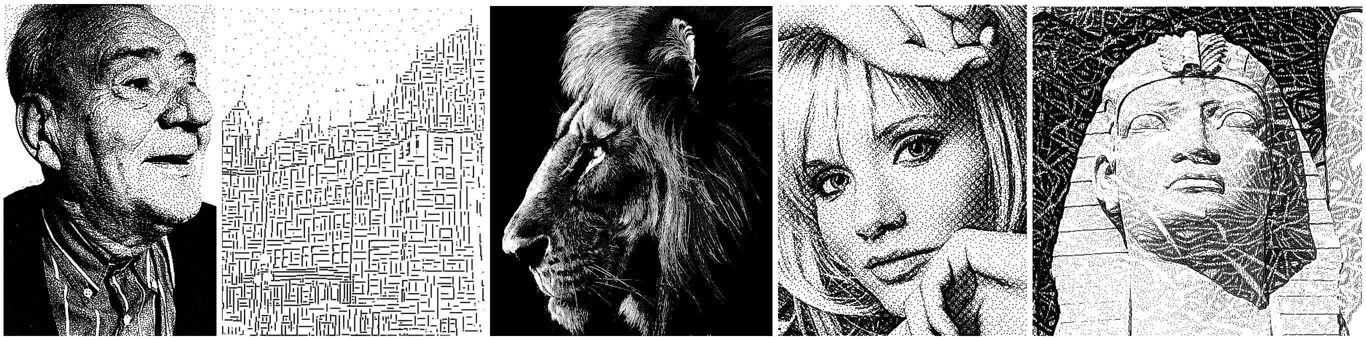
Structure-preserving Stippling by Priority-based Error Diffusion

Hua Li* David Mould†

Carleton University



(a) Our basic method (b) High-level stylization (c) Scratchboard (d) Screening (e) Patterns

Figure 1: Our results.

ABSTRACT

This paper presents a new fast, automatic method for structure-aware stippling. The core idea is to concentrate on structure preservation by using a priority-based scheme that treats extremal stipples first and preferentially assigns positive error to lighter stipples and negative error to darker stipples, emphasizing contrast. We also use a nonlinear spatial function to shrink or exaggerate errors and thus implicitly provide global adjustment of density. Our adjustment respects contrast and hence allows us to preserve structure even with very low stipple budgets. We also explore a variety of stylization effects, including screening and scratchboard, all within the unifying framework of stippling.

Index Terms: Computer Graphics [I.3.3]: Picture/Image Generation—Display algorithms

1 INTRODUCTION

Stippling is a technique of drawing, engraving, or painting using small dots or short strokes. Most computational stippling effects seek either a halftoning effect or an illustrative effect. Other possible effects have received less attention. The concept of evenly-spaced stipples for tone similarity is the mainstream in the literature [3, 5, 10, 17, 18, 20], but irregular spacing is an alternative [9, 12] which is more natural and closer to artist's work. Figure 2 shows two artists' examples and demonstrates the irregularity due to imprecise placement or deliberate preference to satisfying the structural requirement. Automatic methods for structural preservation exist [8, 13] but are not common. However, structure awareness is necessary to preserve fine details in the output images.

Halftoning has strong connections to stippling: black and white stippling results might be treated as a kind of halftoning. The stippled man shown in Figure 2 is a typical halftoning effect. However, nobody will say the stippled woman on the right side is halftoning. As the stipple count decreases, to balance the high frequency part which is the structure and the low frequency part which is the tone, the importance of structure concerns will be raised. Although

*e-mail: hli1@connect.carleton.ca

†e-mail:mould@scs.carleton.ca

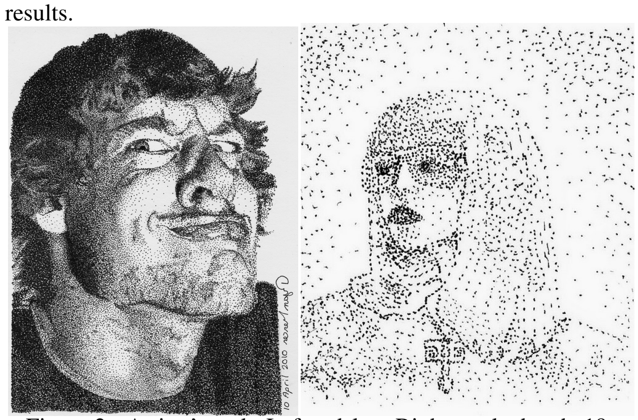
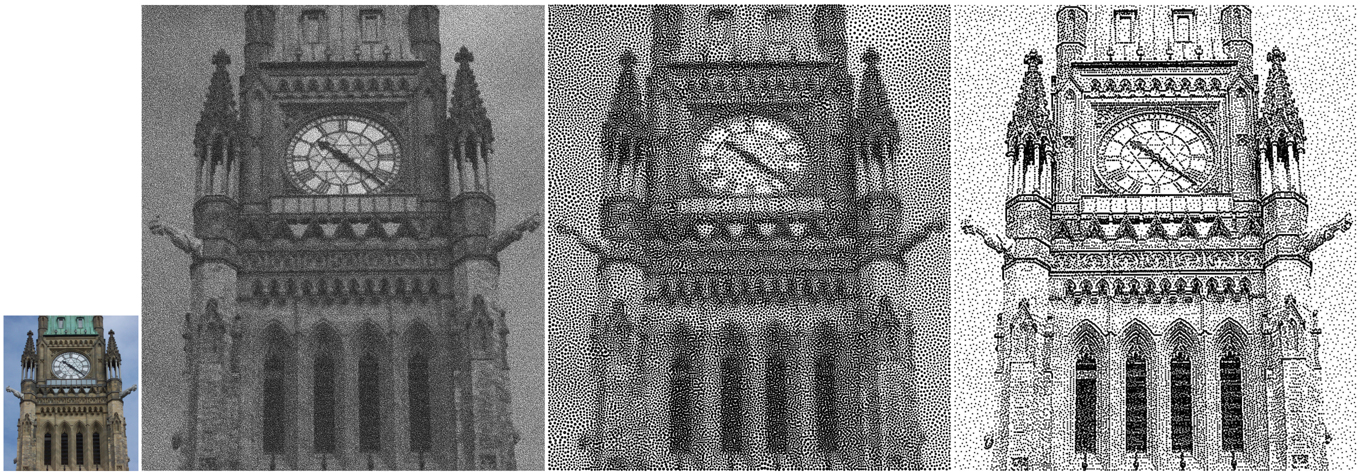


Figure 2: Artists' work. Left: cdslug; Right: makedonche19.

artists favor using very small dots for stippling, a stipple is still larger than a pixel. Deussen et al. characterize stippling in opposition to halftoning, saying that a *smaller number of relatively large dots is used which vary in size and sometimes in shape* [3]. Our interest has been provoked by recent structure-focused halftoning approaches [1, 11, 15] which provide excellent texture preservation as well as good tone matching, and we sought a stippling algorithm that has these properties. We focus on the key challenge of going from a huge number of pixels to a relatively small number of stipples while still preserving structural details with some tone suggestion. This paper presents a novel approach for stippling problems, attacking them from the foundation of priority-based error diffusion. Our contributions include the following:

- We give a mechanism for reducing the huge number of pixels created by contrast-aware halftoning to a relatively small number of stipples while considering structure preservation.
- We propose new weight distribution schemes (masks) for different stippling styles. While designed for stippling, our masks can directly be used for other stylization applications in image processing. Especially, our edge-exclusion mask further enhances structural detail. An Eden-like growth scheme creates irregular stipple distributions.
- We introduce a multiple-stage process to flexibly promote multiple groups of specified features.

