






# Stippling of 2D Scalar Fields

Jochen Görtler , Marc Spicker , Christoph Schulz , *Student Member, IEEE Computer Society*,  
Daniel Weiskopf , *Member, IEEE Computer Society*, and Oliver Deussen 

**Abstract**—We propose a technique to represent two-dimensional data using stipples. While stippling is often regarded as an illustrative method, we argue that it is worth investigating its suitability for the visualization domain. For this purpose, we generalize the Linde–Buzo–Gray stippling algorithm for information visualization purposes to encode continuous and discrete 2D data. Our proposed modifications provide more control over the resulting distribution of stipples for encoding additional information into the representation, such as contours. We show different approaches to depict contours in stipple drawings based on locally adjusting the stipple distribution. Combining stipple-based gradients and contours allows for simultaneous assessment of the overall structure of the data while preserving important local details. We discuss the applicability of our technique using datasets from different domains and conduct observation-validating studies to assess the perception of stippled representations.

**Index Terms**—Stippling, contours, semiotics, evaluation, scalar field visualization, abstraction, sampling

## 1 INTRODUCTION

VISUAL abstraction has been identified as one of the top visualization research problems by Johnson [1]. When it comes to visual representation of univariate data, we usually think of bar charts, line charts or possibly dot plots, where abstraction is often achieved through binning or smoothing. Although binning reduces the amount of visualized data, task performance for continuous geospatial data does not necessarily decrease [2]. However, it has been shown that dot representations significantly outperform contour representations for memorization tasks [3]. Moreover, there is an aesthetic aspect to abstraction which is also an important concern for many applications

In this work, we propose a dot-based technique for visual abstraction of scalar fields based on stippling. Stippling is a form of abstraction, where stipples (dots) are carefully distributed to approximate shading. In that sense, stippling of scalar fields can be seen as a controlled simplification or non-linear smoothing of the dataset. Traditional algorithms for stippling are difficult to steer and therefore challenging to use in a visualization context. We aim to close this gap by modifying a technique from computer graphics to support binned and continuous data simultaneously.

Encoding continuous data with color scales can be problematic because color perception is highly dependent on the surrounding illumination [4]. In contrast, stipples allow us to encode information in their size and density and therefore do not require color. They are usually drawn as black circles on a white background, making them more robust against varying illumination. Another characteristic

of stippling is what has been previously explored as *view-dependent information* in the context of visualization [5]: by looking closely at a region, local differences and patterns can be investigated (in-depth information). With increased viewing distance, larger-scale patterns dominate the visual impression (overview). Stipples can convey information through size and density. Our method provides fine control over these visual variables, which can be used to either encode additional information about the distribution, such as contours, encode information redundantly which can improve the overall visual impression, or combine multiple sources of data, as shown in Figure 1.

Regarding perception, we expect stippling parameters to have a highly non-linear effect on the result. Therefore, we have performed crowdsourcing experiments to better understand the influence of stipple sizes and density on perception. Through these experiments, we also aim to give guidance in choosing the right parameters for our algorithm.

Our contributions are as follows: We modify a stippling algorithm for arbitrary scalar fields with adjustments to control the point size and add restrictions for binning. With this mechanism we introduce structure to our visualization through more deliberate stipple placement. We also verify our approach using psychophysical experiments and discuss implications for usage.

## 2 RELATED WORK

In general, our method shares similarities with different techniques used in cartography [6], [7]. Choropleth maps use lightness of colors to encode quantitative areal data, requiring the data to already be partitioned, for example into administrative areas. However, such data is also well suited to be depicted using symbol representations. Brewer and Campbell [8] explore the usage and perception of such symbols for representing quantitative data on maps.

Proportional symbol maps use the size of glyphs to represent quantitative data either at specific locations or in areas around it. In contrast, dot maps represent the quantity

- Görtler, Spicker, and Deussen are with the University of Konstanz.  
E-mail: { jochen.goertler, marc.spicker, oliver.deussen }@uni-konstanz.de
- Schulz and Weiskopf are with the University of Stuttgart.  
E-mail: { christoph.schulz, daniel.weiskopf }@visus.uni-stuttgart.de



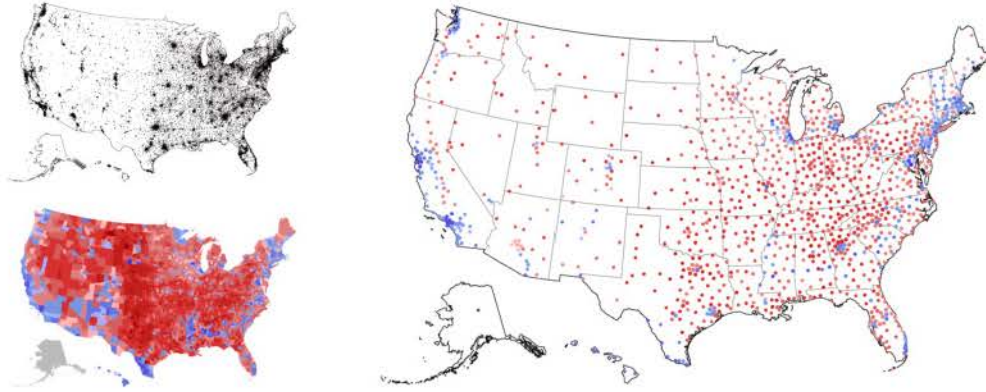


Fig. 1. Stippled illustration of the 2016 election data from the United States (right) that combines the population density as the distribution of stipples (top-left, illustrated as dot map), and the vote difference for each administrative region (bottom-left, depicted using choropleth maps). Note how the stippled abstraction visualizes both aspects of the data simultaneously.

of a feature by sampling the corresponding amount of symbols. Dot maps are sometimes used to depict population distributions, where each dot represents a fixed number of people. This can lead to regions that are very dense or even over-plotted. To solve this problem, De Berg et al. [9] describe an approach to simplify these maps while maintaining the given distribution. Our approach also samples a given distribution, but also respects the size of the point that will be placed at each location.

Non-photorealistic rendering (NPR) is a major field of research in computer graphics and some of the developed techniques have also been explored in the context of visualization [10]. Wood et al. [11] propose sketchy rendering as a visual variable. They conclude that this type of representation can be used to encode additional information and may lead to an increased engagement with the visualization. Kim et al. [12] present a technique for aggregation, abstraction, and stylization in the context of maps. Their visual elements are similar to brush strokes which vary in length, density, color, and orientation depending on the underlying multivariate data. In contrast to our method, both of these approaches use lines as primitives.

Stippling has also been explored in the visualization community. Biological cell maps have been represented using stipples [13] and share visual similarities to our visualization. However, the point positions for these maps are part of the input and do not need to be sampled. Lu et al. [14], [15] propose a stipple and feature enhancement method for volume rendering. Stippling has also been used to visualize brain fibers [16] using diffusion tractograms. These approaches sample stipples in predefined cells of a grid, whereas we use an algorithm that is based on adaptively changing Voronoi cells.

**Stippling Algorithms**—Generating random point sets with almost uniform point-to-point distances is necessary for sampling, halftoning, remeshing, and artistic rendering methods such as stippling [17]. Lloyd’s method [18] is one widely used optimization method based on Voronoi diagrams to generate such point sets. For a given point set, it iteratively moves each point to the centroid of its correspond-

ing Voronoi cell until the points are distributed equally. The method has been extended to create different point densities, for example, based on underlying image information [19]. Stippling with varying point sizes has also been explored [20]. A more recent stippling method is also based on Voronoi diagrams: Weighted Linde-Buzo-Gray (LBG) stippling [21] provides more intuitive parameters to control the final result and can also handle variable point sizes. Some stippling algorithms have been proposed with the goal of recreating or introducing controlled patterns [22], [23]. However, these algorithms focus on specific artistic effects and are limited in their applicability for other purposes. We generalize the algorithm of Deussen et al. [21] to work on scalar fields and extend it to encode additional structure in the results.

