Eurographics Symposium on Rendering (2022)
A. Ghosh and L.-Y. Wei (Editors)

HedcutDrawings: Rendering hedcut style portraits

K. Pena-Pena[1] and G. R. Arce[1]

[1]University of Delaware, Newark, DE 19716, USA

Abstract

Stippling illustrations of CEOs, authors, and world leaders have become an iconic style. Dot after dot is meticulously placed by professional artists to complete a hedcut, being an extremely time-consuming and painstaking task. The automatic generation of hedcuts by a computer is not simple since the understanding of the structure of faces and binary rendering of illustrations must be captured by an algorithm. Current challenges relate to the shape and placement of the dots without generating unwanted regularity artifacts. Recent neural style transfer techniques successfully separate the style from the content information of an image. However, such approach, as it is, is not suitable for stippling rendering since its output suffers from spillover artifacts and the placement of dots is arbitrary. The lack of aligned training data pairs also constraints the use of other deep-learning-based techniques. To address these challenges, we propose a new neural-based style transfer algorithm that uses side information to impose additional constraints on the direction of the dots. Experimental results show significant improvement in rendering hedcuts.

CCS Concepts

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

1. Introduction

Hedcut[†] is a form of stippling illustrations in which the placement of dots is along linear paths. Stippling illustrations are widely used to faithfully represent details on objects' surfaces when printed in black and white. The Wall Street Journal became a pioneer in the extensive use of hedcut illustrations in their articles as the one shown in Figure 1a. The manual process for the elaboration of high-quality stippling images is time-consuming, laborious, and requires artistic training. For this reason, there is an interest in developing algorithms that automatically render this form of style. Several non-photorealistic rendering (NPR) techniques have been already proposed for stippling drawings [RC12].

When rendering stippling drawings, the main challenge is to capture the long experience of stipple artists about the placement and shape of the dots. Early proposed methods are based on centroidal Voronoi diagrams [BSD09, CYC*12, DGBOD12]. However, these methods do not completely represent stippling's artistic style since unwanted regularity artifacts are generated. More recent efforts have tried to improve on these unwanted artifacts, using distribution functions [AML10, SHS02], placement of dots based on examples [KMI*09], and awareness of the structure [LM11]. Only a few works have focused on hedcut stipplings [KSL*08, KWME10, SLKL11a]. Among them, the method proposed by Son et al [SLKL11a], considerably improves on previously proposed NPR techniques. In this method, a structure grid based on a feature vector field of the input image is built. The dots are placed at the intersections of the grid lines and shaped to match the tone map of the image. Even though the results are pleasant, it is easy to tell that it is not hand-made because of the computerized look of the image, as observed in Figure 1b. More information about the state of the art of NPR techniques for digital stippling can be found in [MARI17].

Regarding learning-based techniques, a neural style transfer (NST) algorithm using convolutional neural networks was proposed by Gatys et al. [GEB16]. This technique successfully separates the style and the content information of an image, being able to render the content of an image into many different styles. The content and style feature representations are obtained from the feature maps of a convolutional neural network trained for classification, such as VGG-19, [SZ14]. Nonetheless, NST is limited in the faithfulness of the results for styles such as oil paint, pencil, or watercolors [SID17]. In the case of stippling, neural style transfer produces results with unwanted regularity artifacts. The placement of the dots, however, is completely arbitrary. Hence, NTS is not suitable for the rendering of hedcuts in which the direction of the dots is a key component. Furthermore, hedcuts drawings are not only composed by stipple dots, but also by hatching which is a technique used to represent shading effects with parallel lines. For this reason, spillover artifacts are generated by the NST algorithm

[†] The term "hedcut" comes from a newsroom abbreviation for "headline cut".

DOI: 10.2312/sr.20221160

(a) *Hand-made Hedcut* **(b)** *NPR*

(c) *NST* **(d)** *Proposed method*

Figure 1: *Comparison of (a) hand-made hedcut [Gla], (b) directional stippling [SLKL11a, SLKL11b], (c) NST [GEB16], and (d) our method.*

when using a hedcut like the one shown in Figure 1a as the reference style. This is the case of the image depicted in Figure 1c.

Many learning-based methods that have been proposed for image translation [IZZE17, ZZP*17, WLZ*18] rely on having aligned pair data, which is difficult to obtain in the case of styles such as hedcuts. More recent methods for image-to-image translation that do not depend on paired data, learn the mapping from one domain to another using two conditional adversarial networks with cycle-consistency regularization [ZPIE17]. The WSJ developed an AI tool for the generation of hedcuts [(WS19, Bol19] by designing a model that combines two of these approaches: pix2pix [IZZE17] and CycleGAN [ZPIE17], and training this model on a dataset of 2000 labeled samples. Another technique that focuses on photo-to-pencil translation was introduced in [LFH*19], in which training data pairs are created by extracting clean outlines and tonal shadings from original pencil drawings. Stippling shading is considered as one of the drawing styles generated by this method.

In contrast, the method described in this paper considers neural style transfer as the baseline to capture the features of the style, overcoming unwanted regularity artifacts generated by NPR techniques and requiring only one sample image. It also introduces additional constraints into the objective function that depend on side information obtained from the content image, introducing knowledge about its structure. In the case of hedcut stippling, the placement of the dots along linear paths that depend on the structure of the image is a key component within the drawings that is not captured by traditional neural style transfer [GEB16], in which dots are placed arbitrarily. By adding additional terms into the objective function that depend on a structured grid, the direction of the dots is enforced to follow the linear paths of the grid, achieving more faithful hedcut stippling results and demonstrating the effectiveness of our approach.