The final result of our investigation is a fast, automatic structure-preserving stippling method with diverse new effects unlike those



(a) (b) Martin et al.'s method [12] (c) Secord's method [18] (d) Our stippling

Figure 3: Comparisons. (a) Original image; (b) Huge number of stipples; (c) 30,209 stipples; (d) 30,209 stipples with $P(5,3,0,7)$.

produced by previous methods. Our method can express very complicated and detailed content, extremely difficult to achieve when considering only tone and not structure. Figure 3 shows comparisons with other stippling methods. Our method shows the key details very nicely; for example, look at the frames of the windows and the tops of the arches, which are difficult to discern in the images from Martin et al.'s and Secord's methods.

2 Previous Work on Stippling

Early stippling methods employed iterative relaxation (Lloyd's method) to evenly distribute dots [3]. Secord [18] proposed weighted Voronoi stippling for gentle tone imitation. Much later research [17, 20] also depended on relaxation methods. Essentially, relaxation produces good tone quality by smoothing out high frequency. A lot of researchers agree that high quality distributions are characterized by blue noise, and employ techniques including Wang tiles [10] to enforce blue noise. Stippling created by artists, however, is rather irregular, which presents a more natural look. Thus, some researchers [9, 12] mimicked the statistical properties learned from artists' examples to propose an irregular distribution.

Many stippling methods are focused on tone matching ignoring structural consistency; others segment important regions or lines, perhaps with user assistance, and then distribute dots along features. We favor structure preservation and also prefer not to impose much demands on the user; our goal is an automatic method. Some feature-based methods appear in the literature. Schlechtweg et al. [17] used multi-agent systems to place dots based on image edges. Vanderhaeghe et al. [20] applied an importance map to constrain stroke placement. Kim et al. [8] brought the concept of directional stippling automatically aligned to edges, providing illustration-like effects. Most of these are based on iterative relaxation with different constraints. The structure-guided stippling presented by Mould [13] used a weighted graph and minimized a cost function by path search to choose dots, deliberately paying little attention to tone.

Previous stippling methods typically did not seek to introduce new styles. Historically, stippling has been viewed as its own style, with success measured by tone reproduction [3, 10, 18]. For example, hand-drawn results created by example-based stippling [9, 12] usually favor faithful tone representation. The stippling proposed by Kim et al. [8] is called hedcut stippling, based on even spacing along feature edges. Jang and Hong [6] and Yang and Yang [23] transformed stippling for pointillism effects. There are some previous halftoning methods [14, 16, 22] dealing with halftoning and artistic styles, but still focused on tone, and hence their quality and structure preservation are not very high. Streit and Buchanan [19] presented hatching styles based on importance-driven halftoning, which is close to our work. However, we are more concerned about using fewer stipples to preserve structures.

3 Basic Method for Structure-Aware Stippling

Error diffusion algorithms, mainstream in halftoning techniques, typically process the image pixel by pixel, thresholding the current pixel and then distributing the resulting error in some fashion. If a pixel is thresholded down, positive error is generated; if a pixel is thresholded up, the error is negative. Traditional error diffusion usually updates neighboring pixels with current intensity $I(m,n)$ under a mask as follows:

$$I(m,n)' = I(m,n) + \hat{w}_{mn} * e_{xy}, \tag{1}$$

where $I(m,n)'$ is the new intensity at position (m,n) after diffusing error e_{xy} from a center pixel at position (x,y), and $\hat{w}_{mn}$ is the normalized weight.

In adapting a stippling algorithm from an error diffusion algorithm, we are placing black dots rather than black and white pixels. Naïvely, we could simply place one black stipple for every black pixel, but the resulting stipple count would be extremely high and the image would not resemble a stippled image at all. However, we can adapt error diffusion algorithms to place stipples: processing the pixels in some order, placing a stipple when the intensity is below the threshold, and then distributing the error whenever a pixel is processed. The outcome of processing a pixel is a decision whether to place or not to place a stipple at that location. If the decision is to place a stipple, we incur positive error – the location becomes darker than its actual value, so nearby pixels are lightened. Conversely, if the algorithm decides against placing a stipple, the error is negative – the location is lighter than its true value and we must darken nearby pixels to compensate. Adjustment of error gives control over the density of stipples, since the error is diffused to future pixels and affects the number of stipples to be created. We will use contrast-aware halftoning (CAH) [11] as our basic error distribution algorithm due to its good structure preservation and the flexible priority-based scheme.

3.1 Stippling based on Contrast-aware Halftoning

Our method is a variant of error diffusion, adapting from contrast-aware halftoning. Our structure-preserving property is due to two elements: the use of priority order for processing pixels, and the contrast-preserving way of distributing error in a neighborhood.

Pixels are processed in priority order, where higher priority pixels are those closer in intensity to extreme values (pure white or pure black); note that as error is diffused from processed pixels, a pixel's intensity and hence priority can change.

$$p(x,y) = \begin{cases} 255 - I(x,y) & \text{if } |255 - I(x,y)| > |I(x,y)| \\ I(x,y) & \text{otherwise.} \end{cases} \tag{2}$$

Equation 2 calculates the priority $p(x,y)$ at position (x,y). Error is diffused within a neighborhood according to a set of weights computed based on the error and the destination pixel's current intensity. The policy for weight calculation shown in Equation 3 is that lighter pixels tend to be lightened further, while dark pixels lighten with difficulty but can easily be darkened. The goal here is to respect the pixel's initial predisposition towards dark or light when distributing the error.