**Structure Highlighting**—A popular approach to emphasize structure is image enhancement, which is often used to improve the quality of an image in a post-processing step. Techniques range from simple contrast enhancement methods to more elaborated ones, such as histogram equalization [24], and wavelet-based approaches [25]. Luft et al. [26] use depth information to locally enhance the contrast between objects at different depths with unsharp masking. This improves the perception of the spatial arrangement within a scene. Similar ideas have been proposed in scientific visualization: Bruckner and Gröller [27] present a method that uses halos to emphasize the spatial relationships in volumetric data, making it easier to judge occlusion. Everts et al. [28] also use halos for 3D rendering of dense line data. They highlight tight line bundles while less structured lines are de-emphasized. Hertzmann and Zorin [29] use Mach-bands to improve the perception of surface structure. We employ ideas from these works, extending our stippling algorithm to encode additional structure in the form of contour lines.

**Psychophysics in Visualization**—Many studies of color and brightness perception have been carried out, which is reflected in the overview about color maps by Zhou and Hansen [30]. One of the earliest studies resulted in the well-known MacAdam ellipses [31] in which the color space was measured through repeated pairwise comparisons of



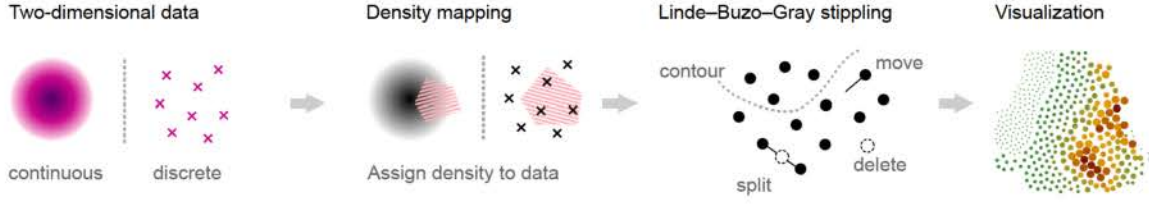


Fig. 2. Overview of our technique. Our approach can handle scalar fields as well as discrete points as input. We use a modified version of the Linde-Buzo-Gray stippling algorithm that provides more control over the final appearance. By locally modifying the stipple distribution, we can encode additional information such as contour lines into the final representation. Our method encodes distributions of the underlying data into corresponding distributions of stipples, leaving other visual variables, such as color, free for encoding additional attributes.

a single subject (method of adjustment). Due to the high effort involved, these measurements provided the basis for standardized color spaces for a long time. Recently, these studies were refined by Froehlich et al. [32] using an adaptive procedure which iteratively attenuates a high-intensity stimulus depending on the observers' mistakes (staircase method).

Because color perception is highly non-linear and context-dependent, specialized studies for information visualization have been conducted. For example, Ware [33] investigated univariate color maps. Later, Mittelstädt et al. [34] developed a technique to compensate for contrast effects. To verify their model, they conducted an experiment in which the participants had to draw a curve based on a non-linear gray scale stimulus (magnitude estimation method). In fact, such psychophysical methods can even be applied to higher levels of perception, as shown by Rensink [35] in his study on the perception of correlation in scatter plots. Recently, Szafir [36] studied color differences in the context of information visualization and questioned some of the well-known ColorBrewer [37] color schemes. Similar to MacAdam, but in a crowdsourcing study, she used a two-alternative forced choice (2AFC) task-based design to obtain psychophysical functions. In our user study we decided to use a similar crowdsourcing-based approach. For a thorough discussion on crowdsourcing in the context of visual representations, we refer to the work of Archambault et al. [38].

### 3 OVERVIEW

Given 2D data, we aim to find a stippled representation of the distribution of its values. To achieve this, we adapt the Linde-Buzo-Gray (LBG) stippling algorithm [21] to continuous and discrete scalar fields. The algorithm is based on Voronoi diagrams: By iteratively moving each stipple to the centroid of its Voronoi cell, global point-to-point distances become more and more equalized. In addition to simply moving the points, the LBG algorithm also splits and merges cells based on the corresponding density function.

Figure 2 provides an overview of the individual steps of our approach. First, the input data is transformed to the target density using a mapping function. This density allows us to establish a relation between the data that falls into the region of a stipple, i.e. its Voronoi cell, and the conceptual amount of ink that a stipple carries. Additionally, non-linear mapping functions can be used to highlight relevant information. For discrete input points in 2D, this mapping determines how multiple data points are aggregated.

We apply our modified version of the LBG stippling algorithm to compute the final representation. The result's point sizes can either be determined adaptively or directly specified by the user. In Section 5, we describe two approaches for adding contour lines to the visualization. These methods can be seen as extensions of our pipeline and require the extraction of contours from the mapped data. The final stippled result reflects the underlying data with the possibility to distinguish density changes locally. Moreover, it can display contours to provide an overall impression of the data simultaneously. Details for each step are described in their respective sections of this paper.

### 4 STIPPLING

The Linde-Buzo-Gray (LBG) stippling algorithm has mainly been described in the context of computer graphics, where it is used to abstract images or for resampling meshes. In the following, we want to generalize this method to scalar fields by means of formalization. Generally, 2D data can be discrete or continuous and potentially comprise a high dynamic range.

We define a 2D scalar field as a function  $\phi : \mathbb{R}^2 \rightarrow \mathbb{R}$ . Similar to color scales, we require a mapping  $\rho : \mathbb{R} \rightarrow [0, 1]$  to transform the co-domain of the scalar field to unit range. We call this the derived scalar field  $\Phi$ :

$$\Phi : \mathbb{R}^2 \rightarrow [0, 1], \quad \Phi = \rho \circ \phi$$

In NPR, different rendering primitives and shapes have been proposed for stippling [39], [40]. Theoretically, our proposed method can handle arbitrary convex shapes, but for now we will use circles with a given extent and position. The goal of the LBG stippling algorithm is to arrange stipples according to a given measure, in our case the values of the transformed scalar field  $\Phi$ . The algorithm starts with a random initial distribution of stipples. Then, it evaluates how well each stipple represents its proximity in  $\Phi$ . The neighborhood of each stipple  $s \in S$  is found by computing the Voronoi diagram over the complete set of stipples.

Conceptually, we want to relate the required amount of ink for a stipple to the values of the scalar field in its vicinity. For continuous scalar fields, we achieve this by integrating over its corresponding Voronoi cell  $V_s$ . The target density for a stipple  $T_s$  is therefore given by:

$$T_s = \iint_{V_s} \Phi(x, y) dA$$



In the case of discrete scalar fields and uniform grids, this is the same as accumulating the density over all points or grid cells that belong to the associated Voronoi cell:

$$T_s = \sum_{i \in V_s} \Phi(i)$$

According to the Linde–Buzo–Gray algorithm, we compare  $T_s$  with the area occupied by the stipple  $A_s$ . More precisely, with the area weighted by the values of the scalar field. Three different cases can occur, as shown in Figure 3: If a stipple represents the target density reasonably well, up to a specified error threshold  $\varepsilon$  ( $T_s \in [A_s - \varepsilon, A_s + \varepsilon]$ ), a relaxation step is performed. This moves the stipple toward the weighted centroid of its Voronoi cell (Lloyd relaxation). In the other two cases the stipple is either split ( $T_s > A_s + \varepsilon$ ) into two separate cells or deleted ( $T_s < A_s - \varepsilon$ ). This procedure is repeated iteratively until no more splits and merges occur and the algorithm converges. To increase the convergence rate, the threshold  $\varepsilon$  is slightly increased with each iteration. This marginally affects the resulting quality in later steps, where very few changes are performed, but guarantees convergence due to the steadily increasing allowed error.

