

Stipple Removal in Extreme-tone Regions

Rosa Azami, Lars Doyle^{ID}, and David Mould

Carleton University, Ottawa, Canada

Abstract

Conventional tone-preserving stippling struggles with extreme-tone regions. Dark regions require immense quantities of stipples, while light regions become littered with stipples that are distracting and, because of their low density, cannot communicate any image features that may be present.

We propose a method to address these problems, augmenting existing stippling methods. We will cover dark regions with solid polygons rather than stipples; in light areas, we both preprocess the image to prevent stipple placement in the very lightest areas and postprocess the stipple distribution to remove stipples that contribute little to the image structure. Our modified stipple images have better visual quality than the originals despite using fewer stipples.

Keywords: Non-photorealistic rendering, Image stylization, Stippling, Stroke-based rendering

CCS Concepts

• Computing methodologies → Non-photorealistic rendering;

1. Introduction

Stippling uses small dots or circles to represent the image content. Stippling and other illustrative visualization techniques such as hatching have been widely used in artistic and scientific applications.

In computer graphics and non-photorealistic rendering (NPR), stippling techniques have focused on optimizing the point distribution and the point size [DHVS00] or tone preservation [CAO09]. Algorithmic stipple placement has tried to maintain some characteristics of handmade stippling images such as regularity and density of the stipples, yet various problems are not solved for stippling.

Stippling methods require enormous numbers of stipples to represent the black areas of an image. An image with extensive dark regions requires vast quantities of stipples and consumes excessive memory in storing them. We will cover each dark region with a polygon instead; doing so does not compromise the stippled appearance, since individual stipples in the dark regions were not visible anyway, and indeed can improve the image: the risk of spurious gaps is eliminated. Dark regions are determined by thresholding, and to eliminate the resulting discontinuities, we employ a Poisson solver to reconstruct a smooth image.

Light areas pose a different problem: the few stipples placed there distract rather than inform the viewer, and as noted by Deussen and Isenberg [DI13], removing stipples placed on white areas produces an image similar to what artists create. This observation had been informally used earlier, where authors such as

Second [Sec02] thresholded away their light background. We apply thresholding, and also supply a postprocessing method that systematically removes additional stipples from light areas when they seem not to contribute to image structure.

We apply our method in conjunction with two existing stippling algorithms: structure-preserving stippling [LM11] and a recent Voronoi-based stippling method called Linde–Buzo–Gray stippling [DSZ17]. The resulting images show marked visual improvement despite the use of fewer stipples. Our main contributions are as follows:

- Illustrating dark regions: We identify the dark regions of the image and cover them with solid polygons instead of stipples. To eliminate discontinuities, we smoothly reconstruct the image in a small area surrounding each dark region.
- Controlling stipple placement in light regions: We eliminate unnecessary stipples on light regions, both through preprocessing to prevent stipple placement in very light areas and through postprocessing to identify and remove such stipples.

Further, we suggest a method to preprocess the image so as to better show small details, measuring local contrast and increasing it for selected elements. Our approach is not necessarily suitable for all images, but shows some improvement in specific instances.

The remainder of our paper is structured as follows. We first review related work in Section 2. Next, in Section 3, we explain our method for dark and light region stippling and an optional preprocessing stage for detail enhancement. We show our results and discuss our findings in Section 4. Finally, we conclude the paper in Section 5.

2. Previous Work

Image stylization techniques are highly varied, and include filter-based methods [SLKD16; DS02], stroke-based techniques [Her98], and most recently, style transfer methods based on deep learning [KNBH12; GEB16]. Here, we concentrate our discussion on stippling methods. The recent survey by Martin et al. [MAAI17] provides additional detail on the available stippling techniques.

Voronoi stippling. Most early stippling methods in computer graphics focussed on achieving an even distribution of points, free of visual artifacts. One method of achieving this outcome is by means of Lloyd’s algorithm [Llo82]. Starting from an initial distribution of generating points, an iterative method computes its Voronoi diagram. A relaxation step then moves each point to the centroid of its Voronoi region. This process continues until the algorithm converges. Since the outcome of this process naturally results in a uniform distribution over the image plane, other factors must be introduced in order to preserve image structure.

Deussen et al. [DHVS00] introduced an interactive stipple editor that used brushes to locally relax and control the number of stipples in local windows. They used manual segmentation to partition the image plane and prevent stipples from migrating across important edges. A fully automatic method was later introduced by Secord [Sec02], who used the local tone of the input image to weight each pixel while computing the centroid of a Voronoi cell. As a consequence, dark regions would attract more stipples, preserving relative tone. Recently, Deussen et al. [DSZ17] proposed a method based on the Linde–Buzo–Gray algorithm. Starting from an initial point distribution, its centroidal Voronoi diagram is refined iteratively by means of hysteresis thresholding: at each iteration, the density of grey values contained in a Voronoi cell determines if its generating point is removed, split or retained. This method represents details better than Secord’s algorithm while converging faster.

Structure-aware stippling. In the methods discussed so far, the intent has been to reproduce the average grey-scale values of the input image while preventing unintended visual artifacts, such as dot chains, from occurring. A byproduct of this goal is an overly smooth image that poorly represents image detail. Structure-aware image reproduction [PQW*08; CAO09] shifts the emphasis away from tone reproduction towards detail and texture representation. Mould [Mou07] proposed a weighted-graph algorithm that favours stipple placement at high-gradient locations. Using a shortest-path algorithm, the image plane is explored while placing stipples at the frontier whenever a distance threshold is exceeded. With a similar objective, Schlechtweg et al. [SGS05] introduced a multi-agent system that attracts *StippleBots* to edges, therefore emphasizing image structure. Li and Mould [LM11] drew on their previous work on error-diffusion based halftoning [LM10] to emphasize contrast and structure in the stippled result. Pixels are processed in a priority order that emphasizes extreme dark or light pixels. When a stipple is placed at a location, then the resulting error, with respect to tone, is distributed amongst neighbouring locations. By varying the distribution pattern, different stylistic effects are achieved.

