Theoretical Bounds of Direct Binary Search Halftoning

Jan-Ray Liao, *Member, IEEE*

***Abstract*— Direct binary search (DBS) produces the images of the best quality among half-toning algorithms. The reason is that it minimizes the total squared perceived error instead of using heuristic approaches. The search for the optimal solution involves two operations: 1) toggle and 2) swap. Both operations try to find the binary states for each pixel to minimize the total squared perceived error. This error energy minimization leads to a conjecture that the absolute value of the filtered error after DBS converges is bounded by half of the peak value of the autocorrelation filter. However, a proof of the bound's existence has not yet been found. In this paper, we present a proof that shows the bound existed as conjectured under the condition that at least one swap occurs after toggle converges. The theoretical analysis also indicates that a swap with a pixel further away from the center of the autocorrelation filter results in a tighter bound. Therefore, we propose a new DBS algorithm which considers toggle and swap separately, and the swap operations are considered in the order from the edge to the center of the filter. Experimental results show that the new algorithm is more efficient than the previous algorithm and can produce half-toned images of the same quality as the previous algorithm.**

***Index Terms*— Halftoning, direct binary search, theoretical bounds.**

I. Introduction

IMAGE halftoning is the process of transforming a continuous-tone image into an image with binary tone levels or a limited number of tone levels. It is useful to reproduce continuous-tone images in devices with only a limited number of output tone levels such as printers or monochrome displays. Based on the low-pass characteristic of human visual system (HVS), the goal of halftoning is to produce a limited-tone image which is visually as close to the original continuous-tone image as possible when observed from a distance.

Halftoning algorithms can be classified into three groups: point processes, neighborhood processes, and iterative processes. The most commonly used algorithms for these three groups are screening or dithering [1] for point processes, error diffusion [2] for neighborhood processes, and direct binary search (DBS) [3]–[5] for iterative processes. Among these methods, the quality of the halftone image produced by DBS is the best and it has been extended to various printer models [7]–[9], different HVS models [10], color halftoning [11]–[13], screen design [14], [15], and clustered dots [16].

The reason that DBS can generate halftone images of the best quality is that it seeks to minimize the total squared perceived error while algorithms in point and neighborhood processes are mostly heuristic. The search of a solution for DBS involves two operations: toggle and swap. The toggle operation finds one of the binary states for each pixel to minimize the squared error while the swap operation tries to switch two pixels of opposite states for the same purpose. Previously proposed DBS algorithms consider toggle and swap together in a limited neighborhood and select the operation with the largest energy reduction in each iteration. Mathematically, given an autocorrelation filter representing the effects of the perceptual HVS model and the printer model, the total squared perceived error can be expressed as the sum of the multiplication between the error per pixel (the difference between the halftone image and the original image) and the filtered error that is obtained by passing the error per pixel through the autocorrelation filter. In turn, the energy difference caused by toggle and swap can be expressed in the filtered error and the criteria to decide whether to proceed with these operations can be expressed as inequalities involving the filtered error. Therefore, minimizing the total squared perceived error is equivalent to minimize the absolute value of the filtered error as has been shown in [5]. It is then conjectured in the same paper that the absolute value of the filtered error is bounded by half of the peak value of the autocorrelation filter. However, a formal mathematical proof of the bound has not yet been found.

In this paper, we present a theoretical analysis that proves the existence of the bound and shows that one side of the bounds is smaller than previously conjectured under the condition that at least one swap occurs after toggle operation converges. The theoretical analysis also leads to three results that have significant impact on DBS implementation. The first is that the range of the bounds is smaller if the first swap occurs at a pixel further away from the center of the autocorrelation filter. The second is that more swaps result in more symmetric bounds. The third is that the filter value can be viewed as the step size in a gradient descent algorithm. Therefore, a toggle operation which uses the peak filter value has a larger step size than a swap operation. In other words, toggle operation

Manuscript received November 20, 2014; revised March 18, 2015 and May 31, 2015; accepted June 21, 2015. Date of publication June 23, 2015; date of current version July 7, 2015. This work was supported by the Ministry of Science and Technology, Taiwan, under Contract MOST 103-2221-E-005-007. The associate editor coordinating the review of this manuscript and approving it for publication was Prof. Jing-Ming Guo.

The author is with the Department of Electrical Engineering, Institute of Communication Engineering, National Chung Hsing University, Taichung 402, Taiwan (e-mail: jrliao@mail.nchu.edu.tw).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TIP.2015.2449077

converges faster but with a coarser resolution. Based on these three results, we propose a new DBS algorithm which has three features. The first is that toggle and swap operations are considered separately with swap being considered after toggle converges. Because a toggle operation has a larger step size than a swap operation, this separation leads to faster convergence. It also results in increased number of toggle operations and decreased number of swap operations during the search. In turn, memory access cost is significantly reduced because the memory access cost of a swap update is twice of a toggle update. The second is that all pixels under the filter coverage are considered in swap. Because of the increased coverage, the new algorithm is more likely to find a better solution. The third is that pixels are separated into groups of the same autocorrelation filter value and each group is processed in the ascending order of the autocorrelation filter value. The reason to process the pixels in groups is to prevent excessive computation when all pixels under the filter coverage are considered. Since the filter value decreases outwardly from the center of the filter, the processing order of the pixel groups can also be viewed as from the edge of the filter toward its center. Because both the first and third features have the advantages of computational efficiency, experiments show that the execution time of the new algorithm is not significantly higher than those of the previous algorithms despite the fact that all pixels under the filter coverage are considered. If we want to further reduce the execution time of the new algorithm, we can process only a subset of pixel groups which occupies the central part of the filter and the resulted halftoned image is still competitive with the previous algorithms.

The remainder of the paper is organized as follows. In the second section, we present the theoretical analysis that proves the existence of the bound. In the third section, we propose a new DBS algorithm based on the results of the theoretical analysis. In the fourth section, we show the experimental results. Finally, in the fifth section, we conclude.

II. Theoretical Analysis

Let $f[\mathbf{m}]$ be the original discrete-space image and $g[\mathbf{m}]$ be the halftone output where $\mathbf{m}$ is a discrete spatial coordinate. The error for each pixel is:

$$e[\mathbf{m}] = g[\mathbf{m}] - f[\mathbf{m}] \tag{1}$$

where $f[\mathbf{m}]$ is in the range of [0, 1] and $g[\mathbf{m}]$ is either 0 or 1.

DBS intends to minimize the total squared perceived error E which can be expressed as:

$$E = \sum_{\mathbf{m}} \sum_{\mathbf{n}} e[\mathbf{m}] e[\mathbf{n}] c_{\tilde{p}\tilde{p}}[\mathbf{m} - \mathbf{n}] \tag{2}$$

where $\mathbf{n}$ is also a discrete spatial coordinate. The function $c_{\tilde{p}\tilde{p}}[\mathbf{m}]$ is an autocorrelation function of the perceptual filter derived from HVS and printer profile. It is an even symmetric function and must satisfy the following two properties:

$$\sum_{\mathbf{m}} c_{\tilde{p}\tilde{p}}[\mathbf{m}] = 1 \tag{3}$$

and

$$|c_{\tilde{p}\tilde{p}}[\mathbf{m}]| \leq c_{\tilde{p}\tilde{p}}[\mathbf{0}]. \tag{4}$$

The first property is required to maintain tonal consistency. The second property is a general property of an autocorrelation function and a proof can be found in [6]. For HVS model, the second property can be made more stringent by ensuring that $c_{\tilde{p}\tilde{p}}[\mathbf{m}]$ is non-negative, i.e., $0 \leq c_{\tilde{p}\tilde{p}}[\mathbf{m}] \leq c_{\tilde{p}\tilde{p}}[\mathbf{0}]$.

The total squared perceived error E in (2) can be rewritten as:

$$E = \sum_{\mathbf{m}} e[\mathbf{m}] c_{\tilde{p}\tilde{e}}[\mathbf{m}] \tag{5}$$

where $c_{\tilde{p}\tilde{e}}[\mathbf{m}]$ is the error $e[\mathbf{m}]$ filtered by $c_{\tilde{p}\tilde{p}}[\mathbf{m}]$:

$$c_{\tilde{p}\tilde{e}}[\mathbf{m}] = \sum_{\mathbf{n}} e[\mathbf{n}] c_{\tilde{p}\tilde{p}}[\mathbf{m} - \mathbf{n}]. \tag{6}$$

The filtered error $c_{\tilde{p}\tilde{e}}[\mathbf{m}]$ is used to determine whether a toggle or a swap can reduce E.

