DOI: 10.1111/cgf.70364

EUROGRAPHICS 2026 / B. Masia and J. Thies
(Guest Editors)

COMPUTER GRAPHICS *forum*
Volume 45 (2026), Number 2

Graph-based Black and White Stylization

Ali Sattari Javid, Jimmy Lord, David Mould

Carleton University, Ottawa, Canada

Figure 1: *Our approach generates detailed black and white images from input photographs, combining stipples with larger regions. Left to right: original; importance map; medium-resolution rendering; high-resolution rendering.*

Abstract

Stippling is an art style forming images from large numbers of small dots. Computer graphics practitioners have proposed numerous algorithms for stippling and stippling variants over the years. In this paper, we propose a black and white image stylization technique in which the image is formed from heterogeneous black and white vector marks, with stipples and larger regions represented by polygons. Our algorithm applies two passes of stippling: first, we apply stippling to an importance map of the input image; then, using the stipples from the first pass as nodes in a planar graph, we create a binary stippling by labeling each graph node as either black or white using error diffusion. A connected component analysis of the resulting labeling, followed by vectorization of the components, yields a high-quality black and white representation of the original image. Isolated nodes, sharing a color with none of their neighbors, are akin to stipples; larger groups are polygons. We provide quantitative and qualitative comparisons with previous work on stippling and black and white rendering.

CCS Concepts

• ***Computing methodologies*** → ***Non-photorealistic rendering;*** *Image processing;*

1. Introduction

The simplicity of black and white rendering gives it aesthetic appeal, and binary reproductions are practical for applications in printing and display. Historically, continuous-tone images were converted to black and white through halftoning methods such as error diffusion and dithering. The related technique of stippling has been much studied in computer graphics, beginning with the pioneering work of Deussen et al. [DHOS99, DHVOS00]. Region-based techniques have also been employed, populating the image plane with large segments of black ink rather than numerous small strokes [MG08, XK08].

Solely using stipples, it is difficult to depict large black areas: huge numbers of stipples are required. Conversely, region-based techniques struggle to represent fine detail and gradations of tone, aspects at which stippling excels. Artists using media such as pen and ink or scratchboard do not restrict themselves to marks of any

particular size; scratchboard, in particular, makes it easy to create both large regions and tiny strokes. In this paper, we propose a unified method that first constructs a graph representation of an image with nodes labeled black or white, then aggregates groups of connected nodes into polygons while allowing isolated nodes to remain as individual strokes. The resulting images are intermediate between region-based and stippled images, and possess the strengths of both. Some sample results appear in Figure 1.

We take a two-pass approach to finding our final labeled graph. First, we compute an importance map over the input image; using importance as intensity, we halftone a regular input graph to label each node as either important or unimportant. The unimportant nodes are discarded, leaving a nonuniform graph with higher node density in important areas, less density in unimportant areas. This second graph undergoes the same graph-based halftoning algorithm to label nodes black or white by error diffusion based on the input image content. Intuitively, one can think of the final graph as a mesh, where a triangle is filled with a solid color when all corners have the same label. In short, we automatically create a heterogeneous, canvas-agnostic rendering; mixing regions and stipples, we retain the ability to show detail using stipples, while being able to cover large regions with polygons instead of huge numbers of individual stipples.

This paper makes the following contributions:

- We propose a novel algorithm for approximating an input greyscale image with a labeled graph, using error diffusion over the graph to assign labels.
- We propose to generate a variable-density graph from a regular input graph and an importance map, applying our error diffusion process with importance as intensity.
- We present a straightforward approach to extract a vector representation of the label-graph image for display.

2. Previous Work

General pen-and-ink illustration [Gup97] was first explored in a computer graphics context in the late 20th century [SABS94, WS94]. The earliest black and white rendering, done for display and reproduction purposes and lacking artistic intent, used halftoning algorithms such as error diffusion [FS76] to convert continuous-tone images into binary images. We build on error diffusion and hence discuss halftoning briefly, but our intent is stylization; there is a substantial literature on stippling, which is the closest past work to ours. We round out the section with a brief discussion of algorithms for the general case of binary image stylization.

2.1. Error Diffusion and Halftoning

Error diffusion converts continuous-tone (greyscale) images into binary representations, seeking to preserve the perceived tone level as much as possible by arranging the binary elements with suitable density. Processing pixels one by one, a given pixel is quantized to black or white and and the resulting quantization error is distributed to unprocessed pixels. By distributing error, the original brightness level is, on average, preserved. The most famous error diffusion method is due to Floyd and Steinberg [FS76]; they process pixels in raster order and propose a fixed-weight distribution function that pushes error into future pixels.

A breakthrough in halftoning came from Pang et al. [PQW*08], who eschewed error diffusion and attempted to directly optimize the distribution of black and white pixels using SSIM; followup work [CAO09, LM10] improved performance while maintaining the high visual quality provided by Pang et al. This paper proposes using error diffusion over a graph, with some elements inspired from contrast-aware halftoning [LM10] and its extension to structure-preserving stippling [LM11]. In particular, this previous work introduced the ideas of priority traversal of the pixel grid, and the notion of weighing error distribution according to some function of color and spatial relationships to the error. We directly use the priority traversal, and we also have dynamic error distribution weights, although the formulation of our weighting scheme is different. We also address the weakness of structure-preserving stippling in restricting stipple positions to a grid; we follow a two-pass approach, using the first pass to create a non-pixel-aligned graph on which to perform the actual halftoning.

2.2. Stippling

Stippling has received considerable attention from computer graphics researchers, beginning with work by Deussen et al. [DHOS99, DHVOS00]. Many methods have appeared in the years since [DKLS06, ADM19, DSZ17, BSD09, MCQS18, KCODL06, Sec02, MALI10, KMI*09]. Several of these methods have involved variations of centroidal Voronoi diagrams [Llo82], and this connection to computational geometry has inspired researchers; the present paper also connects to computational geometry, both in using the Delaunay triangulation to build a graph, and in using a graph as the key data structure rather than having disconnected points in space or points aligned to a pixel grid.

Pure stippling processes, although versatile and effective, struggle with very dark regions of images, as noted by previous authors [ADM19, SKB*21]. Azami et al. suggest replacing dark regions with polygons, while Schulz et al. [SKB*21] have a custom process for identifying gaps and covering them with additional stipples. The results of Schulz et al. are the closest to ours, both in terms of their high quality and because they deploy multiple colors of stipples. Whereas most stippling methods place black stipples on a plain white background, Schulz et al. cover the entire image. We do the same, using both stipples and larger polygonal regions.