Directional stippling is based on traditional hedcut drawings,

where stipples are placed along structured lines which follow abstracted orientation guides. Kim et al. [KSL*08] extracted feature lines from the input image. From these lines, a set of laneways are produced by sampling the resulting distance transform at regular intervals. A constrained Lloyd’s algorithm is then used to produce an even stipple distribution within each lane. Son et al. [SLKL11] achieved a more regular result by using a structure grid for stipple placement. This extension additionally aligns stipples in the direction perpendicular to image features. First, they smoothly deformed a 2D grid to align with local feature directions. Then stipples are placed at the grid intersections. Lines are placed on the grid edges for added effect.

Example-based stippling. Other authors [KMI*09; MALI10; MdSRI15] have focused on aspects of hand-drawn stipple art in order to achieve a more naturalistic look. These aspects include: stipple distributions, dot shape, and grey-scale characteristics (as opposed to binary primitives). Other higher-level artistic processes are considered as well [MALI10], but current NPR algorithms are not able to automate them. Kim et al. [KMI*09] modeled stipple distributions from individual artists, then synthesize new distributions with similar statistical properties using tones maps to guide the stipple density. Martin et al. [MdSRI15] discussed the grey-scale properties of hand-drawn stipples; the interaction between ink and paper, in hand-drawn images, produces many subtle grey-scale tones that are not seen in single-color primitives. This area of research is limited by reproduction technology; screens are typically too low-resolution to reproduce small-scale gradient effects, while printers, though usually higher-resolution, must reproduce stipple grey-scales through halftoning—itsself another binary process.

3. Method

This paper addresses two main issues. First, traditional stippling methods match the tone of dark areas by filling them with huge numbers of stipples; we propose instead covering each dark area with a polygon. Doing this in a naïve fashion would produce visible discontinuities; we reconstruct a small region surrounding each dark region so as to produce a smooth transition. Second, tone-based stippling can produce isolated stipples in light regions; such stipples are distracting and may not represent any specific image content. We propose a postprocessing step that identifies isolated stipples and removes them from the image. The stages of our algorithm are illustrated in Figure 1.

First, we threshold the image to identify the dark regions. We will represent these regions by polygons rather than filling them with numerous stipples. Similarly, we threshold the image’s high intensities to white so as to avoid the necessity of putting stipples there; in such regions, stipples will be infrequent and may appear spurious to a viewer. Figure 1 (a) shows an input image, and (b) shows the dark regions obtained by thresholding. In all cases, when thresholding yields a connected component that is too small (in practice, with an area smaller than 20 pixels) we ignore this region and use the original image values in subsequent stippling. Larger regions are processed as follows.

At cutoff thresholds, the transitions between the black (or white)



Figure 1: Steps of our method: a) original image; b) threshold mask; c) reconstructed image after Poisson blending; d) results from filling polygons; e) final result after stipple removal; f) close-ups of original stippling and final result of the polygon filling followed by stipple removal steps.

regions and the nearby areas are discontinuous. In order to obtain smooth transitions, we reconstruct a narrow boundary zone by solving a Poisson equation. When the reconstructed image is passed to the stippling algorithm, the thresholded pixels are labeled as “done”, thus preventing stipples from being placed there. Figure 1 (c) illustrates the reconstructed gray image and (d) the result of applying stippling to the reconstructed image.

Once the stippling is complete, we remove additional potentially spurious stipples appearing everywhere in the image. We remove stipples through two post-processing steps. In the first step, we remove the most isolated stipples: those that have at most one other stipple nearby. In the second step, we remove additional stipples deemed not to contribute to the image content, identified as follows. We compare two quantities: pixel intensity of the target stipple, and the average intensity of the original image in a range surrounding the stipple location. If the difference between the quantities is large we keep the stipple. If the difference between the quantities is less than a threshold, we do further analysis, basing our decision on the number of stipples nearby: when there are more than five stipples within a given distance, the stipple is considered to be part of the image structure and hence kept, whereas if there are fewer, the stipple is removed. Figure 1 (e) illustrates the final result after stipple removal and (f) shows a close-up of the original stippling on the left and the our final result of the polygon filling and stipple removal steps.

Tone-preserving stippling can straightforwardly show image details in darker regions of an image: there are plenty of stipples, and small variations in their distribution are visible. In lighter regions, however, showing details is more difficult. The requirement of tone preservation enforces a low stipple count. The cat’s white whiskers against a bright background, for example, may no longer be visible after stippling. Although this is a natural outcome of tone-preserving stippling, we would still like to show these details. We propose an optional preprocessing step to identify and amplify these features in the input image, darkening light details when doing so increases the local contrast. Details of this process are given in Section 3.3.

3.1. Illustrating dark regions

We will avoid placing stipples in dark regions and instead cover each region with a polygon. First, we threshold all pixels $I(x) \leq T_{\text{low}}$ to 0. We replace these thresholded regions with solid polygons. To avoid discontinuities between the thresholded regions and their surroundings, we reconstruct a transition region between the thresholded pixels and the unaltered image regions. For reconstruction, we consider an upper threshold T_{high} ; we can then create a transition region Ω , which we define as the set of pixels such that $T_{\text{low}} < I(x) \leq T_{\text{high}}$.