For the convenience of the following discussions, let's redefine $\mathbf{m}$ as the coordinate of the current pixel being considered in the optimization process. A subscript of either 0 or 1 attached to the coordinate indicates the state of that pixel. A minus or plus sign followed after the pixel state distinguishes whether the state occurs before or after a toggle or a swap. For example, $\mathbf{m}_{0-}$ indicates that the pixel state at coordinate $\mathbf{m}$ before an operation is 0 and $\mathbf{n}_{1+}$ indicates that the pixel state at coordinate $\mathbf{n}$ after an operation is 1.

A toggle is accepted if one of the following two inequalities is satisfied:

$$c_{\tilde{p}\tilde{e}}[\mathbf{m}_{0-}] < -\frac{1}{2} c_{\tilde{p}\tilde{p}}[\mathbf{0}], \tag{7}$$

$$c_{\tilde{p}\tilde{e}}[\mathbf{m}_{1-}] > \frac{1}{2} c_{\tilde{p}\tilde{p}}[\mathbf{0}]. \tag{8}$$

After a toggle is accepted, $c_{\tilde{p}\tilde{e}}[\cdot]$ is updated as follows:

$$c'_{\tilde{p}\tilde{e}}[\mathbf{q}] = c_{\tilde{p}\tilde{e}}[\mathbf{q}] + c_{\tilde{p}\tilde{p}}[\mathbf{q} - \mathbf{m}_{0-}], \tag{9}$$

$$c'_{\tilde{p}\tilde{e}}[\mathbf{q}] = c_{\tilde{p}\tilde{e}}[\mathbf{q}] - c_{\tilde{p}\tilde{p}}[\mathbf{q} - \mathbf{m}_{1-}]. \tag{10}$$

In the above two equations, $c_{\tilde{p}\tilde{e}}[\mathbf{q}]$ and $c'_{\tilde{p}\tilde{e}}[\mathbf{q}]$ are the filtered errors before and after the toggle and $\mathbf{q}$ is the coordinate of the pixel where its $c_{\tilde{p}\tilde{e}}[\cdot]$ is affected by the toggle at $\mathbf{m}$.

If convergence is established after the toggle, the opposite of either (7) or (8) must be satisfied so that no more toggle is accepted:

$$c_{\tilde{p}\tilde{e}}[\mathbf{m}_{0+}] \geq -\frac{1}{2} c_{\tilde{p}\tilde{p}}[\mathbf{0}], \tag{11}$$

$$c_{\tilde{p}\tilde{e}}[\mathbf{m}_{1+}] \leq \frac{1}{2} c_{\tilde{p}\tilde{p}}[\mathbf{0}]. \tag{12}$$

It has been shown in [5] that minimizing E is equivalent to minimize $|c_{\tilde{p}\tilde{e}}[\mathbf{m}]|$. Given (11) and (12), it is conjectured that the filtered error $c_{\tilde{p}\tilde{e}}[\mathbf{m}]$ after convergence is bounded by:

$$|c_{\tilde{p}\tilde{e}}[\mathbf{m}]| \leq \frac{1}{2} c_{\tilde{p}\tilde{p}}[\mathbf{0}]. \tag{13}$$

For the swap operation, let $\mathbf{n}^{(k)}$ be the coordinate of the pixel being considered in the neighborhood of $\mathbf{m}$ for the k-th swap where k is a nonnegative integer. Let $g^{(k)}[\cdot]$ be the values of the pixel and $c^{(k)}_{\tilde{p}\tilde{e}}[\cdot]$ be the value of the function $c_{\tilde{p}\tilde{e}}[\cdot]$ after the k-th swap.

A swap is accepted as the k-th swap if one of the following two inequalities is satisfied:

$$\left(c_{\tilde{p}\tilde{p}}[\mathbf{0}] + c_{\tilde{p}\tilde{e}}^{(k-1)}[\mathbf{m}_{0^-}] - c_{\tilde{p}\tilde{e}}^{(k-1)}[\mathbf{n}_{1^-}^{(k)}] - c_{\tilde{p}\tilde{p}}[\mathbf{n}^{(k)} - \mathbf{m}]\right) < 0, \tag{14}$$

$$\left(c_{\tilde{p}\tilde{p}}[\mathbf{0}] + c_{\tilde{p}\tilde{e}}^{(k-1)}[\mathbf{n}_{0^-}^{(k)}] - c_{\tilde{p}\tilde{e}}^{(k-1)}[\mathbf{m}_{1^-}] - c_{\tilde{p}\tilde{p}}[\mathbf{m} - \mathbf{n}^{(k)}]\right) < 0. \tag{15}$$

Because the value $c_{\tilde{p}\tilde{p}}[\mathbf{m} - \mathbf{n}^{(k)}]$ depends only on the distance between the two coordinates, the pixel state is not marked. Let $\Delta c_{\tilde{p}\tilde{p}}^{(k)} = c_{\tilde{p}\tilde{p}}[\mathbf{0}] - c_{\tilde{p}\tilde{p}}[\mathbf{m} - \mathbf{n}^{(k)}]$. Applying the even symmetric property of $c_{\tilde{p}\tilde{p}}[\cdot]$, the above two inequalities can be simplified as:

$$\left(\Delta c_{\tilde{p}\tilde{p}}^{(k)} + c_{\tilde{p}\tilde{e}}^{(k-1)}[\mathbf{m}_{0^-}] - c_{\tilde{p}\tilde{e}}^{(k-1)}[\mathbf{n}_{1^-}^{(k)}]\right) < 0, \tag{16}$$

$$\left(\Delta c_{\tilde{p}\tilde{p}}^{(k)} + c_{\tilde{p}\tilde{e}}^{(k-1)}[\mathbf{n}_{0^-}^{(k)}] - c_{\tilde{p}\tilde{e}}^{(k-1)}[\mathbf{m}_{1^-}]\right) < 0. \tag{17}$$

If convergence is established after the k-th swap, the opposite of either (16) or (17) must be satisfied so that no further swap is accepted. Therefore, one of the following two inequalities must be satisfied:

$$\left(c_{\tilde{p}\tilde{e}}^{(k)}[\mathbf{n}_{1^+}^{(k)}] - c_{\tilde{p}\tilde{e}}^{(k)}[\mathbf{m}_{0^+}]\right) \le \Delta c_{\tilde{p}\tilde{p}}^{(k)}, \tag{18}$$

$$\left(c_{\tilde{p}\tilde{e}}^{(k)}[\mathbf{m}_{1^+}] - c_{\tilde{p}\tilde{e}}^{(k)}[\mathbf{n}_{0^+}^{(k)}]\right) \le \Delta c_{\tilde{p}\tilde{p}}^{(k)}. \tag{19}$$

The update procedure of $c_{\tilde{p}\tilde{e}}^{(k)}[\cdot]$ when a swap is accepted is as follows:

$$c_{\tilde{p}\tilde{e}}^{(k)}[\mathbf{q}] = c_{\tilde{p}\tilde{e}}^{(k-1)}[\mathbf{q}] + c_{\tilde{p}\tilde{p}}[\mathbf{q} - \mathbf{m}_{0^-}] - c_{\tilde{p}\tilde{p}}[\mathbf{q} - \mathbf{n}_{1^-}^{(k)}], \tag{20}$$

$$c_{\tilde{p}\tilde{e}}^{(k)}[\mathbf{q}] = c_{\tilde{p}\tilde{e}}^{(k-1)}[\mathbf{q}] + c_{\tilde{p}\tilde{p}}[\mathbf{q} - \mathbf{n}_{0^-}^{(k)}] - c_{\tilde{p}\tilde{p}}[\mathbf{q} - \mathbf{m}_{1^-}]. \tag{21}$$

In the above two equations, $\mathbf{q}$ is the coordinate of the pixel where its $c_{\tilde{p}\tilde{e}}[\cdot]$ is affected by the swap between $\mathbf{m}$ and $\mathbf{n}^{(k)}$.

