International Journal of Image and Graphics

GAUSSIAN IMAGE BINARIZATION

HENRY KANG
University of Missouri – St. Louis
St. Louis, MO 63121, USA
kangh@umsl.edu

IOANNIS STAMOULIS
University of Oxford
Oxford OX1 2JD, UK
iostamoulis@gmail.com

Line drawing and screentoning are two distinct areas of study in non-photorealistic rendering, where the former emphasizes object contours, while the latter conveys tone and shading information on object surfaces. As these two problems are concerned with different yet equally important features, either method seldom delivers a complete description of the scene when used alone. Yet, research community has largely treated them as separate problems and thus resulted in two entirely different sets of solutions, complicating both implementation and usage. In this paper, we present a stylistic image binarization method called *hybrid difference of Gaussians (HDoG)* that performs both line drawing and screentoning in a unified framework. Our method is based upon two different extensions of DoG operator: one for line extraction, and the other for tone description. In particular, we propose an extension called *adaptive DoG*, that uses luminance as weight to automatically generate screentone that adapts to the local tone. Experimental results demonstrate that our hybrid method effectively generates aesthetically pleasing image binarizations that encompass both line drawing and screentoning, closely resembling professional pen-and-ink illustrations. Also, being based on Gaussian filtering, our method is very fast and also easy to implement.

Keywords: Line drawing; screentoning; stippling; halftoning.

1. Introduction

Non-photorealistic rendering (NPR) aims to convey visual information in a simplified and/or stylized fashion. One of the most fundamental issues in NPR is line drawing, that is, extracting and depicting object boundaries in the scene with a relatively small number of lines, conveying salient structure information. Line drawing often serves as the foundation for many NPR applications [1,2,3,4,5,6,7]. While line drawing is good at portraying the structural elements of the scene, it is not equipped to deliver the tone, shading, and texture information on object surfaces. To account for these missing elements, artists often use *screentoning* methods such as halftoning [8,9,10], hatching [11,12,13], and stippling [14,15,16]. These computerized screentoning techniques convey tonal information using a set of simple black-and-

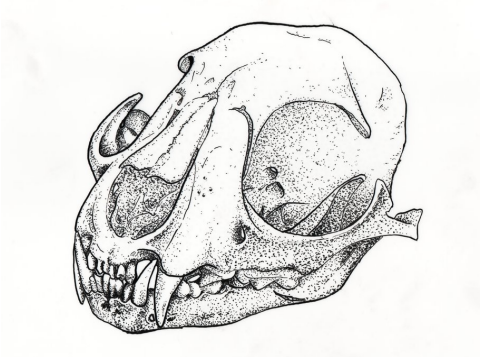


Fig. 1. A pen-and-ink illustration by an artist

white primitives such as dots and line segments. However, they tend to rely on complex algorithms that are difficult to implement. Also, the lack of contours may lead to unclear object structures in complex regions. To avoid this, they would have to deploy a large number of small primitives near object boundaries, while consuming more resources. Relying on a dense set of small primitives also has a downside of making the output look more like a dithered photograph instead of a stylistic illustration.

Instead of pure line drawing or screentoning, we argue that the best way to produce two-tone illustration of a scene is to combine the two methods into a unified framework. Such a framework would allow for both the object contours and the surface tone to be clearly depicted, as we often see in the professional illustrations drawn by artists (see Fig. 1)[a]. In this paper, we present a novel image binarization technique called *hybrid difference of Gaussians (HDoG)* that facilitates automatic generation of such a high quality illustration. It is a unified framework as described above, conveying both object contours and surface tone. Our method is simple, easy to implement, and very fast.

The proposed method, as the name suggests, builds on difference of Gaussians (DoG) [17]. DoG operator is generally known as edge detector, and thus should naturally take care of line drawing side of the task. Our contribution lies in how we use DoG to describe surface tone. We show that DoG introduces a screentoning effect in noisy environments, producing evenly distributed irregular-shaped primitives of a given Gaussian scale. We also propose a simple mechanism to adaptively distribute these screentone primitives to match the local tone. The resulting image is then combined with the DoG lines to complete the illustration. Experimental results show that our framework consistently generates quality black-and-white image stylizations. Moreover, the computational efficiency of DoG, along with its highly

[a]https://deviantart.com/wlbrndl1206/art/Scientific-Illustration-of-Canadian-Lynx-skull-411864668

parallelizable nature, ensures real-time performance even at HD image resolutions.

2. Related Work

2.1. *Line Drawing*

Line drawing has been in the core of 2D NPR, and a variety of methods have been proposed for providing a structure-conveying abstraction of an image comprised of clear and coherent strokes. For example, Canny edge detector [18] is often used for line drawing purposes, which uses Gaussian blurred image gradient to extract thin edge lines of one pixel width. Gooch et al. [29] introduced multiscale DoG operator to detect salient lines from human facial photographs. Unlike Canny's, DoG generates lines of nonuniform width reflecting edge strength, which helps improve structure recognition. Winnemöller et al. [4] subsequently used DoG operator in conjunction with bilateral filter [20] for video abstraction. Kang et al. [21] proposed an anisotropic DoG operator called flow-based Difference of Gaussians (FDoG) for improved line coherence and reduced noise. Winnemöller then developed an extended DoG operator (XDoG) [5] to generate a wider variety of grayscale image stylization effects beyond line drawing.