Black and white rendering [MG08, XK08], potentially converted to a vector representation [LM15], handles large solid-colored regions well. However, past such methods have limited ability to convey fine details and intermediate tones. In this paper, we combine polygons with marks and inverted marks in a unified process.

2.3. Black and White Rendering

Black and white rendering excels at producing clean imagery, typically characterized by large uninterrupted areas of a single binary color. Filter-based methods for strict black and white stylization [MG08, XK08] reject small strokes in favor of detailed raster

structures, including complex region boundaries. The minimal rendering approach of Rosin and Lai [RL10] is also raster-based, offering a combination of content preservation and stylization options, albeit not adhering to a fully black and white representation. The use of sticks filtering in black and white rendering [LM15] promoted adherence to thin features such as tree branches and hair strands. The rendering method of Lin et al. [LYC18] produced tidy black and white images given input geometric models.

While these approaches have their merits, none are as effective as stippling at representing gradients or middle tones, where stippling excels. Our method aims to combine the representational advantages of stippling with the clarity of solid-colored regions.

2.4. Related Developments

In parallel with advancements in stippling, graphics researchers have contributed to developments in point distributions more broadly, with applications in procedural worlds [ENMGC19] and vector texture synthesis [TWZ22]. Fast-trending advances in machine learning and generative AI have touched on illustrative rendering, with SwiftSketch [AFCO*25] being the closest to this work; building on CLIPasso [VPB*22], which generates novel sketch-like vector illustrations from text prompts, SwiftSketch creates perceptually meaningful sketch-like abstractions of input images. The number of strokes is low, compared to primitive counts in typical stipple drawings or the detailed abstractions we propose in this paper. We anticipate that future machine learning approaches will make progress in the high-detail region of design space as well.

Diffusion models [RBL*22] are currently the most capable and general form of generative AI. They are well-suited to creating raster images; vector images are less explored. Diffusion-based style transfer processes such as that of DiffStyler [HZT*25] are an active area of development, transforming an input content image so that it matches a designated style, e.g., painterly or charcoal rendering. Such methods can suffer from poor content preservation and weak adherence to the desired style. The problems are particularly acute for stippled styles, where the low output resolution and strict color constraints raise the difficulty level; significant smearing and undesirable texture can result. Postprocessing could mitigate these issues. Alternatively, diffusion models can be deployed to generate point clouds [ZHM*24] rather than images; recent explorations in this direction could be applied to stippling, although this has not yet been investigated.

3. Method

3.1. Overview

We propose a graph-based method that produces stylized images akin to stipple drawings, mixing individual stipples and polygonal regions of varied size. Primitives are restricted to either black or white. Despite using varied primitives for rendering, our process uses a unified graph labeling, only separating distinct primitives in the final stage.

Our method consists of three distinct phases. In the first two phases, we approximate the input image with a binary labeling over a graph; typically, though not necessarily, the graph will have lower resolution than the input image, i.e., fewer nodes than the image had pixels. The first phase uses an importance map of the input image to create an irregular graph with higher node density in more important regions of the image; in the second phase, we compute a binary labeling over the graph, approximating the local intensities of the input image. In the final phase, the labeled graph is analyzed to extract connected components, with the components being converted into vector elements. Large components are directly replaced with polygons, and small components are replaced with points (stipples).

The underlying algorithm for the first and the second phase are identical: both construct a binary labeling of a graph based on error diffusion of an underlying image. In the first phase, we use a regular grid (e.g., a hexagonal grid or four-connected square grid), and an importance map of the input image provides the brightness levels for dithering. Our importance map comes from the OpenCV "fine-grained saliency" implementation, based on the work of Montabone and Soto [MS10]. This method was chosen chiefly in order to have an automated, reproducible source of importance for the results in this paper; we envision that in practice, users will want to provide a handcrafted salience map, possibly augmented with automatic edge or other detail detection. Having computed a binary labeling of the regular grid, we construct a graph (Delaunay triangulation) over the high-importance nodes, and in the second phase, we apply error diffusion to obtain a binary labeling of this graph.

An illustration of the processing pipeline is given in Figure 2. In the following subsections, we provide details of the graph representation (3.2), of the error diffusion subroutine (3.3), of the graph construction and application of error diffusion (3.4), and finally the vectorization process (3.5) that converts the second labeled graph into a black and white vector image. A sample implementation is available at `https://github.com/gigl-carleton/GraphBasedStippling`.

3.2. Graph Representation

We employ a graph data structure, a planar graph $G = \{N, E\}$ consisting of nodes N and edges E linking the nodes. Edges represent node adjacency and will be used to govern the error diffusion process in label assignment. Nodes have positions on the image plane and each node has some associated data, either derived from the input image or stored as bookkeeping en route to the eventual labeling. The initial goal of our algorithm is to find, for each node, a binary label $L \in \{0, 1\}$ such that the aggregate label assignment best approximates the input image.

Our algorithm makes use of the following per-node data. Each node i stores a current greyscale intensity I_i, modifiable through the error diffusion process; we also store the initial grey level, denoted I_i^0. The local average intensity of the original image is computed and stored, indicated by $\hat{I}_i^0$. Finally, we also have the per-node labels L_i, initially unknown. Note that nodes are not pixels; to map quantities such as intensity from a raster image to a graph, we compute the Voronoi diagram of the nodes and for each node, take the average quantity of pixels within the node's Voronoi region.