$$w_{mn} = \begin{cases} \frac{I(m,n)}{(r_{mn})^b} & \text{if } e_{xy} > 0 \\ \frac{255-I(m,n)}{(r_{mn})^b} & \text{otherwise.} \end{cases} \tag{3}$$

where r_{mn} is the distance of pixel m,n from the mask centre, and b is a parameter ($b = 2$, as in CAH [11]). We have intensity values ranging from 0 to 255 (based on 8-bit images). The weights are normalized in Equation 5, computed by dividing by the sum of weights for all pixels within the mask region that were not previously set to their final values; such pixels are indicated as $M(m,n) = true$:

$$W_{total} = \sum_{(m,n)\in neighbors} w_{mn} \tag{4}$$

$$\hat{w}_{mn} = \frac{w_{mn}}{W_{total}} \tag{5}$$

Deussen et al. [3] argued that artists rarely use stipples which vary in size by more than a factor of two. In Equation 6, we follow this advice and have our maximum stipple twice the size of the minimum, so $r_{\min} = 1$ and $r_{\max} = 2$. Stipple size r varies linearly with the original intensity I_{origin}; a minimum size stipple is placed at a site of intensity 255 (completely white), up to a maximum size stipple when the intensity drops to zero (completely black).

$$r \leftarrow r_{\min} + \frac{(r_{\max} - r_{\min}) \times (255 - I_{\text{origin}})}{255} \tag{6}$$

Our algorithm as presented so far still creates huge numbers of stipples. We propose adjusting the error carried forward in the diffusion process in such a way as to reduce stipple counts. Recall that stipples are only placed when the pixel's present value is below the threshold. Dark pixels will initially be below the threshold, but may be raised above it by accumulating positive error from nearby stipples. Light pixels will be initially above the threshold, but if sufficient negative error accumulates, a stipple will be placed. Our strategy is to reduce the magnitude of the assigned error (called shrinking) when it is negative, and increase it (called exaggeration) when it is positive. Reducing negative error limits production of stipples in lighter portions of the image. Conversely, increasing positive error exaggerates the impact of each stipple: the area near the stipple is lightened by a greater proportion than is actually warranted for faithful tone reproduction and thus fewer stipples will be created. Notice that this adjustment does not prevent us from preserving structure with the stipple distributions, since we still use the priority-based scheme and adhere to the policy of preferentially assigning positive error to lighter pixels and negative error to darker pixels. We propose using gamma correction for the shrinking and exaggeration functions, since gamma correction is effectively a mechanism for tuning contrast. In Equation 7, the spatially-related adjustment factors s_{xy} for negative and positive errors are independently controlled, using G_- and G_+ respectively.

$$s_{xy} = \begin{cases} (1/r)^{G_-} & \text{if } e_{xy} < 0 \\ (r)^{G_+} & \text{otherwise.} \end{cases} \tag{7}$$

With larger powers G_- and G_+, the adjustment brings less negative error for nearby pixels (producing fewer stipples) and more positive error (also producing fewer stipples), satisfying our objective of stipple reduction. The parameters G_- and G_+ adjust the degree of shrinking and exaggeration for light and dark regions respectively, which gives us some flexibility in density control. The net effect is to lighten the image, but we retain structure and contrast.

We need two further elaborations to make the story complete. First, notice that a stipple is usually larger than a pixel, and hence the positive error from placing a stipple is not simply the intensity from the pixel. We introduce a correction term e_0 for the case when a stipple is placed, in which case the intensity updates become

$$I(m,n)' = I(m,n) + \hat{w}_{mn} * (e_{xy} + e_0) * s_{xy}, \tag{8}$$

where e_0 is given by $e_0 = (A_{\text{stipple}} - A_{\text{pixel}}) * k$. The values A_{stipple} and A_{pixel} are the area of a stipple and of a pixel respectively; k is a user-adjustable parameter, which has an influence in black areas. Second, extreme exaggeration of error often causes updated intensity values to fall outside the usual [0,255] range, in which case the values are clamped. Clamping, however, causes some error to be unaccounted for. If a larger mask is used, the increments per pixel are smaller and less clamping occurs. Since exaggeration reduces stipple counts, a larger mask size allows the full effect of exaggeration to take place, and yields fewer stipples. The details are shown in Procedure 1.

Procedure 1 Basic Structure-Aware Stippling Method

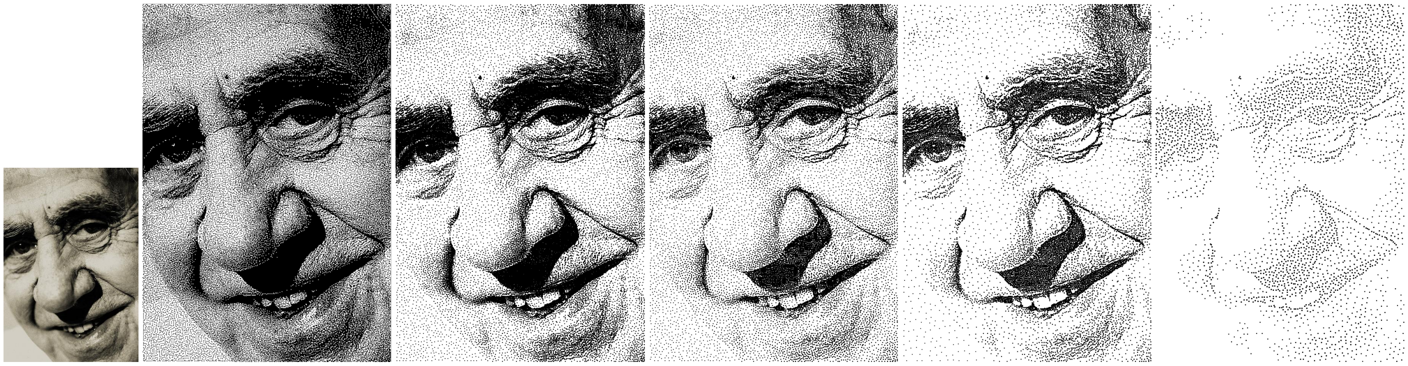
Input: $G_-, G_+, k, D, I_{origin}$ [I_{origin}, a given 8-bit image]
Output: $stippleslist$ [A list of stipples with size information]

$I \leftarrow I_{origin}$
$M \leftarrow false$ [$false$ means unprocessed pixels; $true$ means processed pixels.]
$Heap \leftarrow BuildPriority(I_{origin})$ [Priority]
while !Heap.empty() **do**
  the pixel (x,y) with the highest priority $p_{hi} \leftarrow Heap.pop()$
  $p(x,y) \leftarrow CalculatePriority(I(x,y))$ [Equation 2]
  if $p_{hi} \neq p(x,y)$ [Current priority and old priority not equal] **then**
    $Heap.update(p(x,y))$ [Dynamically update priority]
  else
    if not $M(x,y)$ **then**
      $r \leftarrow Calculatestipplesize(I_{origin}(x,y))$ [Equation 6]
      if $I(x,y) <= Th$ [Intensity below threshold] **then**
        $App \leftarrow 0$ [Black]
        $stippleslist.add(x,y,r)$ [Store the stipple position and the size]
      else
        $App \leftarrow 255$ [White]
      end if
      $e_{xy} \leftarrow I(x,y) - App$ [Calculate error]
      Errordiffusion$(x,y,e_{xy},r,G_-,G_+,k,D)$ [Procedure 2]
      $M(x,y) \leftarrow true$ [Processed this location]
    end if
  end if
end while