In recent years, learning-based approaches have been used to improve semantic accuracy of line drawing. Simo-Serra et al. [22] used convolutional neural network (CNN) for cleaning up rough sketch. Similarly, Li et al. [23] and Kim et al. [24] presented CNN-based methods for extracting lines that are perceptually more meaningful. These approaches, however, require lengthy processes of data collection and machine learning.

Another way to generate lines is via segmentation. There are a variety of approaches to conducting data segmentation, including mean-shift [1,31,32], region growing [33], active contours [34], spectral/wavelet analysis [35,37,38], Markov chain [36], etc. The output of such segmentation may be used to provide a baseline for higher-level image stylizations [1,31,32]. Also related is a topic of vectorization of line drawings [39,40], where the focus is on extracting line primitives devoid of noise from a binary image and converting them into scale-independent vector graphics which may then be further edited, stored, and digitally reproduced.

2.2. *Screentoning*

Screentoning refers to representing shades of grayscale luminance with a repeating pattern of black and white primitives. Computerized screentoning methods vary in the shape of primitives used, such as dots, circles, polygons, and general shapes. Salisbury et al. [11] presented an interactive pen-and-ink illustration system that uses a variety of predefined black-and-white hatching patterns. Similar screentones were later used for interactive generation of hatching illustration based on user-provided direction field [12].

Deussen et al. [14] proposed a stipple drawing method based on Lloyd's algorithm (i.e., construction of a centroidal Voronoi diagram) that uniformly distributes dots

over Voronoi cells. Secord [15] modified this algorithm to produce a weighted centroidal Voronoi diagram that protects image features better. Mould [25] developed a graph-based stippling method for improved protection of image features such as edges. On the other hand, Son et al. [16] used texture synthesis to generate directional stippling and hatching along computer-generated feature flow.

Halftoning, while similar to stippling in many ways, is mainly concerned with visual approximation rather than artistic stylization. Thus, its goal is often to minimize visual error against the source and therefore tends to use smaller dots than stippling. Notable algorithms include the one proposed by Ostromoukhov [8] based on error diffusion. Kopf et al. [9] used recursive Wang tiles for multiscale halftoning. Pang et al. [10] proposed a directional halftoning method that better preserves oriented features.

Kaplan and Bosch [30] developed an innovative screentoning method called *TSP Art* that uses connected line segments to represent grayscale tone. These line segments are formed by connecting neighboring stipple dots that are adaptively distributed to match local tone. Ostromoukhov and Hersch proposed screentoning methods that support arbitrary primitive shapes [26,27]. They define evolving screen dot shape by a description of customized shape contours associated with specific intensity levels, then perform shape blending to interpolate between the predefined shape contours at all other intensity levels.

Technically speaking, our screentoning method is neither *stippling* nor *hatching*. Instead, it uses irregular shaped primitives that are procedurally generated based on the local intensity variation and the noise present. Also, unlike aformentioned approaches, our method does not require any complex algorithm or procedure to meticulously organize the primitive distribution, and thus is easy to implement and highly efficient.

Another related topic is texture/screentone segmentation [41,42,43,44] that tackles the problem of detecting and separating high-frequency details such as texture or screentone from low-frequency structural elements. While these techniques are designed to *analyze* screentone or texture patterns that are already present in the input data, our proposed method is aiming to *synthesize* a screentone pattern that does not exist in the input data.

3. Our Method

Fig. 2 shows the overview of our method. For line drawing, we perform FDoG filtering [21] on Gaussian blurred input. For tone depiction, we introduce a modified DoG called *adaptive DoG (ADoG)* that adjusts DoG filter's sensitivity to noise based on the brightness level in the neighborhood, resulting in a screentoning effect in the filter response. The respective binary filter responses from FDoG and ADoG are then merged via Boolean AND operator to complete the final output.

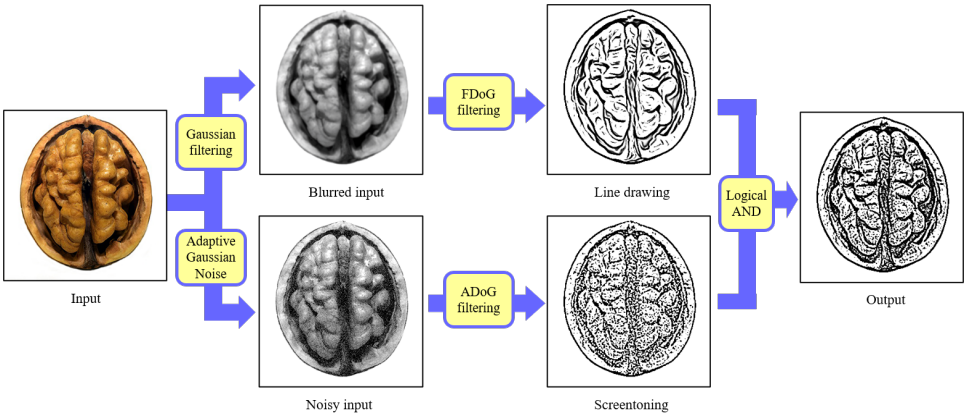


Fig. 2. Process overview

3.1. *Flow-based Difference of Gaussians*

FDoG [21] is an anisotropic DoG filter where the filter kernel is curved along the dominant edge direction in the local neighborhood. This data-dependent kernel shaping delivers enhanced coherence of detected lines as well as reduced spurious edges and noisy responses, making FDoG a powerful line drawing tool. FDoG requires a feature-preserving smooth orientation field called edge tangent flow (ETF). Edge tangent is a vector perpendicular to the dominant image gradient in the neighborhood. For ease of implementation, we employ Gaussian-smoothed structure tensor [28] to construct our ETF.