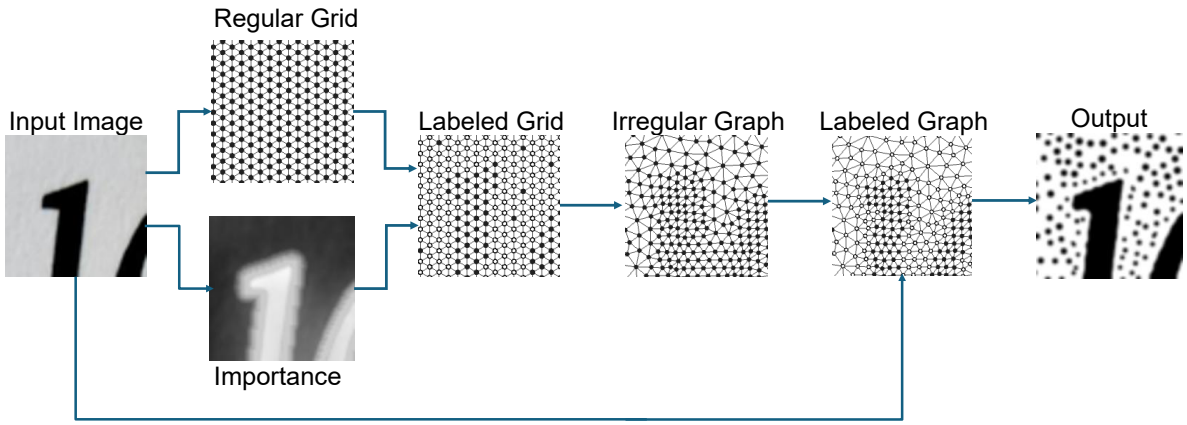

Figure 2: *Overview of approach. From the input image, we compute an importance map, and we conduct error diffusion over a grid to obtain a labeling of nodes as important or unimportant. The unimportant nodes are discarded and the positions are smoothed with centroidal Voronoi relaxation; the smoothed node positions are connected with a Delaunay triangulation, yielding an irregular graph. Error diffusion over this graph approximates the intensities of the input image. From the resulting binary labels, we obtain polygons and stipples, forming our output.*

3.3. Error Diffusion over a Graph

Given a graph G with per-node intensity values, we will use error diffusion to compute a labeling L over G such that $L_i \in \{0,1\}$ for all nodes i, approximating the continuous intensity values. This can be considered a halftoning problem and we use an error diffusion variant with serial priority akin to that proposed by Li and Mould [LM10]: nodes are processed in order of decreasing distance to middle grey, i.e., the brightest and darkest first.

Suppose we are processing node i. We assign a label L_i by thresholding its current intensity value I_i. Then, we determine the resulting error $\varepsilon_i = I_i - L_i$ and distribute it within a local neighborhood, modifying the intensities of the affected nodes. Error can be either positive or negative; positive error (increasing the brightness of the neighbors) arises from a label of zero, and negative error (darkening the neighbors) is produced when a label is set to one.

In distributing error ε_i from node i, each neighboring node j has an intensity update as follows:

$$I'_j \leftarrow I_j + w_{ij}\alpha_i\varepsilon_i, \tag{1}$$

where the weight w_ij is determined by the intensity of node j and within-graph relationships between nodes i and j, and α_i is a per-node error manipulation term, adjusting the propagated error up or down based on whether we want to amplify local details or suppress them. We discuss the weights next, followed by our proposed calculation for parameter α_i.

We compute individual weights for nodes j in the neighborhood of node i, where a node's weight decreases with spatial distance (more error is given to nearby nodes) and where a node's weight decreases with colorspace distance from the labeled node's original intensity level (nodes with more dissimilar starting intensities receive less error). This intention is formalized in the following equation:

$$w_{ij} = \exp(-(k_1 d_c(i,j) + k_2 d_s(i,j))^2), \tag{2}$$

where $d_s(i,j)$ is the Euclidean distance between nodes i and j, and $d_c(i,j)$ is $||I_i^0 - I_j^0||$. This formulation is akin to the bilateral filter, where colorspace distance and spatial distance are combined. We use $k_1 = 1$ and $k_2 = 0.1$ for all results in this paper.

We next turn our attention to the quantity α_i, used for error manipulation. Our aim is to enhance detail while suppressing small variations. By default, the error scaling is set to 0.5, promoting larger solid-colored regions and reducing detail. However, strong deviations from the local average often constitute image content that should be preserved in the binary labeling. Our heuristic approach for α amplifies error when the processed node more greatly differs from its surroundings (enhancing details), as follows:

$$\alpha_i = 0.5 + \begin{cases} \lambda||I_i^0 - \hat{I}_i^0|| & \text{if } \text{sign}(\varepsilon) = \text{sign}(I_i^0 - 0.5) \\ 0 & \text{if } \text{sign}(\varepsilon_i) \neq \text{sign}(I_i^0 - 0.5) \end{cases} \tag{3}$$

We use an empirically-determined $\lambda = 6$ for all results shown in this paper. The results are not especially sensitive to the choice of λ over quite a broad range, however; for approximately $2 < \lambda < 8$ the differences are reasonably subtle and are not discernible without close study. Based on these observations, we did not expend significant effort in attempting to tune λ.

Figure 3 shows the effect of static and dynamic α. On the left and middle respectively, we see two extremes of no error diffusion ($\alpha = 0$) and no error manipulation ($\alpha = 1$). Zero α is equivalent to removing error and thresholding each node individually; large solid-colored regions appear, but many details are lost and the transition from black to white is messy. Setting $\alpha = 1$ yields strict halftoning over the graph; the original greyscale value is preserved,

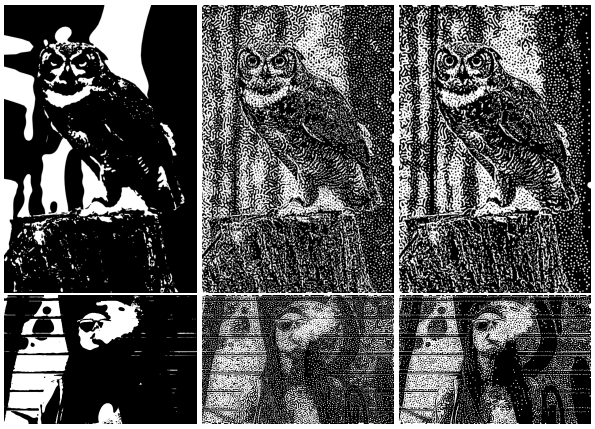

Figure 3: *Effect of error diffusion and manipulation. Left to right:* $\alpha = 0$ *(no error diffusion);* $\alpha = 1$ *(regular error diffusion); dynamic alpha using our proposed method.*

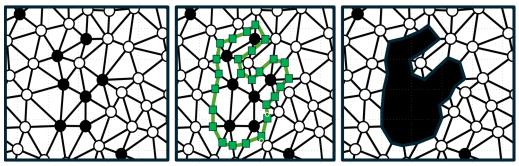

Figure 4: *Extracting a polygon from the labeled graph. The central area is a single connected component; its boundary is obtained from the points halfway along each edge.*

but contrast is low and key details such as edges are unclear. Neither extreme is desirable. The rightmost image shows our proposed dynamic error, using Equation 3; large solid-colored regions appear, while details and boundaries are preserved. Additional results and discussion on the role of α appear in Section 4.3.

