Optimal Dithering Pattern for 3DP

Wonjoon Cho

MIT Design Laboratory

Outline

- Objective
- Overview of Digital Halftoning
- Bayer's Algorithm
- 3DP Application
  - Extraction of 1D Patterns
  - Minimum Run-Length Requirement
  - Pattern Memory
- Generalized Bayer's Algorithm
- Summary

Objective

Development of a *halftoning technique* for local composition control in 3DP

Criteria:

- Minimization of low frequency textures
- Minimum run-length requirement
- Limit of 3DP machine's pattern memories

Overview of Digital Halftoning

Halftone approximation:

Simulation of continuous-tone gray scales for *bi-level* displays and hardcopy devices

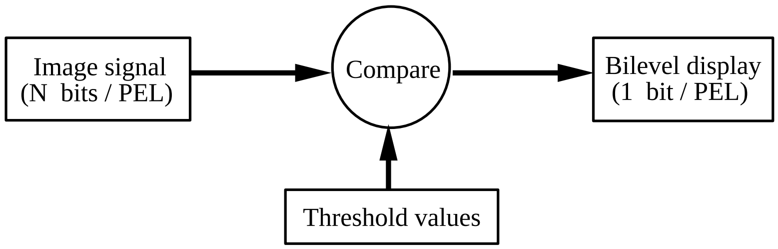


Figure: Basic halftoning process

- Ordered dither approach: A set of *ordered* threshold values τ_{ij} are used.
  - Clustered-dot ordered dither:
    - ∗ On-PELs are *adjacent* to each other.
    - ∗ *Outward growing pattern* $\Longrightarrow$ create an effect of increasing dot size.
    - ∗ Preferred if devices are poor at accommodating isolated on-PELs, e.g. halftone laser printer.
    - ∗ Apparent *low frequency textures* in an area of uniform gray levels.

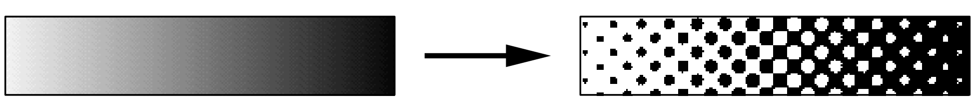


Figure: Gray scale ramp: clustered-dot ordered dither

– Dispersed-dot ordered dither:
  * On-PELs are *not* necessarily *adjacent* to each other.
  * Yield *high frequency* fidelity.

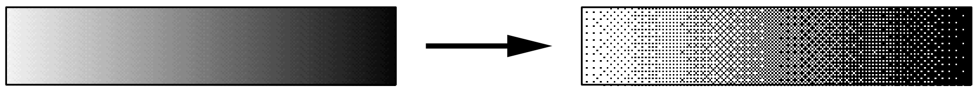


Figure: Gray scale ramp: dispersed-dot ordered dither

- Error diffusion algorithm: Spread halftone approximation error over neighboring PELs to *minimize the global error.*

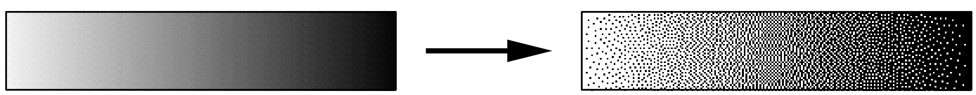


Figure: Gray scale ramp: error diffusion approach

Comments on 3DP application:

- Clustered-dot ordered dither:
  – Less possibility of missing droplet placements
  – Undesirable low frequency texture patterns
- Dispersed-dot ordered dither:
  – High frequency fidelity
    $\Longrightarrow$ desirable material distribution
  – Minimum run-length requirement
- Error diffusion algorithm:
  – Precise description of material composition
  – Hard to utilize pattern memories of 3DP machine

Bayer's Algorithm

- **Given:** The *size*, m, of a 2^m by 2^m dither matrix $\Longrightarrow$ The number of gray levels $= 2^m \times 2^m + 1$.

- **Find:** A rule for *optimum dither pattern* i.e., the optimum *order* in which new dots are added to the halftone cell as gray scale is varied.