Given a grayscale image $I \in [0, 1]$, let (I_x, I_y) denote image gradient. The smoothed structure tensor S is then defined as

$$S = \begin{pmatrix} A & B \\ B & C \end{pmatrix} \tag{1}$$

where $A = G_\sigma * {I_x}^2$, $B = G_\sigma * I_x I_y$, and $C = G_\sigma * {I_y}^2$. The operator $*$ denotes convolution with 2-dimensional Gaussian kernel G_σ of scale σ. We set $\sigma = 5$ by default.

Now, ETF is constructed by taking the minor eigenvector of S. More specifically, the edge tangent direction θ at each pixel is obtained as

$$\theta = \arctan\left(\frac{-2B}{C - A + \sqrt{(C-A)^2 + 4B^2}}\right) \tag{2}$$

Fig. 3b shows an ETF obtained from Fig. 3a.

In the next step, we extract lines by performing anisotropic DoG filtering on image I with respect to ETF. For each pixel $\mathbf{x}$, we set a rectangular filter kernel deformed along the *tangent axis* (see Fig. 3c). Note that tangent axis is a *steamline*

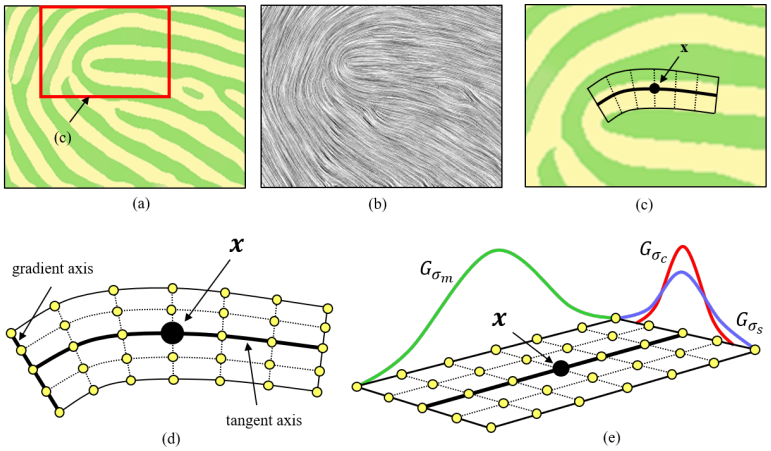


Fig. 3. FDoG: (a) Input (b) ETF (c) Anisotropic kernel at $\mathbf{x}$ (d) Kernel magnified (e) DoG responses accumulated at $\mathbf{x}$

of ETF centered at $\mathbf{x}$. We then apply 1-dimensional DoG filter along *gradient axes* perpendicular to the tangent axis (see Fig. 3d). The DoG operator is defined as:

$$DoG(\mathbf{x}) = G_{\sigma_c}(\mathbf{x}) - \rho \cdot G_{\sigma_s}(\mathbf{x}) \tag{3}$$

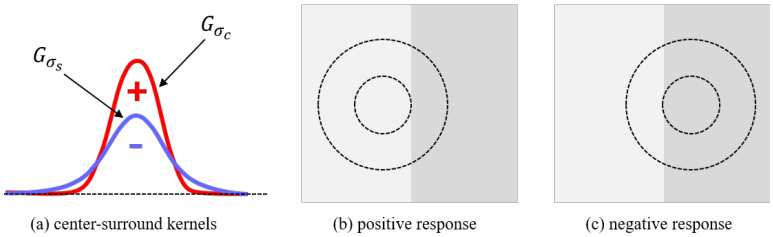
where G_{σ_c} and G_{σ_s} denote center and surrounding Gaussian kernels, respectively (see Fig. 3e). The default scale of the center kernel is set as $\sigma_c = 1.0$, and the surrounding kernel σ_s as 1.6 times the size of σ_c. Parameter ρ controls sensitivity to brightness contrast in the neighborhood. We let it range in $[0.97, 1.0]$ with the default value of $\rho = 0.99$. The higher the value, the noisier the response.

Finally, the individual DoG outputs within the kernel are accumulated along the tangent axis using another Gaussian function G_{σ_m} as blending weight, to form FDoG filter response at $\mathbf{x}$ (see Fig. 3e), which is then converted to a black-and-white line map via binary thresholding. The default value of σ_m is 3.0.

3.2. *Adaptive Difference of Gaussians*

As described in Eq. 3, DoG detects edges using the discrepancy between center-surrounding Gaussian kernels. DoG operator elicits either a positive or negative response where there is a brightness contrast (see Fig. 4). Near an edge, these responses form a line. In a noisy region, an irregular pattern emerges out of rapidly fluctuating positive and negative responses. Moreover, in Eq. 3, the Gaussian scale σ_c controls the size of these irregular primitives, and ρ controls how strongly DoG responds to brightness contrast. We exploit these properties to construct a DoG-based tone descriptor.