We reconstruct all transition regions by solving a Poisson equa-

tion over the image [PGB03]:

$$\Delta I = \text{div} \mathbf{G}, \quad (1)$$

with boundary conditions given by the pixels adjacent to the region; note that the pixel values are taken post-threshold. The Poisson equation uses the image gradient vector field \mathbf{G} . We stretch \mathbf{G} so as to account for the greater range of intensities the transition region needs to cover: \mathbf{G} is scaled by $(a+b)/b$, where $a = T_{\text{low}}$ and $b = T_{\text{high}} - T_{\text{low}}$. Figure 2 shows the pixel intensities near a dark region before and after the reconstruction; the discontinuity produced by thresholding has been eliminated by the reconstruction process. On the figure's right, we can see the effect on a sample image region. Thresholding without reconstruction produces a visible polygon boundary. With reconstruction, no distinct boundary is visible: the stipple density gradually eases as we move away from the polygon. Note that the same polygon is used in both cases. With reconstruction, the fringe around the polygon is darker; this is not usually noticeable without close comparison with the original, but increases the number of stipples in the output, slightly offsetting the gains from eliminating the stipples in the polygon.

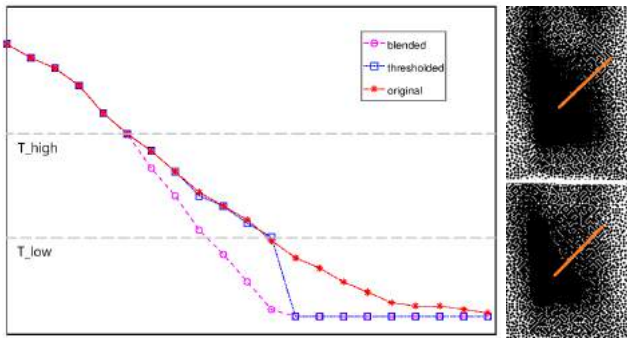


Figure 2: Transition from a dark region to its surroundings. The graph shows the original intensity profile (star), the thresholded values (square), and the results of blending (circle). Top image on the right shows a smooth transition of stipples and the bottom image is result of a hard thresholding.

To obtain a polygon approximating a region, we find the pixels making up the exterior contours and treat each as a vertex; the sequence of non-redundant vertices is the polygon. Figure 3 shows the thresholded regions followed by the resulting stippled image in which the dark regions are shown with solid black polygons.

The blended grayscale image is the input to the stippling algorithm. The pixels in thresholded regions are labeled as “done” so that the stippling algorithm will not place any stipples in these regions. The smooth gradient arising from the Poisson solution guarantees a continuous intensity transition from the thresholded areas towards their surroundings. In Figure 4, we show polygons in different colors, both emphasizing polygon shape and illustrating the smooth transition between polygon boundaries and the surrounding stipples.

Akin to our treatment of the the dark areas, we also threshold the bright image regions. The effect here will be to prevent stipple placement in areas that are close to white, since they will now



Figure 3: Left: thresholded regions; right: stippled image with dark regions covered by polygons.

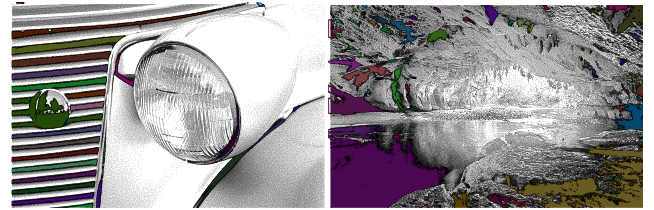


Figure 4: Two examples of dark region polygons illustrated in different colors.

be thresholded to maximum brightness. Note that the light and dark thresholding and reconstruction can be done separately, since their transition regions, consisting of pixels near in intensity to the thresholds, can never overlap.

3.2. Stipple removal in light regions

Isolated stipples contribute little to image content and can distract the viewer. Figure 5 illustrates examples of unnecessary stipples. We remove such stipples in two phases. First, we discard the stipples that have at most one other stipple within a radius r ; these low-density, non-clustered stipples do not indicate any image detail. We run the first step over all stipples. Then, we apply a second round of stipple removal, only considering stipples whose pixel intensities exceed a threshold, say L ; we use $L = 128$ in practice.

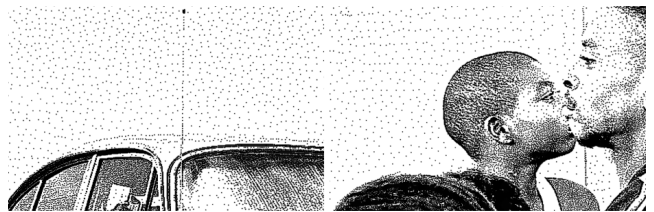


Figure 5: Selected details of the car and New Jersey images. Note the unnecessary stipples in the sky above the car and in the background behind the kid.

Our removal test involves checking two quantities: pixel intensity at the target stipple, and the average intensity of the original

image in a range surrounding the stipple location. The intensity of the pixel where the target stipple is centred is given by I_t and the average intensity of the stipple's neighborhood is I_n . Let $q = |I_n - I_t|$. We decide whether or not to remove the target stipple based on the relationship between q and the threshold T_a :

- a) If $q \geq T_a$, we keep the stipple.
- b) If $T_a > q$, we decide instead based on the number of nearby stipples. The stipple will be removed when the number of nearby stipples is less than some constant c .

We used the threshold $T_a = 25$. We set the search radius $r = 10$ and the stipple count threshold $c = 5$. Parameters c and r together yield a measure of local stipple density and should not be adjusted independently, but note that we cannot simply merge them since we require a parameter for the region size over which count (or density) should be evaluated. An example of the stipple removal process is given in Figure 6.



Figure 6: Stipple removal process. From left to right: result before stipple removal; stipples with 0 or 1 neighbours removed; final result after removing stipples with fewer than c neighbours.

3.3. Detail enhancement

Low-contrast features in bright regions of an image are lost under stippling with faithful tone reproduction: there will simply not be enough stipples to indicate any contrast. We suggest judiciously darkening such features in a preprocessing stage such that the subsequent stippling method will place stipples there.