- **Optimality:** Minimization of *low frequency texture* in an area of *uniform* gray scale.

$$\mathbf{T}^{(2m)} = \begin{pmatrix} 4\mathbf{T}^{(m)} & 4\mathbf{T}^{(m)} + 2\mathbf{U}^{(m)} \\ 4\mathbf{T}^{(m)} + 3\mathbf{U}^{(m)} & 4\mathbf{T}^{(m)} + \mathbf{U}^{(m)} \end{pmatrix}$$

where, $\mathbf{U}^{(m)} = \begin{pmatrix} 1 & 1 & \ldots & 1 \\ 1 & 1 & \ldots & 1 \\ . & . & \ldots & . \\ 1 & 1 & \ldots & 1 \end{pmatrix}$ and $\mathbf{T}^{(2)} = \begin{pmatrix} 0 & 2 \\ 3 & 1 \end{pmatrix}$

Example: $m = 2 \Longrightarrow 4 \times 4 + 1 = 17$ gray levels

$$\mathbf{T}^{(4)} = \begin{pmatrix} 4\mathbf{T}^{(2)} & 4\mathbf{T}^{(2)} + 2\mathbf{U}^{(2)} \\ 4\mathbf{T}^{(2)} + 3\mathbf{U}^{(2)} & 4\mathbf{T}^{(2)} + \mathbf{U}^{(2)} \end{pmatrix} = \begin{pmatrix} 0 & 8 & 2 & 10 \\ 12 & 4 & 14 & 6 \\ 3 & 11 & 1 & 9 \\ 15 & 7 & 13 & 5 \end{pmatrix}$$

3DP Application

(1) Extraction of 1D Patterns

Find all 1D patterns necessary to describe those $2^{2m}+1$ number of 2D Bayer's dither patterns.

Example: If $m = 2$, 12 1D patterns are necessary to describe 17 2D dither patterns.

$\Longrightarrow$ 0000 1000 0100 0010 0001 1010

0101 1110 1101 1011 0111 1111

(2) Minimum Run-Length Requirement

Among those 1D patterns, select patterns that satisfy the minimum run-length requirement.

Example: If $m = 2$ & $minRunLen = 2$, only 5 patterns (1110, 1101, 1011, 0111, 1111) satisfy the minimum run-length requirement.

$\Longrightarrow$ 0000 & 1111 would be done with transition bits, and hence 4 pattern memories may be needed in this case

$\Longrightarrow$ *Only high* intensity levels (levels 15, 16, 17) and the lowest one (level 1) can be described by such patterns and transition instructions.

0 0 0 0	1 1 1 1	1 1 1 1	1 1 1 1
0 0 0 0	1 1 0 1	1 1 1 1	1 1 1 1
0 0 0 0	1 1 1 1	1 1 1 1	1 1 1 1
0 0 0 0	0 1 1 1	0 1 1 1	1 1 1 1
level 1	level 15	level 16	level 17

Minimum Run-Length Requirement (continued)

Construct a $2^m \times minRunLen$ by 2^m halftone cell by replicating each PEL of the 2^m by 2^m halftone cell minimum run-length times along the fast-axis direction.

$$\begin{matrix} 1&0&1&0 \\ 0&1&0&0 \\ 1&0&1&0 \\ 0&0&0&0 \end{matrix} \longrightarrow \begin{matrix} 1&1&0&0&1&1&0&0 \\ 0&0&1&1&0&0&0&0 \\ 1&1&0&0&1&1&0&0 \\ 0&0&0&0&0&0&0&0 \end{matrix}$$

Figure: Level 6 ($m = 2$, $minRunLen = 2$)

- The modified halftone cell still provides $2^{2m} + 1$ dither patterns.

- All the associated 1D patterns will satisfy the minimum run-length requirement.

  Example: 12 1D patterns ($m = 2$, $minRunLen = 2$)

00000000	11000000	00110000	00001100	00000011	11001100
00110011	11111100	11110011	11001111	00111111	11111111

(3) Pattern Memory

If the number N of 1D patterns[a] is greater than the number P of pattern memories of 3DP machine, we consider $\binom{N}{P}$ number of combinations of such 1D patterns.