An ideal screentone descriptor should evenly distribute primitives where the density of distribution is inversely proportional to the brightness of local tone. Thus,



(a) center-surround kernels (b) positive response (c) negative response

Fig. 4. DoG operator

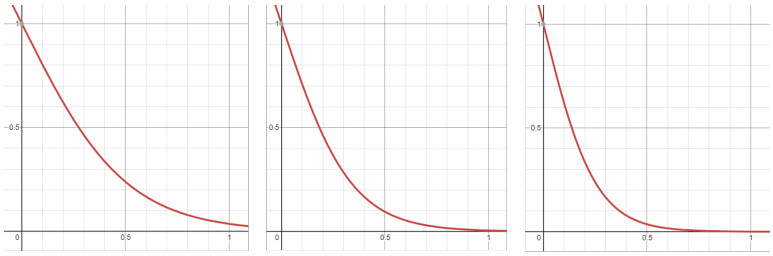
we modify Eq. 3 so that DoG response would produce higher frequency patterns in a dark region. This means the contrast sensitivity parameter ρ should be a function of pixel location $\mathbf{x}$:

$$ADoG(\mathbf{x}) = G_{\sigma_c}(\mathbf{x}) - \rho(\mathbf{x})G_{\sigma_s}(\mathbf{x}) \tag{4}$$

with $\rho(\mathbf{x})$ defined as:

$$\rho(\mathbf{x}) = \tau + (1-\tau)(1 - tanh(s \cdot I(\mathbf{x}))) \tag{5}$$

where τ is the minimum sensitivity ($= 0.99$ by default). The idea is to let the contrast sensitivity range within $[\tau, 1]$ and be inversely proportional to the local tone $I(\mathbf{x})$, which means more primitives will be generated in dark regions. Hyperpolic tangent function, $tanh$, is used to control the density of primitive distribution and thus the overall tone of the illustration. The control parameter s further helps in this regard. Fig. 5 and Fig. 6 illustrate the effect of s. The default value of s is set to 2. Since the contrast sensitivity is now adaptively defined, we call this scheme *adaptive difference of Gaussians (ADoG)*.



(a) s = 2.0 (b) s = 3.0 (c) s = 4.0

Fig. 5. Plots of $1 - tanh(s \cdot x)$

Fig. 7 shows comparison with regular DoG. In case of regular (non-adaptive) DoG, the same degree of contrast sensitivity is used everywhere and thus the number of primitives does not necessarily increase in dark regions (see Fig. 7b). With ADoG, however, the density of primitive distribution adaptively changes with the level

of brightness in the neighborhood, and thus better represents the dark tone (see Fig. 7c).

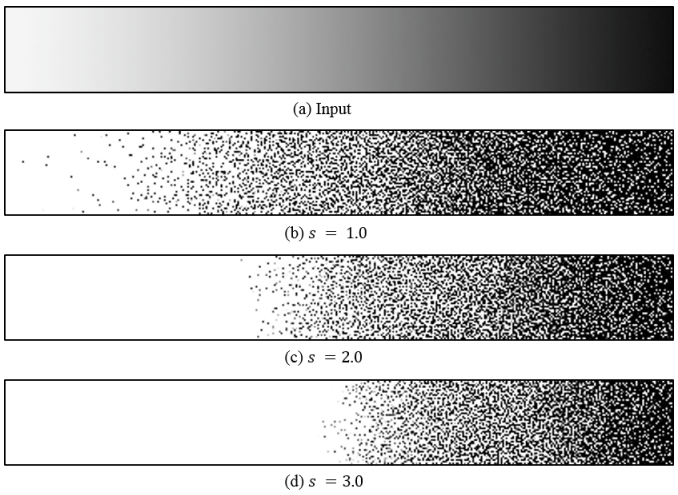


Fig. 6. Effect of scale parameter on tone variation

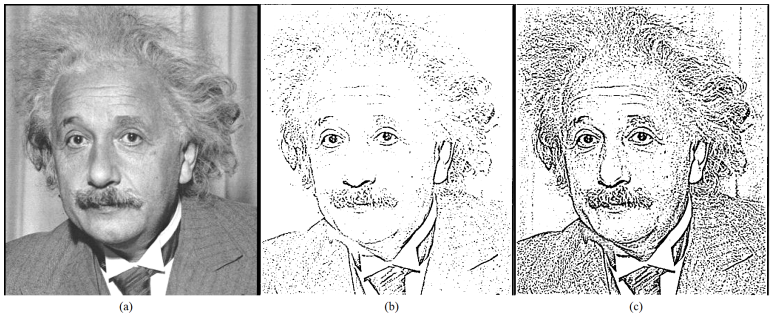


Fig. 7. Regular DoG vs. Adaptive DoG: (a) Input (b) DoG (c) ADoG

The effect of ADoG may be further amplified by adding Gaussian noise to I. Again, we must ensure that the amount of noise added is adaptively determined by the local tone. Specifically, we randomly sample noise from a Gaussian function G_σ, where the scale σ is a function of pixel location $\mathbf{x}$:

$$\sigma(\mathbf{x}) = c \cdot (1 - tanh(s \cdot I(\mathbf{x}))) \tag{6}$$

which uses the same tone-dependent weight as in Eq. 5. Therefore, the amount of intensity perturbation will be inversely proportional to the local tone $I(\mathbf{x})$, meaning

the darker the region, the denser the primitive distribution. The noise scale factor c is typically set to 0.01. This formula helps generate more high-frequency intensity fluctuations in darker regions. Fig. 8 illustrates that with the presence of adaptive noise, the dark regions with little intensity fluctuations are now represented better with increased screentone density.