3.4. Graph Construction and Sequential Error Diffusion

Our black and white stylization uses the error diffusion process twice. First, we apply it to a regular graph (hexagonal grid) whose per-node intensities are taken from an importance map of the image. Call this graph G_1. The error diffusion process yields a binary labeling over G_1 which we can interpret as "more important" and "less important" nodes. We discard the "less important" nodes.

From the "more important" nodes of G_1, we construct a second graph, say G_2. Graph G_2 has greater resolution in areas of higher importance, lower resolution in less-important areas. We run a single pass of Lloyd's algorithm [Llo82] over G_2, then compute the Delaunay triangulation of the resulting node positions. The nodes are populated with intensity values drawn from the original input image, averaged over the Voronoi region of each node.

Our error diffusion process then computes a binary labeling over G_2. This final labeling is is the basis for the stippled image: we extract black and white primitives for rendering, as described next.

Acronym	Method name	Reference
MLBG	Multi-class Linde-Buzo-Gray	[SKB*21]
LBG	Linde-Buzo-Gray stippling	[DSZ17]
CCVT	Capacity-Constrained Voronoi Tesselations	[BSD09]
SPS	Structure-Preserving Stippling	[LM11]
IVS	Incremental Voronoi Sets	[MCQS18]
RWT	Recursive Wang Tiles	[KCODL06]
WCVD	Weighted Centroidal Voronoi Diagrams	[Sec02]
SBE	Stippling By Example	[KMI*09]

Table 1: *Table of methods*

3.5. Primitive Extraction

The labeled graph is converted to a conventional vector representation of the image, as follows.

We perform connected components labeling [GW08] in the labeled graph. With a planar graph from the Delaunay triangulation, adjacency is unambiguous and the segments are well-defined. All isolated segments (components with only one node) are points, or stipples. The remaining segments are considered polygons.

For each polygon, we create a boundary by taking the midpoint of each edge; see Figure 4. The resulting boundary is smoothed and stored as a path to be reproduced at render time.

Note that only simple polygons (i.e., topological discs) are produced with the above method. Although polygons can contain holes, here the holes are represented by separate entities superimposed on simple polygons. For all vector elements, drawing in order of decreasing area (largest first) ensures that visibility is respected, given our context of elements extracted from a planar graph.

4. Results and Evaluation

In the following, we first show several results from our method, comparing against a range of previous stippling approaches (see Table 1) and particularly with Schulz et al. [SKB*21], whose representation and results are closest to ours. Following this, we report the outcome of quantitative measures over a standard image set. Next we give a brief ablation of some key algorithm elements. Finally, we discuss some limitations. Original images used are summarized in Table 2.

4.1. Qualitative Results

Figure 5 shows comparisons with other stippling methods using a public domain illustration by Karen Couch. These images were previously presented by Schulz et al. [SKB*21] and we build on their comparison here; our result appears in the top right of the grid. We particularly call attention to the result of Schulz et al. in the bottom left as having the best quality among previous methods.

Because the original image has a mix of gradients and very dark and light areas, a black and white representation benefits from being able to deploy both stipples and larger primitives. Strong edges,

Image name	Figure	Source
parrot	1	Unsplash
owl	2	Unsplash
smoker	2	Unsplash
frog	5	Karen Couch
portrait	6	public domain
cat	7	Alexas Fotos
lanterns	8	Berkeley segmentation dataset
bridge	8	Carsten Ullrich
headlight	9	NPR benchmark
Oparara	9	NPR benchmark
rim lighting	9	NPR benchmark
toque	9	NPR benchmark
coupe	10	Unsplash
poker	10	Unsplash

Table 2: *Table of image sources. "NPR benchmark" refers to the images compiled by Mould and Rosin [MR17].*

such as the frog's silhouette, are better preserved by sharp boundaries rather than disconnected stipples; note that many stippling methods convey fuzzy boundaries or produce stipples wandering away from the frog. Gradients, on the other hand, benefit greatly from stippling. The original frog has some bright details on a dark background, such as highlights on some spots; Schulz et al.'s result, using white stipples on top of dark ones, conveys these aspects extremely well. Similarly, our ability to place both white and black primitives allows us to preserve bright details.

The high contrast of the frog makes it a straightforward subject for representation in black and white. In Figure 9 we show some more challenging cases, with lower contrast, long gradients, and light details. Only Schulz et al.'s method, along with ours, is able to cope well with the last of these, directly representing light details with primitives as opposed to revealing them implicitly through use of negative space. Our error manipulation produces higher contrast, allowing the background of the rim lighting image to remain entirely dark and reinforcing the details in the Oparara cave. Variable graph density from the importance map allows detail to concentrate on the foreground in the toque image; fine structures on the scarf and the weave pattern on the hat can be indicated, whereas the smoother point distribution of Schulz et al. omits such detail. In the headlight image, however, Schulz et al.'s stipples well convey the gradients, which are attenuated in our method. Our method also sometimes produces distracting patterns across smooth transitions from darker to lighter regions, noticeable in the headlight image. Another comparison is given in Figure 7; both methods well convey the image content. For this and other images, we encourage the reader to zoom in to see small details.

Figure 6 compares our approach with structure-preserving stippling [LM11]. SPS does indeed preserve details, such as in the hair and around the ear, but its sparsity of stipples poses challenges for representing tone. Our approach at similar node density has comparable detail preservation but substantially better tone reproduction, and thus an overall stronger sense of the original image content.