The main parameter of the algorithm is the stipple size and the required number of stipples is automatically determined. Choosing a smaller size results in more points and thus preserves more details, whereas a larger size yields a more abstract representation with fewer points. Because the split and merge operations compare the contained density of a cell to the density represented by the generating stipple, the stipple size can vary locally. It can be set according to its position, attributes of the scalar field, or directly provided by the user. The original LBG stippling algorithm supports variable point sizes by adjusting the size of a stipple (within a provided range) according to  $T_s$ . This, however, does not enforce the point size in any way and there are no guarantees that all sizes within the given range are reached. Our proposed method changes this step of the algorithm: we set the point size according to the scalar field and then compare its area to the target density. Through this modification, information from the scalar field can be encoded independently in the point size as well as the point density. Figure 4a shows how a constant density can be established using constant and variable point sizes. Varying the point size but keeping the density constant does not create a perceptually constant impression (the density of the left looks different compared to the right), even though the density is still accurately reflected. Figure 4b shows a density gradient stippled using constant and variable point sizes. Increasing the point size can be used to boost the perception

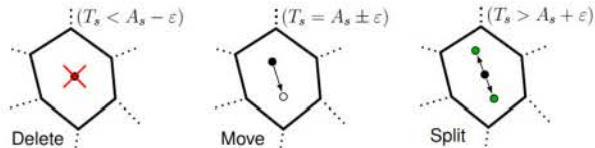


Fig. 3. Schematic illustration of the LBG stippling algorithm. Based on the comparison between the density contained in each cell and the stipple area, cells are either deleted, moved to the centroid of the cell, or split.

of the density. It is important to note that these two channels are not independent: For a given density we might have only a subset of the preferred point sizes and we restrict the amount of required points by enforcing a certain point size. Therefore, it is not readily possible to use these two channels to encode two independent variables of the data.

## 5 EMULATING CONTOURS

It is common to add contour lines to conventional representations of scalar fields to emphasize the overall structure. Therefore, we show how to emulate contours using stipples. Algorithm 1 shows the LBG stippling method with our changes to emulate contours highlighted in red. In the following, we detail the highlighted modifications to emulate contours by carefully adjusting the placement of stipples.

### 5.1 Restricted Stippling

One way of adding contours to stipple drawings is to keep the areas belonging to the same bin visually similar and vice versa. We achieve this by keeping the stipple size constant within a contour. Then, we vary their density to represent the underlying scalar values. To further emphasize the borders between contours, we force stipples to stay within the boundary of their current quantized regions. This results in a visible line between these regions, because no stipples will be placed inside the boundary region. Referring to the Gestalt principles, this technique is also called *negative space*.

To enforce this behavior, we restrict the underlying Voronoi diagrams according to contour boundaries defined by a user-defined map. This approach has already been used to create decorative mosaics, where tiles should not overlap sharp boundaries of regions [41]. While previous approaches propose to delete regions where no tiles should be placed, we aggregate the density of the scalar field into each Voronoi cell for the contours separately instead. During integration of the density of a Voronoi cell, we consider only the density that is part of the contour  $c_i$ :

$$T_s^i = \iint_{V_s} c_i(x, y) \Phi(x, y) dA, \text{ where} \quad (1)$$

$$c_i(x, y) = \begin{cases} 1, & \text{if } (x, y) \text{ is enclosed by contour } i \\ 0, & \text{otherwise} \end{cases}$$

Thus, there can be up to  $n$  centroids per Voronoi cell, depending on the number of given bins  $n$ . Other cell

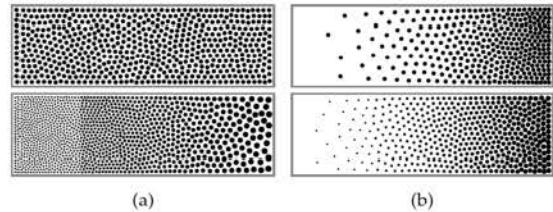


Fig. 4. (a) Stipples distributed with a constant density and either constant point size (top) or variable size (bottom). (b) Increasing density, again with constant point size (top) and variable size (bottom).



properties, such as area and average density are treated in the same fashion. When applying the LBG stippling algorithm, we just consider the sub-cell with the maximum contained density. This forces the stipple to move towards the center of said cell, which in turn means away from the corresponding region boundary. Figure 5 shows an example of this process. The resulting stippled representation is similar to quantized grayscale color, but instead of using different colors, contours are directly encoded into the distribution of the stipples.

## 5.2 Mach-Banding

Our second approach to introduce contours is based on Mach-banding. Mach-bands are an optical illusion that locally amplify the contrast between areas that have slightly different shading. In many applications this effect is undesirable because it changes the perception close to boundaries. In our method we take advantage of this illusion and show how it can be used to emulate stippled contours. Although we coined the proposed approach as ‘Mach-banding’, it would be more precise to think of it as ‘Mach-banding inspired’. Mach-banding has already been used in computer graphics, where it is called *unsharp masking*, to amplify edges or improve contrast by incorporating depth information [26]. We apply this concept in the domain of scalar fields: Starting from a quantized grayscale map  $C_\Phi$  that we want to stipple, we perform a frequency separation, keeping only the high-pass result  $\Delta C_\Phi$ . In practice, this means applying a Gaussian blur  $G_{d \times d}$  to obtain a low-pass filtered version of  $C_\Phi$  and then subtracting it from the original quantized map:

$$\Delta C_\Phi = C_\Phi - G_{d \times d} * C_\Phi$$

The size  $d$  of the Gaussian can be used to control the extent of the local contrast change. Setting its size similar to the stipple size has yielded good results in our experiments. Then, we use this information to modify the scalar field by blending  $\Delta C_\Phi$  with  $\Phi$  using the following blending function where the weight is denoted by  $w \in [0, 1]$ :

$$\Phi' = \begin{cases} \text{clamp}(\Phi + 2w(\Delta C_\Phi - 0.5), 0, 1), & \Delta C_\Phi > 0.5 \\ \text{clamp}(\Phi + 2w(\Delta C_\Phi - 1.0), 0, 1), & \Delta C_\Phi \leq 0.5 \end{cases} \quad (2)$$

The unweighted version of this function is commonly known in image manipulation software as *linear light* blending. Figure 6 shows an example of the complete process. Due to the local nature of the technique, we vary only the representation of the scalar field close to boundaries, while maintaining correct values everywhere else. We discuss the error introduced to the density map and corresponding

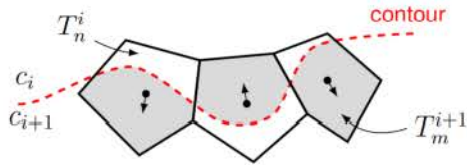


Fig. 5. Contour lines restrict the density integration for each Voronoi cell  $T_s^i$ . Solely the density of the largest area is considered for the stippling algorithm, and the centroid moves accordingly during the relaxation.

Voronoi cells later in Section 9. The size of each stipple is determined by the modified scalar field. By weighing the blending function, we can control how apparent the contours will be in the visualization. It is important to note that for this approach we modify the density channel while keeping the stipple size unmodified to add contours to stipple drawings. This leaves the other channels for encoding additional data.

We will apply our methods to real world data in Section 6, for now we compare the results on a synthetic dataset in Figure 7. Shown are different quadrants of the symmetric *Eggcrate* function. The top left shows the scalar field, and top right the results of the unchanged LBG stippling algorithm. While stippling facilitates estimation and comparison of local densities due to using discrete elements, the overall shape of the function is not retained very well. Our proposed Mach-banding and restricted stippling approaches are shown on the bottom. Our contour emulation approach allows distinguishing local density changes when looking at the visualization from a close viewing distance, yet providing a comprehensive overview when looking from further away. Please note that sometimes the contours are more apparent in the printed version of the paper.