Example: ($m = 2$, $minRunLen = 2$, $N = 10$)

Set $P = 8$, then $\binom{10}{8} = 45$ possible combinations exist.

00000011 00001100 00110000 11000000 00110011

11001100 11001111 11111100 00000000 11111111

Figure: One possible combination ($m = 2$, $minRunLen = 2$)

$\Longrightarrow$ 14 dithering patterns can be described.

$\Longrightarrow$ Levels: 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 17

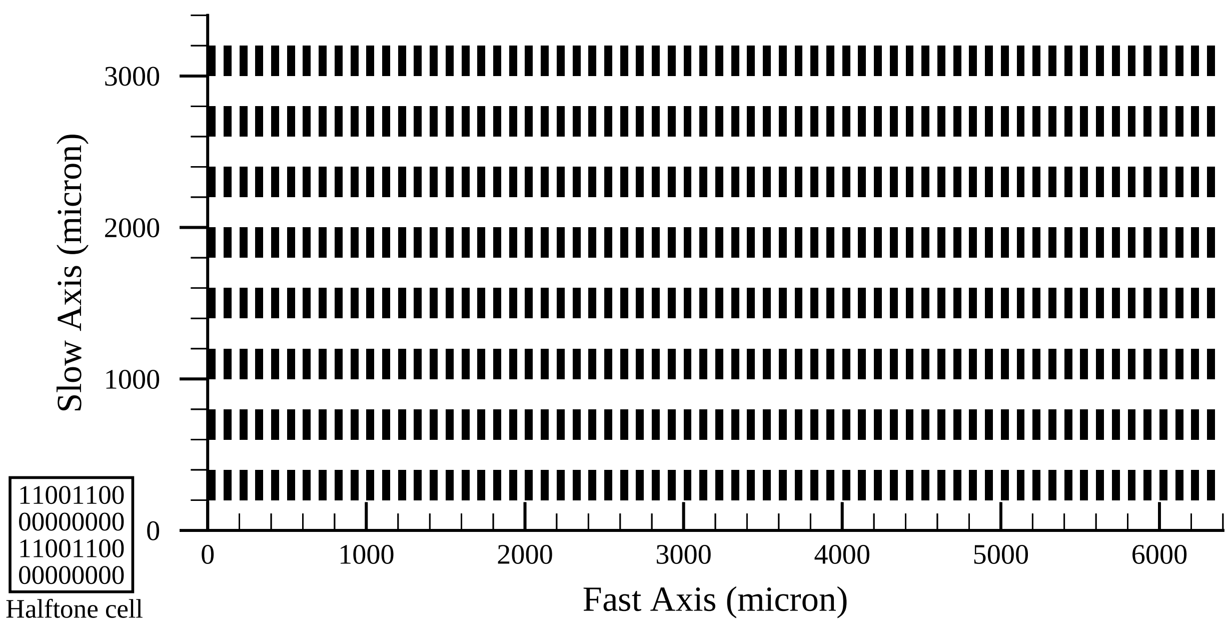


Figure: Level 5 (PEL size = $25\mu \times 200\mu$)

[a] excluding 000...00 and 111...11

Generalized Bayer's Algorithm

- **Given:**
  - m: parameter that controls the size (2^m by 2^m) of threshold matrix $[\tau_{ij}]$
  - L: specified minimum run-length
  - r_0 ($= \Delta y / \Delta x$): aspect ratio of PEL

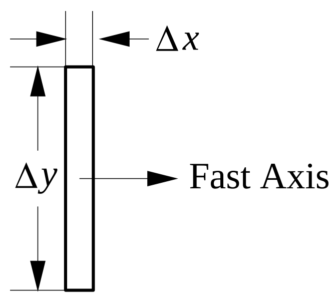


Figure: Non-square PEL

- **Compute:** 2^m by 2^m threshold matrix $[\tau_{ij}]$ satisfying Bayer's criteria for PELs of aspect ratio $r = \Delta Y / \Delta X = r_0 / L$

