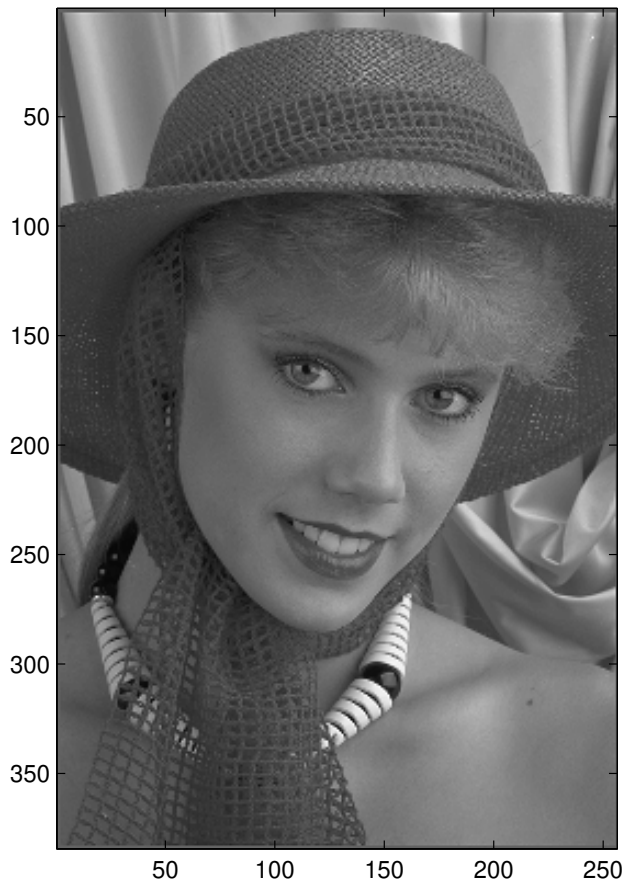
- Jarvis, Judice, and Ninke (1976)

			$7/48$	$5/48$
$3/48$	$5/48$	$7/48$	$5/48$	$3/48$
$1/48$	$3/48$	$5/48$	$3/48$	$1/48$

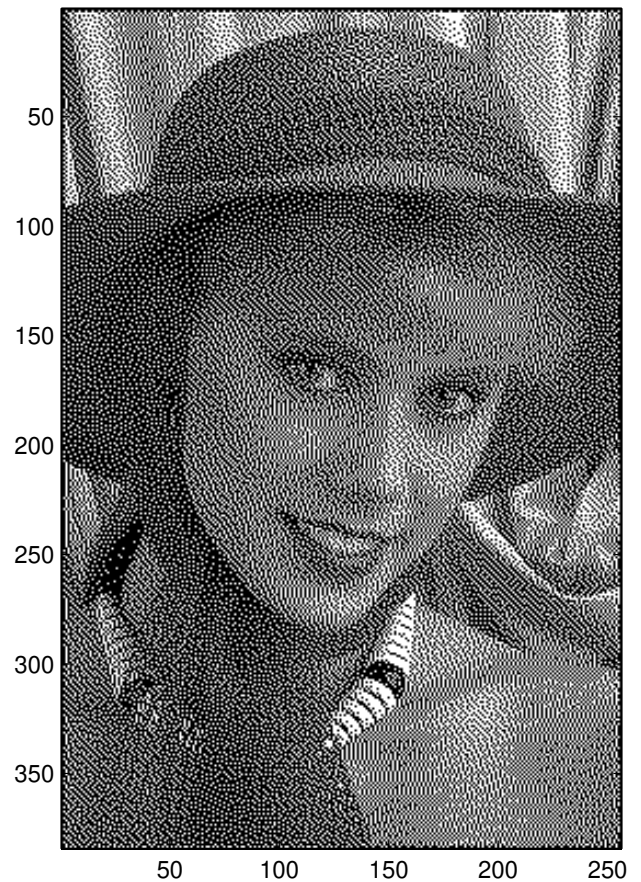
Floyd Steinberg Error Diffusion (1976)

- Process pixels in neighborhoods by “diffusing error” and quantizing.

