Progressive Halftoning via Perona-Malik Diffusion and Stochastic Flipping

Jackie Shen
jhshen@math.umn.edu
University of Minnesota
Minneapolis, MN 55455
www.math.umn.edu/~jhshen

Work is Partially Supported by NSF DMS-0202565
Organization

- A Quick Overview of Existent Halftoning Methods
- Prelude: Diffusion-Based Spatial Regularization

- The New Model:
  - Perona-Malik Diffusion and Edge Adaptivity
  - Stochastic Flipping
  - Progressive Halftoning

- Computational Examples
Popular Error-Diffusion Based Halftoning

Floyd & Steinberg’76: Rastering
Witten & Neal’82: Peano Curve
Numerous Improvements by Allebach’s Purdue Group: 00-04
D. Knuth’87: Dot Diffusion: Path Defined by Class Matrix
Mese & Vaidyanathan’00: Optimal Knuth’s Path Matrix
and by many many other authors

threshold $\mu$

$F(\alpha) = \{ \beta, \gamma, \delta, \ldots \}$
A given mage $u$ determines a random field $b$.

A natural Constraint is

$$E[b_\alpha] = u_\alpha,$$

at each pixel $\alpha$.

The marginal $b_\alpha$ is subject to Bernoulli $B(1, p)$. Then

$$E[b_\alpha] = 1xp + 0x(1-p) =p.$$

**Simplest binary random field: independent $B(1, u_\alpha)$’s.**

Random Field View of Images: Geman-Geman’84; Mumford-Zhu’97

U Maryland, April 13, 2005
Independent Bernoulli Halftoning: Not So Pleasant

- **Pro:** fast and parallelizable

- **Con:** losing spatial coherence

**Lesson:**
- Images are coherent spatial patterns; vital for perception
- Points (or the “spins”) should respect such visual regularity

**Question:**
- How to characterize spatial regularity?

[Independent Bernoulli Halftoning]
Consider a constant shade (a)
\[ u = 0.1 \rightarrow \]
- Ideally 10% on’s and 90% off’s

Consider a 9x9 square
- About 81x10% = 8 on’s.
- Spatial homogeneity (c) looks more visually pleasant than unwanted clustering (b).

Scientific Support:
- Importance of Blue Noise (Ulichney’88)
- Clustering \( \delta(x) \rightarrow \text{Red Noise} \)
Points as a Borel Measure (or Delta Terrain)

- Given n points \( (x_1, \ldots, x_n) \) in a domain \( \Omega \), first form a delta (spiky) terrain (or a delta train in 1-D):
  \[ \phi_0(x) = \delta(x-x_1)+\ldots+\delta(x-x_n). \]

- Or, rather, form a Borel (or Radon) measure, s.t.
  \[ <f, d\phi_0> = f(x_1)+\ldots+f(x_n), \text{ for any test fcn } f(x). \]

- Then the correspondence is one-to-one (i.e., a lossless representation).
Diffusion of a Delta Terrain

Diffuse the delta-terrain

- With some suitable stopping time $\tau$, the terrain is mollified to $\phi^\tau(x) = \phi(x, \tau)$, which is a function, instead of a measure.

- In terms of fundamental solutions,
  $$\phi(x, \tau) = \sum_{i=1}^{n} G(x, \tau; x_i).$$

$$\begin{align*}
\frac{\partial \phi}{\partial x} &= \frac{1}{2} \Delta \phi, \quad x \in \Omega; \\
\phi(x, 0) &= \phi_0(x); \\
\frac{\partial \phi}{\partial \nu} &= 0, \quad \text{along } \partial \Omega.
\end{align*}$$
Points Renormalization: Centroidal Extraction

Thresholding and region extraction: \( V = \{ \mathbf{x} : \phi^t(\mathbf{x}) > \sigma \} \); (In 2-D, the threshold can be \( \sigma = e^{-1/(\pi \tau)} \).

Connected components extraction: \( V = V_1 \cup V_2 \ldots \cup V_m \).

Centroidal points extraction: \( z_k = \text{masscenter}(V_k) \); (A technique used in Centroidal Voronoi Tessellation).