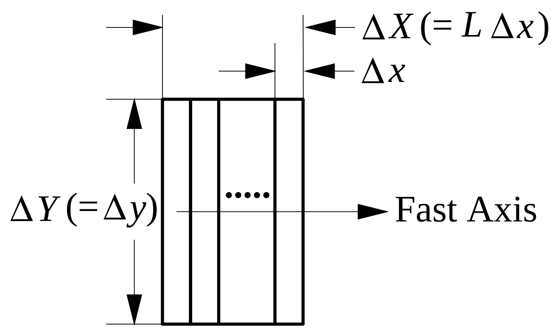


Figure: L times replicated PEL

Generalized Bayer's Algorithm (continued)

- **Bayer's criterion**

Minimization of the *longest finite wavelength* of the nonzero sinusoidal components that describe the dot pattern of an area of uniform intensity.

- **Construction of threshold matrix $[\tau_{ij}]$**

Suppose that an area of uniform intensity is represented by repeating both horizontally and vertically a 2^m by 2^m subarray of rectangular elements of width ΔX, height ΔY.

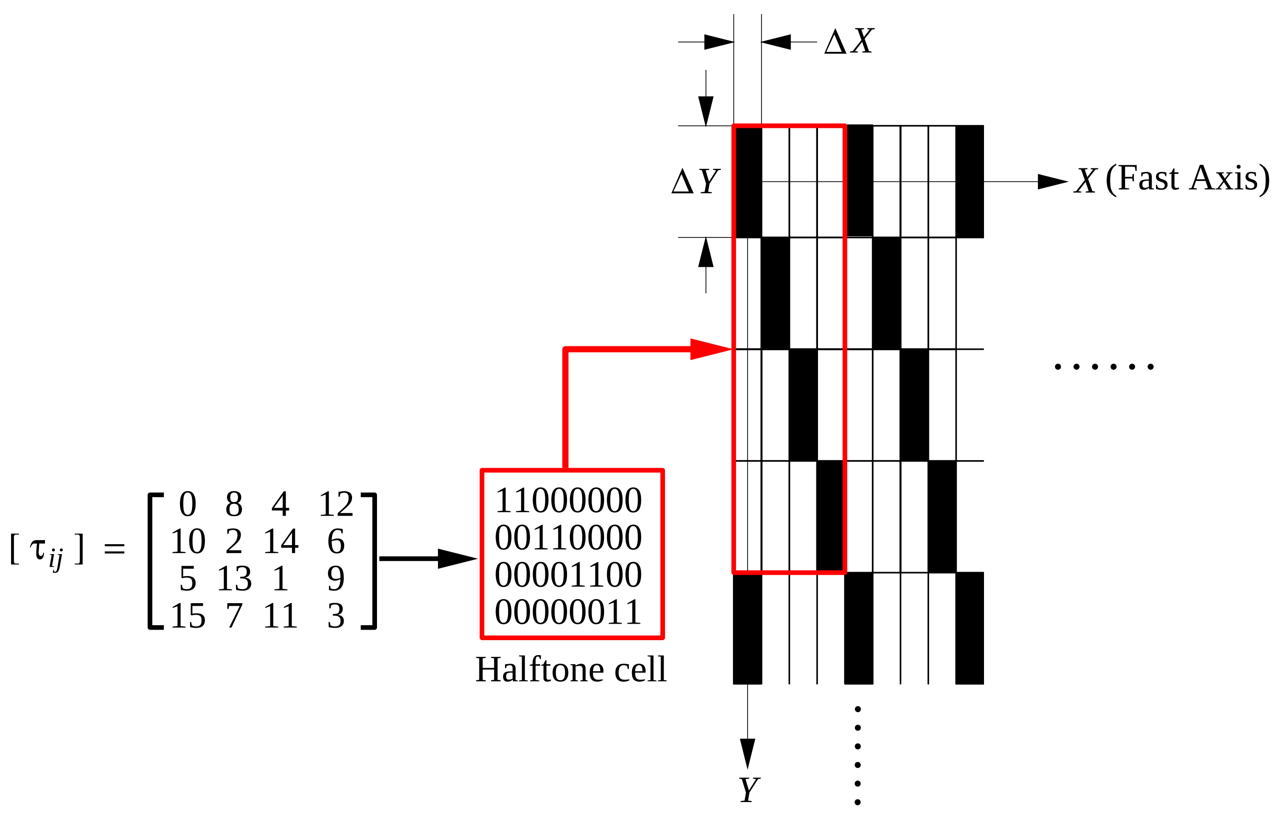


Figure: An area of uniform intensity (level 5, $m = 2$, $r = 4$)

Construction of threshold matrix (continued)

Let $I(X,Y)$ be the intensity value ($= 0$ or 1) at $(X,Y) = (k\Delta X, l\Delta Y)$, where integers k, $l \in [0, 2^m)$, then:

$$I(X + 2^m \Delta X,\ Y) = I(X,\ Y + 2^m \Delta Y) = I(X,Y), \tag{1}$$

i.e.,

$$I(\,(k + 2^m)\Delta X,\ l\Delta Y) = I(k\Delta X,\ (l + 2^m)\Delta Y) = I(k\Delta X, l\Delta Y). \tag{1'}$$

The basic subarray can be specified by all combinations of such k, l and hence,

$$I(X,Y) = I(k\Delta X, l\Delta Y) := \sum_{u=-2^{m-1}+1}^{2^{m-1}} \sum_{v=-2^{m-1}+1}^{2^{m-1}} J(\frac{u}{2^m \Delta X}, \frac{v}{2^m \Delta Y}) e^{-i2\pi(\frac{uk+vl}{2^m})} \tag{2}$$

where,

$$J(\frac{u}{2^m \Delta X}, \frac{v}{2^m \Delta Y}) := \frac{1}{2^{2m}} \sum_{k=0}^{2^m-1} \sum_{l=0}^{2^m-1} I(k\Delta X, l\Delta Y) e^{i2\pi(\frac{uk+vl}{2^m})}. \tag{3}$$

Construction of threshold matrix (continued)