Fig. 8. Effect of adaptive noise: (a) Input (b) Without noise (c) With noise

Fig. 9 shows that the rotationally symmetric nature of center-surrounding Gaussians (as shown in Fig. 4) ensures regular spacing of irregular primitives even in a highly noisy environment, where the size of spacing is determined by the Gaussian scale. This regular spacing of primitives is key to why ADoG output looks stylistic rather than noisy.

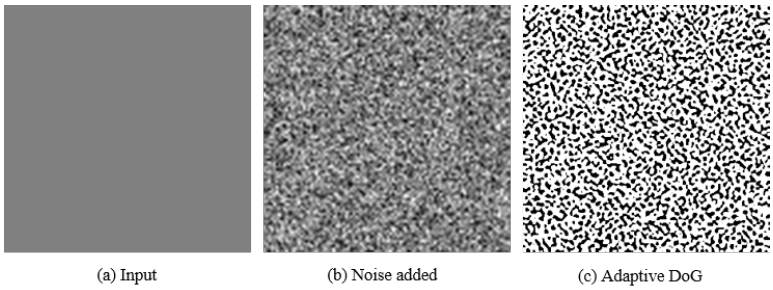


Fig. 9. Primitive spacing by ADoG: (a) Dark region (b) Gaussian noise added (c) ADoG

3.3. *Hybrid Difference of Gaussians*

With FDoG and ADoG output images generated and binarized, we combine them into the final image as follows:

$$HDoG(\mathbf{x}) = FDoG(\mathbf{x}) \wedge ADoG(\mathbf{x}) \tag{7}$$

where we use pixel-wise logical *AND* operator ($\wedge$). This is to preserve all the black primitives present in the two images, considering they are both binarized into either

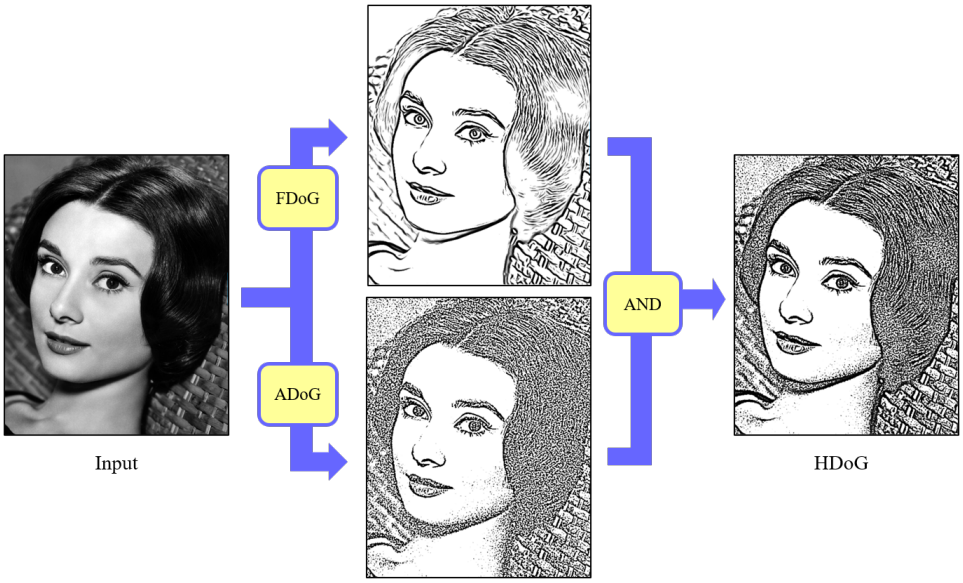


Fig. 10. The operation of HDoG

0 (black) or 1 (white). Note that this would be the same as applying logical *OR* operator ($\vee$) on the inverted images (where 1 would represent lines and primitives), then invert the result back. That is,

$$\sim(\sim FDoG \ \vee \sim ADoG) = FDoG \wedge ADoG \tag{8}$$

which is a well-known De Morgan's law ($\sim$ denotes logical negation). Fig. 10 illustrates this operation, which we call *hybrid difference of Gaussians (HDoG)*. Note that HDoG output conveys both structure and tone information well, while either ADoG or FDoG comes up short in this regard.

When creating black-and-white art, artists often seek to find the optimal scale for the primitives whether they are dots, polygons or arbitrary shapes. If primitives are too large, it becomes harder to protect important structures and features in the scene. If primitives get too small, the output may end up looking too realistic to pass as non-photorealisic art (e.g., like a dithered photograph on a newspaper). Thus, it is prudent to avoid using primitives that are too big or too small. One challenge, though, is that the dark tone is harder to represent with medium or large sized primitives. This is because the bigger the primitives, the larger the spacing between them, and our framework is no exception. To address this, we propose a simple extension to Eq. 7:

$$HDoG = FDoG \wedge ADoG_s \wedge ADoG_{s'} \tag{9}$$

where s and s' denote the scale parameter in Eq. 5. That is, we run a second ADoG filter with a larger scale factor s', in order to generate additional screentone in some

of the darkest regions without affecting brighter ones (see Fig. 6). We empirically set $s' = 4s$. Fig. 11 illustrates the effect of the second ADoG, with which the darkest tone in the picture is represented more effectively (see Fig. 11c).