We apply an edge-preserving filtering algorithm [Mou13] and compute the residual. Regions where the residual is positive correspond to features brighter than their surroundings. Such regions are candidates for darkening; to darken an area, we will add its reversed residual to the original image, akin to unsharp masking. Note that only regions with positive residuals are considered, so that only darkening can occur, not lightening.

We cannot darken all positive-residual regions indiscriminately, however. In textured areas, for example, darkening the bright parts and leaving the rest alone will ruin the image detail. Instead, we test whether reversing the residual will increase or decrease contrast; if contrast is increased with reversal, the reversal is beneficial and we accept it.

The test operates as follows. We calculate two average RMS contrasts [Pel90] for each group of connected components within a neighbourhood: one with the original values and the other with positive residuals reversed. If the original contrast is greater, we do nothing, otherwise we reverse the positive residuals in that area.

Figure 7 shows an example image and the corresponding positive residuals. When using this process, some windows of the building are made clearer. The drawback is that the original image is no longer precisely reproduced.

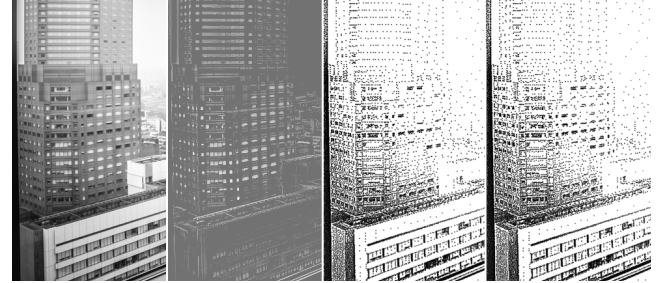


Figure 7: Effect of detail enhancement. From left to right: original gray image; positive residuals; original structure-aware stippling; stippling after detail enhancement.

4. Results and Discussion

In this section we show results from our method and discuss the advantages and limitations of the approach. Most results from SPS [LM11] use the following parameters: $G_+ = G_- = 5$; mask with radius $D/2 = 10$. For most images, we set $T_{\text{low}} = 10$ and $T_{\text{high}} = 50$, with a symmetric set of thresholds to deal with light regions, an upper threshold of 245 and a lower threshold of 205. We used different thresholds for the *parrot* and *headlight* images, where $T_{\text{low}} = 20$ and the threshold for bright areas was 235. In general, thresholding operations are sensitive to the choice of threshold, so the user may wish to select different values for a particular input image. We also show the effect of our method on LBG stippling. We set the minimum and maximum stipple size to 2 and 8, respectively, to get comparable contrast to the input image. In the following section, we first discuss some results from structure-preserving stippling, then a few results from LBG stippling.

4.1. Visual evaluation

Structure-preserving stippling. Our main contribution involved representing the darkest areas of an image with solid polygons rather than filling them with stipples. We use polygons for the dark background behind the girl in *toque* in Figure 8, black stripes in *headlight* and the dark woods behind *parrot*; these results are shown in Figure 9. The visual effect is quite good: the polygons are not intrusive and can occasionally be improvements over the stippled black areas, e.g., in the rare case where an inopportune arrangement of stipples leaves a tiny gap. The transition from the polygon exterior to the rest of the image appears natural, preserving the surrounding detail and edge placements. The top detail panel in the first row of Figure 8 shows sample transitions.

Stipples have been eliminated from the brightest regions of the image through a similar thresholding process. Few stipples existed in such regions in the first place, and the transition away from such regions also exhibits few stipples, so the effect is not very dramatic.



Figure 8: Smooth transitions and pattern preservation. Left column: original stipplings; middle: our results; right: details, in which we see smooth transitions from dark to light regions and preservation of facial features in *toque*, and the kid's face and tattoo in *New Jersey*.

Nonetheless, removing these few stipples can improve the image; the sky seen through the big window of the *city* image in Figure 10 provides an example, indicated by the blue circle in the middle. Removing distracting stipples in light areas also reduced the number of stipples; however, more importantly, it improves the clarity of the image. The girl's face in *toque* has been largely cleared, eliminating distractions. The stipples that had been sparsely distributed over the top of the *headlight* represented neither gradients nor structure, so removing them made the image clearer.

Our method can reveal patterns and preserve some thin structures that were hidden under the original stippling. For example, in Figure 8, elements in the *New Jersey* image have become more prominent, including the kid's face, the tattoo on the man's arm, and the text on the window.

We partially preserved low-intensity but structural features by reversing residuals. In Figure 10, some of the small windows on the high building of the *city* and in *cat* the whiskers on the left and some details in the fur have been darkened; these were light details in light regions and would not normally be feasible to show with dark stipples. The two circles indicate some places where differences between the original and enhanced images appear; the reader may wish to zoom in for a better comparison. Note that the whiskers on the right were not darkened since doing so would have reduced the contrast of the region. Similarly, not all windows of the *city* were darkened.

Linde-Buzo-Gray stippling. Figure 11 shows LBG stippling before and after applying our algorithm. The black polygons cover the dark chair backs in *theater* and water in the cave in the *Oparara* image. Filling the gaps between stipples in the dark regions has in-

creased the image contrast and improved the appearance. We also removed spurious stipples from the theater screen. Intermediate regions like the carpet remained unchanged.

4.2. Performance

Tables 1 and 2 show the number of stipples for selected images before and after applying our algorithm on the SPS [LM11] and LBG [DSZ17] methods, respectively. Images with large dark areas experienced significant reductions in stipple count. For example, the *old man* image generated almost 82k stipples under the original structure-preserving stippling method; this number decreased to 26k after applying our stipple removal method. The tables also show the number of polygons; a small number of polygons were added relative to the number of stipples eliminated.