The relationship between $c_{\tilde{p}\tilde{e}}^{(k)}[\mathbf{m}]$ and $c_{\tilde{p}\tilde{e}}^{(k-1)}[\mathbf{m}]$ can be derived from (20) and (21) by letting $\mathbf{q}$ equal to either $\mathbf{m}_{0-}$ or $\mathbf{m}_{1-}$:

$$c_{\tilde{p}\tilde{e}}^{(k)}[\mathbf{m}_{1^+}] = c_{\tilde{p}\tilde{e}}^{(k-1)}[\mathbf{m}_{0^-}] + \Delta c_{\tilde{p}\tilde{p}}^{(k)}, \tag{22}$$

$$c_{\tilde{p}\tilde{e}}^{(k)}[\mathbf{m}_{0^+}] = c_{\tilde{p}\tilde{e}}^{(k-1)}[\mathbf{m}_{1^-}] - \Delta c_{\tilde{p}\tilde{p}}^{(k)}. \tag{23}$$

A. Theoretical Bounds for Toggle Operation

In the current DBS implementation, toggle and swap operations are considered simultaneously and only the one with the largest energy reduction is kept. For the sake of considering the theoretical bounds, we will consider the two operations separately. Therefore, the system will reach two convergent states. We will call the state after the toggle operation converges as "toggle convergent" state and the state after the swap operation converges as "swap convergent" state.

In toggle convergent state, all toggles are not accepted and the bounds are established by (11) and (12). These two inequalities indicate that $c_{\tilde{p}\tilde{e}}[\mathbf{m}]$ is bounded only on one end when only toggle operation is considered. The reason is that toggle operation can only move $c_{\tilde{p}\tilde{e}}[\mathbf{m}]$ in one direction. Therefore, the bound on the other end has to be maintained by the neighboring pixels but not the pixel itself. For example, it is impossible to reduce the value of $c_{\tilde{p}\tilde{e}}[\mathbf{m}_0]$ by toggling because a toggle to 1 can only increase its value by $c_{\tilde{p}\tilde{p}}[\mathbf{0}]$. If we want to reduce $c_{\tilde{p}\tilde{e}}[\mathbf{m}_0]$, it can only be helped by toggling those neighboring pixels with their values equal to 1. This means that the bound on the other end has to be established by the swap operation.

B. Theoretical Bounds for the First Swap

Following the previous definition of $g^{(k)}[\mathbf{m}]$ and $c_{\tilde{p}\tilde{e}}^{(k)}[\mathbf{m}]$, we can see that $g^{(0)}[\mathbf{m}]$ and $c_{\tilde{p}\tilde{e}}^{(0)}[\mathbf{m}]$ represent the values after toggle operation converges.

Assuming convergence is established after the first swap, we need to establish that the toggle bounds in (11) and (12) remain valid. Let $g^{(1)}[\mathbf{m}] = 0$, $g^{(0)}[\mathbf{m}] = 1$, $g^{(1)}[\mathbf{n}^{(1)}] = 1$, and $g^{(0)}[\mathbf{n}^{(1)}] = 0$. From (18), we have:

$$(c_{\tilde{p}\tilde{e}}^{(1)}[\mathbf{n}^{(1)}] - c_{\tilde{p}\tilde{e}}^{(1)}[\mathbf{m}]) \le \Delta c_{\tilde{p}\tilde{p}}^{(1)} \tag{24}$$

which leads to $c_{\tilde{p}\tilde{e}}^{(1)}[\mathbf{m}] \ge (c_{\tilde{p}\tilde{e}}^{(1)}[\mathbf{n}^{(1)}] - \Delta c_{\tilde{p}\tilde{p}}^{(1)})$. Then, applying (22) to the right-hand side, we have:

$$c_{\tilde{p}\tilde{e}}^{(1)}[\mathbf{m}] \ge c_{\tilde{p}\tilde{e}}^{(0)}[\mathbf{n}^{(1)}] \ge -\frac{1}{2}c_{\tilde{p}\tilde{p}}[\mathbf{0}] \tag{25}$$

where (11) is applied on the right-hand side. Similarly, for $g^{(1)}[\mathbf{m}] = 1$, $g^{(0)}[\mathbf{m}] = 0$, $g^{(1)}[\mathbf{n}^{(1)}] = 0$, and $g^{(0)}[\mathbf{n}^{(1)}] = 1$, we have:

$$c_{\tilde{p}\tilde{e}}^{(1)}[\mathbf{m}] \le c_{\tilde{p}\tilde{e}}^{(0)}[\mathbf{n}^{(1)}] \le \frac{1}{2}c_{\tilde{p}\tilde{p}}[\mathbf{0}]. \tag{26}$$

Therefore, we prove that the bounds established in toggle convergent state remain valid after the first swap.

To establish the bounds on the other end, consider $g^{(0)}[\mathbf{m}] = 0$ and $g^{(0)}[\mathbf{n}^{(1)}] = 1$ as in the derivation of (26). The swap convergent state happens in two cases. The first case is that it is reached immediately after toggle convergent state, i.e.,

$$c_{\tilde{p}\tilde{e}}^{(0)}[\mathbf{m}] \ge \left(c_{\tilde{p}\tilde{e}}^{(0)}[\mathbf{n}^{(1)}] - \Delta c_{\tilde{p}\tilde{p}}^{(1)}\right). \tag{27}$$

If this is the case, no swap will occur and no further bounds can be derived. The second case is that convergence is reached after the first swap, (26) can be rewritten as:

$$c_{\tilde{p}\tilde{e}}^{(1)}[\mathbf{m}] \le c_{\tilde{p}\tilde{e}}^{(0)}[\mathbf{n}^{(1)}] = \left(c_{\tilde{p}\tilde{e}}^{(1)}[\mathbf{n}^{(1)}] + \Delta c_{\tilde{p}\tilde{p}}^{(1)}\right) \le \frac{1}{2}c_{\tilde{p}\tilde{p}}[\mathbf{0}]. \tag{28}$$

Using the inequality on the right-hand side, we can derive the following relationship:

$$c_{\tilde{p}\tilde{e}}^{(1)}[\mathbf{n}^{(1)}] \le \left(\frac{1}{2}c_{\tilde{p}\tilde{p}}[\mathbf{0}] - \Delta c_{\tilde{p}\tilde{p}}^{(1)}\right). \tag{29}$$

Replacing $\mathbf{n}^{(1)}$ with $\mathbf{m}$ and combining with (25), we can establish the following statement:

If $g[\mathbf{m}] = 0$ and its value is reached due to a swap after the establishment of toggle convergent state, the bounds on $c_{\tilde{p}\tilde{e}}[\mathbf{m}]$ are:

$$-\frac{1}{2}c_{\tilde{p}\tilde{p}}[\mathbf{0}] \le c_{\tilde{p}\tilde{e}}^{(1)}[\mathbf{m}] \le \left(\frac{1}{2}c_{\tilde{p}\tilde{p}}[\mathbf{0}] - \Delta c_{\tilde{p}\tilde{p}}^{(1)}\right). \tag{30}$$

Similarly, for the case of $g^{(1)}[\mathbf{m}] = 0$, $g^{(0)}[\mathbf{m}] = 1$, $g^{(1)}[\mathbf{n}^{(1)}] = 1$, and $g^{(0)}[\mathbf{n}^{(1)}] = 0$, the following inequality can be derived from (25):

$$c^{(1)}_{\tilde{p}\tilde{e}}[\mathbf{n}^{(1)}] \geq \left(-\frac{1}{2}c_{\tilde{p}\tilde{p}}[\mathbf{0}] + \Delta c^{(1)}_{\tilde{p}\tilde{p}}\right). \quad (31)$$

Combining the above inequality with (26) by replacing $\mathbf{n}^{(1)}$ with $\mathbf{m}$, we can establish the following statement:

If $g[\mathbf{m}] = 1$ and its value is reached due to a swap after the establishment of toggle convergent state, the bounds on $c_{\tilde{p}\tilde{e}}[\mathbf{m}]$ are:

$$\left(-\frac{1}{2}c_{\tilde{p}\tilde{p}}[\mathbf{0}] + \Delta c^{(1)}_{\tilde{p}\tilde{p}}\right) \leq c^{(1)}_{\tilde{p}\tilde{e}}[\mathbf{m}] \leq \frac{1}{2}c_{\tilde{p}\tilde{p}}[\mathbf{0}]. \quad (32)$$

Thus, (30) and (32) prove that the bounds exist on both ends after the first swap.

Two special cases arise when there is not any neighbor available for swap. The first is that $g[\mathbf{m}_0] = 0$ and all its neighbors $g[\mathbf{n}] = 0$ for all $\mathbf{n}$. In this case, $e[\mathbf{n}] \leq 0$ for all $\mathbf{n}$. Combining with (11), we can get the following bounds:

If $g[\mathbf{n}] = 0$ for all $\mathbf{n}$, then

$$-\frac{1}{2}c_{\tilde{p}\tilde{p}}[\mathbf{0}] \leq c_{\tilde{p}\tilde{e}}[\mathbf{m}_0] \leq 0. \quad (33)$$

The second special case is that $g[\mathbf{m}_1] = 1$ and all its neighbors $g[\mathbf{n}] = 1$ for all $\mathbf{n}$. Applying $e[\mathbf{n}] \geq 0$ to (12), we get the following bounds:

If $g[\mathbf{n}] = 1$ for all $\mathbf{n}$, then

$$0 \leq c_{\tilde{p}\tilde{e}}[\mathbf{m}_0] \leq \frac{1}{2}c_{\tilde{p}\tilde{p}}[\mathbf{0}]. \quad (34)$$

For both special cases, the bounds exist after toggle convergent state. There is no need for a swap to occur in order for the bounds to be valid.