## 6 EXAMPLES

In this section, we showcase our technique using real-world geographic, election, and multivariate data to demonstrate the effectiveness as visualization and abstraction technique. Moreover, we aim to convey the potential design space of stipple visualizations. For this, we also refer to the supplemental material, which contains additional examples.

---

### Algorithm 1: Modified LBG stippling with contours

---

**Input** : Target density function  $\Phi$

**Output**: List of stipples  $S$

---

```

1  $S \leftarrow$  initialize random stipples
2 repeat
3    $V \leftarrow$  compute Voronoi diagram of  $S$  using  $\Phi$ 
4   foreach  $s \in S$  do
5      $T_s \leftarrow \iint_{V_s} \Phi(x, y) dA$  // density for each cell
6     if restricted stippling then
7        $T_s \leftarrow \arg\max_i \iint_{V_s} c_i(x, y) \Phi(x, y) dA$  // (1)
8     else if Mach-banding then
9        $T_s \leftarrow \iint_{V_s} \Phi'(x, y) dA$  // (2)
10    end
11    foreach Voronoi cell  $V_s \in V$  and resp. stipple  $s$  do
12      if Density  $T_s$  too low then
13        remove  $s$  from  $S$ 
14      else if Density  $T_s$  too high then
15        split  $s$  into two and add to  $S$ 
16      else
17        move  $s$  to weighted centroid of  $V_s$ 
18      end
19    end
20 until no more splits and merges
21 return  $S$ 

```

---



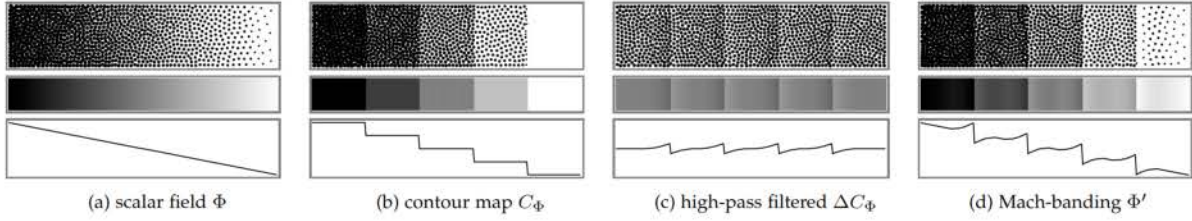


Fig. 6. The different steps that are required to emulate contours using the Mach-banding effect. Starting from a scalar field input (a) together with the corresponding quantized grayscale map (b), we can compute a high-pass filtered version (c). The final result (d) can be obtained by stippling a combined version of (a) and (c). Essentially, we manipulate the underlying density in the vicinity of the contours, shown in the top image of (c).

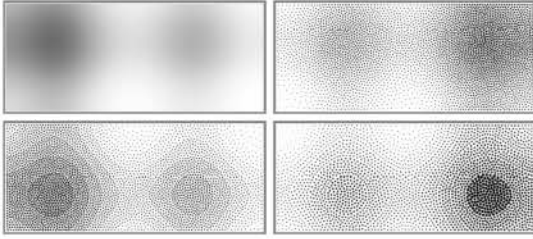


Fig. 7. Stippling techniques applied to different quadrants of the symmetric *Eggcrate* function. Top-left: target density, top-right: stippled with the unmodified algorithm, bottom-left: contours using Mach-banding, bottom-right: contours using restricted stippling approach.

**Topographical Data**—Figure 8 shows geographic height data of Australia. The dataset [42] is based on measurements collected during the *Shuttle Radar Topography Mission (SRTM)*. We use a quantized height map with four bins to guide our restricted stippling approach. Here we invert the usual point size mapping by using larger points in lower areas and vice versa. This allows us to display the (usually small) high altitude regions with more detail. The resulting visualization shows not only the different densities, but also emphasizes the contours and thus provides an overview of the topography, similar to contour lines on maps. Additionally, we encode non-quantized height information as stipple color to make differences more apparent. While double encoding is usually avoided in visualization, it can help people with color vision deficiency distinguish heights.

**Cartography**—In cartography, it is common to show the relationship between multiple variables. If one of these variables is continuous, our stippling approach can be used to compute a simplified, but still visually representative illustration of the underlying co-domain. Because our technique supports multiple channels, we can encode additional variables to create a bivariate visualization. An example of this approach is shown in Figure 1: The stipple distribution reflects the population density of the United States of America, as recorded by the 2010 census [43], whereas the color of the stipples shows the approximate vote share during the 2016 presidential election [44]. As we can see, the population density has a big influence on the result of the election. Here, our approach has certain advantages over existing techniques. Election results are often visualized with choropleth maps, where the color of an entire administrative region is based on vote share. However, this representation

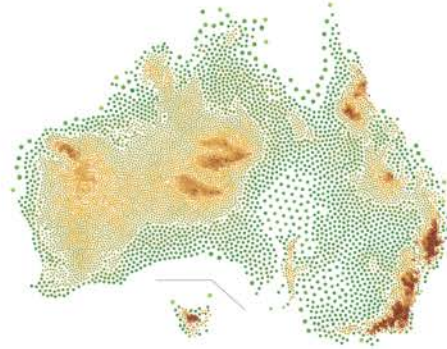


Fig. 8. Elevation data of Australia created with our restricted stippling approach. The data is quantized to four bins; each level is represented with a discrete point size to make the distinction more clear. Additionally, the non-quantized height is used to color-code each stipple.

can be misleading because some regions might be large in area, and therefore visually dominant, but comprise only a small amount of the entire population. Our method samples the population and creates an abstract representation of the distribution and thus does not share this problem. Stipples aggregate the underlying data, therefore the distribution of colors contained in the Voronoi cell of each stipple needs to be carefully sampled as well. Our visualization was inspired in parts by the 2016 *Election Map* webcomic from *xkcd*.

Bivariate proportional symbol maps are often used to alleviate the area problem of choropleth maps—they allow us to encode information in the color and size of the symbols. Figure 9a shows an excerpt of the same dataset with circles of different radii representing the population for each county and color representing the difference between the vote percentages. Figure 9b depicts the corresponding stipple drawing for comparison. A common problem with proportional symbol maps becomes immediately visible: the overlap between nearby symbols. Finding the right drawing order to reduce the overlap is a non-trivial problem [45]. Overlap cannot occur with our approach (or to a very limited extent) due to the space-partitioning property of Voronoi diagrams used by our algorithm.

**Isocontours**—In Figure 10, we show a comparison of our contouring techniques using a stippled heightmap without color of *Mount Fuji* in Japan [42]. Both techniques emphasize the differences in height. This helps to highlight the slope of Mount Fuji and the large plateaus in its surrounding;



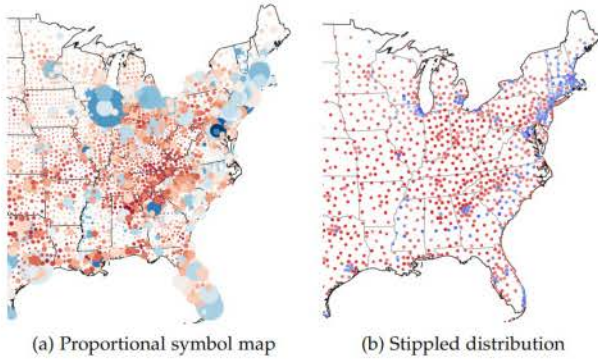


Fig. 9. Comparison between a bivariate proportional symbol map (a) and the corresponding stippled representation (b). Proportional symbol maps can suffer from overlap, making it hard to make out the underlying distribution. The stippled representation does not have this problem, however the total maximum of the distribution is harder to spot.