All our results were generated on a laptop with an Intel Core i7-4510U CPU 2.0 GHz. Reconstructing the blended gray image takes roughly 0.9 sec for an image of size 1600×1200 pixels. Stippling process takes almost one second longer with the stipple removal steps to produce 107K stipples. Timing directly depends on the size of the image and the number of stipples. Including our method in the stippling creation increases the overall time by no more than a few seconds, a small fraction of the overall cost for most stippling methods. We see a significant reduction in size of the vector graphics files of many images; for example, the scalable vector graphics file for the *old man* declined from 3.6MB to 1.2MB based on SPS method and the *toque* file size went from 6.25MB to 4.9MB. However, the darkened fringe around black polygons requires more stipples than the same region in the original image; for sufficiently small polygons, this effect outweighs the savings from eliminating

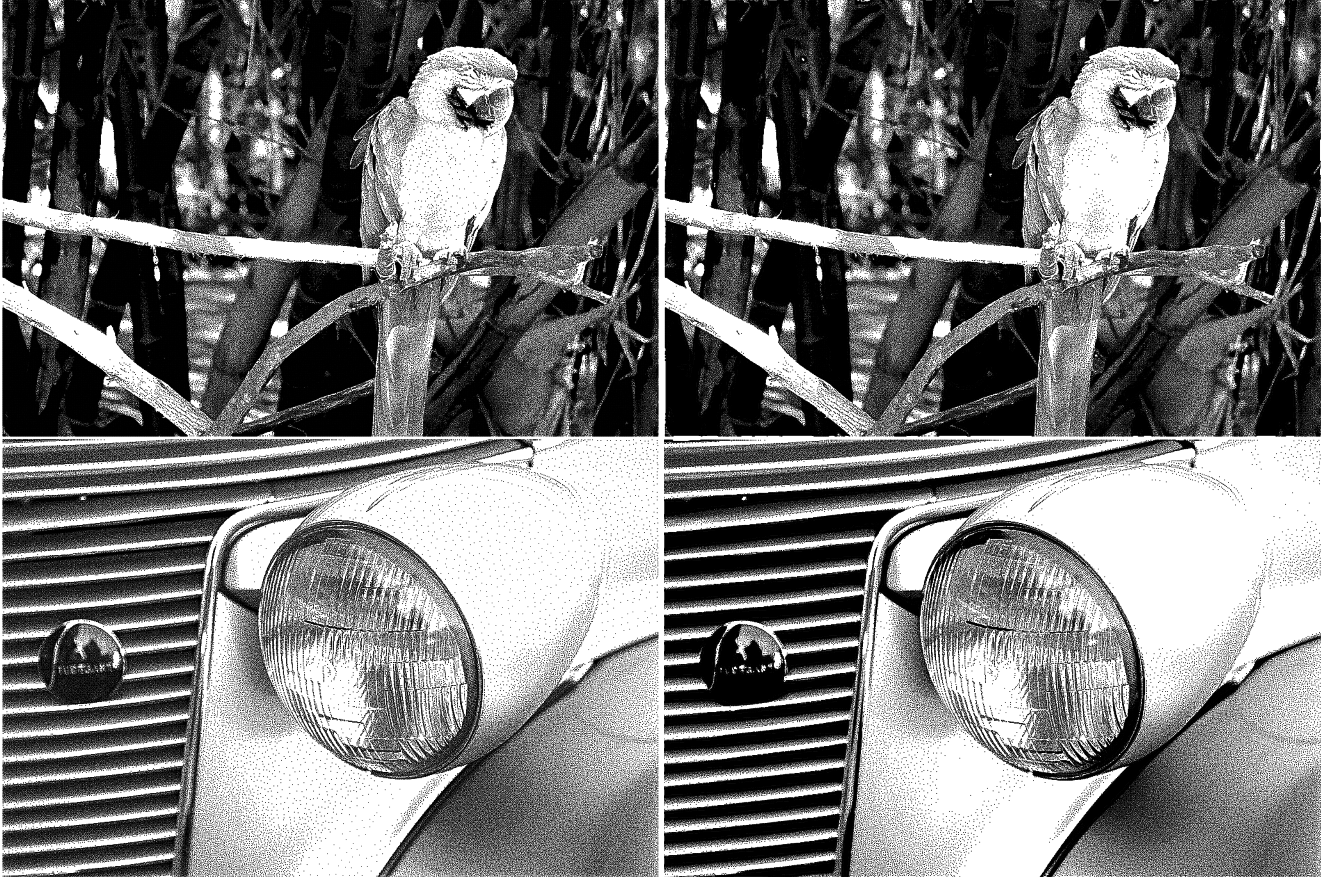


Figure 9: Results after applying our method on parrot and headlight based on structure-preserving stippling [LM11]. Left: original stipplings; right: our results.

Image	# Stipples (SPS)	# Stipples (ours)	Reduction	# Polygons
Toque	139K	107K	23%	121
Headlight	148K	125K	15%	32
Old man	82K	26K	67%	22
Oparara	293K	206K	30%	140
Theater	265K	255K	4%	96

Table 1: Stipple and polygon counts from SPS before and after applying our method.

Image	# Stipples (LBGS)	# Stipples (ours)	Reduction	# Polygons
Toque	10.6K	8.8K	17%	121
Headlight	13.26K	8.4K	37%	32
Old man	3.40K	2.01K	39%	22
Oparara	14.9K	11.6K	22%	140
Theater	17.35K	14.7K	15%	96

Table 2: Stipple counts from LBG stippling before and after applying our method.

stipples within the polygon, and in rare cases the overall stipple count for the image increases.