(a) **(b)** **(c)**

Fig. 11. Effect of second ADoG: (a) Input (b) Without second ADoG (c) With second ADoG

4. Results

Fig. 13 presents HDoG results from the test images of Fig. 12. The test results show that our HDoG filter converts an arbitrary photograph into an artistic stylization composed of line drawing and computer generated screentone, which closely resembles human art (such as Fig. 1). In particular, the combination of FDoG and ADoG complements each other and effectively conveys both the structual elements and the shading information, making the resulting art look more complete. Note that either FDoG or ADoG, when used alone, would have been lacking in this regard. Our method works well on a wide variety of images of different subjects, such as humans, animals, and inanimate objects.

Fig. 14 shows how the results of ADoG and FDoG are combined to form HDoG binarizations with some additional sample images. ADoG nicely captures a wide range of shades that exist in the scene, which FDoG is not equipped to do. See how ADoG and FDoG respond differently on the inside of the shell (middle row). FDoG however does a good job of capturing structure and object boundaries with clear and coherent lines. It should be noted that ADoG itself is an edge detector as it was derived from DoG operator. However, the edge detection capability of ADoG is not as strong as that of FDoG. See for an example the flower image in the top row, where the boundaries of individual petals are clearly captured with FDoG, but not as much with ADoG. Then the combined HDoG result on the right delivers the best of both worlds.

Fig. 15 provides comparison with some of the existing image binarization methods. The second column shows results of a multi-scale stippling technique developed by Kopf et al [9]. This technique uses a set of carefully planned recursive Wang tiles and thus obtains faithful reproduction of local tone as well as rigorously spaced dot primitives everywhere. One limitation of such pure stippling is that it is not

Fig. 12. Test images

Fig. 13. Results

well-equipped to clearly depict the structural elements, such as surface contours and object boundaries. It typically requires a large number of primitives near edges to do so, resulting in an output looking more like a dithered photograph than an artistic stylization.

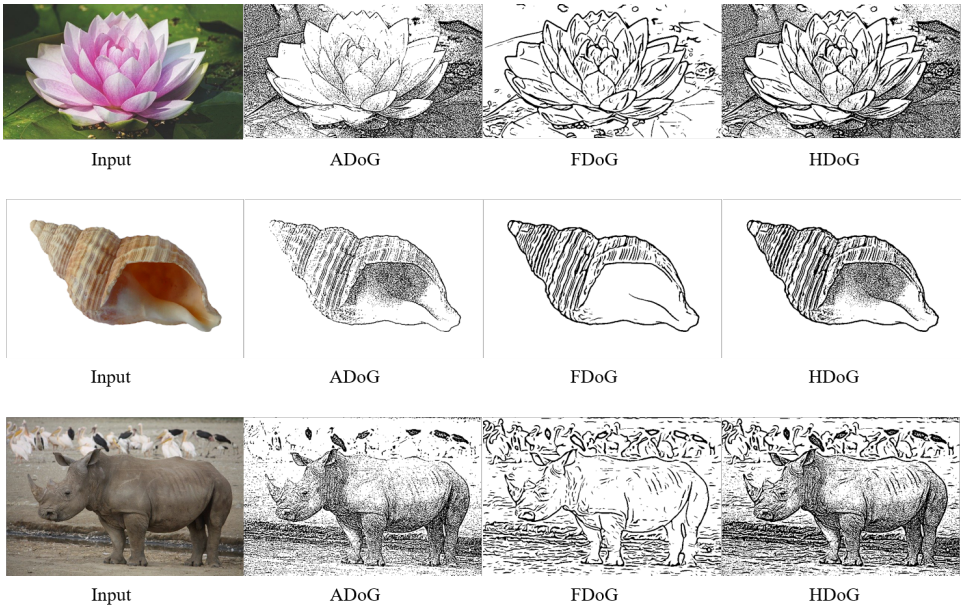


Fig. 14. Example ADoG, FDoG, and HDoG results

The third column shows results of *TSP Art* algorithm by Kaplan et al [30]. This algorithm generates an artistic screentone pattern by connecting the dots that are carefully distributed to match local tone. It typically consumes a less number of primitives than pure stippling and thus leaves, somewhat deliberately, many of the structural elements in the scene vague. For example, in the second row of TSP Art, the shape of Audrey Hepburn's nose is quite unclear.

On the other hand, our algorithm (4th column) does not go through a lengthy procedure to meticulously place each primitive in the right place. Yet, as demonstrated by the experimental results, our method is quite effective in capturing and delivering both *structural and tonal* elements that are perceptually important, not unlike hand-drawn professional illustrations.

Operation	**Time complexity**
ETF construction (Eq. 1)	$O(N)$
FDoG filtering (Eq. 3)	$O(N)$
ADoG filtering (Eq. 4)	$O(N)$
Adaptive Gaussian noise (Eq. 6)	$O(N)$
HDoG filtering (Eq. 7)	$O(N)$
Overall	$O(N)$

Table 1. Time complexity of each step of our algorithm.