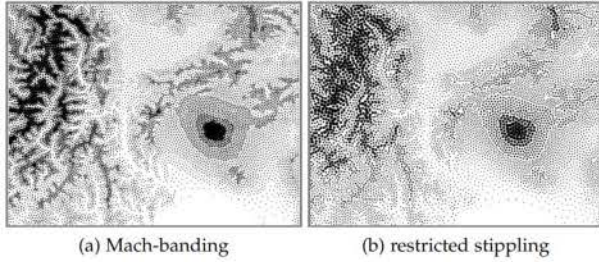


Fig. 10. Elevation data of the Akaishi mountain range and Mount Fuji in Japan. The Mach-banding technique (a) highlights ridges and valleys, while the restricted stippling approach (b) shows contours more clearly.

the ocean in the south is the lowest part of this region and therefore not covered with stipples. We use five isocontours for both results with the minimum and maximum point size set to 3 and 12 respectively. Here, a low number of contours facilitates comparability between our contouring techniques. Creating boundaries too close to each other in the restricted stippling approach would result in empty regions because the algorithm is not able to distribute points within the small region between adjacent boundaries. In contrast, the Mach-banding approach does not share this problem because boundaries are created implicitly—points of different contours can freely move between them. While borders are more emphasized with restricted stippling, finer details of the height differences are more apparent when using Mach-banding, as can be seen on the left. However, due to the contrast enhancement, these details are also exaggerated. Compared to Figure 8, the point size mapping here is inverted. We use visually more prominent larger points for higher altitude regions to steer the attention towards the peak of Mount Fuji, while preserving the structure of the mountain range to the left.

**Stippled Scatter Plots**—Traditional scatter plots often suffer from over-plotting: many points are plotted on top of each other in dense regions, obscuring the underlying distribution. A standard technique to tackle this problem employs kernel

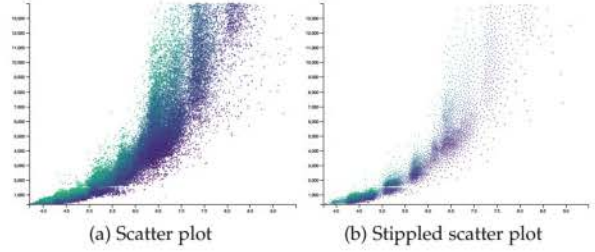


Fig. 11. Visualization of the diamond dataset, showing *price* over *width*. Color represents *clarity* of each diamond. In (a) the dataset is depicted using a regular scatter plot with all 51,772 data points, while (b) shows the result of our stippling technique, reducing the amount of points to 3,389, showing the actual distribution more clearly.

density estimation or transforms an existing continuous data domain, e.g., velocity, to the image domain [46]. The resulting density field is then colored using a transfer function which conveys an impression of depth and opacity. However, this approach has a major drawback: areas with low point density suffer from low contrast, making it difficult to spot outliers.

Our proposed technique can be used to avoid both of these problems by sampling the data points. Figure 11a shows a traditional scatter plot of diamond properties, namely *width* and *price*. Color is used to encode the *clarity*. In this example, the representation suffers from a lot of over-plotting in the lower regions. The stippled representation behaves differently, as can be seen in Figure 11b. Please note how our technique reveals clusters in the data, which were concealed by visual clutter in the traditional scatter plot. With our method, we were able to reduce the number of points from 51,772 points in the scatter plot down to 3,389 points. Admittedly, the points in the stippled scatter plot do not necessarily fall onto points from the original dataset. However, other methods from information visualization, such as kernel density estimation, make the same compromise. Nevertheless, each point is a representative of the underlying distribution. One possible extension here would be to incorporate actual data locations into our algorithm. Sparse regions could use the actual data positions, whereas regions with higher density could maintain the intended stipple distribution, and everything in-between could use a compromise of both. This presents a trade-off between the accuracy of individual points and their overall distribution.

## 7 EVALUATION

In previous work, evaluation of NPR stippling techniques either targeted technical verification [21] or measuring perceived quality [47]. Therefore, our evaluation targets the area between: the relation between parameters and low-level perception. In contrast to the far more common user studies [48], our goal is not to prove the usefulness or acceptance of our technique but to discover possible problems and opportunities for improvement. Thus, we strive to answer a set of research questions related to the verification and validation of our technique [49]:

**RQ 1** How linear is the perception of stippling?

**RQ 2** How does stippling compare to grayscale?

**RQ 3** How does stippling diminish regarding perception?



Our questions cannot be answered by means of a traditional user study, because we want to acquire stimulus-response functions. Therefore, we do a combination of leveraging the idea of generative data models [50] and psychophysical methods (Section 2).

### 7.1 Experiment Design

We conducted our experiments using the crowd-sourcing platform *CrowdFlower* (now *Figure Eight*) in separate sets of browser-based micro-tasks. In all of our experiments, one question corresponds to one micro-task and instructions briefly give users an introduction to stippling using artistic images. An important factor in crowdsourcing is quality control without scratching validity: We asked test questions with obvious stimuli (far above the just-noticeable difference threshold), but still difficult to distinguish from regular stimuli. The test questions were randomly inserted, and the participant received a notification in case of an invalid answer. Participants with an accuracy rate below 70% on these questions were removed from the experiment. At the beginning of all experiments, each participant had to complete a quiz that was randomly compiled from test questions to verify the participants task comprehension.

### 7.2 Experiment: Just-Noticeable Differences

We address questions RQ1-RQ3 by determining and comparing the *just-noticeable differences* (JND) for stippled and grayscale images. To determine the JNDs, we chose a classic two-alternative forced choice (2AFC) experiment and the method of constant stimuli as known from psychophysics. Note that we consider only the original LBG stippling algorithm using three different constant point sizes to reduce the parameter space.

**Procedure**—All participants were instructed to locate the dark side of linear gradient stimuli, i.e., judge the direction of the slope between a reference point and a comparison point. We sampled 11 even-distributed reference points between 0 (white) and 1 (black), each with up to 9 comparison points at a distance of  $\{0, 0.0375, 0.075, 0.15, 0.3\}$  around the corresponding reference point. Out of bounds sample points below 0 and above 1 were dropped. We verified the usefulness of the range around the reference points in a small-scale test run. Comparison points with the maximum possible distance of 0.3 were used as test questions. Each stimulus was surrounded by a gray border to reduce the influence of contrast. Moreover, we suggested participants optimizing their viewing angle, viewing distance, and avoid light shining directly on their screens. For each stimulus, the participants were asked to decide whether the left or right side were appearing darker.

**Analysis**—We collected 23,753 responses from 841 participants in total. In a first step, we plotted absolute and relative value differences along with their average detection rate to check data quality. Then, we joined symmetric difference pairs around each reference point to determine left/right biases. This was a precautionary measure as our sampling method does not necessarily produce a symmetrical result under symmetrical parameter transformations. We fitted logistic functions to our data with a common random guess rate of 0.5 for 2AFC experiments. Based on the fitted

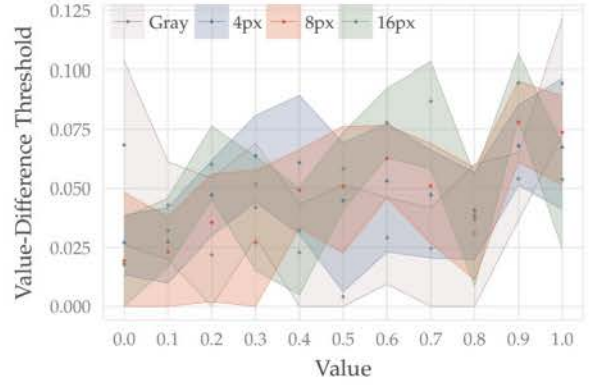


Fig. 12. Visualization of the perceivable difference for increasing grayscale values in comparison to stippling from our user study. The ribbons show the 95% confidence intervals and mean values at measurement points for a detection rate of 80%.

functions we can determine difference thresholds for 80% detection rate, depicted in Figure 12.