C. Theoretical Bounds for the Second Swap and Beyond

For the case that more than one swap occurs for a pixel, we first consider a pixel $g^{(2)}[\mathbf{m}] = 0$ that converges after two swaps. The following inequality holds from (18):

$$c^{(2)}_{\tilde{p}\tilde{e}}[\mathbf{m}] \geq \left(c^{(2)}_{\tilde{p}\tilde{e}}[\mathbf{n}^{(2)}] - \Delta c^{(2)}_{\tilde{p}\tilde{p}}\right). \quad (35)$$

The right hand side of the above inequality is equal to $c^{(1)}_{\tilde{p}\tilde{e}}[\mathbf{n}^{(2)}]$ with $g^{(1)}[\mathbf{n}^{(2)}] = 0$. From (30), we can establish that

$$c^{(2)}_{\tilde{p}\tilde{e}}[\mathbf{m}] \geq c^{(1)}_{\tilde{p}\tilde{e}}[\mathbf{n}^{(2)}] \geq -\frac{1}{2}c_{\tilde{p}\tilde{p}}[\mathbf{0}]. \quad (36)$$

The above inequality can be further extended to the case of $c^{(k)}_{\tilde{p}\tilde{e}}[\mathbf{m}]$ and the case of $g^{(k)}[\mathbf{m}] = 1$. It means that no matter how many swaps occurs, it cannot invalidate the bounds established after the toggle convergent state, i.e., (11) and (12) are always valid after the toggle convergent state.

Applying (32) to $c^{(2)}_{\tilde{p}\tilde{e}}[\mathbf{m}] = c^{(1)}_{\tilde{p}\tilde{e}}[\mathbf{m}] - \Delta c^{(2)}_{\tilde{p}\tilde{p}}$ with $g^{(1)}[\mathbf{m}] = 1$, we have

$$\begin{aligned}\left(-\frac{1}{2}c_{\tilde{p}\tilde{p}}[\mathbf{0}] + \Delta c^{(1)}_{\tilde{p}\tilde{p}} - \Delta c^{(2)}_{\tilde{p}\tilde{p}}\right) &\leq c^{(2)}_{\tilde{p}\tilde{e}}[\mathbf{m}] \\ &\leq \left(\frac{1}{2}c_{\tilde{p}\tilde{p}}[\mathbf{0}] - \Delta c^{(2)}_{\tilde{p}\tilde{p}}\right).\end{aligned} \quad (37)$$

The above inequality means that the boundaries of $c_{\tilde{p}\tilde{e}}[\mathbf{m}]$ are shifted by the amount of $-\Delta c^{(2)}_{\tilde{p}\tilde{p}}$ after the second swap but the range remains the same as that of the first swap.

Combining (36) and (37), the lower bound of $c^{(2)}_{\tilde{p}\tilde{e}}[\mathbf{m}]$ for $g^{(2)}[\mathbf{m}] = 0$ can be expressed as:

$$\max\left\{-\frac{1}{2}c_{\tilde{p}\tilde{p}}[\mathbf{0}], \left(-\frac{1}{2}c_{\tilde{p}\tilde{p}}[\mathbf{0}] + \Delta c^{(1)}_{\tilde{p}\tilde{p}} - \Delta c^{(2)}_{\tilde{p}\tilde{p}}\right)\right\} \quad (38)$$

and its upper bound is $((1/2)c_{\tilde{p}\tilde{p}}[\mathbf{0}] - \Delta c^{(2)}_{\tilde{p}\tilde{p}})$.

If we assume that the swap operation is being considered in the order that

$$\Delta c^{(1)}_{\tilde{p}\tilde{p}} \geq \Delta c^{(2)}_{\tilde{p}\tilde{p}} \geq \cdots \geq \Delta c^{(k)}_{\tilde{p}\tilde{p}}. \quad (39)$$

The lower bound in (38) can be simplified as $(-(1/2)c_{\tilde{p}\tilde{p}}[\mathbf{0}] + \Delta c^{(1)}_{\tilde{p}\tilde{p}} - \Delta c^{(2)}_{\tilde{p}\tilde{p}})$. If the above assumption does not hold, $c_{\tilde{p}\tilde{e}}[\mathbf{m}]$ is still bounded by $\pm(1/2)c_{\tilde{p}\tilde{p}}[\mathbf{0}]$. However, this assumption is certainly a condition that we want to keep because it maintains a tighter bound. Let us call this assumption "monotonically non-decreasing swap condition" or MNDS condition for short.

Similarly, the lower bound of $c^{(2)}_{\tilde{p}\tilde{e}}[\mathbf{m}]$ for $g^{(2)}[\mathbf{m}] = 1$ is $(-(1/2)c_{\tilde{p}\tilde{p}}[\mathbf{0}] + \Delta c^{(2)}_{\tilde{p}\tilde{p}})$ and the upper bound is $((1/2)c_{\tilde{p}\tilde{p}}[\mathbf{0}] - \Delta c^{(1)}_{\tilde{p}\tilde{p}} + \Delta c^{(2)}_{\tilde{p}\tilde{p}})$ under MNDS condition.

We can easily extend the above results to the k-th swap under MNDS condition. For $g^{(k)}[\mathbf{m}] = 0$ when k is odd and for $g^{(k)}[\mathbf{m}] = 1$ when k is even, the lower bound of $c^{(k)}_{\tilde{p}\tilde{e}}[\mathbf{m}]$ is:

$$-\frac{1}{2}c_{\tilde{p}\tilde{p}}[\mathbf{0}] + \sum_{i=2}^{k}(-1)^i \Delta c^{(i)}_{\tilde{p}\tilde{p}} \quad (40)$$

and the upper bound is:

$$\frac{1}{2}c_{\tilde{p}\tilde{p}}[\mathbf{0}] + \sum_{i=1}^{k}(-1)^i \Delta c^{(i)}_{\tilde{p}\tilde{p}}. \quad (41)$$

For $g^{(k)}[\mathbf{m}] = 1$ when k is odd and for $g^{(k)}[\mathbf{m}] = 0$ when k is even, The lower bound of $c^{(k)}_{\tilde{p}\tilde{e}}[\mathbf{m}]$ is:

$$-\frac{1}{2}c_{\tilde{p}\tilde{p}}[\mathbf{0}] - \sum_{i=1}^{k}(-1)^i \Delta c^{(i)}_{\tilde{p}\tilde{p}} \quad (42)$$