The real part of each term in Eq. (2) is a sinusoidal plane wave with an *amplitude* A_{uv}, given by

$$A_{uv} = \sqrt{J(\frac{u}{2^m \Delta X}, \frac{v}{2^m \Delta Y})\ J(\frac{-u}{2^m \Delta X}, \frac{-v}{2^m \Delta Y})} \tag{4}$$

and a *wave length* λ_{uv} measured at right angles in the wave front, given by

$$\lambda_{uv} = \frac{2^m \Delta Y}{\sqrt{(ru)^2 + v^2}}, \qquad r \geq 1 \tag{5}$$

in XY-plane.

An *index of texture* Λ in a uniform area is defined by

$$\Lambda := \max_{-M+1 \leq u,v \leq M} \{\lambda_{uv} | A_{uv} \neq 0, \lambda_{uv} < \infty\}, \tag{6}$$

where $M = 2^{m-1}$.

Example: For $m = 2$, $r = 4 \Longrightarrow \underline{\Lambda = 4\Delta Y = \lambda_{0,-1} = \lambda_{01}} > \lambda_{02} > \lambda_{-1,0} = \lambda_{10}$
$> \lambda_{-1,-1} = \lambda_{-1,1} = \lambda_{1,-1} = \lambda_{11} > \lambda_{-1,2} = \lambda_{1,2} > \lambda_{20} > \lambda_{2,-1} = \lambda_{21} > \lambda_{22}$

$\Longrightarrow$ *Our problem consists of choosing a sequence of* 2^{2m} *positions within* 2^m *by* 2^m *lattice that minimize* Λ *for each level of intensity.*

Construction of threshold matrix (continued)

Remark: From Eq.'s (3) and (5), we can have $\Lambda < 2^m \Delta Y$ if and only if $J_{0,-1} = J_{01} = 0$, where $J_{uv} := J(\frac{u}{2^m \Delta X}, \frac{v}{2^m \Delta Y})$. However, we can not have $\Lambda < 2^m \Delta Y$ for *two successive* values of N_{dot}, where N_{dot} is the number of black dots in the basic 2^m by 2^m lattice.