**Results**—Regarding RQ2, we can see that stippling outperforms grayscale maps in the lower range of values (0.0 to 0.2). This supports our claim that stippling is useful in low density regions because of the representation with discrete elements. The explanation for this is quite obvious: black dots have a much higher contrast on white background compared to a gray gradient. Grayscale outperforms stippling in the mid range of values (0.4 to 0.8) and there are almost no differences in the upper value range.

Concerning RQ1, all curves describing the perception of stippling are non-linear, with a stipple size of 8 being the closest to linear. Moreover, the dependency between the threshold and stipple size is non-linear, as can be seen by the bump (0.3 for size 4, 0.5 for 8 and 0.6 for 16). Note how the bump moves from the lower third to the upper third with increasing point size.

For RQ3 we expected stippling to perform worse with increasing point density, however this kind of bump was surprising. We suspect that there are several reasons for this: On the one hand, the algorithm tends to create regular patterns at the border due to the handling of Voronoi cells. As a result, stipples at the border become countable with increasing size. On the other hand, perceived density might also be related to frequency perception in our stipple visualizations. In Figure 13, we transform two stipple sets, having low and high constant density, to frequency space using the Fourier transform. In contrast to standard sampling in computer graphics, we do not transform the point positions but the final images that also contain the stipple sizes. The resulting frequency plots show rings that represent frequencies relating to reoccurring distances between stipples. In the case of lower density, these rings are clearly visible because the point density allows distribution of stipples with similar distances. The opposite is true for higher densities, where placement of stipples becomes more difficult for the stippling algorithm, to the point where individual stipples become almost indistinguishable. This manifests as blurred rings in the frequency plot because undesired frequencies are introduced. We expect this effect in frequency space to



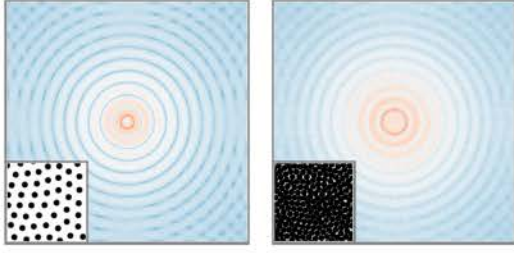


Fig. 13. Stipples with low and high density in image and frequency space. The blurred rings in the high density case (right) are due to frequencies introduced by our algorithm in dense regions. We suspect this to be detrimental for the performance in distinguishing local densities.

translate to a worse performance in distinguishing local densities. Lastly, we cannot control parameters such as display resolution, rasterization, and viewing distance of each study participant. Therefore, we assume that some combinations of these factors combined with point sizes might have led to the bump artifact. In practical terms, this means mid-range value differences should be boosted for example by employing our contouring techniques.

## 8 PARAMETRIZATION

In this section, we give guidance by discussing the impact of different parameters, provide some reasoning behind the parameter selection, and list trade-offs to help others in creating their own stippled visualizations.

**Stippling**—Choosing a smaller stipple size improves the spatial resolution and amount of preserved details, whereas a larger stipple size increases the degree of abstraction. With increasing viewing distance, smaller points become less visible, filtering out details represented with such point sizes. Choosing a variable point size, either akin to the original LBG stippling algorithm, or by specifying it via a point size map, increases the contrast in the stippled result. Typically, we used a factor of 3 to 5 between maximum and minimum point size to create a reasonable contrast in the results.

The computation time is similar to the original stippling algorithm. It mainly depends on the number of created stipples and therefore the point size. The algorithm required several seconds on commodity hardware (Intel Core i7-4790K CPU at 4.0 GHz, Nvidia GTX 980 Ti GPU) for a typical visualization in this paper ( $\sim 10k$  points). The initial number of points and their distribution affect only the runtime of the algorithm, not the quality of the result: The algorithm converges to a similar number of stipples, usually after 50 to 100 iterations. In general, our method does not significantly change the runtime of the original algorithm and adds only a small overhead. The mach-banding technique requires a pre-processing of the input, whereas the restricted stippling approach increases the memory requirement proportional to the number of contours. For a more in-depth runtime analysis, we refer to the original LBG stippling paper [21].

**Contours**—Apart from the number of contours, the Mach-banding technique has two parameters that influence the result. Recall that we use a Gaussian blur as part of our technique to modify the scalar field locally. The radius of this

Gaussian blur corresponds directly to the affected adjacent stipples. We found that using the maximum stipple size as blur radius yields good results. The weight  $w$  of the blending function used to combine the high-pass filtered version with the scalar field, on the other hand, is used to steer how much the density changes in the vicinity of the contour, making it more or less visually prominent. We used values between 0.3 and 0.8 depending on the desired emphasis of the contours.

Restricted stippling does not require parameters, but since the stipple sizes are constant for each contour, setting suitable point sizes becomes even more important. This was either done with a user-provided point size map, or by distributing the point sizes equally between the minimum and maximum point size of the LBG stippling algorithm.

**Encoding**—Our presented stippling approach allows different forms of encoding in each element (cf. Figure 14). In this paper, we mainly used size and sometimes color as visual variables. Additional information can also be encoded in shape, or in Gestalt principles when looking at groups of stipples. We already use Gestalt principles for introducing contours, which become apparent because of proximity and similarity. Combinations of these visual variables can be used to encode different information or encode the same information redundantly to further increase the perceived difference and improve the visual impression. The core of the algorithm works by comparing the density of a representational element, in our case filled circles. However, the algorithm supports arbitrary convex shapes without any changes apart from an adapted calculation of the Voronoi diagram. The application of glyphs in context of visualization has been investigated thoroughly [51]. However, we consider the exploration of this large design space for stippling to be beyond the scope of this work.

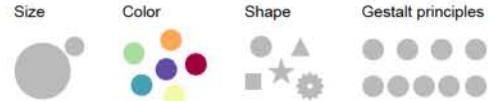


Fig. 14. Different forms of encoding that can be utilized in the stipple visualization. Combinations can be used to encode additional information or introduce redundancy to further increase the visual discrepancy.

**Density Mapping**—In Section 7 we have already derived the 80% JNDs for three constant stipple sizes (Figure 12). From this, we can give some advice regarding the representation of the data. The density mapping should be root or logarithmically scaled, because depiction works well in the lower to mid range (0.0 to 0.5)—then the bump effect occurs, as discussed in the evaluation. This effect has an unfavorable impact on the value range above 0.5 with increasing stipple size, making these densities more difficult to distinguish. As already mentioned, we can use contours to alleviate this problem. The ideal solution to this problem would be an exhaustive psychophysical evaluation of the parameter space in order to linearize it, similar to what has been achieved for color with CIELAB. Therefore, we see our experiments as preliminary work for further research.

## 9 DISCUSSION AND LIMITATIONS

Prior work [47] evaluated the relationship between the number of stipples and the perceived quality of the resulting



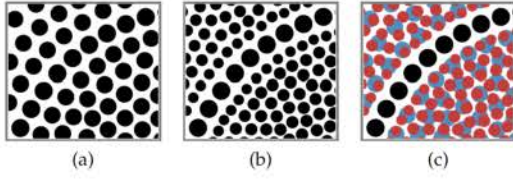


Fig. 15. Features embedded into stipplings strongly depend on the local context. For both (a) and (b), the space between the feature and the surrounding stipples is the same, but the space appears greater for the smaller stipples in (b). Both stipple densities are juxtaposed in (c).

image. Their finding is that an abstraction with more stipples leads to a higher perceived quality, however at an decreasing rate for higher number of stipples. We were able to confirm this connection in Section 7. Similar to other visual variables, such as color, we have shown that the perception of stipples as visual variable is non-linear. An important aspect of stippled images is that the local position of the stipples and their context can have a strong influence on how we understand the visualization. Figure 15 shows an example of this effect: Although the spacing between the feature and the surrounding stipples is the same, the gap is perceived larger when stipples in the neighborhood are smaller. We suspect that this difference in perception is due to how humans perceive frequency or more precisely frequency changes.