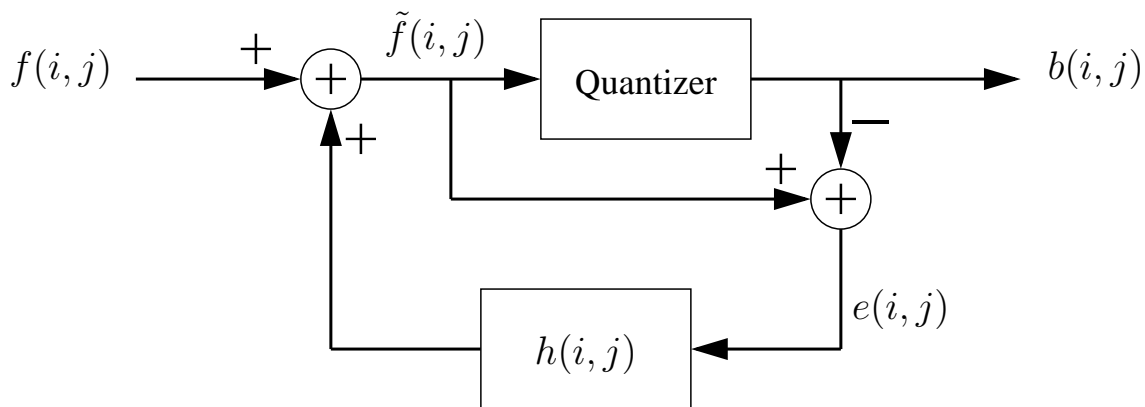
Original Image



Floyd and Steinberg Error Diffusion



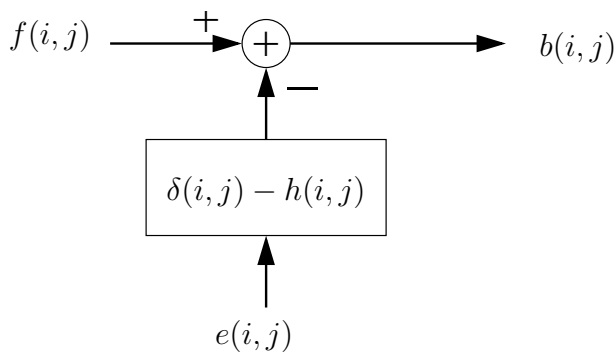
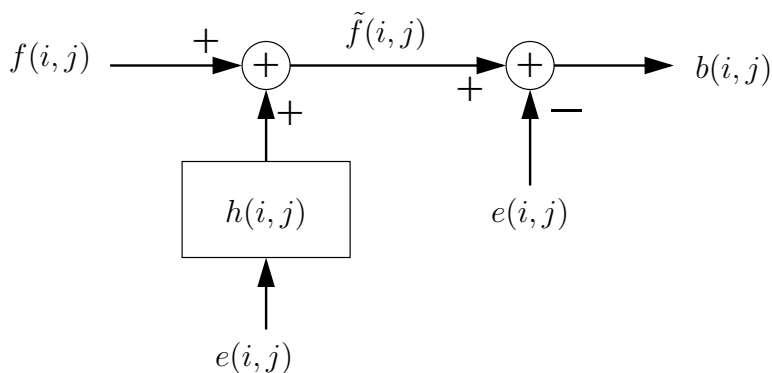
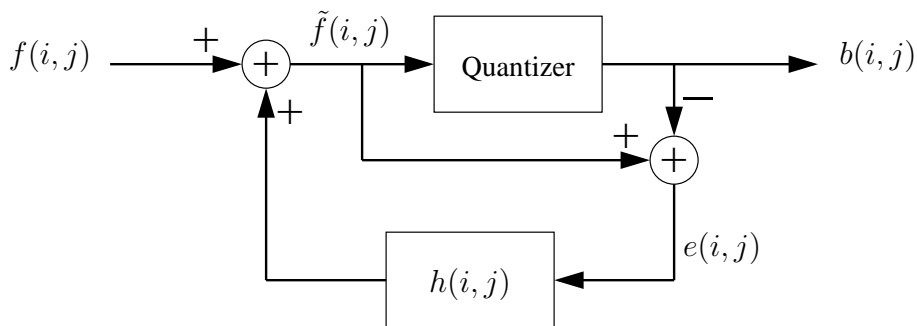
Quantization Error Modeling for Error Diffusion