$\Longrightarrow$ Suppose that J_{uv} is zero for some value of N_{dot}, then adding one dot to an empty position $(X_\alpha, Y_\alpha) = (k_\alpha \Delta X, l_\alpha \Delta Y)$ results in adding a nonzero term to Eq. (3). Thus J_{uv} can not equal zero for $(N_{dot} + 1)$ black dots.

$\Longrightarrow$ We can, however, make $\Lambda < 2^m \Delta Y$ for $(N_{dot} + 2)$ black dots by computing *appropriate positions* (X_α, Y_α) and (X_β, Y_β) for $(N_{dot} + 1)^{\text{th}}$ and $(N_{dot} + 2)^{\text{th}}$ black dots, respectively.

Construct-Threshold-Matrix ($[\tau_{ij}]$)

1. **for each** $i, j \in [0, 2^m)$ **set** $\tau_{ij} \leftarrow 0$, $d_{ij} \leftarrow 0$
   ▷ *Initialize threshold matrix* $[\tau_{ij}]$, *dither pattern* $[d_{ij}]$.
2. **set** $counter \leftarrow 0$
3. **while** $(\exists(\alpha, \beta) | d_{\alpha\beta} = 0)$ **do:**
   ▷ *Repeat until every dither pattern is computed.*
4. **set** $n \leftarrow 0$
5. **for each** $i, j \in [0, 2^m)$ **such that** $d_{ij} = 0$ **do:**
6. **set** $d_{ij} \leftarrow 1$ ▷ *Add a dot to an empty position.*
7. **for each** $k, l \in [0, 2^m)$ **such that** $d_{kl} = 0$ **and** $(2^m k + l > 2^m i + j)$ **do:**
8. **set** $d_{kl} \leftarrow 1$
   ▷ *Add another dot to an empty position.*
9. **insert** ij, kl **into** $List[n]$
10. **for each** $u, v \in (-2^{m-1}, 2^{m-1}]$ **and** $(u \neq 0 \lor v \neq 0)$ **do:**
11. **compute** J_{uv} ▷ *Compute Fourier coeff's.*
12. **if** $J_{uv} \neq 0$ **do:**
13. **compute** λ_{uv}, $A_{uv} \Longrightarrow$ **let** $Wave$
14. **insert** $Wave$ **into** $List[n]$
15. **reset** $d_{kl} \leftarrow 0$ ▷ *For the next trial* (k, l)
16. **reset** $d_{ij} \leftarrow 0$ ▷ *For the next trial* (i, j)
17. **set** $n \leftarrow n + 1$
18. Find-Optimal-Position $(List[0, 1 \ldots n-1], ij, kl)$
    ▷ *Find* (ij, kl) *satisfying Bayer's criteria.*
19. **set** $\tau_{ij} \leftarrow counter$, $\tau_{kl} \leftarrow counter + 1$, $counter \leftarrow counter + 2$
    ▷ *Assign threshold values.*
20. **set** $d_{ij} \leftarrow 1$, $d_{kl} \leftarrow 1$
    ▷ *Add two successive dots to optimal positions.*

Find-Optimal-Position ($List[0, 1, \cdots, n-1]$, i^*j^*, k^*l^*)

1. **for** $\alpha \leftarrow 0$ **to** $n-1$ **sort** λ_{uv}'s **in** $List[\alpha]$
   ▷ *For each trial pair of dot positions, sort* λ_{uv}*'s in descending order*
2. **for each** $List[\alpha]$ & $List[\beta]$ **do:**
3.   **if** Which-Is-Better ($List[\alpha], List[\beta]$) $= List[\alpha]$
         ▷ *Compare two pairs of trial dot positions.*
         then remove $List[\beta]$
4.   **else remove** $List[\alpha]$
5. **set** i^*j^*, $k^*l^* \leftarrow ij$, kl **in** $List[0]$
   ▷ *Optimal positions for two successive dots* d_{ij}, d_{kl}*.*

