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.
- $255 \Rightarrow \text{Black}$ and $0 \Rightarrow \text{White}$
- 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
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.
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.

\[ T(i \mod N, j \mod N) \]

• The ordered dither algorithm is then applied via thresholding.

\[ b(i, j) = \begin{cases} 
255 & \text{if } X(i, j) > T(i \mod N, j \mod N) \\
0 & \text{otherwise}
\end{cases} \]
Clustered Dot Screens

• Definition: If the consecutive thresholds are located in spatial proximity, then this is called a “clustered dot screen.

• Example for $8 \times 8$ matrix:

<p>| | | | | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>62</td>
<td>57</td>
<td>48</td>
<td>36</td>
<td>37</td>
<td>49</td>
<td>58</td>
<td>63</td>
</tr>
<tr>
<td>56</td>
<td>47</td>
<td>35</td>
<td>21</td>
<td>22</td>
<td>38</td>
<td>50</td>
<td>59</td>
</tr>
<tr>
<td>46</td>
<td>34</td>
<td>20</td>
<td>10</td>
<td>11</td>
<td>23</td>
<td>39</td>
<td>51</td>
</tr>
<tr>
<td>33</td>
<td>19</td>
<td>9</td>
<td>3</td>
<td>0</td>
<td>4</td>
<td>12</td>
<td>24</td>
</tr>
<tr>
<td>32</td>
<td>18</td>
<td>8</td>
<td>2</td>
<td>1</td>
<td>5</td>
<td>13</td>
<td>25</td>
</tr>
<tr>
<td>45</td>
<td>31</td>
<td>17</td>
<td>7</td>
<td>6</td>
<td>14</td>
<td>26</td>
<td>40</td>
</tr>
<tr>
<td>55</td>
<td>44</td>
<td>30</td>
<td>16</td>
<td>15</td>
<td>27</td>
<td>41</td>
<td>52</td>
</tr>
<tr>
<td>61</td>
<td>54</td>
<td>43</td>
<td>29</td>
<td>28</td>
<td>42</td>
<td>53</td>
<td>60</td>
</tr>
</tbody>
</table>
Example: $8 \times 8$ Clustered Dot Screening

- Only supports 65 gray levels.
Example: $16 \times 16$ 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.

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

\[
I_{2n} = \begin{bmatrix}
4 \cdot I_n + 1 & 4 \cdot I_n + 2 \\
4 \cdot I_n + 3 & 4 \cdot I_n
\end{bmatrix}
\]

• Examples

\[
\begin{array}{cccccc}
1 & 2 & 3 & 0 \\
5 & 9 & 6 & 10 \\
13 & 14 & 2 & 12 \\
7 & 11 & 4 & 8 \\
15 & 3 & 12 & 0
\end{array}
\]

\[
\begin{array}{cccccc}
21 & 37 & 25 & 41 & 22 & 38 \\
53 & 5 & 57 & 9 & 54 & 6 \\
29 & 45 & 17 & 33 & 30 & 46 \\
61 & 13 & 49 & 1 & 62 & 14 \\
23 & 39 & 27 & 43 & 20 & 36 \\
55 & 7 & 59 & 11 & 52 & 4 \\
31 & 47 & 19 & 35 & 28 & 44 \\
63 & 15 & 51 & 3 & 60 & 12 \\
\end{array}
\]

\[
2 \times 2 \quad 4 \times 4 \quad 8 \times 8
\]

• Yields finer amplitude quantization over larger area.

• Retains good detail rendition within smaller area.
Example: $8 \times 8$ Bayer Dot Screening

- 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: $128 \times 128$ 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

\[ f(i, j) \xrightarrow{+} \tilde{f}(i, j) \xrightarrow{+} b(i, j) \]

\[ h(i, j) \]

\[ e(i, j) \]

- 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

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

   $\tilde{f}(i + 1, j + 1) = h(1, 1) \times e$

   $\tilde{f}(i + 1, j) = h(1, 0) \times e$

   $\tilde{f}(i + 1, j - 1) = h(1, -1) \times e$

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

3. Display binary image $b(i, j)$
Commonly Used Error Diffusion Weights

- Floyd and Steinberg (1976)

- Jarvis, Judice, and Ninke (1976)
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

\[ b(i, j) = f(i, j) - (\delta(i, j) - h(i, j)) \ast e(i, j) \]

rewriting ...

\[ f(i, j) - b(i, j) = (\delta(i, j) - h(i, j)) \ast e(i, j) \]

- 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.
Correlation of Quantization Error and Image

- Quantizer error spectrum is unknown
- Quantizer error model

\[ E(\mu, \nu) = \rho F(\mu, \nu) + R(\mu, \nu) \]

\[ = \rho \text{(Image)} + \text{(Residual)} \]

- \( \rho \) represents correlation between quantizer error and image

<table>
<thead>
<tr>
<th>Weight</th>
<th>( \rho )</th>
</tr>
</thead>
<tbody>
<tr>
<td>1-D</td>
<td>0.0</td>
</tr>
<tr>
<td>Floyd and Steinberg</td>
<td>0.55</td>
</tr>
<tr>
<td>Jarvis, Judice, and Ninke</td>
<td>0.8</td>
</tr>
</tbody>
</table>

- Using this model, we have

\[ B(\mu, \nu) = F(\mu, \nu) - (1 - H(\mu, \nu)) E(\mu, \nu) \]

\[ = [1 - \rho (1 - H(\mu, \nu))] F(\mu, \nu) + \text{noise} \]

- 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