Fig. 15. Comparison with existing binarization methods

Table 1 shows performance analysis of our algorithm. As shown in the table, the overall time complexity of our algorithm is $O(N)$, where N denotes the number of pixels in the source image. This is due to the fact that every major step in our algorithm involves Gaussian filtering, which is a well-known *linear* operator. Even a 2-dimensional Gaussian, such as the ones used in Eq. 1 and Eq. 4, is mathematically separable into two 1-dimensional Gaussians along x and y axes, respectively, without sacrificing accuracy. Other steps, such as the logical *AND* ($\wedge$) in Eq. 7 and the subtraction ($-$) of center-surrounding Gaussians in Eq. 4, are all pixel-wise local operators. Thus, the entire process of our algorithm, both the Gaussian-based components and the local operations, is highly parallelizable and GPU-friendly. Not

surprisingly, the GPU implementation of HDoG filter runs in real time, processing images of HD resolutions (of over $1M$ pixels) at 1000+ fps on a system equipped with Intel Core i7-7700HQ and Nvidia GeForce GTX 1050.

5. Conclusions

We have presented a simple yet effective binary image stylization method based on Gaussian filtering. We proposed a novel hybrid difference of Gaussians (HDoG) framework that relies on DoG operator for both line drawing and tone description. In particular, our adaptive DoG filter is designed to quickly generate screentone that captures local tone and shading information in the image. The generated line map and tone map are then combined to produce a high quality binary stylization that resembles professional pen-and-ink drawing. Our algorithm is highly efficient and easily implementable on GPU.

Possible future research directions include exploring different styles of screentone pattern that can be generated within the current framework, and finding ways to parameterize the control of the primitive shapes. Extending current framework to video while preserving temporal coherence between frames would be another interesting yet challenging task.

References

1. D. DeCarlo and A. Santella, "Stylization and Abstraction of Photographs", *ACM Transactions on Graphics*, 21(3), p. 769-776, 2002.
2. J. Hays and I. Essa, "Image and video based painterly animation", *Proc. Non-photorealistic animation and rendering*, 21(3), p. 113-120, 2004.
3. H. Kang, S. Lee, and C. Chui, "Flow-Based Image Abstraction", *IEEE Transactions on Visualization and Computer Graphics*, 15(1), p. 62-76, 2009.
4. H. Winnemöller, S. Olsen, and B. Gooch, "Real-time Video Abstraction", *ACM Transactions on Graphics*, 25(3), p. 1221–1226, 2006.
5. H. Winnemöller, "XDoG: Advanced Image Stylization with eXtended Difference-of-Gaussians", *Proc. Non-Photorealistic Animation and Rendering*, 21(3), p. 147–156, 2011.
6. T. Lindemeier, S. Pirk, and O. Deussen, "Image stylization with a painting machine using semantic hints", *Computers and Graphics*, 37(5), p. 293–301, 2013.
7. J. Lu, C. Barnes, C. Wan, P. Asente, R. Mech, and A. Finkelstein, "DecoBrush: drawing structured decorative patterns by example", *ACM Transactions on Graphics*, 33(3), 90:1-90:9, 2014.
8. V. Ostromoukhov, "A Simple and Efficient Error-diffusion Algorithm", *Proc. ACM SIGGRAPH*, 21(3), p. 567–572, 2001.
9. J. Kopf, D Cohen-Or, O. Deussen, and D. Lischinski, "Recursive Wang Tiles for Real-Time Blue Noise", *ACM Transactions on Graphics*, 25(3), p. 509-518, 2006.
10. W. Pang, Y. Qu, T. Wong, D. Cohen-Or, and P. Heng, "Structure-Aware Halftoning", *ACM Transactions on Graphics*, 27(3), 89:1-89:8, 2008.
11. M. Salisbury, S. Anderson, R. Barzel, and D. Salesin, "Interactive pen-and-ink illustration", *Proc. ACM SIGGRAPH*, 21(3), p. 101-108, 1994.
12. M. Salisbury, M. Wong, J. Hughes, and D. Salesin, "Orientable textures for image-based pen-and-ink illustration", *Proc. ACM SIGGRAPH*, 21(3), p. 401-406, 1997.