Which-Is-Better ($List[\alpha]$, $List[\beta]$)

1. **set** $Wave_\alpha \leftarrow head(List[\alpha])$, $Wave_\beta \leftarrow head(List[\beta])$
2. **while** ($Wave_\alpha$ & $Wave_\beta \neq$ NIL) **do:**
3.   **set** $\lambda_\alpha \leftarrow \lambda$ **of** $Wave_\alpha$, $\lambda_\beta \leftarrow \lambda$ **of** $Wave_\beta$
4.   **if** $\lambda_\alpha < \lambda_\beta$ **then return** $List[\alpha]$
5.   **else if** $\lambda_\alpha > \lambda_\beta$ **then return** $List[\beta]$
6.   **else do:**
7.   **set** $Wave_\alpha \leftarrow next(Wave_\alpha)$ $Wave_\beta \leftarrow next(Wave_\beta)$
8. **reset** $Wave_\alpha \leftarrow head(List[\alpha])$, $Wave_\beta \leftarrow head(List[\beta])$
9. **while** ($Wave_\alpha$ & $Wave_\beta \neq$ NIL) **do:**
10.   **set** $A_\alpha \leftarrow A$ **of** $Wave_\alpha$, $A_\beta \leftarrow A$ **of** $Wave_\beta$
11.   **if** $A_\alpha < A_\beta$ **then return** $List[\alpha]$
12.   **else if** $A_\alpha > A_\beta$ **then return** $List[\beta]$
13.   **else do:**
14.   **set** $Wave_\alpha \leftarrow next(Wave_\alpha)$ $Wave_\beta \leftarrow next(Wave_\beta)$
15. **return** $List[\alpha]$

Example For $r_0 = 8$, $L = 2 \Longrightarrow r = 4$:

(1) $m = 1$:

$$\begin{bmatrix} 0 & 2 \\ 3 & 1 \end{bmatrix} \qquad \begin{bmatrix} 0 & 2 \\ 3 & 1 \end{bmatrix}$$

(2) $m = 2$:

$$\begin{bmatrix} 0 & 8 & 4 & 12 \\ 10 & 2 & 14 & 6 \\ 5 & 13 & 1 & 9 \\ 15 & 7 & 11 & 3 \end{bmatrix} \qquad \begin{bmatrix} 0 & 8 & 2 & 10 \\ 12 & 4 & 14 & 6 \\ 3 & 11 & 1 & 9 \\ 15 & 7 & 13 & 5 \end{bmatrix}$$

(3) $m = 3$:

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

(a) Generalized algorithm (b) Original algorithm

Example: $(m = 2,\ L = 2,\ r_0 = 8 \Longrightarrow r = 4)$

$$\underline{\Lambda = \frac{2\Delta Y}{\sqrt{5}}} = \lambda_{-1,2} = \lambda_{12} > \lambda_{2,-1} = \lambda_{21} \tag{7}$$

We note:

$$\underline{\Lambda = 2\Delta Y} = \lambda_{02} > \lambda_{-1,0} = \lambda_{10} > \lambda_{-1,2} = \lambda_{12} \tag{8}$$

if original Bayer's $[\tau_{ij}]$ is used.