**Stippling artifacts**—There are several important characteristics of our stippling algorithm that are carried over into the visualization and could potentially introduce uncertainty regarding interpretation of the visualization. Due to how the Voronoi diagrams are calculated, stipples at the border of the visualization can end up in a configuration with a higher density than it is present in the data. To varying degree, an example of this can be seen at the border of Figure 4. One potential solution to alleviate this problem would be to use either smaller or partial stipples in boundary regions. However, it has to be investigated how much this idea affects the density perception.

Another artifact that can be introduced by the LBG stippling algorithm are hexagonal patterns. This happens when the algorithm converges slowly (due to parametrization) and the additional steps of Lloyd relaxation lead to stipples arranging in a regular hexagonal grid. While this effect is certainly unwanted in the domain of rendering, it has to be investigated how disruptive these patterns are for information visualization. The effect can be reduced by jittering the positions in a post-processing step.

**Contour techniques**—We proposed two approaches for adding structure to the stippled representation (Section 5). Both contouring techniques differ not only in their implementation but also in the visual result. Mach-banding leads to surface-like regions, whereas using the Voronoi diagram to restrict stipple position leads to lines that appear because stipples are constrained from moving too close to the region boundaries. The latter technique uses negative space as a means of adding information. Figure 16 shows the error that is introduced by the standard version of the LBG stippling algorithm for each Voronoi cell (left). The relative size error is color-coded so that red means a cell is too large and blue means too small. The error for the Mach-banding

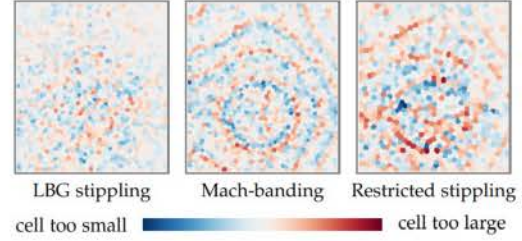


Fig. 16. Relative error of Voronoi cells visualized for an excerpt of the bottom left quadrant of the Eggcrate function (see Figure 7). The error of Mach-banding follows the changes in density, while the error for restricted stippling is more localized.

approach is shown in the middle. A higher error occurs close to the contours due to the local change in density. On the right, the error for the restricted stippling approach is shown. Since stipple movement and density integration is constrained and cannot cross a contour, an additional error is introduced which is less uniform compared to the Mach-banding approach. As discussed previously, the number of potential contours differs between the two approaches. While Mach-banding can even be used for a high number of contours, the restricted stippling approach creates artifacts when contours come too close to each other, creating regions where the stippling algorithm cannot place points.

## 10 CONCLUSION

We have presented a novel method to visualize and abstract continuous and discrete 2D data using stippling. The presented algorithm allows us to encode information in different channels of stipples, such as density, size, color, and shape. For example, we have shown how to encode additional data in the form of contour lines in the stipple drawings. This enables users to perceive information at different levels of abstraction and also in a viewing distance-dependent manner: Details are preserved at close range and can be inspected when viewing the visualization from close-up, whereas the overall structure can become more apparent at a higher distance or when viewed in a smaller-than-usual size, for example when used in thumbnails.

For future work, we want to further investigate how stippled visualizations are perceived and interpreted by the user. We see the potential for research on the influence that illumination changes might have on stippling and how task performance behaves in comparison to conventional representations such as color scales. We also imagine interesting extensions to the process of stippling. One idea is to not restrict the representation to black stipples on a white background, but also use white stipples on a dark background in dense areas, further increasing the contrast.

## ACKNOWLEDGMENTS

Funded by the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation)—Projektnummer 251654672—TRR 161 (Projects A01 and A04).