Figure 8 shows comparisons with two methods for region-based

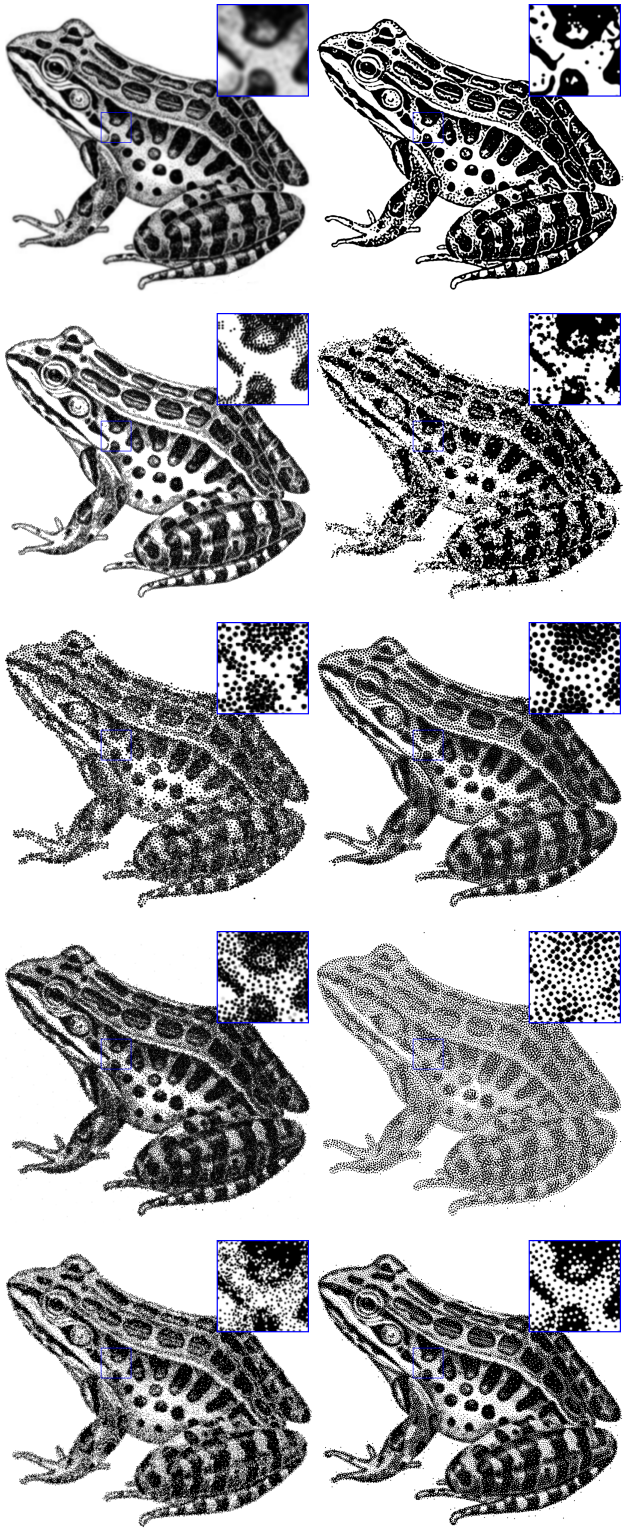

Figure 5: *Comparisons with various approaches. In raster order: original; ours; structure-preserving stippling [LM11]; CCVT [BSD09]; IVS [MCQS18]; LBG [DSZ17]; Wang tiles [KCODL06]; weighted Voronoi stippling [Sec02]; stippling by example [KMI*09]; MLBG [SKB*21]. All subfigures except ours adapted from Schulz et al. [SKB*21].*

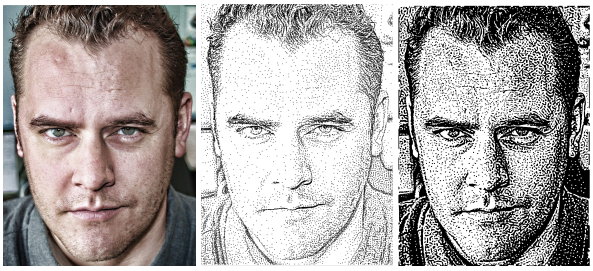

Figure 6: *Comparison of our method with structure-preserving stippling [LM11]. Left to right: original; stippled result; our result.*

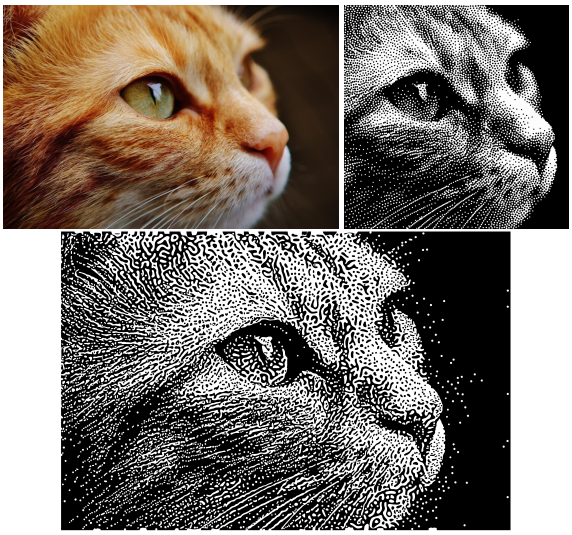

Figure 7: *Comparison of our method with MLBG stippling [SKB*21]. Above: original, and stippled result created by MLBG authors; below: our result.*

Figure 8: *Some comparisons with pure black and white rendering. Top: lanterns; bottom: bridge. Left to right: Xu and Kaplan; Li and Mould; ours.*

Figure 9: *Selected images from the benchmark. Top to bottom: headlight; Oparara; rim lighting; toque. Left: MLBG [SKB*21]; right: ours.*

black and white rendering, those of Xu and Kaplan [XK08] and Li and Mould [LM15]. The segmentation-based approach of Artistic Thresholding fares poorly when image content does not segment neatly (e.g., the man's face in the lanterns image). Both region-based results have limited ability to portray gradients, such as the shape of the lanterns or the illuminated water under the bridge. Mixing stipples and regions allows us to capture both large-scale shapes and fine details or gradients.

4.2. Quantitative Results

Table 3 shows image similarity metrics: perceptual similarity as measured by LPIPS, and image similarity as measured by SSIM and PSNR. We compare with other methods, reporting the average scores for images on the NPR-benchmark image set [MR17]. By far the most relevant past method is that of Schulz et al. [SKB*21] and we provide comparisons at multiple scales. Resolution is given as the length of the longest side (not all images are square). CCVT and SPS use these image dimensions directly; for Schulz et al.'s method, we double the image dimension and use stipple size range 2-4 in the authors' tool, downsampling to the original resolution before comparing; for our method, we create an initial hex grid with nodes matching the pixel count, and rasterize our final vector image at the original image resolution.