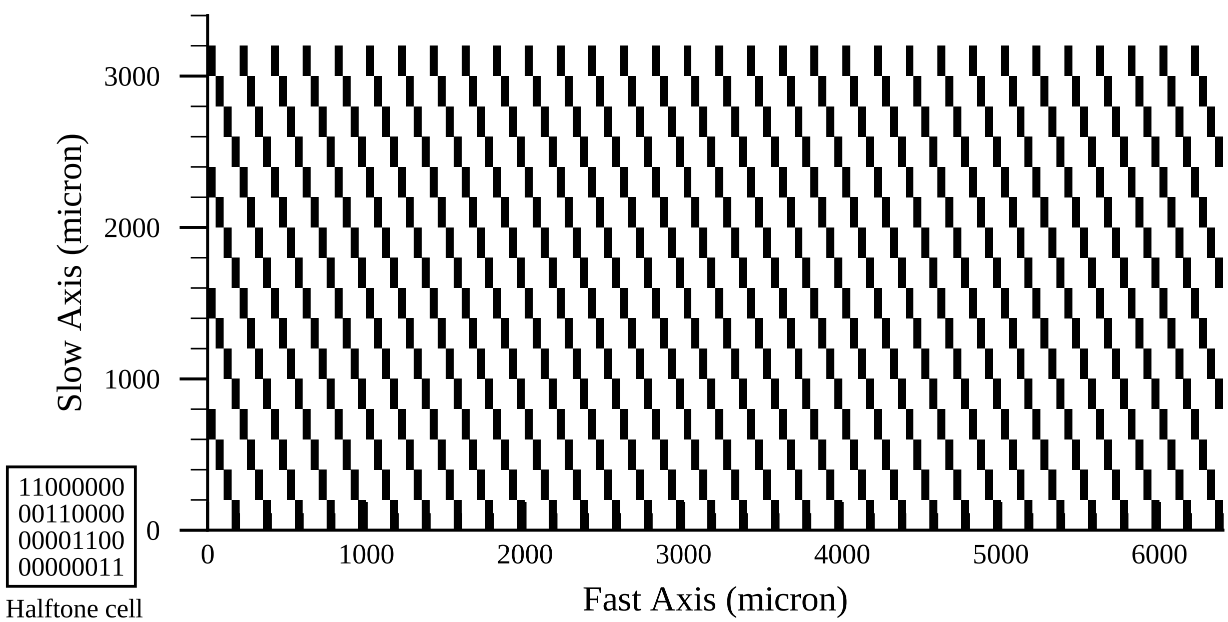
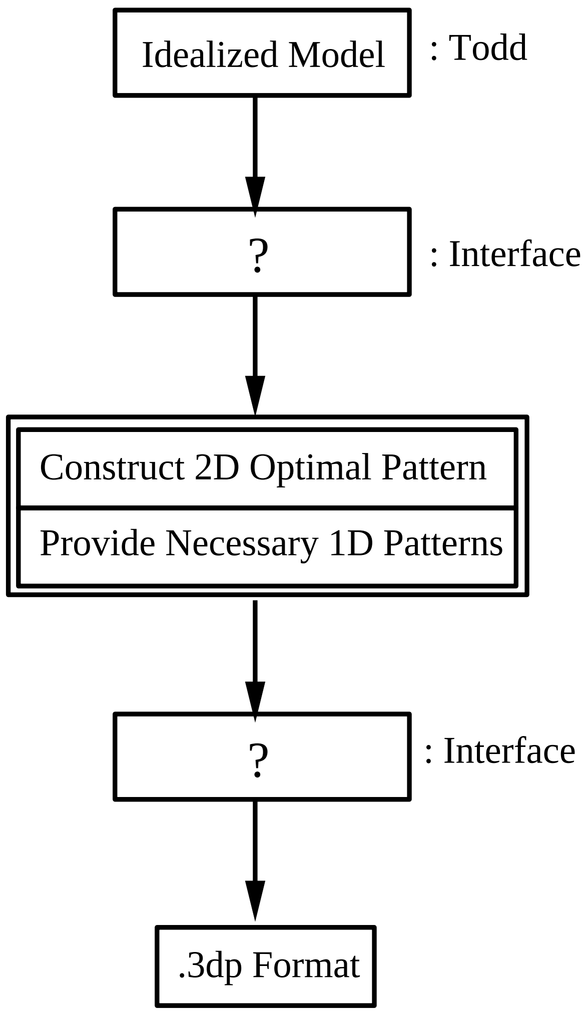


Figure: Level 5 (PEL size = $25\mu \times 200\mu$)

Summary



- Extension to optimal dithering for composition of *arbitrary* number of materials.

  $\Longrightarrow$ Utilize *color halftoning* technique based on Bayer's criteria.

- 3D halftoning for a volume of uniform material composition