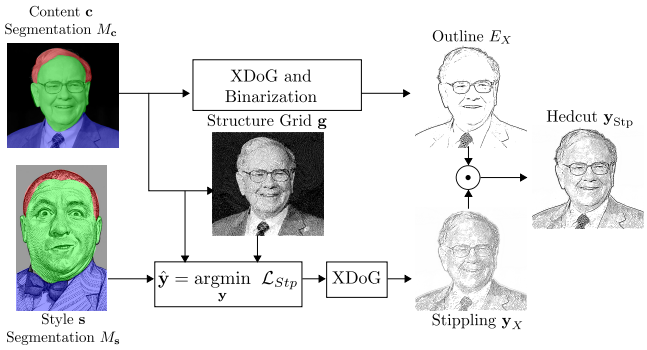


Figure 2: *General scheme for the rendering of hedcut style. A content image* **c** *and a style image* **s** *[Gla] are used for the generation of the structure grid* **g** *and the segmentation masks* $M_{\mathbf{c}}$ *and* $M_{\mathbf{s}}$*, which are needed for the minimization of the neural-based loss function* $\mathcal{L}_{Stp}$*. The obtained stylized image* $\hat{\mathbf{y}}$ *is post-processed by the XDoG algorithm [WKO12] to obtain the stippling* $\mathbf{y}_X$*. The Hadamard product* $\odot$ *between the stippling* $\mathbf{y}_X$ *and the outline* E_X *generates the hedcut stylized image* $\mathbf{y}_{Stp}$*.*

2. Method

Our algorithm takes as inputs: a content image **c** which is generally a portrait and a style image **s** which is a hand-made stippling to generate a structure grid image **g** from the content image and semantic segmentation masks ($M_{\mathbf{c}}$ and $M_{\mathbf{s}}$) from the content and style images as it is shown in Figure 2. Next, all these images are used as input to the neural-based algorithm, in which the stylized image $\hat{\mathbf{y}}$ is obtained by minimizing the following objective function:

$$\mathcal{L}_{Stp} = \alpha\mathcal{L}_{s+} + \beta\mathcal{L}_{c+} + \omega\mathcal{L}_{G_s} + \mu\mathcal{L}_{G_c}, \tag{1}$$

where $\mathcal{L}_{s+}$ and $\mathcal{L}_{c+}$ are the augmented style and content loss. $\mathcal{L}_{G_c}$ and $\mathcal{L}_{G_s}$ are a content loss and a style loss based on the grid **g**. $\alpha, \beta, \omega, \mu$ are tuning parameters that balance the effect of each component within the loss function. The stylized image $\hat{\mathbf{y}}$ is post-processed and then multiplied by the outline of the content image to generate the final hedcut style image $\mathbf{y}_{\text{Stp}}$. Detail information about each step in the algorithm will be explained later in this section. The main goal of our method is to achieve a more realistic rendering of hand-made hedcut stippling drawings which have been identified to be challenging in the state of art [MARI17]. Our approach builds upon the method proposed by Gatys et al. [GEB16] by adding additional constraints depending on side information obtained from the content image. The main contributions of this work can be summarized as follows:

- A neural-based hedcut stippling generation process that de-

pends on a structure grid generated from the content image, providing not only a directional flow to the stipple dots but also keeping the quality and spacing of the dots.

- Spillover effects generated by traditional neural style transfer methods are alleviated by using semantic information from the content and the style image as well as correctly post-processing the output.

2.1. Background

Content Loss:
The content loss, also known as perceptual loss, measures the dissimilarity between high-level features of the content image $\mathbf{c}$ and the synthesized image $\mathbf{y}$. Let $F^l(\mathbf{x}) \in \mathbb{R}^{D_l \times N_l}$ be a matrix that stores the features of an image $\mathbf{x}$ obtained from the l-th convolutional layer of a pre-trained convolutional network and D_l be the number of feature maps. Each vectorized feature map is represented as $F_i^l(\mathbf{x}) \in \mathbb{R}^{N_l}$ where N_l is the width times the height of the feature map. Thus, the content loss is computed as:

$$\mathcal{L}_c = \sum_{l=1}^{L} \sigma_l ||F^l(\mathbf{c}) - F^l(\mathbf{y})||_F^2, \tag{2}$$

where L is the number of convolutional layers of the neural network, σ_l is a weighting parameter that configures the contribution of each layer to the content loss, and $||\cdot||_F$ is the Frobenius norm. In our implementation, the pre-trained VGG-19 [SZ14] network was used.

Style Loss:
The style loss measures the dissimilarity between the style image and the generated image $\mathbf{y}$ in a feature space that captures texture information [GEB16], like the correlation between feature maps in each layer of the neural network. For an image $\mathbf{x}$, these style features are given by the Gram matrix $G^l(\mathbf{x}) \in \mathbb{R}^{D_l \times D_l}$ whose entries G_{ij}^l are computed as the correlation

$$G_{ij}^l(\mathbf{x}) = \sum_{k=1}^{N_l} F_{i,k}^l(\mathbf{x}) F_{k,j}^l(\mathbf{x}), \tag{3}$$

where $F_{i,k}^l$ is the vectorized activation map of the i-th filter at position k of layer l. The style loss is then computed as:

$$\mathcal{L}_s = \sum_{l=1}^{L} \frac{\tau_l}{N_l^2} ||G^l(\mathbf{s}) - G^l(\mathbf{y})||_F^2, \tag{4}$$

where L is the number of convolutional layers of the neural network and τ_l is a weighting parameter that configures layer preferences.