- Quantization error is commonly assumed to be:
 - Uniformly distributed on $[-0.5, 0.5]$
 - Uncorrelated in space
 - Independent of signal $\tilde{f}(i, j)$
 - $E[e(i, j)] = 0$
 - $E[e(i, j)e(i + k, j + l)] = \frac{\delta(k, l)}{12}$

Modified Error Diffusion Block Diagram

- The error diffusion block diagram can be rearranged to facilitate error analysis



Error Diffusion Spectral Analysis

- So we see that

$$b(i, j) = f(i, j) - (\delta(i, j) - h(i, j)) * e(i, j)$$

rewriting ...

$$f(i, j) - b(i, j) = \underbrace{(\delta(i, j) - h(i, j))}_{\text{high pass filter}} * \underbrace{e(i, j)}_{\text{quantization error}}$$

- Display error is $f(i, j) - b(i, j)$
- Quantization error is $e(i, j)$
- Display error is a high pass version of quantization error
- Human visual system is less sensitive to high spatial frequencies

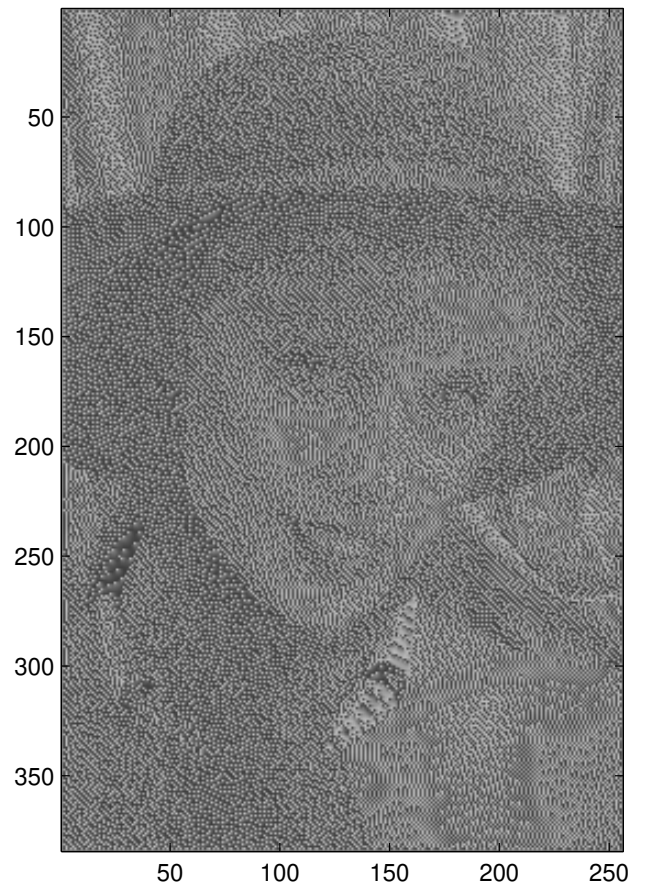
Error Image in Floyd Steinberg Error Diffusion

- Process pixels in neighborhoods by “diffusing error” and quantizing.

Original Image



Quantizer Error Image



Correlation of Quantization Error and Image

- Quantizer error spectrum is unknown
- Quantizer error model

$$\begin{aligned} E(\mu, \nu) &= \rho F(\mu, \nu) + R(\mu, \nu) \\ &= \rho(\text{Image}) + (\text{Residual}) \end{aligned}$$

- ρ represents correlation between quantizer error and image

Weight	ρ
1-D	0.0
Floyd and Steinberg	0.55
Jarvis, Judice, and Ninke	0.8

- Using this model, we have

$$\begin{aligned} B(\mu, \nu) &= F(\mu, \nu) - (1 - H(\mu, \nu)) E(\mu, \nu) \\ &= [1 - \rho(1 - H(\mu, \nu))] F(\mu, \nu) + \text{noise} \end{aligned}$$

- This is unsharp masking

Additional Topics

- Pattern Printing
- Dot Profiles
- Halftone quality metrics
 - Radially averaged power spectrum (RAPS)
 - Weighted least squares with HVS contrast sensitivity function
 - Blue noise dot patterns
- Error diffusion
 - Unsharp masking effects
 - Serpentine scan patterns
 - Threshold dithering
 - TDED
- Least squared halftoning
- Printing and display technologies
 - Electrophotographic
 - Inkjet