## REFERENCES

- [1] C. R. Johnson, "Top scientific visualization research problems," *IEEE Comput. Graph. Appl.*, vol. 24, no. 4, pp. 13–17, 2004.
- [2] L. Padilla, P. S. Quinan, M. Meyer, and S. H. Creem-Regehr, "Evaluating the impact of binning 2d scalar fields," *IEEE Trans. Vis. Comput. Graph.*, vol. 23, no. 1, pp. 431–440, 2017.
- [3] M. Tory, C. Swindells, and R. Dreezer, "Comparing dot and landscape spatializations for visual memory differences," *IEEE Trans. Vis. Comput. Graph.*, vol. 15, no. 6, pp. 1033–1040, 2009.
- [4] D. H. Brainard and B. A. Wandell, "Asymmetric color matching: How color appearance depends on the illuminant," *Journal of the Optical Society of America A*, vol. 9, no. 9, pp. 1433–1448, 1992.
- [5] P. Isenberg, P. Dragicevic, W. Willett, A. Bezerianos, and J.-D. Fekete, "Hybrid-image visualization for large viewing environments," *IEEE Trans. Vis. Comput. Graph.*, vol. 19, no. 12, pp. 2346–2355, 2013.
- [6] B. D. Dent, J. Torguson, and T. W. Hodler, *Cartography: Thematic Map Design*. William C. Brown, 2008.
- [7] S. Nusrat and S. Kobourov, "The state of the art in cartograms," *Comput. Graph. Forum*, vol. 35, no. 3, pp. 619–642, 2016.
- [8] C. Brewer and A. Campbell, "Beyond graduated circles: Varied point symbols for representing quantitative data on maps," *Cartographic Perspectives*, no. 29, pp. 6–25, 1998.
- [9] M. de Berg, P. Bose, O. Cheong, and P. Morin, "On simplifying dot maps," *Computational Geometry*, vol. 27, no. 1, pp. 43–62, 2004.
- [10] I. Viola and T. Isenberg, "Pondering the concept of abstraction in (illustrative) visualization," *IEEE Trans. Vis. Comput. Graph.*, vol. 24, no. 9, pp. 2573–2588, 2018.
- [11] J. Wood, P. Isenberg, T. Isenberg, J. Dykes, N. Boukhelifa, and A. Slingsby, "Sketchy rendering for information visualization," *IEEE Trans. Vis. Comput. Graph.*, vol. 18, no. 12, pp. 2749–2758, 2012.
- [12] S. Kim, R. Maciejewski, A. Malik, Y. Jang, D. S. Ebert, and T. Isenberg, "Bristle maps: A multivariate abstraction technique for geovisualization," *IEEE Trans. Vis. Comput. Graphics*, vol. 19, no. 9, pp. 1438–1454, 2013.
- [13] M. Meyer, T. Munzner, A. DePace, and H. Pfister, "MulteeSum: A tool for comparative spatial and temporal gene expression data," *IEEE Trans. Vis. Comput. Graph.*, vol. 16, no. 6, pp. 908–917, 2010.
- [14] A. Lu, C. Morris, D. Ebert, P. Rheingans, and C. Hansen, "Non-photorealistic volume rendering using stippling techniques," in *IEEE Visualization, 2002. VIS. IEEE*, 2002, pp. 211–218.
- [15] A. Lu, C. Morris, J. Taylor, D. Ebert, C. Hansen, P. Rheingans, and M. Hartner, "Illustrative interactive stipple rendering," *IEEE Trans. Vis. Comput. Graph.*, vol. 9, no. 2, pp. 127–138, 2003.
- [16] M. Goldau, A. Wiebel, N. S. Gorbach, C. Melzer, M. Hlawitschka, G. Scheuermann, and M. Tittgemeyer, "Fiber stippling: An illustrative rendering for probabilistic diffusion tractography," in *2011 IEEE Symp. on Biological Data Visualization (BioVis)*, 2011, pp. 23–30.
- [17] D. Martin, G. Arroyo, A. Rodríguez, and T. Isenberg, "A survey of digital stippling," *Computers & Graphics*, vol. 67, pp. 24–44, 2017.
- [18] S. Lloyd, "Least squares quantization in PCM," *IEEE Trans. Inf. Theory*, vol. 28, no. 2, pp. 129–137, 1982.
- [19] A. Secord, "Weighted Voronoi stippling," in *Proc. 2nd Int. Symp. on Non-photorealistic Animation and Rendering (NPAR)*, 2002, pp. 37–43.
- [20] O. Deussen, S. Hiller, C. V. Overveld, and T. Strothotte, "Floating points: A method for computing stipple drawings," *Comput. Graphics Forum*, vol. 19, no. 3, pp. 41–50, 2001.
- [21] O. Deussen, M. Spicker, and Q. Zheng, "Weighted Linde-Buzo-Gray stippling," *ACM Trans. Graphics*, vol. 36, no. 6, pp. 233:1–233:12, 2017.
- [22] D. Kim, M. Son, Y. Lee, H. Kang, and S. Lee, "Feature-guided Image Stippling," *Comput. Graphics Forum*, vol. 27, no. 4, pp. 1209–1216, 2008.
- [23] H. Li and D. Mould, "Structure-preserving Stippling by Priority-based Error Diffusion," in *Proceedings of Graphics Interface 2011*, 2011, pp. 127–134.
- [24] J. A. Stark, "Adaptive image contrast enhancement using generalizations of histogram equalization," *IEEE Trans. Image Process.*, vol. 9, no. 5, pp. 889–896, 2000.
- [25] J. L. Starck, F. Murtagh, E. J. Candes, and D. L. Donoho, "Gray and color image contrast enhancement by the curvelet transform," *IEEE Trans. Image Process.*, vol. 12, no. 6, pp. 706–717, 2003.
- [26] T. Luft, C. Colditz, and O. Deussen, "Image enhancement by unsharp masking the depth buffer," in *ACM SIGGRAPH 2006 Papers*, 2006, pp. 1206–1213.
- [27] S. Bruckner and E. Gröller, "Enhancing depth-perception with flexible volumetric halos," *IEEE Trans. Vis. Comput. Graphics*, vol. 13, no. 6, pp. 1344–1351, 2007.
- [28] M. H. Everts, H. Bekker, J. B. T. M. Roerdink, and T. Isenberg, "Depth-dependent halos: Illustrative rendering of dense line data," *IEEE Trans. Vis. Comput. Graphics*, vol. 15, no. 6, pp. 1299–1306, 2009.
- [29] A. Hertzmann and D. Zorin, "Illustrating smooth surfaces," in *Proc. 27th Annu. Conf. on Comput. Graph. and Interact. Techn.* ACM, 2000.
- [30] L. Zhou and C. Hansen, "A survey of colormaps in visualization," *IEEE Trans. Vis. Comput. Graphics*, vol. 22, no. 8, pp. 2051–2069, 2016.
- [31] D. L. MacAdam, "Visual sensitivities to color differences in daylight," *J. Opt. Soc. Am.*, vol. 32, no. 5, pp. 247–274, 1942.
- [32] J. Froehlich, T. Kunkel, R. Atkins, J. Pytlarz, S. Daly, A. Schilling, and B. Eberhardt, "Encoding color difference signals for high dynamic range and wide gamut imagery," in *Color and Imaging Conf.*, vol. 1, 2015, pp. 240–247.
- [33] C. Ware, "Color sequences for univariate maps: Theory, experiments, and principles," *IEEE Comput. Graph. Appl.*, vol. 8, no. 5, pp. 41–49, 1988.
- [34] S. Mittelstädt, A. Stoffel, and D. A. Keim, "Methods for compensating contrast effects in information visualization," *Comput. Graphics Forum*, vol. 33, no. 3, pp. 231–240, 2014.
- [35] R. A. Rensink, "The nature of correlation perception in scatterplots," *Psychonomic Bulletin & Review*, vol. 24, no. 3, pp. 776–797, 2017.
- [36] D. A. Szafr, "Modeling color difference for visualization design," *IEEE Trans. Vis. Comput. Graphics*, vol. 24, no. 1, pp. 392–401, 2017.
- [37] M. Harrower and C. A. Brewer, "Colorbrewer.org: An online tool for selecting colour schemes for maps," *The Cartographic Journal*, vol. 40, no. 1, pp. 27–37, 2003.
- [38] D. Archambault, H. C. Purchase, and T. Hoßfeld, *Evaluation in the Crowd. Crowdsourcing and Human-Centered Experiments*, ser. Lecture Notes in Computer Science. Springer, 2017, no. 10264.
- [39] S. Hiller, H. Hellwig, and O. Deussen, "Beyond stippling - methods for distributing objects on the plane," *Comput. Graphics Forum*, vol. 22, no. 3, pp. 515–522, 2003.
- [40] K. Dalal, A. W. Klein, Y. Liu, and K. Smith, "A spectral approach to NPR packing," in *Proc. 3rd international symposium on Non-photorealistic animation and rendering (NPAR)*, 2006, pp. 71–78.
- [41] A. Hausner, "Simulating decorative mosaics," in *Proc. 28th Annu. Conf. on Comput. Graph. and Interactive Techniques*, ser. SIGGRAPH '01, 2001, pp. 573–580.
- [42] J. de Ferranti and C. Hoffmann, "Digital Elevation Data," <http://viewfinderpanoramas.org/dem3.html>, 2014, last Accessed 2018-03-22.
- [43] United States Census Bureau, "Census of population and housing," <https://www.census.gov/prod/www/decennial.html>, 2010, last Accessed: 2018-03-30.
- [44] Federal Election Commission (United States of America), "Election Results for the U.S. President, the U.S. Senate, and the U.S. House of Representatives," <https://transition.fec.gov/pubrec/electionresults.shtml>, 2016, last Accessed: 2018-03-30.
- [45] S. Cabello, H. Haverkort, M. van Kreveld, and B. Speckmann, "Algorithmic aspects of proportional symbol maps," *Algorithmica*, vol. 58, no. 3, pp. 543–565, 2009.
- [46] S. Bachthaler and D. Weiskopf, "Continuous scatterplots," *IEEE Trans. Vis. Comput. Graph.*, vol. 14, no. 6, pp. 1428–1435, 2008.
- [47] M. Spicker, F. Hahn, T. Lindemeier, D. Saupe, and O. Deussen, "Quantifying visual abstraction quality for stipple drawings," in *Proc. Symp. on Non-Photorealistic Animation and Rendering*, ser. NPAR '17. ACM, 2017, pp. 8:1–8:10.
- [48] H. Lam, E. Bertini, P. Isenberg, C. Plaisant, and S. Carpendale, "Empirical studies in information visualization: Seven scenarios," *IEEE Trans. Vis. Comput. Graph.*, vol. 18, no. 9, pp. 1520–1536, 2012.
- [49] T. Isenberg, P. Isenberg, J. Chen, M. Sedlmair, and T. Möller, "A systematic review on the practice of evaluating visualization," *IEEE Trans. Vis. Comput. Graph.*, vol. 19, no. 12, pp. 2818–2827, 2013.
- [50] R. Schulz, A. Nocaj, M. El-Assady, S. Frey, M. Hlawatsch, M. Hund, G. Karch, R. Netzel, C. Schätzle, M. Butt, D. A. Keim, T. Ertl, U. Brandes, and D. Weiskopf, "Generative data models for validation and evaluation of visualization techniques," in *Proc. 6th Workshop on Beyond Time and Errors on Novel Evaluation Methods for Visualization (BELIV)*, 2016, pp. 112–124.
- [51] R. Borgo, J. Kehler, D. H. S. Chung, E. Maguire, R. S. Laramée, H. Hauser, M. Ward, and M. Chen, "Glyph-based Visualization: Foundations, Design Guidelines, Techniques and Applications," in *Eurographics 2013 - State of the Art Reports*, M. Sbert and L. Szirmay-Kalos, Eds. The Eurographics Association, 2013.