Augmented Style Loss with Semantic Segmentation:
The Gram matrices in the style loss globally capture texture information within the image but they do not keep information about the spatial arrangement of the features, causing spillovers as it can be seen in Figure 1c. In [LPSB17], semantic information about the content image and style image was incorporated to the loss. Given the segmentation mask of the content and the style image ($M_\mathbf{c}$ and $M_\mathbf{s}$), let the augmented style loss for layer l be defined as:

$$\mathcal{L}_{s+}^l = \frac{1}{N_{l,c}^2} \sum_{p=1}^{P} ||G_p^l(\mathbf{s}) - G_p^l(\mathbf{y})||_F^2, \tag{5}$$

where $G_p^l(\mathbf{s}) = F^{l,p}(\mathbf{s}) F^{l,p}(\mathbf{s})^T$ is the Gram matrix of the style image. The vectorized feature maps at layer l are segmented based on label p, i.e. $F^{l,p}(\mathbf{s}) = F^l(\mathbf{s}) \odot M_\mathbf{s}^{l,p}$, where $M_\mathbf{s}^{l,p} \in \mathbb{R}^{N^l}$ denotes the vectorized binary segmentation mask corresponding to label p in layer l. For the remaining of this paper, $\odot$ represents the Hadamard product. In the same way, for the synthesized image $\mathbf{y}$, $F^{l,p}(\mathbf{y}) = F^l(\mathbf{y}) \odot M_\mathbf{c}^{l,p}$. The masks are down-sampled to match the resolution of the feature maps. The segmentation masks used for the results in this paper were manually generated. However, there are many works which are devoted to the semantic segmentation of faces [KML15, NMT*18] that could be used for their automatic generation. This augmented style loss with semantic segmentation, $\mathcal{L}_{s+}^l$, corresponds to the first term in the proposed objective function (Eq. 1).

2.2. Augmented Content Loss

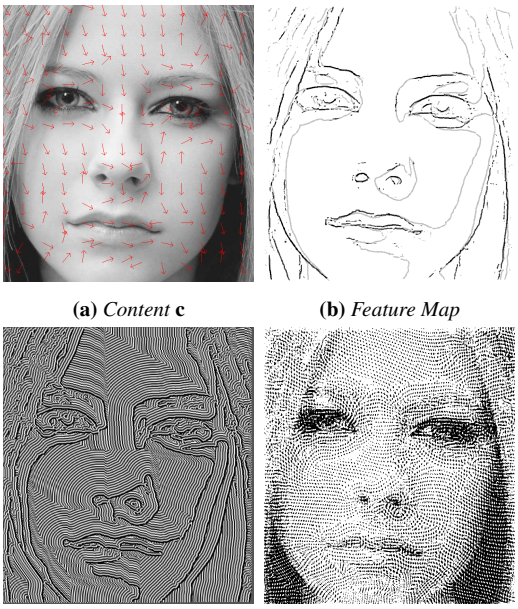
For analyzing the structure of hand-made stippling drawings, two different components can be identified: the outline that closely depends on the content image, and even parts of the drawing, which vary depending on the style and the artist. In traditional style transfer [GEB16], when the content of the images is selected to match features in lower layers of the network, much of the pixel information of the content image is preserved and the texture of the style image is barely present on the synthesized image. On the other hand, when matching the content information in higher layers a lot of the structure of the image like the edges are modified to match the style image. Considering these two facts, we proposed as the second term of our objective function, Eq. 1, an augmented content loss that is weighted by an edge mask, matching the content of the images in lower layers of the network to keep the outline information but allowing the even regions to change mostly depending on the style loss. The augmented content loss is computed as

$$\mathcal{L}_{c+} = \sum_{l=1}^{L} \sigma_l ||(F^l(\mathbf{c}) - F^l(\mathbf{y})) \odot E(\mathbf{c})||_F^2, \tag{6}$$

where $E(\mathbf{c}) \in \mathbb{R}^{N^l}$ is the vectorized edge priority mask computed using the algorithm proposed in [DZ13]. The edges of the content image are identified with higher values within the mask. In this way, the error between the content features of the edges are considered to be more important for the contribution of the content loss. The edge mask is also down-sampled to match the resolution of the feature maps.

2.3. Grid Content and Style Loss

Given that the Gram matrix does not allow direct control over the spatial arrangement of the style features and the direction of the dots is key when rendering hedcut stippling, we imposed additional constraints in the proposed objective function, Eq. 1, that depend on a structure grid as the one shown in Figure 3d. Hence, we propose the computation of a grid loss defined as the linear combination of



(a) *Content* **c** **(b)** *Feature Map*

(c) *JFA [RT06]* **(d)** *Structure grid* **g**

Figure 3: *Grid construction. A feature map (b) is first extracted from (a) to generate a feature vector field (red arrows in (a)). Isophote curves are the gray curves in (b). The stripe pattern following tangential directions is initialized by using JFA (c). The final structure grid* **g** *in (d) is obtained by binarizing the combination of tangential and orthogonal stripe patterns.*

the traditional content loss and style loss between the grid image, **g**, and the synthesized image, **y**:

$$\mathcal{L}_G = \omega\mathcal{L}_{G_s} + \mu\mathcal{L}_{G_c}, \tag{7}$$

$$\mathcal{L}_{G_s} = \sum_{l=1}^{L} \sigma_l ||F^l(\mathbf{g}) - F^l(\mathbf{y})||_F^2 \tag{8}$$

$$\mathcal{L}_{G_c} = \frac{1}{N_l^2} \sum_{l=1}^{L} \tau_l ||G^l(\mathbf{g}) - G^l(\mathbf{y})||_F^2. \tag{9}$$

Such that, the grid loss enforces the appearance of the synthesized image to be similar to the grid but not equal as it is in [SLKL11a]. $\mathcal{L}_{G_s}$ and $\mathcal{L}_{G_c}$ correspond the last two terms in the proposed objective function (Eq. 1). As will be demonstrated in our experiments, this combination gives the desired direction to the stipple dots but reduces the computerized appearance of the grid.

Grid Generation:
We generated the grid image, **g**, from the content image, **c**, by implementing the method proposed in [SLKL11a], in which a feature vector field is first generated from the input image and a grid is built by synthesizing a set of stripes. One following the tangential direction of the feature vector field and the other following the normal direction.