Point loss due to merging (that is desired !!):
\[
S = \{ \mathbf{x}_1, \ldots, \mathbf{x}_n \} \rightarrow Z = \{ \mathbf{z}_1, \ldots, \mathbf{z}_m \}, \quad m =< n.
\]
Where and HOW to deposit the \( n-m \) new points:

a. Let \( V^c = \Omega \setminus V \) be the complement pixel domain.

b. Set probability \( p = (n - m) / \# V^c \).

c. Draw a random UNIFORM i.i.d. field \( F \) on \( V^c \).

\textbf{d. Add} any pixel \( \beta \) of \( V^c \) into \( Z \) iff \( F_\beta < p \).

e. (minor deterministic correction if necessary).

- Repeat the diffusion process on the new set \( Z \)...
Halftoning Real Images

- **Real images** are not constant. The above points manipulation is not straightforward globally.

- **Change of Mind Set:**
  - **Keep:** the diffusion idea (& path independent)
  - **Dump:** windowing and thresholding

- **Features of the New Model:**
  - No windows, no paths, and no hard thresholding
  - Progressive
  - Parallelizable
  - Combining deterministic and stochastic processes
Given: \( b = (b_\alpha) \)-current halftone field of \( u = (u_\alpha) \).

Instead of the preceding diffusion and regularization process on \( b \), one diffuses the error field

\[
e = u - b; \quad e_\alpha = u_\alpha - b_\alpha, \quad \alpha \text{ in } \Omega.
\]

Let \( P_\tau \) denote the diffusion operator, and \( e(\tau) = P_\tau e \).

If halftoning is already satisfactory, then

\[
u_\alpha = E[b_\alpha] \sim = \langle b_\alpha \rangle \text{ (spatial) } \sim = (P_\tau b)_\alpha;
\]

\[
e_\alpha(\tau) = P_\tau (u - b)_\alpha \sim = 0.
\]

\[
|e_\alpha(\tau)| \text{ characterizes how good } b \text{ has been.}
\]
Fast Forward (B): Info of Diffused Error $e(\tau)$

- **POSITIVE** error $e_\alpha(\tau) \sim= u_\alpha - <b>_\alpha > 0$:
  - over-off near $\alpha$ $\rightarrow$ turn on more pixels.

- **NEGATIVE** error $e_\alpha(\tau) \sim= u_\alpha - <b>_\alpha < 0$:
  - over-on near $\alpha$ $\rightarrow$ turn off more pixels.

**Conclusion:**

Use $e_\alpha(\tau)$ to update the halftone $b \rightarrow b_{\text{new}}$.

**Questions for (A) and (B):**

(A) how to diffuse?  (B) how to update?
New Model/Algorithm: PM-SF

Answers to the previous two questions:

A. Diffusion: Perona-Malik Diffusion, \( e = u - b \),
\[ e(\tau) = \text{PM}(e | u, \tau). \]
Why: Edge-adaptive; diffusion only within patches.

B. Updating: Stochastic Flipping (based on \( e(\tau) \))
\[ b_{\text{new}} = \text{SF}(b | e(\tau), f), \]
where \( f \) is a randomly drawn i.i.d. reference field.

The entire progressive halftoning process:
1. Starting with any initial guess \( b^0; e = u - b^0 \);  
2. PM-diffusing the error \( e \) to \( e(\tau) \);  
3. Updating \( b^0 \) to \( b^1 \) by stochastic flipping; and repeat.
Flow-Chart of the New Model: Progressive PM-SF
Example: A bright full moon against a dimmer sky:
- Ideally, FIRST segment V₁ (moon) and V₂ (background),
- And THEN apply halftoning to each homogeneous patch.

Challenge: For real complex images, segmentation is challenging, computationally expensive, and slow.

Solution: As far as DIFFUSION is concerned, Perona-Malik (1990, PAMI) is a simple and effective solution!
Perona-Malik’s Anisotropic Diffusion (1990)

Key: Edge Signature Function:

\[ D = g \left( |\nabla u|^2 \right) = \begin{cases} \frac{1}{\sqrt{1 + a |\nabla u|^2}} \\ \exp \left( - \frac{|\nabla u|^2}{2 \sigma^2} \right) \end{cases} \]

PM Nonlinear Diffusion:

\[ \begin{cases} \frac{\partial u}{\partial t} = \nabla \cdot \left[ g \left( |\nabla u|^2 \right) \nabla u \right] \\ u(x, 0) = u_0 \end{cases} \]

Effective Action:
- No mess-up among objects;
- Sharp edges are NOT smeared.
Perona-Malik Error Diffusion: $e^n(\tau) = \text{PM}(e^n | u, \tau)$.