Procedure 2 Errordiffusion$(x,y,e_{xy},r,G_-,G_+,k,D)$ [x,y the position, e_{xy} the error, r the stipple size, D the mask size, G_-,G_+,k,D parameters for density control]

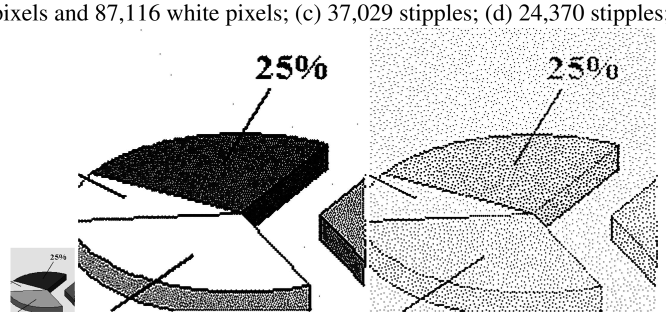
for all pixels $I(m,n)$ under the circular mask with the center (x,y) and the radius $D/2$ **do**
  if not $M(m,n)$ **then**
    $r_{mn} \leftarrow$ the distance from (m,n) to (x,y)
    $\hat{w}_{mn} \leftarrow CalculateWeight(I(m,n), r_{mn}, e_{xy})$ [Equation 5]
    $s_{xy} \leftarrow Shrinkorexaggerate(r, e_{xy}, G_-, G_+)$[Equation 7]
    $e_0 \leftarrow (A_{\text{stipple}} - A_{\text{pixel}}) * k$ [Correction]
    $I(m,n) \leftarrow I(m,n) + \hat{w}_{mn} * (e_{xy} + e_0) * s_{xy}$ [Update]
    $I(m,n) \leftarrow max(0, min(I(m,n), 255))$ [Clamping]
  end if
end for

Figure 4 gives a simple example for parameter choices. The notation $P(9,2,0,7)$ means $G_- = 9$, $G_+ = 2$, $k = 0$, and a $D = 7 \times 7$



(a) (b) CAH (c) $P(5,3,0,7)$ (d) $P(5,5,0,7)$ (e) $P(5,5,0.1,15)$ (f) $P(10,10,0.1,15)$

Figure 5: Our basic method: transition from CAH to stippling with stipple budget decreasing. (a) Original image; (b) CAH with 95,156 black pixels and 87,116 white pixels; (c) 37,029 stipples; (d) 24,370 stipples; (e) 16,980 stipples; (f) 3,085 stipples.



(a) (b) $P(9,2,0,7)$ (c) $P(2,9,0,7)$

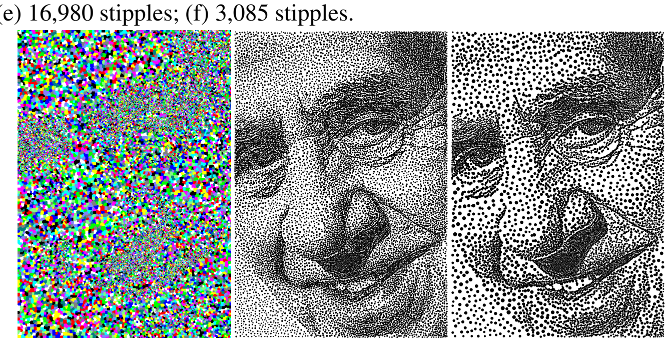
Figure 4: Parameter control. (a) Original image; (b) 12,705 stipples; (c) 6,133 stipples.

mask is used. Figure 4 (b) has larger G_- and smaller G_+ and (c) has smaller G_- and larger G_+; both draw the image with reduced stipple counts, but the former lightens the lighter areas while the latter lightens the darker areas. Usually, we suggest using $G_+ = G_-$ or G_+ a little smaller than G_-. Larger mask size D brings sparser distribution. The value for k usually is zero or very small.

Figure 5 shows further results of error adjustment on an image: the stipple count can be dramatically reduced without major impact on structure quality. With little shrinking or exaggeration of errors (small G_- and G_+), we have 37,029 stipples in (c) and, with further shrinking or exaggeration of errors (larger G_- and G_+), fewer stipples in (d) are used. In (e), we double the mask size and reduce this number to 16,980 while still being able to show the face wrinkles quite well. A further reduction is possible by very large values with very large mask size, shown in (f); at this point there are only 3,085 dots yet the face is still discernible. The use of k results in fewer stipples for the black shadow under the nose.

3.2 Tone Matching with Stipple Resizing

It is clear from inspecting Figure 5 that the tone can be improved by making more dramatic adjustments to stipple size. One option is a size calculation similar to that of Secord [18], computed in post-processing once stipple locations have been established. Secord set the size of each stipple based on input image darkness integrated over the stipple's Voronoi region. The result of our implementation of this process is shown in Figure 6, in which (a) shows the stipple regions and (b) gives the corresponding result with tone improved. Figure 6 (c) shows another result: large dots are used on the face, which is unattractive and invokes the connotation of blemishes and freckles. Although the tone has indeed been improved, there has been a detrimental impact on structure and visual appearance. This suggests that if the budget is very limited and we still want to preserve the key information, more dots should be used to support structure, not tone. If structure preservation is desired,



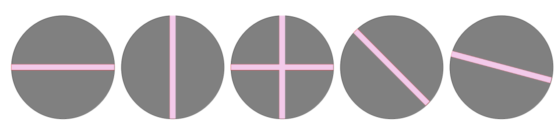
(a) $P(5,5,0,7)$ (b) $P(5,5,0,7)$ (c) $P(5,5,0.1,15)$

Figure 6: Our modified size strategy. (a) stipple influence areas; (b) result of (a); (c) another result. See also Figure 5 (d) and (e).

small stipples are better and hence tone loss is unavoidable under a very low budget.

4 Styles based on Structure-aware Stippling

We next turn our attention to explorations of variations including mask elements, mask shapes, priority configuration, and color to present diverse new styles for stippling such as stylization, irregular arrangements, stipple-based patterning, scratchboard, heightening, and painterly effects. The same parameters for density control are available for all of those styles, thus providing different levels of abstraction.