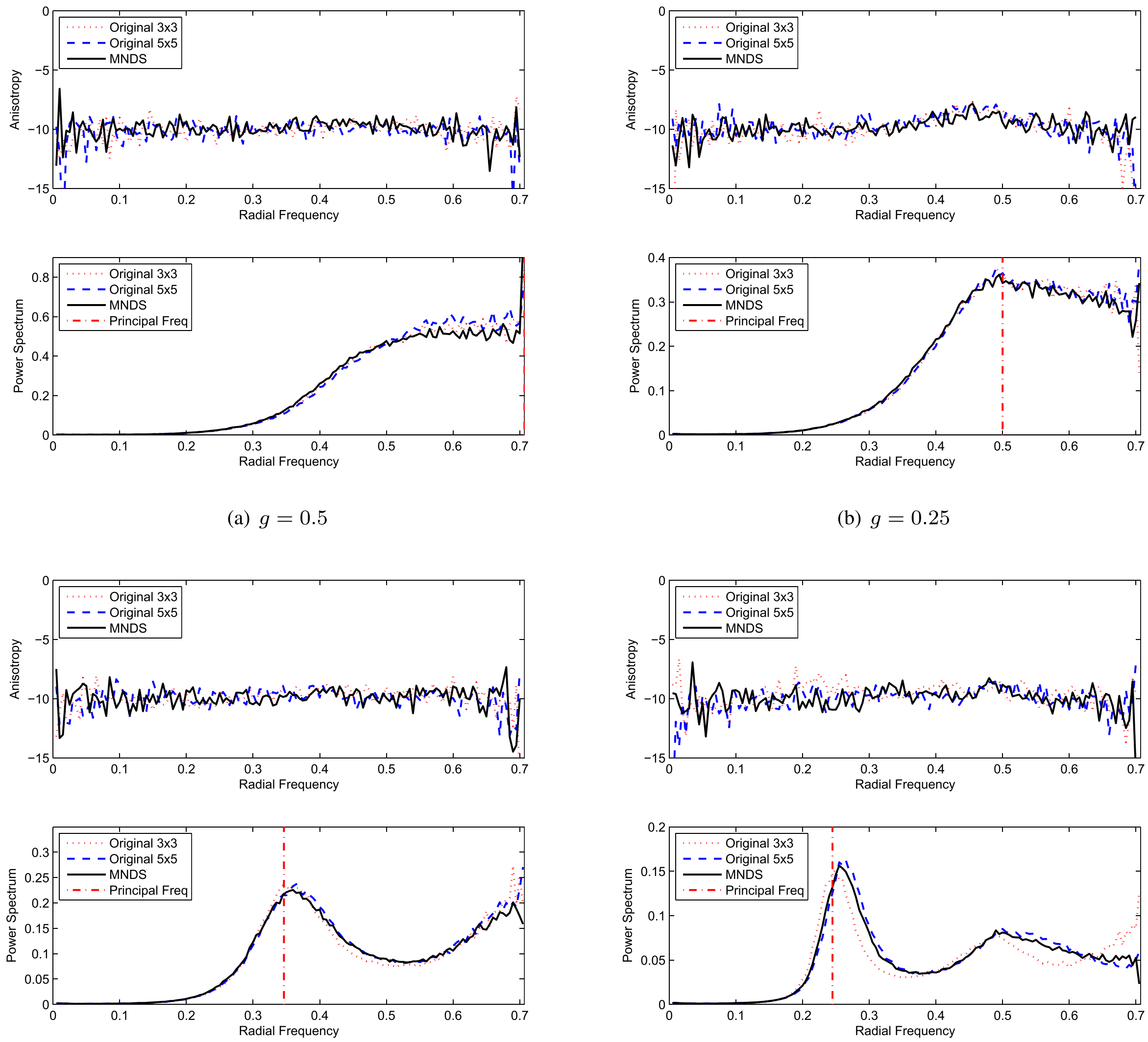
and the upper bound is:

$$\frac{1}{2}c_{\tilde{p}\tilde{p}}[\mathbf{0}] - \sum_{i=2}^{k}(-1)^i \Delta c^{(i)}_{\tilde{p}\tilde{p}}. \quad (43)$$

In summary, the range of $c_{\tilde{p}\tilde{e}}[\mathbf{m}]$ is equal to $(c_{\tilde{p}\tilde{p}}[\mathbf{0}] - \Delta c^{(1)}_{\tilde{p}\tilde{p}})$ if at least one swap occurs after toggle convergent condition and all the swaps obey MNDS condition. The bounds are shifted in alternating direction with monotonically non-increasing step after each swap and the midpoint between the two bounds is:

$$\pm\left(-\Delta c^{(1)}_{\tilde{p}\tilde{p}} + 2\sum_{i=2}^{k}(-1)^i \Delta c^{(i)}_{\tilde{p}\tilde{p}}\right). \quad (44)$$

Intuitively, the midpoint gradually approaches zero as the number of swaps increases. When the midpoint is not zero,



(a) $g = 0.5$ (b) $g = 0.25$ (c) $g = 0.12$ (d) $g = 0.06$

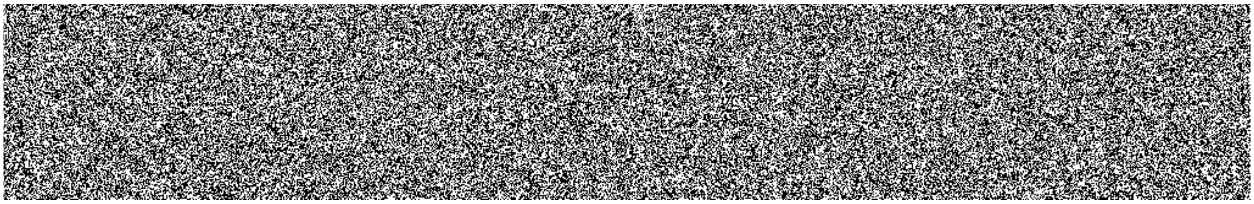
Fig. 1. RAPSD and anisotropy for all the methods tested at the following gray levels: (a) 0.5, (b) 0.25, (c) 0.12, and (d) 0.06.

it implies that a tonal bias may exist and the bias is the most significant when no swap occurs. For example, the two special cases when $g[\mathbf{n}] = 0$ or 1 for all $\mathbf{n}$ exhibit tonal bias as discussed in [5]. However, we cannot come up with a formal proof of the tonal bias at this time.

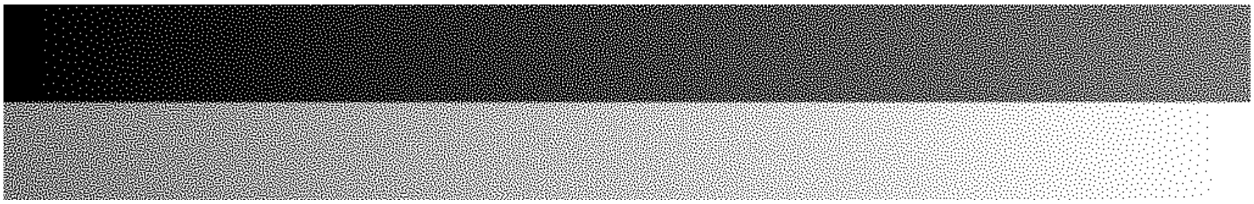
III. Monotonically Non-Decreasing Swap DBS Algorithm

We can make three observations from the theoretical analysis and apply them to improve the implementation of DBS algorithm. The first observation is that the behavior of the bounds is similar to a gradient descent algorithm with $c_{\tilde{p}\tilde{p}}[\mathbf{0}]$ being the step size for toggle operation and $\Delta c^{(k)}_{\tilde{p}\tilde{p}}$ being the step size for the k-th swap operation. Because $c_{\tilde{p}\tilde{p}}[\mathbf{0}] > \Delta c^{(k)}_{\tilde{p}\tilde{p}}$, we should consider only toggle operation until the solution converges before we consider swap operation to achieve faster convergence. The fact that swap operation converges much slower than toggle operation has also been observed in previous researches [16]. The second observation is that $\Delta c^{(1)}_{\tilde{p}\tilde{p}}$ determines the range of $c_{\tilde{p}\tilde{e}}[\mathbf{m}]$ and a swap with a pixel further away from the center of the filter results in a smaller range. Therefore, when considering candidates for swap operation, they should be considered in either the ascending order of $c_{\tilde{p}\tilde{p}}[\mathbf{m}]$ or the descending order of the distance to the center of the filter so that the largest $\Delta c^{(1)}_{\tilde{p}\tilde{p}}$ could be picked and the maximum number of pixels could be considered. The third is that more swaps result in more symmetric bounds. In practice, a smoothly varying $c_{\tilde{p}\tilde{p}}[\mathbf{m}]$ encourages more occurrences of swaps and is beneficial in generating more symmetrical bounds.

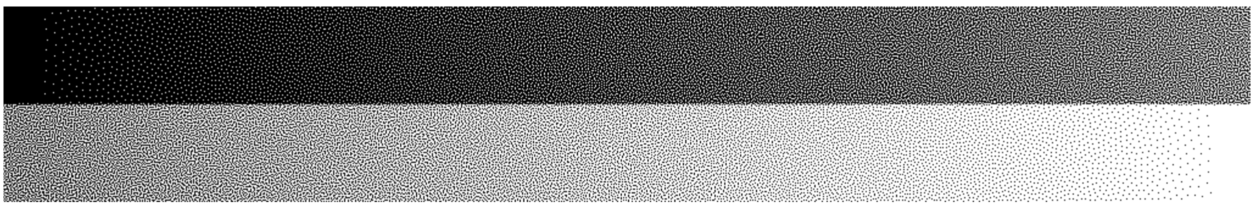
The current implementation of DBS considers toggle and swap together and only the immediate neighbors are considered in swap. Based on the above observations, we propose a new implementation which obeys the MNDS condition and may result in smaller total perceived squared error. The new algorithm works as follows: In the preprocessing stage,



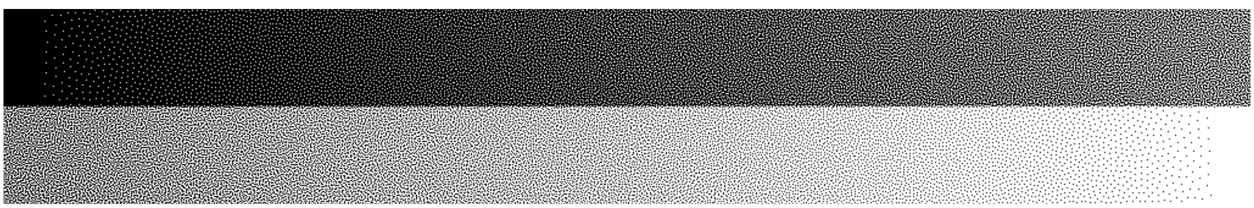
(a)



(b)



(c)



(d)

Fig. 2. The halftone output with randomly distributed dots as the initial halftone. (a) The initial halftone. (b) Original DBS considering 3 neighborhood. (c) Original DBS considering 5 neighborhood. (d) MNDS DBS.