- At step n, suppose
  - halftone image: $b^n$
  - halftone error: $e^n = u - b^n$

- Given image $u$: information source of edges/objects

- How to CONFINE error diffusion within each object:
  \[
  \begin{cases}
  \frac{\partial e}{\partial t} = \nabla \cdot \left[ g(\| \nabla u \|) \nabla e \right] \\
  e(x, 0) = e^n
  \end{cases}
  \]

- How to conciliate digital & analog views: $u(x)$ vs. $u_\alpha$
  - Analogous to the spectral graph theory (Chung & Yau, 1994)
Stochastic Flipping $b^{n+1} = \text{SF}(b^n \mid e^n(\tau))$,

- After PM error diffusion, the diffused error is $e^n(\tau)$.

- To minimize changes/achieve statistical convergence, first **copy** $b^{n+1}$ from $b^n$.

- From Slide “Fast Forward (B),” **qualitatively,**
  - $e^n_\alpha(\tau) > 0 \rightarrow$ over off $\rightarrow$ turn **on** more pixels
  - $e^n_\alpha(\tau) < 0 \rightarrow$ over on $\rightarrow$ turn **off** more pixels

- **Challenge:** What’s the **quantitative** rule?
Turning/Flipping Rate (positive error case)

- Take a local imaginary window $W$.
- IDEALLY : $\#\text{on's} = \#W \times u_\alpha$.
- ACTUALLY : $\#\text{on's} = \#W \times <b^n>_\alpha = \#W \times b^n_{\alpha}(\tau)$
- SHORT of on’s: $\#W \times [u_\alpha - b^n_{\alpha}(\tau)] = \#W \times e^n_{\alpha}(\tau)$
- TURNING rate (from the off’s):

$$p^+ = \frac{\#\text{on's short of}}{\#\text{off's in } W} = \frac{e^n_{\alpha}(\tau) \#W}{\#W - b^n_{\alpha}(\tau) \#W} = \frac{e^n_{\alpha}(\tau)}{1 + e^n_{\alpha}(\tau) - u_\alpha}$$

U Maryland, April 13, 2005
Flipping Rates (both positive & negative errors)

(from the preceding slide) off $\rightarrow$ on turning rate:

$$p^+ = \frac{\text{#on's short of}}{\text{#off's in W}} = \frac{e^n_\alpha(\tau)\#W}{\#W - b^n_\alpha(\tau)\#W} = \frac{e^n_\alpha(\tau)}{1 + e^n_\alpha(\tau) - u_\alpha}$$

- Similarly, when $e^n_\alpha(\tau) < 0$, on $\rightarrow$ off turning rate:

$$p^- = \frac{\text{#off's short of}}{\text{#on's in W}} = \frac{[-e^n_\alpha(\tau)]\#W}{b^n_\alpha(\tau)\#W} = \frac{-e^n_\alpha(\tau)}{u_\alpha - e^n_\alpha(\tau)}$$

- In the stochastic view of turning/flipping:

$$\begin{align*}
\text{Prob}(b^{n+1}_\alpha = 1 \mid b^n_\alpha = 0 \text{ and } e^n_\alpha(\tau) > 0) &= p^+ \\
\text{Prob}(b^{n+1}_\alpha = 0 \mid b^n_\alpha = 1 \text{ and } e^n_\alpha(\tau) < 0) &= p^-
\end{align*}$$

U Maryland, April 13, 2005
Flipping Rates (cont’d)

(from the preceding slide)

\[
\begin{align*}
 p^+ &= \frac{e^n_\alpha(\tau)}{1 + e^n_\alpha(\tau) - u_\alpha}, \\
 p^- &= \frac{-e^n_\alpha(\tau)}{u_\alpha - e^n_\alpha(\tau)}.
\end{align*}
\]

➢ define a univariate function: for any real \(x\),

\[
p_\alpha(x) = \frac{x}{1_{x \geq 0} + x - u_\alpha}, \text{ logic variable } 1_{x \geq 0} = \text{True}(x \geq 0)
\]