(a) (b) (c) (d) (e)

Figure 7: Exclusion-based Masks. (a) H-exclusion; (b) V-exclusion; (c) Cross-exclusion; (d) 45°-exclusion; (e) Edge-exclusion.

4.1 Stylization by Exclusion-based Masks

Our basic method uses a priority-based scheme. We apply a small modification to our basic method, applying an exclusion-based mask instead of the usual circular mask. Some possible exclusion-based masks are shown in Figure 7. For example, if a 45°-exclusion mask is used (Figure 7 (d)), the error will not propagate into the pixels along one diagonal. The priorities of neighboring pixels will generally be lowered after error diffusion [11]. However, the excluded diagonal pixels will not change priorities, and with their relatively higher priorities they are more likely to be chosen as stipples: diagonal patterns then form, as shown in Figure 8 (top). Fur-

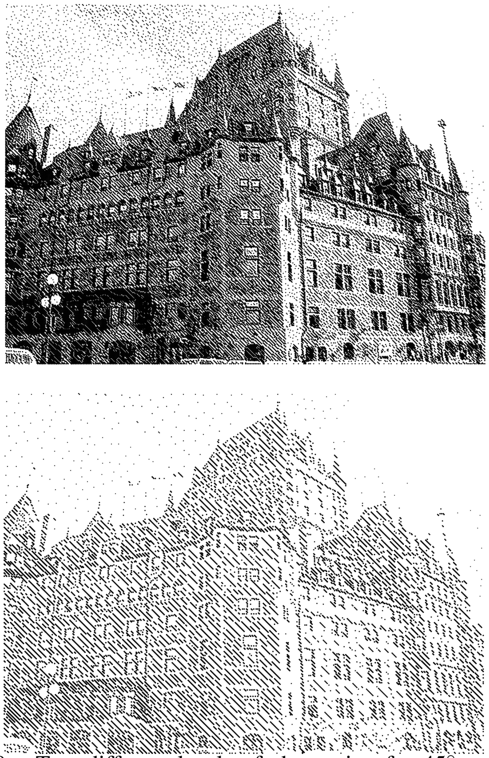


Figure 8: Two different levels of abstraction for 45°-exclusion. Top: $P(5,3,0,7)$; Bottom: $P(6,6,0,21)$.

ther abstraction with fewer stipples, combining with existing structure and patterns in the output image, is also shown in Figure 8 (bottom). Similarly, if an H-exclusion, V-exclusion, or cross-exclusion is used, the resulting image will contain respectively horizontal, vertical, or crossed patterns; see Figures 1 (b) and 9. Even small stipple counts can indicate both patterns and content clearly; it is very difficult for previous stippling methods to attain this.

Similarly, another possibility is to exclude a line of pixels oriented along the edge direction. Doing this further promotes structure details and textures by attracting stipples along edges. We exploit this combined with simple color variations for scratchboard effects: extended linear structures such as hair are well preserved with this approach. See Figure 1 (c) and 17. The quality from our edge-exclusion results will also be difficult to achieve with previous stippling methods because highly-textured images are a challenge.

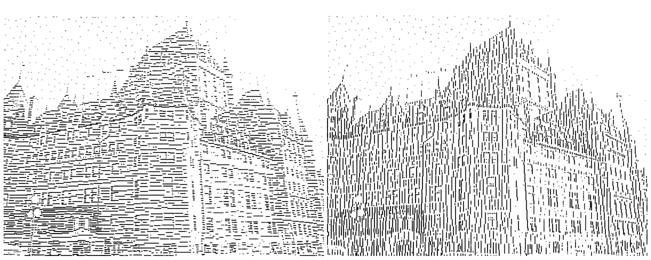


Figure 9: Left: H-exclusion; Right: V-exclusion.

4.2 Irregular Distribution with Irregular Masks

To break the excessive regularity of stipple distribution, we propose an Eden-like growth scheme [4] for irregular mask generation. Instead of using a circular mask, a fixed-size mask is generated by randomly adding 8-connected pixels to grow an irregular shape until the target pixel count is reached. Figure 10 demonstrates the evolution of a 26-pixel irregular mask. In each step, blue pixels are possibilities for the next selection. Our scheme is similar to the Eden growth model, randomly clustering on the boundary. The difference between circular masks and our irregular masks is shown in Figure 11 (the upper row). The stipple distribution on the face of the clock expresses tone in a less ordered way compared to the regular distribution of the circular mask. Importantly, the irregularity does not reduce our ability to preserve structure. Though the generation is time-consuming (it takes around 50 seconds to process the clock) the idea of varying shapes for masks might inspire another approach for hand-drawn stippling and also might give an initial clue to introduce stipple shapes into an error diffusion scheme. This exploration introduces another way for hand-drawn stippling and also shows the flexibility of our system.

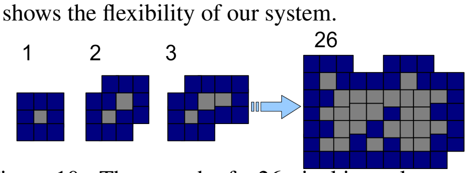


Figure 10: The growth of a 26-pixel irregular mask.