TABLE I

COMPARISON OF MNDS AND ORIGINAL DBS ALGORITHMS

Algorithm	Toggles	Swaps	Time (s)	E
Original 3×3	80924	131225	136.50	14.0488
Original 5×5	72895	125061	305.98	13.0435
MNDS	299904	26445	331.36	13.2596

pixels under the coverage of $c_{\tilde{p}\tilde{p}}[\mathbf{m}]$ are sorted by the value of $c_{\tilde{p}\tilde{p}}[\mathbf{m}]$. These pixels are separated into groups with the same value of $c_{\tilde{p}\tilde{p}}[\mathbf{m}]$. We first consider only toggle operation until it converges. For swap operation, pixel groups of the same filter value are considered in ascending order. Each group is iterated until it converges. We call this new algorithm "MNDS DBS" algorithm.

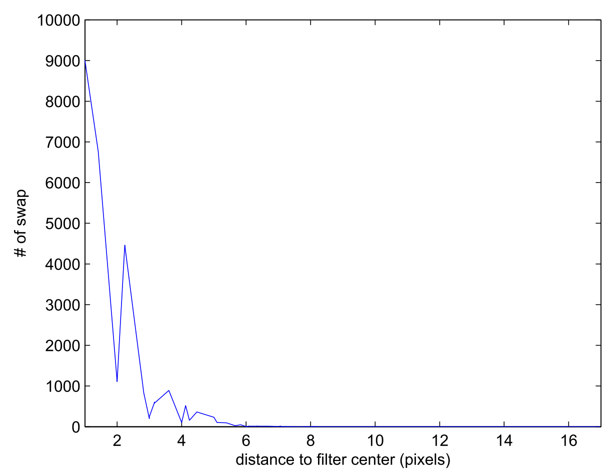
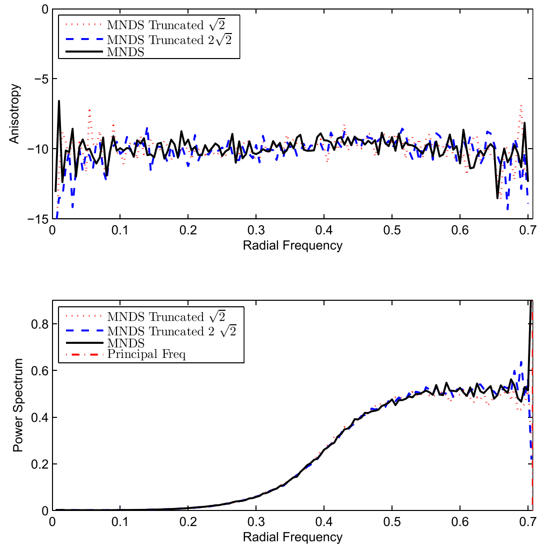


Fig. 3. The number of swap for each pixel group vs. the distance to the center of the filter for MNDS DBS algorithm.

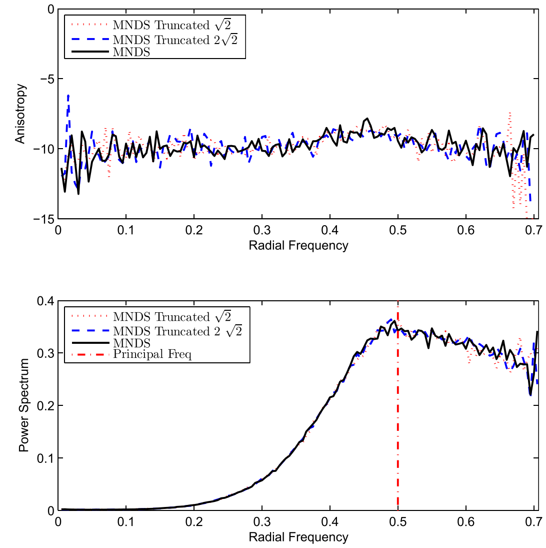
A. Complexity Analysis

The complexity analysis of DBS algorithm can be divided into two parts. The first part is the decision process to determine whether to proceed with an operation and select the best operation. The second part is the update process after an operation is committed. From the complexity analysis, we can conclude that there are two advantages for the new MNDS DBS algorithm.

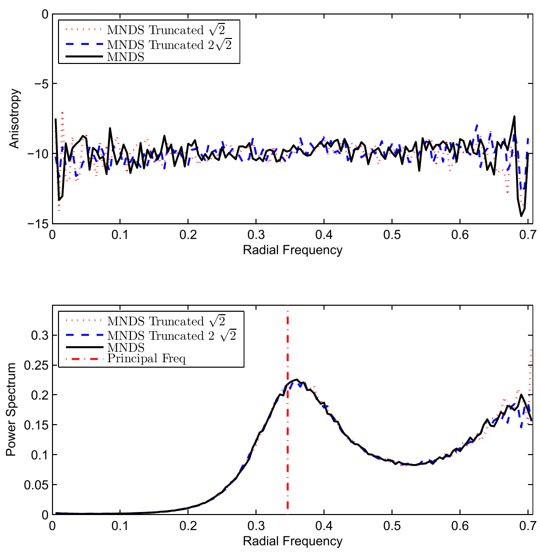
The first advantage is in the decision process where the selection of the best operation is simplified. For toggle operation, only the comparison operation in (7) or (8) is necessary for each pixel in the new algorithm while we need to calculate $\Delta E = c_{\tilde{p}\tilde{p}}[\mathbf{0}] \pm 2 \cdot c_{\tilde{p}\tilde{e}}[\mathbf{m}]$ and compare it with other operations in the original algorithm. Therefore, MNDS DBS saves one addition for every toggle decision.



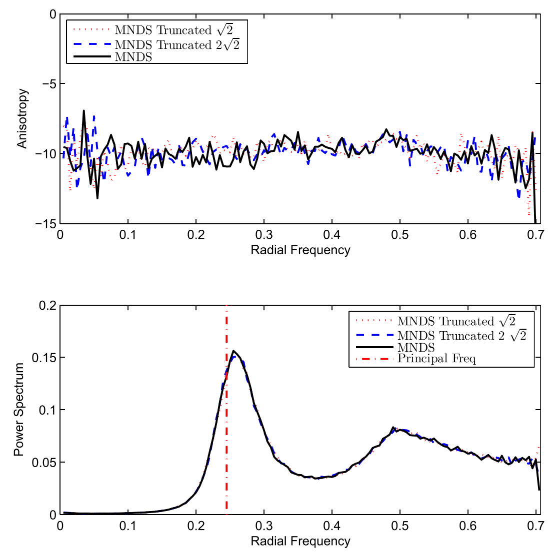
(a) $g = 0.5$



(b) $g = 0.25$



(c) $g = 0.12$



(d) $g = 0.06$

Fig. 4. RAPSD and anisotropy for the truncated MNDS DBS algorithm at the following gray levels: (a) 0.5, (b) 0.25, (c) 0.12, and (d) 0.06.