➢ regardless of the sign of errors, one has

\[
p^{+\cdot-} = p_\alpha(e^n_\alpha(\tau)) := p^n_\alpha.
\]

➢ Computational implementation is very simple (next) ...
Matlab Implementation: $b^{n+1} = SF(b^n | e^n(\tau))$

\[
\begin{align*}
\text{Prob}(b^{n+1}_{\alpha} = 1 | b^n_{\alpha} = 0 \text{ and } e^n_{\alpha}(\tau) > 0) &= p^n_{\alpha} \\
\text{Prob}(b^{n+1}_{\alpha} = 0 | b^n_{\alpha} = 1 \text{ and } e^n_{\alpha}(\tau) < 0) &= p^n_{\alpha}
\end{align*}
\]

- Matlab Codes for $b^{n+1} = SF(b^n | e^n(\tau), f)$, (version I):
  - Draw any i.i.d. random field $f$ of Uniform(0,1).
  - $b^{n+1} = b^n$
  - $b^{n+1}(e^n(\tau) > 0 \text{ and } b^n_{\alpha} = 0 \text{ and } f < p^n) = 1$; % turning on
  - $b^{n+1}(e^n(\tau) < 0 \text{ and } b^n_{\alpha} = 1 \text{ and } f < p^n) = 0$; % turning off

- Simplification (version II):
  - $b^{n+1}(e^n(\tau) > 0 \text{ and } f < p^n) = 1$; % turning on
  - $b^{n+1}(e^n(\tau) < 0 \text{ and } f < p^n) = 0$; % turning off
The Entire Progressive Algorithm PM-SF

- Inputs: a given contone image $u$, stopping time $\tau$.
- Initial halftone: $b^0$.
- For $n=0, 1, 2, ...$
  - Current halftone error: $e^n = u - b^n$;
  - PM (Perona-Malik diffusion): $e^n(\tau) = \text{PM}(e^n \mid u, \tau)$;
  - SF (stochastic flipping): $b^{n+1} = \text{SF}(b^n \mid e^n(\tau))$.

Main Features:
- Object-adapted halftoning without explicit segmentation
- No artificial mosaics, windows, paths, or thresholds
- Parallel implementation is straightforward
**Definition.** [Flipping Rate Per Pixel (frpp)]

At each step $n$, the frpp $R^n$ (a random variable) is defined by

$$R^n = \frac{\|b^{n+1} - b^n\|_1}{|\Omega|} = \frac{1}{N \times M} \sum_{\alpha \in \Omega} |b^{n+1}_\alpha - b^n_\alpha|$$

**Theorem 1.** At each step $n$, the expected frpp is given by

$$E[R^n] = \frac{1}{|\Omega|} \sum_{\alpha \in \Omega} \chi(b^n_\alpha + \text{sign}(e^n_\alpha(\tau))) p^n_\alpha; \quad \chi(t) = 1_{[0,1]}(t).$$

**Theorem 2.** Suppose the given image $u$ takes values from $[\delta, 1-\delta]$ for some $0<\delta<1$. Then,

$$E[R^n] \leq \frac{\|e^n(\tau)\|_1}{\delta |\Omega|}.$$
Computational Examples (I)
Stochastic Convergence: frpp at each step

frpp at each step: (for the preceding Lenna image)
Blue Noise Feature: Constant Image $u=0.35$

Power spectra of Floyd-Steinberg's halftones: $u = 0.35$

Power spectra of the PM-SF halftones: $u = 0.35$
Blue Noise Feature: Constant Image $u=0.5$

Power spectra of Floyd-Steinberg's halftones: $u=0.5$

Power spectra of the PM-SF halftones: $u=0.5$
Acknowledgments

- Center of Scientific Computing and Mathematical Modeling (CSCMM) at U. Maryland, founded by Eitan Tadmor.
- Ron DeVore (USC), Ingrid Daubechies (Princeton), Sinan Güntürk (Courant), and Chai-Wah Wu (IBM)
- Gil Strang (MIT), Tony Chan and Stan Osher (UCLA), David Mumford and Stu Geman (Brown)
- NSF, Division of Mathematical Sciences