4.3. Limitations

Our method usually improves image quality, according to our subjective assessment. Nonetheless, it suffers from some limitations, generally arising from the use of solid colors (whether black or white) to represent large image regions. By thresholding the dark regions, we eliminate any details that might have been present there; we use a conservative default threshold, but for particular images, the user may need to adjust this setting. Detail in light regions is also eliminated through thresholding, but this is of less concern, since so few stipples would have been placed there that such details would not have been communicated in the first place. Our stipple removal through postprocessing is helpful, but imperfect: it might remove stipples that were useful, or fail to remove stipples that could safely have been eliminated. We have sought to avoid wrongly eliminating stipples and largely succeeded by limiting removal to regions with both few stipples and low contrast. In our observation, long bright gradients are the main problematic case; the *mac* image is an example where too many stipples have been removed, as the shape of the Mac is no longer apparent. We showed this case in Figure 12. The result can be improved by assigning a smaller value to the parameter c .

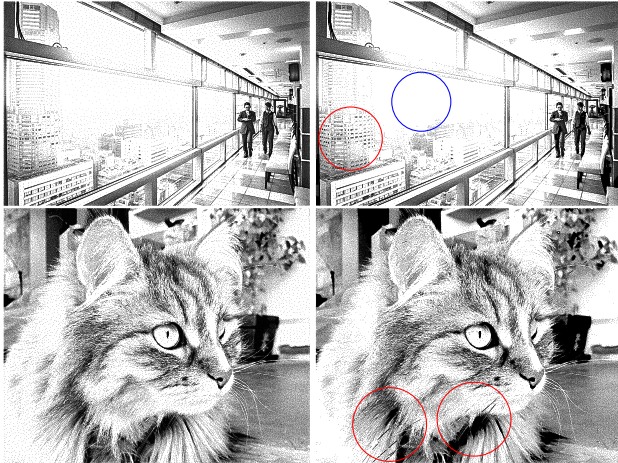


Figure 10: Results of detail enhancement. Left: original SPS; right: detail enhancement. Red circles show particular cases of detail enhancement and the blue circle shows stipple removal. Zoom in to see details.

Our suggested detail enhancement process is likely the most contentious part of our method. We identified details through computing the residuals from edge-preserving smoothing and then checking whether reversing the residual increased or decreased contrast; this process does not always succeed in improving visibility of details. Incorporating some notion of salience into the computation could be helpful. In addition, reversing the residuals and then stippling does not necessarily reveal the missing detail; the area might still be fairly light, so that too few stipples are placed. Overall, we suggest that more attention should be paid to the problem of stippling light details in light regions.

5. Conclusions

This paper described a method for improving stippled images in extreme-tone regions. We threshold images in two directions, setting sufficiently light pixels to maximum intensity and sufficiently dark pixels to minimum intensity; clusters of dark pixels are then represented by polygons rather than by a multitude of stipples. The intensity discontinuities resulting from thresholding are repaired by a Poisson blending operation over a narrow band surrounding each thresholded region. Despite thresholding, the stippling algorithm may still place isolated stipples that contribute little to image content; we postprocess the stipple distribution to remove such stipples.

The resulting images are clearer than the original stippled images, and use fewer stipples, dramatically so when large dark areas are replaced with polygons. Showing detail in light areas remains a problem, though. We suggest a preliminary approach to darken light details in light areas, making them feasible to show with a tone-preserving stippling algorithm; this portion of our method is effective in some cases but we do not recommend applying it blindly to all images. One direction for future work is to further address stipple representation of light details.

We deployed our pre- and postprocessing for structure-preserving stippling as well as LBG stippling. We believe that it would also benefit other stippling algorithms, but have not tested this supposition. Our approach might also be adapted to improve other stroke-based rendering methods or line drawing methods.