In our implementation, the algorithm proposed in [DZ13] followed by the XDoG [WKO12] was first used to build a feature map from the content image **c**, identifying clean edges even from highly-textured inputs as shown in Figure 3b. Additionally, isophote curves are detected to provide further information about the surface of the faces. To this end, the K-means [M*67] algorithm was used to quantize the image, detecting the isophote curves along the boundaries of each quantization level.

Once the feature map from the content image **c** is determined, we compute the gradient for each feature point (black pixels) as well as the tangential vector. In order to obtain the directions for non-feature pixels (white pixels), the multi-level-B-spline interpolation [LWS97] is applied, as it was performed in [SLKL11a]. Finally, for the generation of stripes, following normal and tangential directions of the features of the image, the algorithm proposed in [SLKL11a] was implemented. For the tangential stripes, the algorithm is initialized using a periodic transformation of the distances obtained by the jump-flooding-algorithm (JFA) [RT06] as observed in Figure 3c. After several iterations, both strip patterns are combined to generate a structure grid pattern. For our algorithm, the structure grid **g** is obtained from thresholding the obtained structure grid pattern based on the tone map of the content image **c**. The tone map is determined by the Gaussian blurring of the gray-scale transformation of the input content image.

2.4. Real Stipple Dots

After minimizing the augmented neural style loss with side information, $\mathcal{L}_{Stp}$ (Eq. 1), the stylized image $\hat{\mathbf{y}}$ is obtained. Usually, a pen with black ink is used by artists to draw stippling illustrations. In [MDSRI15], from several perceptual studies, the authors concluded that real stipple dots are not black circles, but they have different tones, shapes and size ranges, which depends on the pen and paper used. To achieve more realistic stipple dots representation in our results, the grayscale image $\hat{\mathbf{y}}$ generated by our neural-based algorithm is not binarized but an Extended Difference-of-Gaussians (XDoG) filter is applied instead. In this method, the input image **x** is filtered by two Gaussians with different standard deviations σ and $k\sigma$. The Difference of Gaussians (DoG) is then determined by

$$D_{\sigma,k,\tau}(\mathbf{x}) = G_\sigma - \tau G_{k\sigma}(\mathbf{x}). \tag{10}$$

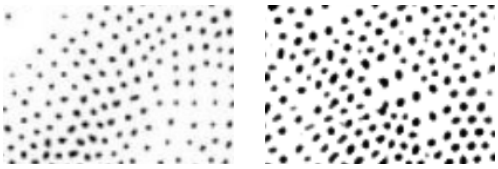
Next, instead of directly binarizing the DoG, a continuous ramp is applied:

$$T_{\varepsilon,\phi}(D) = \begin{cases} 1 & \text{if } D \geq \varepsilon, \\ 1 + \tanh(\phi(D-\varepsilon)) & \text{otherwise .} \end{cases} \tag{11}$$

Let $\mathbf{y}_X$ be the XDoG of the synthesized image $\hat{\mathbf{y}}$, as it is shown in Figure 2. In this way, as it can be observed in Figure 4, the stipple dots are black but with softened continuous color degradation, irregular shapes, size, and tones.

2.5. Outline Extraction and Final Result

In the proposed method, the outline of the content image **c** is extracted separately such that important details that might have been lost during the optimization are highlighted at the end. To this end, we use again the Extended Difference-of-Gaussians (XDoG) [WKO12] since it generates outlines that are very similar to the



(a) *Real hand-made* **(b)** *Rendered stipple dots*

Figure 4: *Stipple dots. Comparison of hand-made stipple dots (a) with the stippled dot rendered by our algorithm. The parameters for the XDoG filter [WKO12] were set to* $\tau = 0.96, \Phi = 20, \varepsilon = 0.02, k = 1.4$, *and* $\sigma = 1.1$.

hand-made illustrations. Then, the final synthesized image is obtained by

$$\mathbf{y}_{\text{Stp}} = \mathbf{y}_X \odot E_X, \tag{12}$$

where $\mathbf{y}_X$ is the XDoG filter of the synthesized image $\hat{\mathbf{y}}$ as introduced in Section 2.4. E_X is the outline image obtained by binarizing the XDoG filter of the content image $\mathbf{c}$.

3. Implementation Details

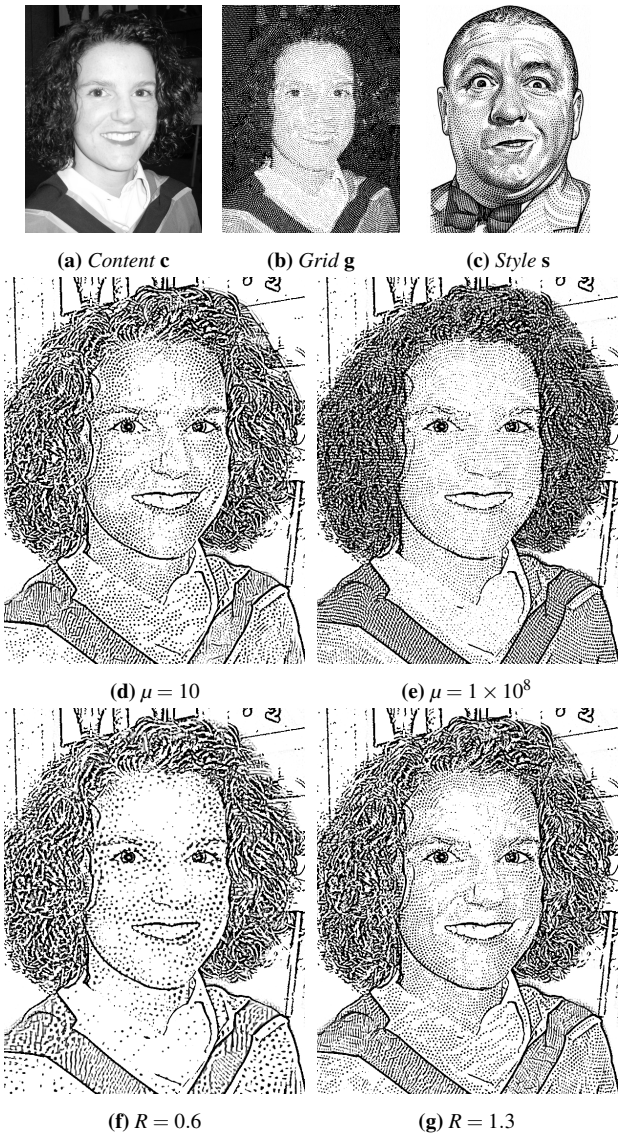
In this section, the experiments performed show the effectiveness of our method. We first discuss the effects of the additional constraints added to the traditional neural style transfer loss [GEB16]. Then, we compare our method with several state-of-the-art techniques.