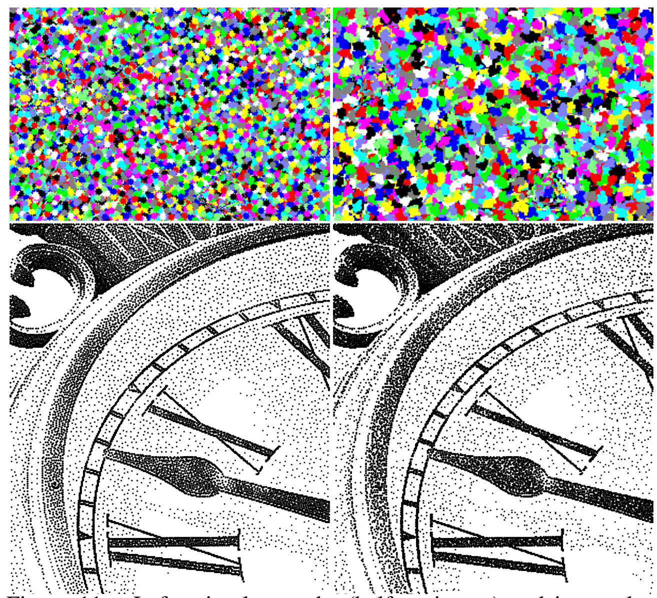
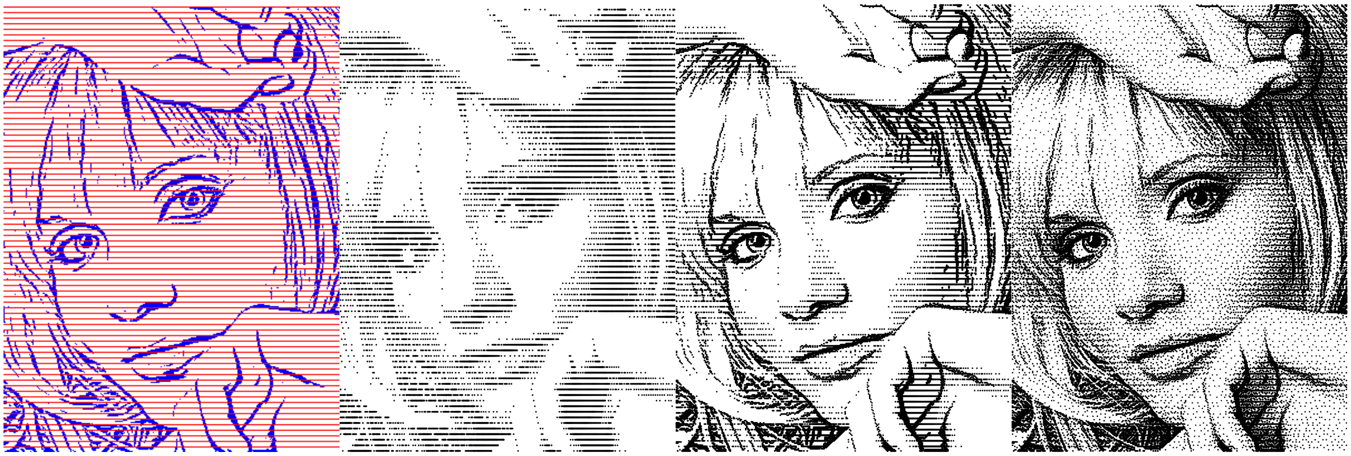


Figure 11: Left: circular masks (half an image) and its result; Right: irregular masks (half) and irregular distribution.

4.3 Multiple-Stage Stippling

We now present a variant of our priority scheme involving an extended priority configuration. Now, the pixels ($S = I$) will be processed in several stages: first, all pixels belonging to the first-stage priority set (S_1) will be processed; second, the pixels belonging to the second-stage priority set (S_2) will be processed; and so on. After the system processes all priority sets, it deals with any remaining pixels (S_N) in a final stage. Density control can be applied independently to each stage to give varied levels of emphasis. Regions of the image designated unimportant can be processed with more aggressive stipple reduction, while larger numbers of stipples can be dedicated to the important areas. We will have parameters $G_{1,-}$ and $G_{1,+}$ to control local density in the first phase; in general, we will write $P_j(G_{j,-}, G_{j,+}, k_j, D_j)$ for the jth stage. The separate parameters for different stages provide flexible density control over each stage. Users can promote the interesting stages with low parameters and deemphasize the uninteresting stages with higher values.

Broadly speaking, the priority set can be generated in two ways. We could generate it automatically, using low-level features such as edges or high-level features such as faces; or, we could enlist user assistance and create it manually. The minor drawback is that the outcome is dependent on the quality of the priority field, but it has the advantage of allowing us to use our stipples effectively and



(a) A priority map (b) $P_1(1,1,0,7)$ (c) $P_2(1,1,0,7)$ (d) $P_3(5,3,0,7)$

Figure 13: Screening from geometrical curves. (a) A map for priority configuration; (b) the first-stage result; (c) the second-stage result; (d) the final result.

stylistically. In the following, we suggest some specific ways of exploiting multiple priority levels.

(a) Dense ETF edges (b) Sparse ETF edges

Figure 12: Our ETF method for two densities. (a) $P_1(1,1,0,7)$ and $P_2(5,3,0,7)$; (b) $P_1(3,3,0,7)$ and $P_2(5,3,0,7)$.

Promotion for ETF Edges, called our ETF method: In a two-stage process, we use a first-stage priority set drawn from the edges using the flow-based difference of Gaussian for the edge tangent field (ETF) [7]. Figure 12 demonstrates two two-stage results for two different stipple distributions with ETF edges promoted. Both express the ETF edges clearly.

Promotion for Geometric Curves: Here we demonstrate a three-stage process in Figure 13. The colored map with different stages for priority configuration is shown in (a). The first priority set (S_1 in red) is a geometric arrangement of lines; the second set (S_2 in blue) is the collection of ETF edges; and the third set (S_3 in white). Figure 13 (b) is an intermediate result for the first stage, where the first-stage pixels have been processed based on tone and structure. The second-stage result is shown in (c). Figure 13 (d) is the final result with promotion for both horizontal lines and ETF edges. Another example with a different priority map is shown in Figure 1 (d).

Promotion for Textural Edges: An alternative is to take the priority map from an input image, such as a texture. We use a three-stage process as in the previous section. Different examples with different textures are shown in Figure 1 (e) and Figure 19.

4.4 Three Minor Styles with Color Variations

We here briefly discuss three additional artistic styles: *scratchboard*, *heightening*, and *painterly* effects. **Scratchboard** is a technique to generate highly detailed, textured artwork by scratching a thin black layer to uncover inside white clay. Our edge-exclusion-based method is very good for structural images, common in the artistic medium of scratchboard. Results appear in Figure 1 (c) and 17; highly-detailed hair textures for both the lion and the old man are clearly preserved by white stipples. **Heightening** is a striking effect to raise contrast: most of the image is drawn with dark shades on a neutral background, and a very few regions are drawn in light shades (for example, using white chalk). DeCarlo and Rusinkiewicz [2] demonstrated heightening with drawing style. The majority of stipples are black, providing tonal and structural components, with (say) 4% of the stipples to be drawn in white. In Figure 18: the heightening gives some details a shinier appearance. Also intrigued by Pointillism, we color our dots instead of using only black or white; Figure 18 shows the resulting **painterly effect**.

5 Results and Discussion

Here, we compare and analyze stippling methods to help evaluate our approach. Our results are based on our basic method with $P(5,3,0,7)$ unless otherwise stated. Some original images are shown in Figure 14. Figure 15 shows analysis for tone. Our inten-

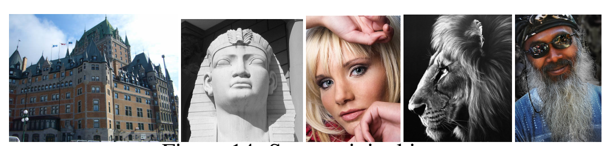
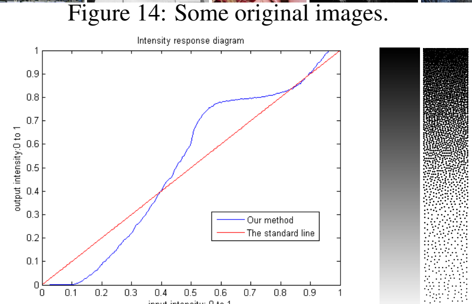


Figure 14: Some original images.