As Schulz et al. comment in their paper, strict adherence to the original intensity (as measured by SSIM and PSNR) is not particularly exciting. Many methods can achieve very high fidelity by rendering at very high resolutions, but in this case would no longer be characterized as stylization. More interesting is the middle ground where the stylization is evident, but the original image content is still apparent. At all scales, we obtain superior LPIPS scores to all

Method	LPIPS ↓	PSNR ↑	MSSIM ↑
Ours (1024)	**0.566**	16.77	*0.505*
Schulz et al. (1024)	*0.607*	**19.56**	**0.599**
CCVT (1024)	0.761	9.61	0.075
SPS (1024)	0.739	5.37	0.008
LBG (1024)	0.647	*17.97*	0.468
RWT (1024)	0.664	15.92	0.336
WCVD (1024)	0.706	10.82	0.177
SBE (1024)	0.668	15.34	0.300
Ours (512)	0.550	16.75	0.405
Schulz (512)	0.594	19.59	0.391
Ours (256)	0.530	16.80	0.440
Schulz (256)	0.566	19.42	0.397

Table 3: *Quantitative comparisons for benchmark image set for various methods. Best scores for 1024 resolution are indicated in bold, second best in italics.*

Image	#Stipples	#Polygons	#Vertices
parrot (medium)	1623	522	50k
parrot (high)	7346	1743	165k
poker	1474	169	24k
coupe	2125	781	62k
frog	788	362	54k
portrait	2765	927	80k
toque	2904	604	59k
Oparara	2422	516	26k
rim	1016	251	24k
headlight	1931	364	53k

Table 4: *Primitive counts for selected images. "Vertices" includes all polygon corners and stipple positions.*

previous methods, indicating good recognizability of content; our LPIPS scores are somewhat stable with respect to scale. We have deliberately increased contrast, and consequently have lower PSNR than do Schulz et al. Since they aim to reproduce tone accurately, and also do not distinguish between different regions of the image, Schulz et al. benefit more than we do from greater primitive density. As an aside, we comment that by removing error manipulation (reverting to conventional error diffusion, albeit over our nonuniform graph structure rather than a grid) we can obtain higher MSSIM and PSNR scores, at the expense of worse perceptual outcomes as measured by LPIPS.

Table 4 reports primitive counts for several of our test images. We separately report the number of stipples, the number of polygons, and the total number of vertices in all paths and objects in a vector graphics representation of our images. Unsurprisingly, the number of primitives rises with image complexity, with simpler images such as "poker" requiring substantially fewer primitives than a complex, high-salience image such as "toque". Most of our cost comes from representing the polygons at the resolution of the second graph; reductions in vertex count are available by decimating the polygons. Results took approximately one minute to generate, on unoptimized CPU-only code, on a single-threaded program running on a 2920x Threadripper equipped with 64 GB of RAM.

Variant	LPIPS ↓	PSNR ↑	MSSIM ↑
Constant importance	0.550	17.07	0.399
No relaxation	0.550	16.75	0.405
No error manipulation	0.564	**17.59**	0.389
Full method	**0.543**	16.96	**0.420**

Table 5: *Quantitative scores for ablated algorithm variants, computed over the benchmark set. Best scores are indicated in bold.*

4.3. Ablation Results

Here, we ablate three elements of our algorithm: the use of salience to modify density; the use of Lloyd's method to relax the node distribution before the final round of error diffusion; and our error manipulation. Representative images are shown in Figure 10, and quantitative results over the NPR benchmark set are given in Table 5.

Running with fixed importance produces a slightly irregular graph, as nodes are removed from a regular pattern and the surviving nodes smoothed; the result is plausible, but some resolution is wasted in irrelevant areas. For example, notice how the edges of the symbols on the cards are distorted compared to the edges in the full method, where primitives concentrate near such details and the reproduction is more accurate. The no-relaxation approach produces results very similar to the full method, but in some places (e.g., vertical alignment of stipples in the sky; spurious vertical structure in the shadow beneath the coupe) the underlying regular structure becomes visible. Note that the regular structures are less objectionable than they would be if rendered solely with stipples, as the extracted polygons conceal much of the grid.

Running the process without any error manipulation (fixed $\alpha = 1$) is akin to halftoning. It produces quite reasonable results in cases where the original image was already high-contrast (the poker image is one such), but for more natural images with lower contrast and subtle variations of intensity, the contrast enhancement from manipulating error produces a clearer subject. In the coupe image, many aspects of the content are clearer: the horizon; the highway lines; even the coupe itself stands out more.

Table 5 shows MSSIM, PSNR, and LPIPS scores for the variants. As in Section 4, we emphasize the LPIPS scores as most appropriate for evaluating content preservation in stylization applications. MSSIM and PSNR indicate fidelity to the original image, which is not so important in stylization; maximum scores here could be achieved by doing nothing, returning the original image unchanged. Our error modification scheme (varying α) increases contrast, leading to more identifiable content and thus improved LPIPS scores; fixing $\alpha = 1$ more faithfully represents the original, as indicated by higher PSNR, but makes the content harder to see (worse LPIPS) as we also noted in the qualitative assessment.

This partial ablation demonstrates the effect of different elements of the algorithm. The most important ingredient is the error manipulation, making the image content more clear in low-contrast cases. The full two-pass method with importance weighting allows us to deploy stipples in the most relevant areas, plus removes occasional artifacts caused by the initial grid. An immediate direction for future development would be to look at improving the importance

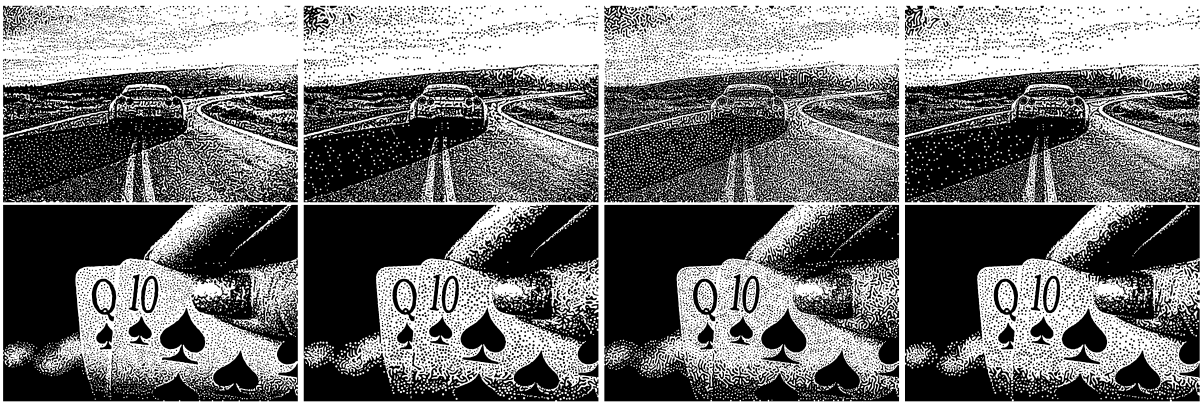

Figure 10: *Ablations. Left to right: constant importance; no relaxation; no error manipulation; full method. Above: coupe; below: poker.*

map itself, either choosing a method better suited to this application, or even devising a dedicated method focused on the case of black and white rendering.