In our implementation, we use the network VGG-19 [SZ14], pretrained for classification, to extract both the content features and the style features from the content image $\mathbf{c}$, the style image $\mathbf{s}$, and the grid image $\mathbf{g}$. We adapted the method proposed in [SLKL11a] for the generation of the structure grid images and we used the publicly available neural style transfer code in TensorFlow [Ten18] as baseline for our implementation. Adam optimizer was used for the optimization. After several experiments, we selected the feature maps at layer conv2_2 of the VGG-19 as the content features of the content image $\mathbf{c}$ and the feature maps from layers conv1_2, conv3_2 to compute the style features of the style image $\mathbf{s}$. The features of the grid image were extracted from the layer conv1_1, for both the style and content loss. The weight parameters for the style image and the content image were set to $\alpha = 5000$ and $\beta = 10$. The effect of the weight parameters in the grid loss are analyzed next. Our algorithm is run for 1000 iterations.

We initialized our algorithm with a blending between the grid mask and the content mask weighted by the edge priority mask $E(\mathbf{c})$ introduced in Section 2.2.

3.1. Tuning Parameters

In Figure 5, the grid parameters that can be controlled are varied. In the first row, Figure 5a-5c, the content image $\mathbf{c}$, the grid image $\mathbf{g}$ and the style image $\mathbf{s}$ used for this experiment are depicted. In the second row, Figure 5d-5e, the weight parameter of the style grid loss is set to zero ($\omega = 0$) and the weight of the content grid loss μ is increased. We can observe how the presence of the grid is predominant for large values of μ. Another parameter that will influence the



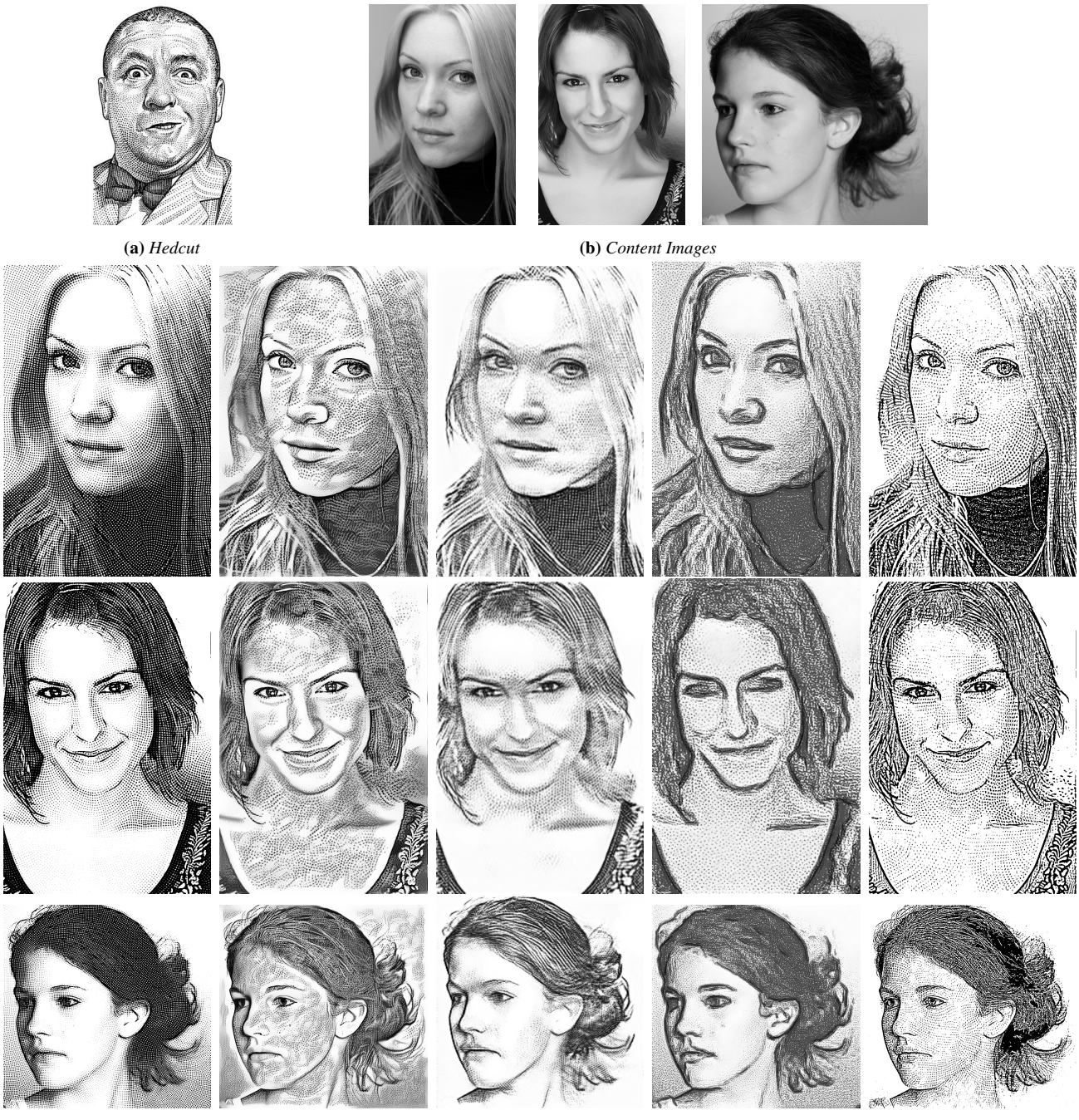
(a) *Content* $\mathbf{c}$ **(b)** *Grid* $\mathbf{g}$ **(c)** *Style* $\mathbf{s}$