References

- [CAO09] CHANG, JIANGHAO, ALAIN, BENOÎT, and OSTROMOUKHOV, VICTOR. "Structure-aware error diffusion". *ACM Trans. Graph.* 28.5 (2009), 162:1–162:8. DOI: [10.1145/1618452.1618508](https://doi.org/10.1145/1618452.1618508). URL: <http://doi.acm.org/10.1145/1618452.1618508> 1, 2.
- [DHVS00] DEUSSEN, OLIVER, HILLER, STEFAN, VAN OVERVELD, CORNELIUS, and STROTHOTTE, THOMAS. "Floating Points: A Method for Computing Stipple Drawings". *Computer Graphics Forum* (2000). ISSN: 1467-8659. DOI: [10.1111/1467-8659.00396](https://doi.org/10.1111/1467-8659.00396) 1, 2.
- [DI13] DEUSSEN, OLIVER and ISENBERG, TOBIAS. "Halftoning and stippling". *Image and Video-Based Artistic Stylisation*. Ed. by ROSIN, PAUL and COLLOMOSSE, JOHN. Computational Imaging and Vision 42. London: Springer London, 2013, 45–61. ISBN: 978-1-4471-4518-9. DOI: [10.1007/978-1-4471-4519-6_3](https://doi.org/10.1007/978-1-4471-4519-6_3) 1.
- [DS02] DECARLO, DOUG and SANTELLA, ANTHONY. "Stylization and Abstraction of Photographs". *ACM Trans. Graph.* 21.3 (July 2002), 769–776. ISSN: 0730-0301. DOI: [10.1145/566654.566650](https://doi.org/10.1145/566654.566650). URL: <http://doi.acm.org/10.1145/566654.566650> 2.
- [DSZ17] DEUSSEN, O., SPICKER, M., and ZHENG, Q. "Weighted Linde-Buzo-Gray Stippling". *ACM Transactions on Graphics* 36.6 (Nov. 2017), 233:1–233:12. DOI: [10.1145/3130800.3130819](https://doi.org/10.1145/3130800.3130819). URL: <http://doi.acm.org/10.1145/3130800.3130819> 1, 2, 6.
- [GEB16] GATYS, LEON A., ECKER, ALEXANDER S., and BETHGE, MATTHIAS. "Image Style Transfer Using Convolutional Neural Networks". *CVPR*. IEEE Computer Society, 2016, 2414–2423 2.
- [Her98] HERTZMANN, AARON. "Painterly Rendering with Curved Brush Strokes of Multiple Sizes". *Proceedings of the 25th Annual Conference on Computer Graphics and Interactive Techniques*. SIGGRAPH '98. New York, NY, USA: ACM, 1998, 453–460. ISBN: 0-89791-999-8. DOI: [10.1145/280814.280951](https://doi.org/10.1145/280814.280951). URL: <http://doi.acm.org/10.1145/280814.280951> 2.
- [KMI*09] KIM, SUNGYE, MACIEJEWSKI, ROSS, ISENBERG, TOBIAS, et al. "Stippling By Example". *Proceedings of the Seventh International Symposium on Non-Photorealistic Animation and Rendering (NPAR, August 1–2, New Orleans, USA)*. Ed. by ISENBERG, TOBIAS, KAPLAN, CRAIG, and SPENCER, STEPHEN N. New York: ACM, 2009, 41–50. DOI: [10.1145/1572614.1572622](https://doi.org/10.1145/1572614.1572622) 2.
- [KNBH12] KALOGERAKIS, EVANGELOS, NOWROUZSAHRAI, DEREK, BRESLAV, SIMON, and HERTZMANN, AARON. "Learning hatching for pen-and-ink illustration of surfaces". *ACM Trans. Graph.* 31.1 (2012), 1:1–1:17 2.
- [KSL*08] KIM, DONGYEON, SON, MINJUNG, LEE, YUNJIN, et al. "Feature-guided Image Stippling". *Comput. Graph. Forum* 27.4 (2008), 1209–1216 2.
- [Llo82] LLOYD, STUART P. "Least squares quantization in PCM". *IEEE Transactions on Information Theory* 28 (1982), 129–137 2.
- [LM10] LI, HUA and MOULD, DAVID. "Contrast-aware Halftoning". *Comput. Graph. Forum* 29.2 (2010), 273–280 2.
- [LM11] LI, HUA and MOULD, DAVID. "Structure-preserving Stippling by Priority-based Error Diffusion". *Proceedings of Graphics Interface 2011*. GI '11. St. John's, Newfoundland, Canada: Canadian Human-Computer Communications Society, 2011, 127–134. ISBN: 978-1-4503-0693-5. URL: <http://dl.acm.org/citation.cfm?id=1992917.1992938> 1, 2, 5–7, 9.

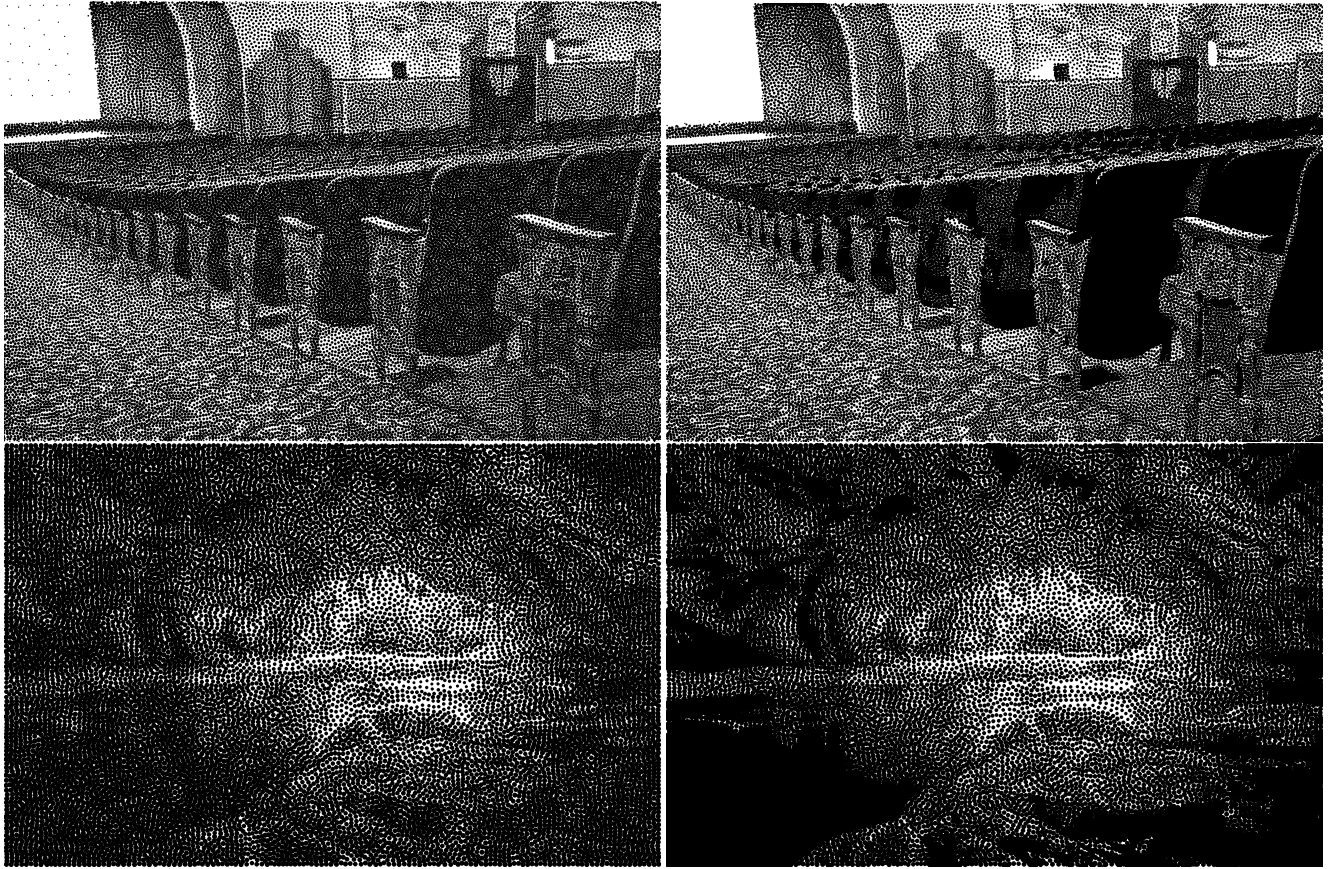


Figure 11: LBG stippling with our algorithm on Theater and Oparara images. Left column: original LBG stippling and right column: LBG stippling after applying our method.