4.4. Limitations

Our high-quality results notwithstanding, our method has some drawbacks. The output images often contain distracting labyrinthine structures in middle-tone regions. Similarly, occasional regions are created where by chance nearby nodes are given the same label, and it would be better to keep them as separate stipples.

Our approach is presently aimed only at black and white results. We would like to extend it to multiple labels, and to labels other than black and white for colorful binary images.

We rely on an importance map to estimate where to expend more attention, in the form of graph nodes. We used the salience detection method of Montabone and Soto [MS10]; a more semantically aware approach would benefit us greatly. Stipple artists often disregard the background of an image entirely, which existing stippling algorithms do not do automatically.

Lastly, although our vertex counts are not wildly greater than stipple counts from many past methods, we would like to reduce them to increase the practicality of our approach. We have not investigated smoothing or otherwise downsampling our polygons, and we believe this will straightforwardly improve the memory footprint of vector images generated using our method.

5. Conclusions

We presented a method for stylizing input images as black and white vector graphics, mixing polygons and individual stipples in a unified underlying graph representation. The method works by performing two rounds of graph-based error diffusion, first on a regular grid to approximate an input importance map, then on the Delaunay triangulation of the points resulting from the first pass to approximate the input image intensities. The resulting binary graph labeling is converted to vector form by identifying connected components; very small components are stipples, and larger components are polygons.

The approach is capable of creating large polygons to cover light or dark areas, with detail in the form of small polygons or stipples added on top. Mixing light and dark primitives was previously proposed by Schulz et al. [SKB*21] for stipples; our structures attain better LPIPS scores than previous methods, which we attribute to our error modification process promoting locally extremal points in the image.

Our approach lends itself to followup work. We used an existing salience estimator for the importance map; further investigation of importance maps tailored to the black and white rendering application could be productive. Our vectorization of the labeled graph is a straightforward boundary traversal and additional detail could be added, or extra constraints or stylization placed on the boundary. We can also consider generalizing the binary labeling to a wider palette for applications such as posterization.

6. Acknowledgements

Thanks to previous authors who provided source code, executables, and images for comparison. Many test images were taken from Unsplash (unsplash.com) and are used under their licensing terms. Thanks to the reviewers who provided many helpful suggestions. This work was funded in part by the NSERC Discovery Grant program, RGPIN-2018-06298.

References

[ADM19] AZAMI R., DOYLE L., MOULD D.: Stipple removal in extreme-tone regions. In *ACM/Eurographics Expressive Symposium* (Goslar, DEU, 2019), Expressive '19, Eurographics Association, pp. 123–132. doi:10.2312/exp.20191083. 2

[AFCO*25] ARAR E., FRENKEL Y., COHEN-OR D., SHAMIR A., VINKER Y.: SwiftSketch: A diffusion model for image-to-vector sketch generation. In *Proc. SIGGRAPH* (New York, NY, USA, 2025), SIGGRAPH Conference Papers '25, ACM. doi:10.1145/3721238.3730612. 3

[BSD09] BALZER M., SCHLÖMER T., DEUSSEN O.: Capacity-constrained point distributions: a variant of Lloyd's method. *ACM Trans.*

Graph. 28, 3 (July 2009). doi:10.1145/1531326.1531392. 2, 5, 6

[CAO09] CHANG J., ALAIN B., OSTROMOUKHOV V.: Structure-aware error diffusion. In *SIGGRAPH Asia 2009* (New York, NY, USA, 2009), SIGGRAPH Asia '09, ACM. doi:10.1145/1661412.1618508. 2

[DHOS99] DEUSSEN O., HILLER S., OVERVELD C. V., STROTHOTTE T.: Computer-generated stipple drawings. In *Workshop on Vision, Modelling, and Visualization (VMV)* (1999), pp. 329–338. 1, 2

[DHVOS00] DEUSSEN O., HILLER S., VAN OVERVELD C., STROTHOTTE T.: Floating points: A method for computing stipple drawings. *Computer Graphics Forum* (2000). doi:10.1111/1467-8659.00396. 1, 2

[DKLS06] DALAL K., KLEIN A. W., LIU Y., SMITH K.: A spectral approach to NPR packing. In *International Symposium on Non-photorealistic Animation and Rendering* (New York, NY, USA, 2006), NPAR '06, ACM, pp. 71–78. doi:10.1145/1124728.1124741. 2

[DSZ17] DEUSSEN O., SPICKER M., ZHENG Q.: Weighted Linde-Buzo-Gray stippling. *ACM Transactions on Graphics 36*, 6 (nov 2017), 233:1–233:12. doi:10.1145/3130800.3130819. 2, 5, 6

[ENMGC19] ECORMIER-NOCCA P., MEMARI P., GAIN J., CANI M.-P.: Accurate synthesis of multi-class disk distributions. *Computer Graphics Forum 38*, 2 (2019), 157–168. doi:10.1111/cgf.13627. 3

[FS76] FLOYD R. W., STEINBERG L.: An adaptive algorithm for spatial gray-scale. *Proc. Soc. Inf. Disp. 17* (1976), 75–77. 2

[Gup97] GUPTILL A. L.: *Rendering in Pen and Ink*. Watson-Guptill Publications, New York, NY, USA, 1997. 2

[GW08] GONZALEZ R. C., WOODS R. E.: *Digital image processing*, 3rd ed. Prentice Hall, Upper Saddle River, N.J, 2008. 5

[HZT*25] HUANG N., ZHANG Y., TANG F., MA C., HUANG H., DONG W., XU C.: DiffStyler: Controllable dual diffusion for text-driven image stylization. *IEEE Transactions on Neural Networks and Learning Systems 36*, 2 (2025), 3370–3383. doi:10.1109/TNNLS.2023.3342645. 3