(d) $\mu = 10$ **(e)** $\mu = 1 \times 10^8$

(f) $R = 0.6$ **(g)** $R = 1.3$

Figure 5: *Additional constraints added to NST. When the content weight parameter* μ *is increased in (d) and (e), the presence of the grid is enforced. The variation of the ratio between the size of the content image and the style image influences the size of the dots as shown in (f) and (g). Only one parameter vary each time while the others are fixed* ($\alpha = 5000, \beta = 10, R = 1.1, \mu = 1 \times 10^5, \omega = 10$).

size and shape of the stipple dots is the ratio between the size of the content image and the style image R. As shown in Figures 5f-5g, the size of the dots gets smaller as the ratio increases. The amount of dots then increases to achieve the same global tone. In our algorithm, these parameters can be controlled by the user. The weight of the style grid loss ω, in contrast, is kept fixed. Preferences can vary depending on the desired output device and personal choice. In our experiments, we found a good trade-off using the following parameters: $\alpha = 5000, \beta = 10, \mu = 1 \times 10^5, \omega = 10$, and $R \in [0.7, 1.3]$.



(a) *Hedcut* **(b)** *Content Images*

(c) *Son et al. [SLKL11a]* **(d)** *NST [GEB16]* **(e)** *CycleGAN [ZPIE17]* **(f)** *Im2Pencil [LFH*19]* **(g)** *Proposed method*

Figure 6: *Visual comparison of the proposed method with both NPR state-of-the-art (c) and deep learning-based (d)-(g) algorithms. (a) Reference style image by Randy Glass [Gla] used for the neural-based style transfer (d) and (g).*

3.2. Comparisons

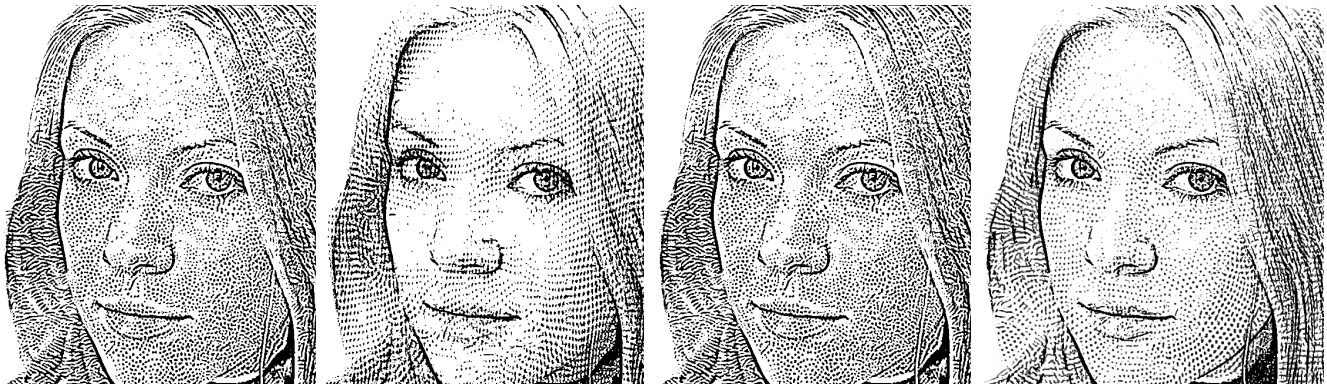
We also compare our results with four different algorithms, one that uses traditional NPR [SLKL11a], and three different learning-based algorithms: Gatys [GEB16], CycleGAN [ZPIE17], and Im2Pencil [LFH*19]. The content images (Fig. 6b) and directional stipplings (Fig. 6c) generated with the method proposed in [SLKL11a] were taken from the authors' website [SLKL11b]. For traditional style transfer [GEB16], we tuned the parameters to obtain the results more similar to the target style as possible. Features maps were extracted from layers conv2_2 for the content image and layers conv1_1, conv2_1, and conv3_1 for the style im-

Figure 7: *Comparison of HedcutDrawings generated using different style images. The reference style images [Gla] (a) and (e) were use to generate the HedcutDrawings (b)-(d) and (f)-(h), respectively.*

age. The content weight and style weight for the traditional loss proposed in [GEB16] were set to $\alpha = 1000$ and $\beta = 10$. The results from this method [GEB16] were always generated using the same target-style image as in our method. In the case of unpaired Image-to-Image translation using cycle-consistent adversarial networks [ZPIE17], we prepared an unpaired data set, collecting 110 content images which are portraits randomly selected from the Helen dataset [LBL*12] and 110 hedcuts obtained from online sources. Authors' publicly available code was used to train the CycleGAN network. In a similar way, the online pre-trained model for the Im2Pencil method [LFH*19] was used to generate the stippling shaded samples used here for comparison.

The five methods are visually compared in Figure 6. The style image used for both neural-based techniques, Gatys [GEB16] and our method, is shown in Figure 6a. Even though the stippling images generated by the method proposed by Son et al [SLKL11a] (Fig. 6c) are pleasant, their grid structure is very regular providing a computerized look that is not present in traditional hand-made drawings. Gatys' method generates images with a similar texture to hand-made stippling illustrations, as shown in the second column of images i.e. Fig. 6d. The stipple dots, however, do not follow linear paths along the image features. Additionally, spillover effects can be observed, in which dots are not distinguishable. In CycleGAN results, dark areas are at random locations within the portraits, as it can be observed in the part of the eyes. The synthesized texture is somehow similar to the target style. Nonetheless, stipples dots are not clearly defined either and spillover areas are also perceived. Even though the results of Im2Pencil [LFH*19] are the more realistic to be hand-made illustrations, stippling points in certain areas are not distinguishable. The outline strokes differ from the target style. Moreover, the placement of the points does not resemble the characteristics of the hedcut style images. In these three methods, the size of the dots barely changes compare to the variation observed in hand-made hedcuts.