(a) (b) (c)

Figure 15: Tone analysis: (a) Our intensity response diagram; (b) original ramp; (c) Our ramp.

Image	baby	balloon	clock	kid	man	w1	w2	w3	w4
Our basic method	0.29	0.52	0.52	0.58	0.42	0.33	0.59	0.55	0.15
Our ETF method	0.30	0.55	0.55	0.61	0.44	0.36	0.63	0.56	0.15
Stipple resizing	0.05	0.44	0.30	0.20	0.10	0.12	0.30	0.17	0.08
Kim et al.'s	0.33	0.55	0.46	0.33	0.38	0.17	0.49	0.32	0.18
Secord's	0.13	0.52	0.32	0.20	0.12	0.10	0.34	0.29	0.07
Martin et al.'s	0.07	0.45	0.19	0.07	0.06	0.07	0.18	0.07	0.08

Table 1: Structure similarity; 'w' is short for 'woman'.

sity response shows continuous contrast enhancement: light input tones have even lighter output, while darker ones are darkened even further. This is a necessary property given our structure-based objectives. Visually, the effect is quite good in Figure 15 (c).

Our main concern is not tone matching, however, but structure preservation. Figure 3 shows our method preserves structure better than previous tone-focused stippling methods. Here, Figure 16 gives a comparison against the results from Mould [13], who shares our goal of using stipples for structure. We are able to achieve significantly better tone reproduction and better structure than Mould's method, both with high and very low budget of stipples, for an overall much better effect. We employ the structural similarity index

Figure 16: Comparison with Mould's method. Top left: Mould's method, approx. 3,000 stipples; Top right: our method, 3,271 stipples, $P(10,10,0,17)$; Lower left: Mould's method, approx. 11,000 stipples; lower right: our method, 10,550 stipples, $P(6,6,0,17)$.

measure (SSIM) [21] to quantify the structure difference between the stippling result and the original grayscale image. Table 1 gives the data for structure similarity; tone similarity by peak signal-to-noise ratio is shown in Table 2. Higher values indicate higher similarity. To be a fair comparison, all test images are from Kim et al.'s paper [8] and all results are output as vector graphics and converted into raster images (using Inkscape) at the original image resolution. Secord's output uses the same number of stipples as we do. Thanks to our structure awareness, both our basic method ($P(5,3,0,7)$) and our ETF method ($P_1(0,0,0,7)$ and $P_2(5,3,0,7)$) have higher SSIM values than other previous methods, thus higher structure similarity. We also have better tone matching than others, apart from Secord's method which was carefully honed to match tone exactly.

Most of the effects in Section 4 are new for stippling. Due to our structure awareness, priority-based scheme, and flexibility in style

Image	baby	balloon	clock	kid	man	w1	w2	w3	w4
Our basic method	12.4	17.7	14.0	13.7	13.7	13.0	15.1	16.8	14.3
Our ETF method	13.1	19.0	15.4	15.2	14.6	14.4	16.5	17.4	14.5
Stipple resizing	21.4	19.0	21.3	22.5	24.2	21.4	23.4	20.8	18.3
Kim et al.'s	13.0	10.8	13.3	16.3	12.8	10.9	14.3	13.8	8.7
Secord's	27.4	21.3	23.8	29.5	28.4	23.8	25.7	21.4	18.7
Martin et al.'s	8.0	12.9	9.2	8.6	8.2	8.2	9.9	9.9	9.7

Table 2: Tone similarity.

transition, the diverse effects still present the content details nicely and are gracefully unified with other style elements.

This method is an evolved error diffusion scheme whose computation cost depends on the image size and the mask size. It ensures an upper bound on computational complexity and hence enjoys high efficiency and fast speed. Based on an Intel Core Duo CPU E8400@ 3.0GHz with 3GB RAM, most of our results are produced in only a few seconds to tens of seconds. The exception is the irregular masks, which take nearly a minute. This is a distinct advantage over previous stippling methods, which are usually slower. Secord [18] reported that drawing up to 40,000 stipples takes 20 minutes (albeit on much older hardware). Kim et al. [8] required several minutes to place 8,000 to 12,000 stipples. Mould's and Martin et al.'s methods [12, 13] are a little faster, but cannot achieve the same quality of structure and tone similarity. One limitation of our method is that density control is indirect, through adjusting nonlinear parameters. In summary, our new stippling method provides very good structure and also suggests the original image tone while attaining high processing speed.

6 Conclusions

In this paper, we presented a new technique for structure-preserving stippling by a priority-based scheme. We proposed a nonlinear error adjustment function to reduce stipple counts. Thanks to the high quality of structure preservation and flexible priority-based scheme, our diverse evolved styles present image content clearly. Our method might provide an initial distribution for other primitive placement algorithms in non-photorealistic rendering.

Acknowledgements

We'd like to thank our anonymous reviewers for many insightful comments. We also want to thank artists from Flickr.com. This work was supported by grants from both Carleton University and NSERC.

References

[1] J. Chang, B. Alain, and V. Ostromoukhov. Structure-aware error diffusion. *ACM Trans. Graph.*, 28:162:1–162:8, December 2009.

[2] D. DeCarlo and S. Rusinkiewicz. Highlight lines for conveying shape. In *Proceedings of the 5th international symposium on Non-photorealistic animation and rendering*, NPAR '07, pages 63–70, New York, NY, USA, 2007. ACM.

[3] O. Deussen, S. Hiller, C. van Overveld, and T. Strothotte. Floating points: A method for computing stipple drawings. *Computer Graphics Forum*, 19:40–51, 2000.

[4] M. Eden. A two-dimensional growth process. In *Proceedings of Fourth Berkeley Symposium on Mathematics, Statistics, and Probability*, volume 4, pages 223–239, 1961.

[5] S. Hiller, H. Hellwig, and O. Deussen. Beyond stippling – methods for distributing objects on the plane. *Computer Graphics Forum*, 22(3):515–522, 2003.

[6] S. Jang and H.-K. Hong. Stippling technique based on color analysis. In *PCM (2)*, pages 782–793, 2005.

[7] H. Kang, S. Lee, and C. K. Chui. Coherent line drawing. In *ACM Symposium on Non-Photorealistic Animation and Rendering (NPAR)*, pages 43–50, Aug. 2007.

[8] D. Kim, M. Son, Y. Lee, H. Kang, and S. Lee. Feature-guided image stippling. *Computer Graphics Forum*, 27(4):1209–1216, 2008.