[KCODL06] KOPF J., COHEN-OR D., DEUSSEN O., LISCHINSKI D.: Recursive Wang tiles for real-time blue noise. *ACM Trans. Graph. 25*, 3 (July 2006), 509–518. doi:10.1145/1141911.1141916. 2, 5, 6

[KMI*09] KIM S. Y., MACIEJEWSKI R., ISENBERG T., ANDREWS W. M., CHEN W., SOUSA M. C., EBERT D. S.: Stippling by example. In *International Symposium on Non-Photorealistic Animation and Rendering* (New York, NY, USA, 2009), NPAR '09, ACM, pp. 41–50. doi:10.1145/1572614.1572622. 2, 5, 6

[Llo82] LLOYD S. P.: Least squares quantization in PCM. *IEEE Transactions on Information Theory 28* (1982), 129–137. 2, 5

[LM10] LI H., MOULD D.: Contrast-aware halftoning. *Computer Graphics Forum 29*, 2 (2010), 273–280. doi:https://doi.org/10.1111/j.1467-8659.2009.01596.x. 2, 4

[LM11] LI H., MOULD D.: Structure-preserving stippling by priority-based error diffusion. In *Graphics Interface 2011* (University of Waterloo, Waterloo, Ontario, Canada, 2011), GI '11, CHCCS, pp. 127–134. 2, 5, 6, 7

[LM15] LI H., MOULD D.: Contrast-enhanced black and white images. *Computer Graphics Forum 34*, 7 (2015), 235–245. doi:10.1111/cgf.12752. 2, 3, 7

[LYC18] LIN Y.-E., YANG Y.-L., CHU H.-K.: Scale-aware black-and-white abstraction of 3D shapes. *ACM Trans. Graph. 37*, 4 (July 2018). doi:10.1145/3197517.3201372. 3

[MALI10] MARTÍN D., ARROYO G., LUZÓN M. V., ISENBERG T.: Example-based stippling using a scale-dependent grayscale process. In *International Symposium on Non-Photorealistic Animation and Rendering* (2010), pp. 51–61. doi:10.1145/1809939.1809946. 2

[MCQS18] MA L., CHEN Y., QIAN Y., SUN H.: Incremental Voronoi sets for instant stippling. *The Visual Computer 34*, 6–8 (June 2018), 863–873. doi:10.1007/s00371-018-1541-7. 2, 5, 6

[MG08] MOULD D., GRANT K.: Stylized black and white images from photographs. In *International Symposium on Non-Photorealistic Animation and Rendering* (New York, NY, USA, 2008), NPAR '08, ACM, pp. 49–58. doi:10.1145/1377980.1377991. 1, 2

[MR17] MOULD D., ROSIN P. L.: Developing and applying a benchmark for evaluating image stylization. *Computers & Graphics 67* (2017), 58–76. doi:https://doi.org/10.1016/j.cag.2017.05.025. 6, 7

[MS10] MONTABONE S., SOTO A.: Human detection using a mobile platform and novel features derived from a visual saliency mechanism. *Image Vision Comput. 28*, 3 (Mar. 2010), 391–402. doi:10.1016/j.imavis.2009.06.006. 3, 9

[PQW*08] PANG W.-M., QU Y., WONG T.-T., COHEN-OR D., HENG P.-A.: Structure-aware halftoning. In *ACM SIGGRAPH 2008 Papers* (New York, NY, USA, 2008), SIGGRAPH '08, ACM. doi:10.1145/1399504.1360688. 2

[RBL*22] ROMBACH R., BLATTMANN A., LORENZ D., ESSER P., OMMER B.: High-resolution image synthesis with latent diffusion models. In *CVPR* (2022), IEEE, pp. 10674–10685. doi:10.1109/CVPR52688.2022.01042. 3

[RL10] ROSIN P. L., LAI Y.-K.: Towards artistic minimal rendering. In *International Symposium on Non-Photorealistic Animation and Rendering* (New York, NY, USA, 2010), NPAR '10, ACM, pp. 119–127. doi:10.1145/1809939.1809953. 3

[SABS94] SALISBURY M. P., ANDERSON S. E., BARZEL R., SALESIN D. H.: Interactive pen-and-ink illustration. In *Proc. SIGGRAPH* (New York, NY, USA, 1994), SIGGRAPH '94, ACM, pp. 101–108. doi:10.1145/192161.192185. 2

[Sec02] SECORD A.: Weighted Voronoi stippling. In *International Symposium on Non-photorealistic Animation and Rendering* (New York, NY, USA, 2002), NPAR '02, ACM, pp. 37–43. doi:10.1145/508530.508537. 2, 5, 6

[SKB*21] SCHULZ C., KWAN K. C., BECHER M., BAUMGARTNER D., REINA G., DEUSSEN O., WEISKOPF D.: Multi-class inverted stippling. *ACM Trans. Graph. 40*, 6 (Dec. 2021). doi:10.1145/3478513.3480534. 2, 5, 6, 7, 9

[TWZ22] TU P., WEI L.-Y., ZWICKER M.: Clustered vector textures. *ACM Trans. Graph. 41*, 4 (July 2022). doi:10.1145/3528223.3530062. 3

[VPB*22] VINKER Y., PAJOUHESHGAR E., BO J. Y., BACHMANN R. C., BERMANO A. H., COHEN-OR D., ZAMIR A., SHAMIR A.: CLIPasso: semantically-aware object sketching. *ACM Trans. Graph. 41*, 4 (July 2022). doi:10.1145/3528223.3530068. 3

[WS94] WINKENBACH G., SALESIN D. H.: Computer-generated pen-and-ink illustration. In *Proc. SIGGRAPH* (New York, NY, USA, 1994), SIGGRAPH '94, ACM, pp. 91–100. doi:10.1145/192161.192184. 2

[XK08] XU J., KAPLAN C. S.: Artistic thresholding. In *International Symposium on Non-Photorealistic Animation and Rendering* (New York, NY, USA, 2008), NPAR '08, ACM, pp. 39–47. doi:10.1145/1377980.1377990. 1, 2, 7

[ZHM*24] ZHENG X., HUANG X., MEI G., HOU Y., LYU Z., DAI B., OUYANG W., GONG Y.: Point cloud pre-training with diffusion models. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2024, Seattle, WA, USA, June 16-22, 2024* (2024), IEEE, pp. 22935–22945. doi:10.1109/CVPR52733.2024.02164. 3