In contrast, our method achieves stippling points that are clearly defined and placed along linear paths that depend on the structure



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

Figure 8: *To measure the contribution of each term in the cost function we let the cost function be: (a)* $\alpha\mathcal{L}_{s+} + \beta\mathcal{L}_{c+}$*; (b)* $\beta\mathcal{L}_{c+} + \omega\mathcal{L}_{G_s} + \mu\mathcal{L}_{G_c}$*; (c)* $\alpha\mathcal{L}_{s+} + \beta\mathcal{L}_{c+} + \omega\mathcal{L}_{G_s}$*; (d)* $\alpha\mathcal{L}_{s+} + \beta\mathcal{L}_{c+} + \omega\mathcal{L}_{G_s} + \mu\mathcal{L}_{G_c}$.

of the face, which resemble the key characteristics of hedcuts. Furthermore, the shape of the dots is not regular, providing a more realistic hand-made appearance. Outlines are designed to match the target style. The size of the dots changes depending not only on the tone of the image but also on the reference target style. In addition, a much better representation of the eyes within the portrait with respect to the other three methods is achieved, being pretty much like the original hedcut drawings. As shown in Figure 7, similar high-quality stippling illustrations are obtained when different style images are used as a reference in the proposed method. More results are included as part of the supplemental material, where the style image and the size of the stipple dots are varied for the generation of hedcuts from a wide variety of content images. Additionally, in Fig. 8, we can visually evaluate the individual impact of each loss term in Eq. 1. Among the limitations of the current approach and motivations for future research are that some results still contain some artifacts. Moreover, even though the stippling dots are very similar to the the handmade hedcuts, the hatching of the hair and the clothing characteristic of hand-made hedcuts is not completely captured by the proposed algorithm.

Finally, since the synthesis of hedcut style is a subjective task, we also conducted a user study in order to compare our results with the state of the art. In this way, 45 users were first asked to get familiar with the style, showing them several samples of hand-made hedcut stippling illustrations. Next, they were provided with synthesized results randomly organized side-by-side for three different content images, asking them to select which image resembled the most the style previously defined. Synthesized hedcuts using the proposed method were chosen to be more similar to the hand-made hedcuts 60.7% of the times.

4. Conclusion

A method for the generation of the iconic hedcuts using neural style transfer was proposed. In this method, additional constraints depending on a structure grid and binarization techniques improve on state-of-the-art results for the generation of directional stipplings. Style transfer allows to learn from the experience of the professional artists through the style image and the grid constraints provide the missing information about the structure of the content image. Results show a less computerized look than state-of-the-art NPR results and the placement of the dots still follow linear paths just like hand-made hedcut stipplings. These drawings can be used for the generation of smart QR codes [GALV14, Gar14, PPLAA22] to be used in businesses cards, press, and publicity designs. The rendering of other styles in which the placement of the style primitives is a key component of the drawings such as intaglio are being explored as part of our future work.

References

[AML10] ARROYO G., MARTÍN D., LUZÓN M. V.: Stochastic generation of dots for computer aided stippling. *Computer-Aided Design and Applications 7*, 4 (2010), 447–463. 1

[Bol19] BOLTON E.: Can a Machine Illustrate WSJ Portraits Convincingly?, 2019. URL: https://medium.com/the-wall-street-journal/can-a-machine-illustrate-wsj-portraits-convincingly-3f76a10ee9ae. 2

[BSD09] BALZER M., SCHLÖMER T., DEUSSEN O.: *Capacity-constrained point distributions: a variant of Lloyd's method*, vol. 28. ACM, 2009. 1

[CYC*12] CHEN Z., YUAN Z., CHOI Y.-K., LIU L., WANG W.: Variational blue noise sampling. *IEEE Transactions on Visualization and Computer Graphics 18*, 10 (2012), 1784–1796. 1

[DGBOD12] DE GOES F., BREEDEN K., OSTROMOUKHOV V., DESBRUN M.: Blue noise through optimal transport. *ACM Transactions on Graphics (TOG) 31*, 6 (2012), 171. 1

[DZ13] DOLLÁR P., ZITNICK C. L.: Structured forests for fast edge detection. In *Proceedings of the IEEE international conference on computer vision* (2013), pp. 1841–1848. 3, 4

[GALV14] GARATEGUY G. J., ARCE G. R., LAU D. L., VILLARREAL O. P.: Qr images: optimized image embedding in qr codes. *IEEE transactions on image processing 23*, 7 (2014), 2842–2853. 8

[Gar14] GARATEGUY G. J.: *Optimal embedding of QR codes into color, gray scale and binary images*. University of Delaware, 2014. 8

[GEB16] GATYS L. A., ECKER A. S., BETHGE M.: Image style transfer using convolutional neural networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (2016), pp. 2414–2423. 1, 2, 3, 5, 6, 7

[Gla] GLASS R.: Randy glass illustrations. https://randyglassstudio.com/, Last accessed on 2019-11-14. 2, 6, 7

[IZZE17] ISOLA P., ZHU J.-Y., ZHOU T., EFROS A. A.: Image-to-image translation with conditional adversarial networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (2017), pp. 1125–1134. 2

[KMI*09] KIM S. Y., MACIEJEWSKI R., ISENBERG T., ANDREWS W. M., CHEN W., SOUSA M. C., EBERT D. S.: Stippling by example. In *Proceedings of the 7th International Symposium on Non-Photorealistic Animation and Rendering* (2009), ACM, pp. 41–50. 1