13. T. Umenhoffer, L. Szecsi, and L. Szirmay–Kalos, "Hatching for Motion Picture Production", *Computer Graphics Forum*, 30(2), p. 533-542, 2011.
14. O. Deussen, H. Stefan, V. Cornelius, and S. Thomas, "Floating Points: A Method for Computing Stipple Drawings", *Computer Graphics Forum*, 19(3), p. 41-50, 2000.
15. A. Secord, "Weighted Voronoi Stippling", *Proc. Non-photorealistic Animation and Rendering*, p. 37–43, 2002.
16. M. Son, Y. Lee, H. Kang, and S. Lee, "Structure grid for directional stippling", *Graphical Models*, 73(3), p. 74 - 87, 2011.
17. D. Marr and E. Hildreth, "Theory of edge detection", *Proc. Royal Society of London B: Biological Sciences*, 207(1167), p. 187–217, 1980.
18. J. Canny, "A Computational Approach to Edge Detection", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 8(6), p. 679-698, 1986.
19. D. Comaniciu and P. Meer, "Mean shift: A robust approach toward feature space analysis", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24(5), p. 603-619, 2002.
20. C. Tomasi and R. Manduchi, "Bilateral filtering for gray and color images", *Proc. International Conference on Computer Vision*, p. 839–846, 1998.
21. H. Kang, S. Lee, and C. Chui, "Coherent Line Drawing", *Proc. Non-photorealistic Animation and Rendering*, p. 43–50, 2007.
22. E. Simo-Serra, S. Iizuka, K. Sasaki, and H. Ishikawa, "Learning to Simplify: Fully Convolutional Networks for Rough Sketch Cleanup", *ACM Transactions on Graphics*, 35(4), 121:1–121:11, 2016.
23. C. Li, X. Liu, and T. Wong, "Deep Extraction of Manga Structural Lines", *ACM Transactions on Graphics*, 36(4), 117:1–117:12, 2017.
24. B. Kim, O. Wang, A. Oztireli, and M. Gross, "Semantic Segmentation for Line Drawing Vectorization Using Neural Networks", *Computer Graphics Forum*, 37(2), p. 329-338, 2018.
25. D. Mould, "Stipple placement using distance in a weighted graph", *Proc. Computational Aesthetics*, p. 45–52, 2007.
26. V. Ostromoukhov and R. Hersch, "Artistic Screening", *Proc. ACM SIGGRAPH*, p. 219–228, 1995.
27. V. Ostromoukhov and R. Hersch, "Multi-color and artistic dithering", *Proc. ACM SIGGRAPH*, p. 101–108, 1999.
28. J. Kyprianidis and H. Kang, "Image and Video Abstraction by Coherence-Enhancing Filtering", *Computer Graphics Forum*, 30(2), p. 593–602, 2011.
29. B. Gooch, E. Reinhard, A. Gooch, "Human facial illustrations", *ACM Transactions on Graphics*, 23(1), p. 27-44, 2004.
30. C. Kaplan, R. Bosch, "TSP Art", *Proc. Bridges: Mathematical Connections in Art, Music and Science*, p. 310-308, 2005.
31. J. Collomosse, D. Rowntree, P. Hall, "Stroke Surfaces: Temporally Coherent Artistic Animations from Video", *IEEE Transactions on Visualization and Computer Graphics*, 11(5), p. 540–549, 2005.
32. J. Wang, Y. Xu, H. Shum, M. Cohen, "Video Tooning", *ACM Transactions on Graphics*, 23(3), p. 574–583, 2004.
33. N. Shrivastava, J. Bharti, "Automatic Seeded Region Growing Image Segmentation for Medical Image Segmentation: A Brief Review", *International Journal of Image and Graphics*, 20(3), 2020.
34. W. Wang, C. Chung, "Image Segmentation with Complementary Use of Edge and Region Information", *International Journal of Image and Graphics*, 11(4), p. 549-570, 2011.

35. E. Regentova, D. Yao, S. Latifi, J. Zheng, "Image Segmentation using Ncut in the Wavelet Domain", *International Journal of Image and Graphics*, 6(4), p. 569-582, 2006.
36. Atiampo, Armand Kodjo and Loum, Georges Laussane, "Unsupervised Image Segmentation with Pairwise Markov Chains Based on Nonparametric Estimation of Copula Using Orthogonal Polynomials", *International Journal of Image and Graphics*, 16(4), p. 1-15, 2016.
37. A. Ouahabi, C. Depollier, L. Simon, D. Koume, "Spectrum Estimation from Randomly Sampled Velocity Data", *IEEE Transactions on Instrumentation and Measurement*, 47(4), p. 1005-1012, 1998.
38. A. Ouahabi, "Review of wavelet denosing in medical imaging", *Proc. IEEE international workshop on systems, signal processing and their applications*, p. 19-26, 2013.
39. X. Hilaire, K. Tombre, "Robust and accurate vectorization of line drawings", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 28(6), p. 890-904, 2006.
40. M. Alvarez, M. Algorri, "Vectorization and Line Detection for Automatic Image Recognition", *International Journal of Image and Graphics*, 11(3), p. 439-470, 2011.
41. C. Yao, S. Hung, G. Li, I. Chen, R. Adhitya Y. Lai, "Manga Vectorization and Manipulation with Procedural Simple Screentone", *IEEE Transactions on Visualization and Computer Graphics*, 23(2), p. 1070-1084, 2017.
42. K. Ito, Y. Matsui, T. Yamasaki, K. Aizawa, "Separation of Manga Line Drawings and Screentones", *Proc. Eurographics - Short Papers*, p. 73-76, 2015.
43. D. Meriem, O. Abdeldjalil, B. Hadj, B. Adrian, K. Denis, "Discrete wavelet for multifractal texture classification: application to medical ultrasound imaging", *Proc. IEEE International Conference on Image Processing*, p. 637-640, 2010.
44. A. Ouahabi, "Multifractal analysis for texture characterization: A new approach based on DWT", *Proc. International Conference on Information Science, Signal Processing and their Applications*, p. 698-703, 2010.

Photo and Bibliography

Henry Kang is an Associate Professor of Computer Science at the University of Missouri – St. Louis, USA. He received his M.S. and Ph.D. in computer science from the Korea Advanced Institute of Science and Technology (KAIST) in 1996 and 2002, respectively. His current research interests include computer graphics, data visualization, image/video processing, and computer animation.

Ioannis Stamoulis graduated the International Baccalaureate Diploma Program at Anatolia College, Greece, with the second highest overall grade in 2019. Since October 2019, he is an undergraduate student at the University of Oxford, reading for a degree in Mathematics and Computer Science under the Department of Computer Science.