Figure 12: Effect of parameter c in bright images. Left to right: original stippling [LM11], result with $c = 3$, $c = 4$, and $c = 5$.

[MAA117] MARTÍN, DOMINGO, ARROYO, GERMÁN, AGUILERA, ALEJANDRO RODRÍGUEZ, and ISENBERG, TOBIAS. “A survey of digital stippling”. *Computers & Graphics* 67 (2017), 24–44. DOI: [10.1016/j.cag.2017.05.001](https://doi.org/10.1016/j.cag.2017.05.001). URL: <https://doi.org/10.1016/j.cag.2017.05.001>.

[MALI10] MARTÍN, DOMINGO, ARROYO, GERMÁN, LUZÓN, M. VICTORIA, and ISENBERG, TOBIAS. “Example-Based Stippling using a Scale-Dependent Grayscale Process”. *Proceedings of the 8th International Symposium on Non-Photorealistic Animation and Rendering*

2010, Annecy, France, June 7–10, 2010. 2010, 51–61. DOI: [10.1145/1809939.1809946](https://doi.org/10.1145/1809939.1809946). URL: <http://doi.acm.org/10.1145/1809939.1809946>.

[MdSRI15] MARTÍN, DOMINGO, del SOL, VICENTE, ROMO, CELIA, and ISENBERG, TOBIAS. “Drawing characteristics for reproducing traditional hand-made stippling”. *NPAR*. Eurographics Association, 2015, 103–115 2.

[Mou07] MOULD, DAVID. “Stipple Placement Using Distance in a Weighted Graph”. *Proceedings of the Third Eurographics Conference*

- on *Computational Aesthetics in Graphics, Visualization and Imaging*. Computational Aesthetics'07. Alberta, Canada: Eurographics Association, 2007, 45–52. ISBN: 978-3-905673-43-2. DOI: [10.2312/COMPAESTH/COMPAESTH07/045-052](https://doi.org/10.2312/COMPAESTH/COMPAESTH07/045-052). URL: <http://dx.doi.org/10.2312/COMPAESTH/COMPAESTH07/045-052>.
- [Mou13] MOULD, DAVID. “Special Section on Expressive Graphics: Image and Video Abstraction Using Cumulative Range Geodesic Filtering”. *Comput. Graph.* 37.5 (Aug. 2013), 413–430. ISSN: 0097-8493. DOI: [10.1016/j.cag.2013.03.002](https://doi.org/10.1016/j.cag.2013.03.002). URL: <http://dx.doi.org/10.1016/j.cag.2013.03.002>.
- [Pel90] PELI, ELI. “Contrast in complex images”. *J. Opt. Soc. Am. A* 7.10 (Oct. 1990), 2032–2040. DOI: [10.1364/JOSAA.7.002032](https://doi.org/10.1364/JOSAA.7.002032). URL: <http://josaa.osa.org/abstract.cfm?URI=josaa-7-10-2032>.
- [PGB03] PFFDFFDREZ, P., GANGNET, M., and BLAKE, A. “Poisson image editing”. *ACM Transactions on Graphics (SIGGRAPH'03)* 22.3 (2003), 313–318.
- [PQW*08] PANG, WAI-MAN, QU, YINGGE, WONG, TIEN-TSIN, et al. “Structure-Aware Half-Toning”. *ACM Transactions on Graphics (SIGGRAPH 2008 issue)* 27.3 (2008), 89:1–89:8.
- [Sec02] SECORD, ADRIAN. “Weighted Voronoi Stippling”. *Proceedings of the 2Nd International Symposium on Non-photorealistic Animation and Rendering*. NPAR '02. Annecy, France: ACM, 2002, 37–43. ISBN: 1-58113-494-0. DOI: [10.1145/508530.508537](https://doi.org/10.1145/508530.508537). URL: <http://doi.acm.org/10.1145/508530.508537>.
- [SGS05] SCHLECHTWEG, STEFAN, GERMER, TOBIAS, and STROTHOTTE, THOMAS. “RenderBots – Multi-Agent Systems for Direct Image Generation”. *Computer Graphics Forum* 24.2 (2005), 137–148. DOI: [10.1111/j.1467-8659.2005.00838.x](https://doi.org/10.1111/j.1467-8659.2005.00838.x). eprint: <https://onlinelibrary.wiley.com/doi/pdf/10.1111/j.1467-8659.2005.00838.x>. URL: <https://onlinelibrary.wiley.com/doi/abs/10.1111/j.1467-8659.2005.00838.x>.
- [SLKD16] SEMMO, AMIR, LIMBERGER, DANIEL, KYPRIANIDIS, JAN ERIC, and DÖLLNER, JÜRGEN. “Image Stylization by Interactive Oil Paint Filtering”. *Comput. Graph.* 55.C (Apr. 2016), 157–171. ISSN: 0097-8493. DOI: [10.1016/j.cag.2015.12.001](https://doi.org/10.1016/j.cag.2015.12.001). URL: <https://doi.org/10.1016/j.cag.2015.12.001>.
- [SLKL11] SON, MINJUNG, LEE, YUNJIN, KANG, HENRY, and LEE, SE-UNGYONG. “Structure Grid for Directional Stippling”. *Graphical Models* 73.3 (2011), 74–87.