[KML15] KHAN K., MAURO M., LEONARDI R.: Multi-class semantic segmentation of faces. In *2015 IEEE International Conference on Image Processing (ICIP)* (2015), IEEE, pp. 827–831. 3

[KSL*08] KIM D., SON M., LEE Y., KANG H., LEE S.: Feature-guided image stippling. In *Computer Graphics Forum* (2008), vol. 27, Wiley Online Library, pp. 1209–1216. 1

[KWME10] KIM S., WOO I., MACIEJEWSKI R., EBERT D. S.: Automated hedcut illustration using isophotes. In *International Symposium on Smart Graphics* (2010), Springer, pp. 172–183. 1

[LBL*12] LE V., BRANDT J., LIN Z., BOURDEV L., HUANG T. S.: Interactive facial feature localization. In *European conference on computer vision* (2012), Springer, pp. 679–692. 7

[LFH*19] LI Y., FANG C., HERTZMANN A., SHECHTMAN E., YANG M.-H.: Im2pencil: Controllable pencil illustration from photographs. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (2019), pp. 1525–1534. 2, 6, 7

[LM11] LI H., MOULD D.: Structure-preserving stippling by priority-based error diffusion. In *Proceedings of Graphics Interface 2011* (2011), Canadian Human-Computer Communications Society, pp. 127–134. 1

[LPSB17] LUAN F., PARIS S., SHECHTMAN E., BALA K.: Deep photo style transfer. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (2017), pp. 4990–4998. 3

[LWS97] LEE S., WOLBERG G., SHIN S. Y.: Scattered data interpolation with multilevel b-splines. *IEEE transactions on visualization and computer graphics 3*, 3 (1997), 228–244. 4

[M*67] MACQUEEN J., ET AL.: Some methods for classification and analysis of multivariate observations. In *Proceedings of the fifth Berkeley symposium on mathematical statistics and probability* (1967), vol. 1, Oakland, CA, USA, pp. 281–297. 4

[MARI17] MARTÍN D., ARROYO G., RODRÍGUEZ A., ISENBERG T.: A survey of digital stippling. *Computers & Graphics 67* (2017), 24–44. 1, 2

[MDSRI15] MARTÍN D., DEL SOL V., ROMO C., ISENBERG T.: Drawing characteristics for reproducing traditional hand-made stippling. In *Proceedings of the workshop on Non-Photorealistic Animation and Rendering* (2015), Eurographics Association, pp. 103–115. 4

[NMT*18] NIRKIN Y., MASI I., TUAN A. T., HASSNER T., MEDIONI G.: On face segmentation, face swapping, and face perception. In *2018 13th IEEE International Conference on Automatic Face & Gesture Recognition (FG 2018)* (2018), IEEE, pp. 98–105. 3

[PPLAA22] PENA-PENA K., LAU D. L., ARCE A. J., ARCE G. R.: Qrnet: fast learning-based qr code image embedding. *Multimedia Tools and Applications* (2022), 1–20. 8

[RC12] ROSIN P., COLLOMOSSE J.: *Image and video-based artistic stylisation*, vol. 42. Springer Science & Business Media, 2012. 1

[RT06] RONG G., TAN T.-S.: Jump flooding in gpu with applications to voronoi diagram and distance transform. In *Proceedings of the 2006 symposium on Interactive 3D graphics and games* (2006), ACM, pp. 109–116. 4

[SHS02] SECORD A., HEIDRICH W., STREIT L.: Fast primitive distribution for illustration. In *Rendering Techniques* (2002), pp. 215–226. 1

[SID17] SEMMO A., ISENBERG T., DÖLLNER J.: Neural style transfer: a paradigm shift for image-based artistic rendering? In *Proceedings of the Symposium on Non-Photorealistic Animation and Rendering* (2017), ACM, p. 5. 1

[SLKL11a] SON M., LEE Y., KANG H., LEE S.: Structure grid for directional stippling. *Graphical Models 73*, 3 (2011), 74–87. 1, 2, 4, 5, 6, 7

[SLKL11b] SON M., LEE Y., KANG H., LEE S.: Structure grid for directional stippling - more results, 2011. http://cg.postech.ac.kr/research/structure_grid/results.php, Last accessed on 2019-11-03. 2, 6

[SZ14] SIMONYAN K., ZISSERMAN A.: Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556* (2014). 1, 3, 5

[Ten18] TENSORFLOW: Neural style transfer with eager execution, 2018. https://colab.research.google.com/github/tensorflow/models/blob/master/research/nst_blogpost/4_Neural_Style_Transfer_with_Eager_Execution.ipynb, Last accessed on 2019-05-30. 5

[WKO12] WINNEMÖLLER H., KYPRIANIDIS J. E., OLSEN S. C.: Xdog: an extended difference-of-gaussians compendium including advanced image stylization. *Computers & Graphics 36*, 6 (2012), 740–753. 2, 4, 5

[WLZ*18] WANG T.-C., LIU M.-Y., ZHU J.-Y., TAO A., KAUTZ J., CATANZARO B.: High-resolution image synthesis and semantic manipulation with conditional gans. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (2018), pp. 8798–8807. 2

[(WS19] (WSJ) W. S. J.: AI Portrait, 2019. URL: https://aiportrait.wsj.com/. 2

[ZPIE17] ZHU J.-Y., PARK T., ISOLA P., EFROS A. A.: Unpaired image-to-image translation using cycle-consistent adversarial networks. In *Proceedings of the IEEE international conference on computer vision* (2017), pp. 2223–2232. 2, 6, 7

[ZZP*17] ZHU J.-Y., ZHANG R., PATHAK D., DARRELL T., EFROS A. A., WANG O., SHECHTMAN E.: Toward multimodal image-to-image translation. In *Advances in Neural Information Processing Systems* (2017), pp. 465–476. 2