[9] S. Y. Kim, R. Maciejewski, T. Isenberg, W. M. Andrews, W. Chen, M. C. Sousa, and D. S. Ebert. Stippling by example. In *Proceedings of the 7th International Symposium on Non-Photorealistic Animation*

Figure 18: Heightening and Painterly effects.

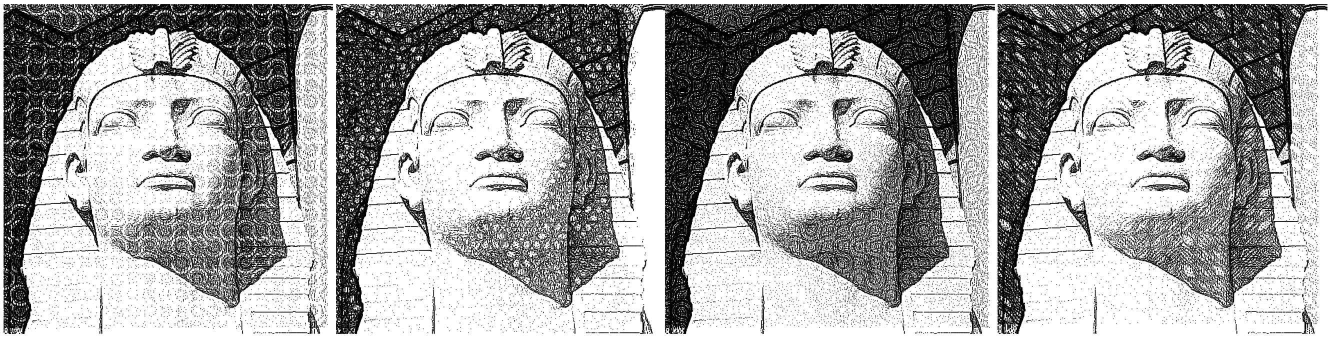


Figure 19: Screenings from texture input.

and Rendering, NPAR '09, pages 41–50, New York, NY, USA, 2009. ACM.

[10] J. Kopf, D. Cohen-Or, O. Deussen, and D. Lischinski. Recursive Wang tiles for real-time blue noise. *ACM Transactions on Graphics (SIGGRAPH 2006 issue)*, 25(3):509–518, 2006.

[11] H. Li and D. Mould. Contrast-aware halftoning. *Computer Graphics Forum*, 29(2):273–280, 2010.

[12] D. Martin, G. Arroyo, M. V. Luzon, and T. Isenberg. Example-based stippling using a scale-dependent grayscale process. *In Proceedings of the 8th international symposium on Non-photorealistic animation and rendering (NPAR)*, pages 51–61, 2010.

[13] D. Mould. Stipple placement using distance in a weighted graph. In *Computational Aesthetics*, pages 45–52, 2007.

[14] V. Ostromoukhov and R. D. Hersch. Multi-color and artistic dithering. In *Proceedings of the 26th annual conference on Computer graphics and interactive techniques*, SIGGRAPH '99, pages 425–432, New York, NY, USA, 1999. ACM Press/Addison-Wesley Publishing Co.

[15] W.-M. Pang, Y. Qu, T.-T. Wong, D. Cohen-Or, and P.-A. Heng. Structure-aware halftoning. *ACM Transactions on Graphics (SIGGRAPH 2008 issue)*, 27(3):89:1–89:8, 2008.

[16] T. Scheermesser and O. Bryngdahl. Control of texture in image halftoning. *J. Opt. Soc. Am. A*, 13:1645–1652, 1996.

[17] S. Schlechtweg, T. Germer, and T. Strothotte. Renderbots–multi-agent systems for direct image generation. *Computer Graphics Forum*, 24(2):137–148, 2005.

[18] A. Secord. Weighted Voronoi stippling. In *Proceedings of the 2nd international symposium on Non-photorealistic animation and rendering (NPAR)*, pages 37–43, 2002.

[19] L. Streit and J. Buchanan. Importance driven halftoning. *Computer Graphics Forum*, 17(3):207–217, 1998.

[20] D. Vanderhaeghe, P. Barla, J. Thollot, and F. Sillion. Dynamic point distribution for stroke-based rendering. In *Rendering Techniques 2007 (Proceedings of the Eurographics Symposium on Rendering)*, pages 139–146, 2007.

[21] Z. Wang, A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli. Image quality assessment: from error visibility to structural similarity. *Image Processing, IEEE Transactions on*, 13(4):600–612, 2004.

[22] S. Weissbach and O. Bryngdahl. Control of halftone texture by error diffusion. *Optics Communications*, 103(3-4):174 – 180, 1993.

[23] C.-K. Yang and H.-L. Yang. Realization of Seurat's pointillism via non-photorealistic rendering. *Vis. Comput.*, 24(5):303–322, 2008.

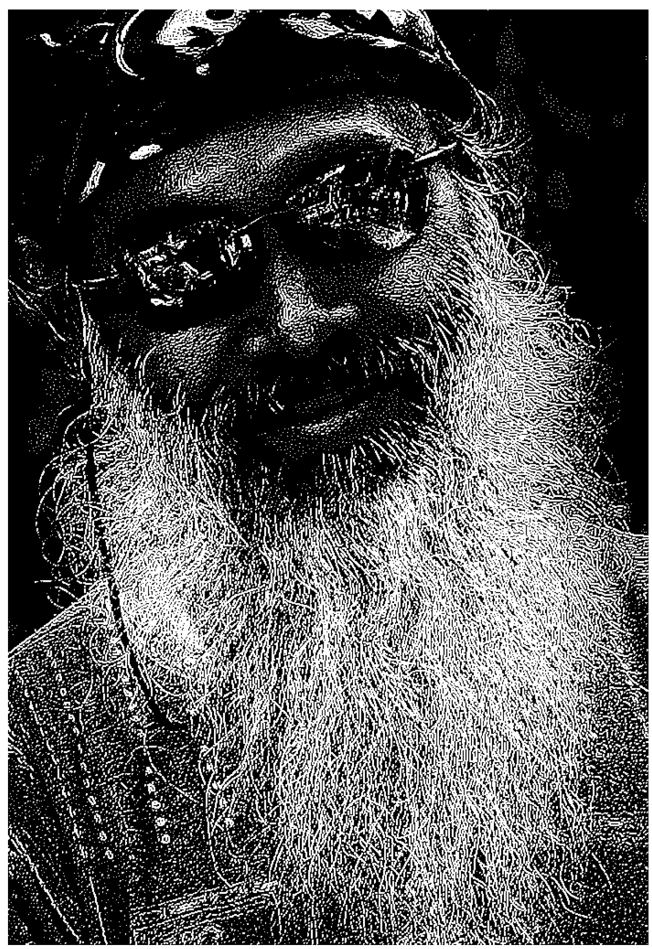


Figure 17: A scratchboard effect for an old man.