For swap operation, because $\Delta c_{\tilde{p}\tilde{p}}^{(k)}$ is the same for each pixel group, MNDS DBS first determine whether the condition $(c_{\tilde{p}\tilde{e}}[\mathbf{m}_1] - c_{\tilde{p}\tilde{e}}[\mathbf{m}_0]) > \Delta c_{\tilde{p}\tilde{p}}^{(k)}$ is satisfied. Only pixels passed the test are compared to find the best pixel with the maximum $(c_{\tilde{p}\tilde{e}}[\mathbf{m}_1] - c_{\tilde{p}\tilde{e}}[\mathbf{m}_0])$. On the contrary, we need to calculate $\Delta E = \Delta c_{\tilde{p}\tilde{p}} + c_{\tilde{p}\tilde{e}}[\mathbf{m}_0] - c_{\tilde{p}\tilde{e}}[\mathbf{m}_1]$ and compare them in the original algorithm. Let the probability that a pixel satisfies $(c_{\tilde{p}\tilde{e}}[\mathbf{m}_1] - c_{\tilde{p}\tilde{e}}[\mathbf{m}_0]) > \Delta c_{\tilde{p}\tilde{p}}^{(k)}$ be p_s. For every swap decision, MNDS DBS needs one addition and $1 + p_s$ comparison while the original DBS needs two additions and one comparison. Therefore, MNDS DBS needs one less addition and p_s more comparison for every swap decision. Since $p_s \leq 1$, we can conclude that both toggle and swap decisions in the new algorithm require less computation than the original algorithm.

The second advantage is the reduced cost of memory access when updating $c_{\tilde{p}\tilde{e}}[\mathbf{m}]$. From (20) and (21), a swap needs to update $c_{\tilde{p}\tilde{e}}[\mathbf{m}]$ in two areas under the coverage of $c_{\tilde{p}\tilde{p}}[\mathbf{m}]$ while a toggle only needs to update in one area as in (9) or (10). This means that the memory access cost of a swap update is twice of a toggle update. Let the number of toggles be N_t and the number of swaps be N_s. The memory access cost is directly proportional to $N_t + 2N_s$. The separation of toggle and swap leads to increased N_t and reduced N_s as compared to the original algorithm. Thus, the memory access cost is reduced. The reason can be explained from

TABLE II
COMPARISON OF TRUNCATED MNDS DBS ALGORITHMS

Distance	Swaps	Time (s)	E
$\sqrt{2}$	25898	40.99	13.2136
2	25425	44.89	13.1381
$2\sqrt{2}$	25845	80.33	13.1696
4	26558	116.32	13.1986
5	26551	148.87	13.2812
6	26733	185.21	13.2530
7	26608	204.82	13.2718

the viewpoint of a gradient descent algorithm as previously mentioned. A swap operation occurred prior to toggle convergent state can be viewed as a movement with a smaller step before the solution reaches the target area. Therefore, more swaps are needed to achieve the same movement of one toggle before toggle convergent state. This can also be explained by the fact that many local minima exist for halftoning and moving toward a local minimum (i.e., choosing the largest energy reduction) may mean taking a zigzag path instead of a more direct path to the target. From the viewpoint of theoretical analysis, a swap operation occurred prior to toggle convergent state cannot contribute in the establishment of the swap convergent bound because it is counted as two toggles instead of one swap in the theoretical analysis. Therefore, separating toggle and swap is advantageous in terms of speeding up convergence or reducing memory access cost as will be shown in the experiments.

B. Truncation of Monotonically Non-Decreasing Swap Condition

The execution time of the MNDS DBS algorithm can be further reduced by utilizing the fact that the number of swap at the edge of $c_{\tilde{p}\tilde{p}}[\mathbf{m}]$ is small because the difference between $c_{\tilde{p}\tilde{e}}[\mathbf{m}_1]$ and $c_{\tilde{p}\tilde{e}}[\mathbf{m}_0]$ has to be larger in order to satisfy the swap criteria (16) or (17). Therefore, those pixel groups located at the edge of the filter can be skipped without significantly increasing the total squared perceived error. This truncated MNDS DBS algorithm will be further examined in the following experiments.

IV. EXPERIMENTAL RESULTS

We compare the proposed MNDS DBS algorithm with the original DBS algorithm that considers pixels in a 3×3 or 5×5 neighborhood. The original DBS algorithm is implemented as described in [5] but without block-based speed-up strategy [4]. The HVS filter used is the one described by Kim et al. in [10] with $(\alpha, \beta) = (6.65, 1.73)$ which produces a more homogeneous texture in the midtone areas. Both algorithms are implemented in Matlab R2012a (v7.14.0.739) and tested on a computer running 64-bit Windows 7 with Intel Core i7-3770 3.4GHz CPU and 8GB RAM. It is worth mentioning that if the algorithms are implemented in compiled programming language such as C, the performance will be much better than the Matlab implementation.

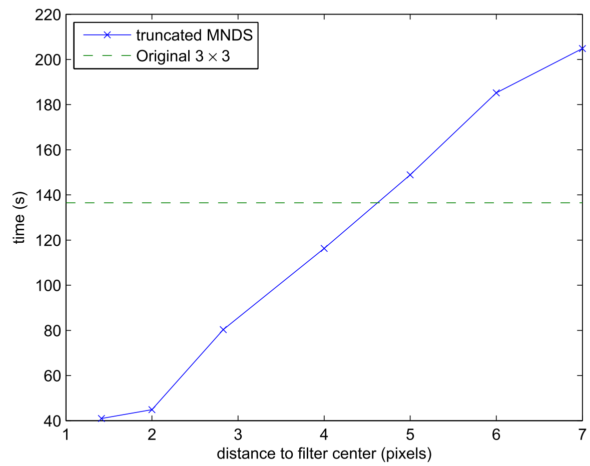


Fig. 5. The execution time vs the truncated distance to the center of the filter for truncated MNDS DBS algorithm.

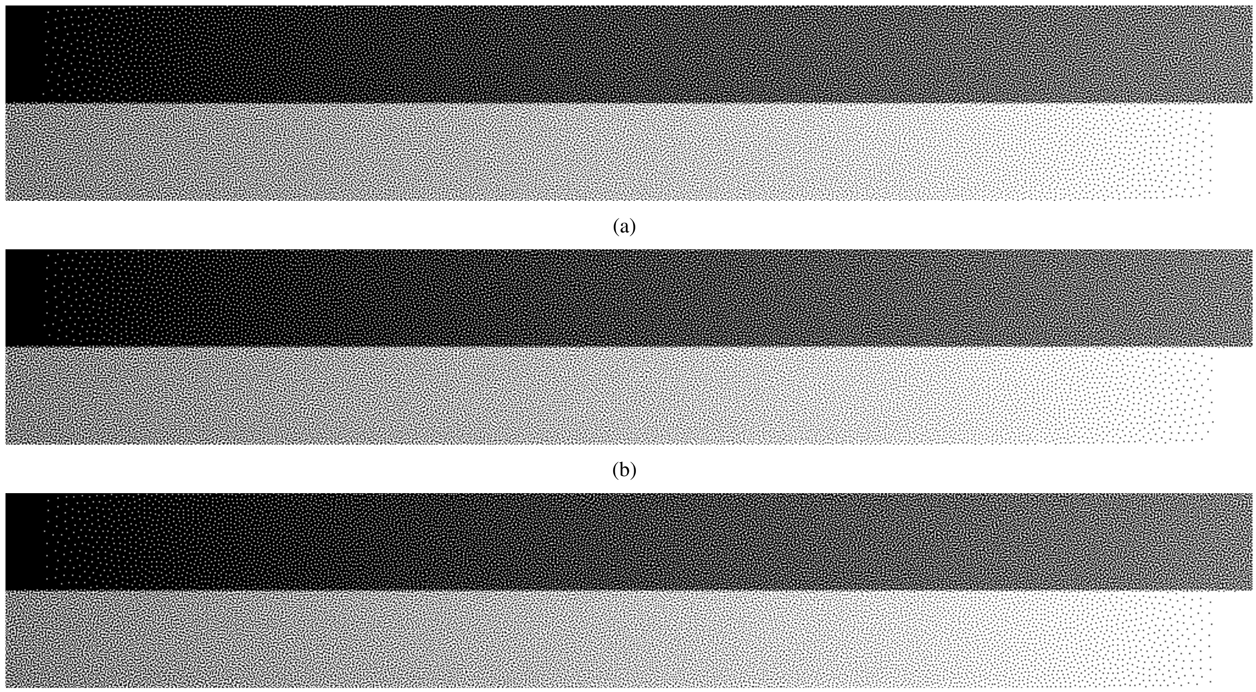
A. Radially Averaged Power Spectral Density and Anisotropy

Fig. 1 shows the radially averaged power spectral density (RAPSD) and anisotropy at four different gray levels for all the methods tested. From the figure, we can see that the behaviors for both the original DBS and MNDS DBS are similar except for gray level 0.06. At $g = 0.06$ where the distance between "on" pixels is larger, RAPSD's for both MNDS DBS and original 5×5 DBS shows less oscillation than original 3×3 DBS and RAPSD of MNDS DBS is slightly flatter at high frequency region than the original DBS covering 5×5 neighborhood. The reason that power spectral density of MNDS DBS is only slightly different from the original DBS is that the number of swapped pixels with distance larger than $2\sqrt{2}$ (the distance for 5×5 neighborhood) is less than 3% of the total number of swapped pixels in MNDS DBS for all the cases shown in the figure. The reason for the small number of swapped pixels at large distance is that only neighbors with distance of $1/\sqrt{g}$ change with higher possibilities and the majority of these pixels has already been set in the toggle stage when the image has uniform gray level.

B. Ramp Image

Fig. 2 shows the DBS halftone outputs of a gray-level ramp image using randomly distributed dots as the initial halftone. The size of the ramp image is 1024×160. As the results of RAPSD and anisotropy indicated, the visual quality of the three algorithms are very close.

Table I summarizes the number of toggles, the number of swaps, the execution time for each algorithm, and the total squared perceived error. From the table, we can see that the number of toggles is much larger while the number of swaps is much smaller for MNDS DBS algorithm as compared to the original DBS algorithm. In these experiments, the number of pixels considered in MNDS DBS is 25×25 pixels which is 25 times more than the 5×5 neighborhood considered in the original DBS. From the complexity analysis in Section III and considering the significantly increased coverage of pixels



(a)

(b)

(c)

Fig. 6. The halftone output of truncated MNDS DBS algorithm for truncated distance of (a) $\sqrt{2}$ pixels, (b) $2\sqrt{2}$ pixels, and (c) 5 pixels.

in MNDS DBS, the execution time in MNDS DBS algorithm is only 8.3% more than that of the original 5×5 DBS. Therefore, MNDS DBS does appear to be much more efficient than the original DBS algorithm. While the total squared perceived errors for both original 5×5 DBS and MNDS DBS are less than that of original 3×3 DBS, the total squared perceived error for MNDS DBS is slightly larger than that of original 5×5 DBS. It is worth noting that the total squared perceived error depends on the HVS filter used. When using a simple Gaussian filter, we observe a 30% reduction on the total squared perceived error for MNDS DBS as compared to original 5×5 DBS.

Fig. 3 shows the number of swaps occurred in each pixel group. We can see that there is no swap when the distance to the center of the filter is larger than $\sqrt{65} = 8.0623$. The number of swaps for pixel groups with distance larger than $2\sqrt{2}$, i.e., beyond the 5×5 neighborhood, is 16% of the total number of swaps in this case.

C. Truncation of Monotonically Non-Decreasing Swap Condition

The results in Fig. 3 indicate that the truncated version of MNDS DBS proposed in Section III-B is indeed practical. We perform the same experiments in the previous two subsections but skip the pixel groups with their distance to the center of the filter being larger than $\sqrt{2}$ to 7 pixels. Fig. 4 shows the RAPSD and anisotropy for the truncated MNDS DBS. From the figure, we can see that the performance of the truncated version is very similar to the full search. Only when the truncated distance reduced to $\sqrt{2}$, the RAPSD shows more oscillations at the high frequency.

For the ramp image, Table II summarizes the number of swaps, the execution time, and the total squared perceived error for each truncated distance. The number of the toggles for all the cases is 299,904 which is the same as the full search. Fig. 5 plots the execution time vs. the truncated distance. The execution time for the original 3×3 DBS algorithm is also plotted in Fig. 5 as a reference. Fig. 6 shows the halftone outputs at three different truncated distances.

From Table II, we can see that the total squared perceived error does not vary much for all the truncated distances. As a result, there is almost no perceived differences among the halftoned images in Fig. 6. However, from Fig. 5, we can see that MNDS DBS can cover distance of more than 4 pixels or more than twice the number of pixels in the 3×3 neighborhood for the same execution time. In other words, MNDS DBS can cover 3×3 neighborhood (i.e., distance of $\sqrt{2}$) in 30% of execution time of the original 3×3 DBS algorithm. This confirms our previous argument that separating toggle and swap is more efficient than considering them together.

V. Conclusion

In this paper, we present the theoretical analysis that proves the conjecture: the filtered error $|c_{\tilde{p}\tilde{e}}[\mathbf{m}]|$ is bounded by the value $(1/2)c_{\tilde{p}\tilde{p}}[\mathbf{0}]$. We show that one side of the bounds is established by the toggle operation and the other side of the bounds is established by the swap operation. Additionally, two important results are derived from the theoretical analysis. First, if a swap occurs further away from the center of the filter, the range of the bounds is smaller. Second, if more swaps occurs, the bounds become more symmetrical.

Based on these two results, we propose a new DBS algorithm called "monotonically non-decreasing swap" (MNDS) DBS which considers toggle and swap separately and the swap operations are considered in the order from the edge to the center of the filter. In the experiments, we show that the MNDS DBS algorithm is more computationally efficient than the original DBS algorithm while producing halftoned image of the same quality.

REFERENCES

[1] R. Ulichney, *Digital Halftoning*. Cambridge, MA, USA: MIT Press, 1987.

[2] R. W. Floyd and L. Steinberg, "An adaptive algorithm for spatial gray scale," *J. Soc. Inf. Display*, vol. 17, no. 2, pp. 75–77, 1976.

[3] M. Analoui and J. P. Allebach, "Model-based halftoning using direct binary search," *Proc. SPIE*, vol. 1666, pp. 96–108, Aug. 1992.

[4] D. J. Lieberman and J. P. Allebach, "Efficient model based halftoning using direct binary search," in *Proc. IEEE Int. Conf. Image Process.*, vol. 1. Oct. 1997, pp. 775–778.

[5] D. J. Lieberman and J. P. Allebach, "A dual interpretation for direct binary search and its implications for tone reproduction and texture quality," *IEEE Trans. Image Process.*, vol. 9, no. 11, pp. 1950–1963, Nov. 2000.

[6] A. Leon-Garcia, *Probability, Statistics, and Random Processes for Electrical Engineering*, 3rd ed. Upper Saddle River, NJ, USA: Prentice-Hall, 2009, p. 534.

[7] D. Kacker, T. Camis, and J. P. Allebach, "Electrophotographic process embedded in direct binary search," *IEEE Trans. Image Process.*, vol. 11, no. 3, pp. 243–257, Mar. 2002.

[8] F. A. Baqai and J. P. Allebach, "Halftoning via direct binary search using analytical and stochastic printer models," *IEEE Trans. Image Process.*, vol. 12, no. 1, pp. 1–15, Jan. 2003.

[9] J.-H. Lee and J. P. Allebach, "Inkjet printer model-based halftoning," *IEEE Trans. Image Process.*, vol. 14, no. 5, pp. 674–689, May 2005.

[10] S. H. Kim and J. P. Allebach, "Impact of HVS models on model-based halftoning," *IEEE Trans. Image Process.*, vol. 11, no. 3, pp. 258–269, Mar. 2002.

[11] J.-H. Lee and J. P. Allebach, "Colorant-based direct binary search halftoning," *J. Electron. Imag.*, vol. 11, no. 4, pp. 517–527, Oct. 2002.

[12] A. U. Agar and J. P. Allebach, "Model-based color halftoning using direct binary search," *IEEE Trans. Image Process.*, vol. 14, no. 12, pp. 1945–1959, Dec. 2005.

[13] Z. He, "Hierarchical colorant-based direct binary search halftoning," *IEEE Trans. Image Process.*, vol. 19, no. 7, pp. 1824–1836, Jul. 2010.

[14] G.-Y. Lin and J. P. Allebach, "Multilevel screen design using direct binary search," *J. Opt. Soc. Amer. A*, vol. 19, no. 10, pp. 1969–1982, Oct. 2002.

[15] C. Lee and J. P. Allebach, "The hybrid screen—Improving the breed," *IEEE Trans. Image Process.*, vol. 19, no. 2, pp. 435–450, Feb. 2010.

[16] P. Goyal, M. Gupta, C. Staelin, M. Fischer, O. Shacham, and J. P. Allebach, "Clustered-dot halftoning with direct binary search," *IEEE Trans. Image Process.*, vol. 22, no. 2, pp. 473–487, Feb. 2013.

Jan-Ray Liao received the B.S. degree in electrical engineering from National Taiwan University, Taipei, Taiwan, in 1989, and the M.S. and Ph.D. degrees in electrical engineering from Stanford University, Palo Alto, CA, in 1993 and 1997, respectively. From 1997 to 2001, he was an Assistant Professor with the Department of Electrical Engineering, National Chung Hsing University, Taichung, Taiwan. Since 2001, he has been an Associate Professor with the Department of Electrical Engineering.