First publ. in: Computer Graphics Forum, Vol. 19 (2000), Number 3

Floating Points: A Method for Computing Stipple Drawings

Oliver Deussen[1], Stefan Hiller[1], Cornelius van Overveld[2], Thomas Strothotte[1]

[1]Faculty of Computer Science, University of Magdeburg, Germany
[2]Department of Mathematics and Computer Science, Eindhoven University of Technology, The Netherlands

Abstract

We present a method for computer generated pen-and-ink illustrations by the simulation of stippling. In a stipple drawing, dots are used to represent tone and also material of surfaces. We create such drawings by generating an initial dot set which is then processed by a relaxation method based on Voronoi diagrams. The point patterns generated are approximations of Poisson disc distributions and can also be used for integrating functions or the positioning of objects. We provide an editor similar to paint systems for interactively creating stipple drawings. This makes it possible to create such drawings within a matter of hours, instead of days or even weeks when the drawing is done manually.

1. Introduction

Creating drawings by using mostly dots is a powerful and widely used illustration method. Typically, several tens of thousands of dots are manually arranged to generate a single drawing. A variety of methods for placing dots and controling their size are used to represent different materials. The technique is referred to as stippling (see Figure 1).

In comparison to a photograph, artists are often able to represent a given object much clearer by a stipple drawing. Carefully placing dots allows to represent the specific surface structure as well as many kinds of artifacts. In some textbooks on archeology almost 80% of the illustrations are stipple drawings[12], even though good photographs are available. By their very nature, the drawings are easy to reproduce and inexpensive to print.

The only reason which prevents stipple drawings from being used much more widely is their expensive creation. To generate a good and large stipple drawing is a very time consuming process which might take days and weeks[13]. In some cases the artist must be able to place dots of constant density without visible patterns over a large area, in other cases slight changes of tone must be created. To use the full dynamic range of ink on paper, very dark areas have to be maintained where the small spaces between dots have to be distributed carefully. A dot which is set to the wrong position must be removed by a cutting knife; often large areas are destroyed by some misplaced dots.

Though expensive to create manually and powerful to use, computer scientists working on non-photorealism have not focused explicitly on this topic so far. In this work we close this gap by providing an interactive system that helps artists to create stipple drawings efficiently. Before discussing the related work in computer graphics and image science, we first want to sketch the different scenarios in which an artist has to create a stipple drawing:

Sometimes an image of the object to be stippled is given. In this case the process of creating a stipple drawing can be seen as a kind of halftoning. In this paper we describe a semi-automatic way of generating stipple drawings which have halftoning property when such a reference image is given. This is achieved by applying a specialized halftoning method that allows to place initial dots of the stipple drawing. The initial dot set is later refined during the artistic work on the drawing.

An important application of stippling is found in archeology. As the method is good for drawing materials like stone or cloth, it is widely used for the illustration of ancient scenes[12]. In this case, no initial image is given and stippling is an interactive, drawing-like process.

Based on our algorithm for computer generated stipple drawings we developed an interactive system that enables

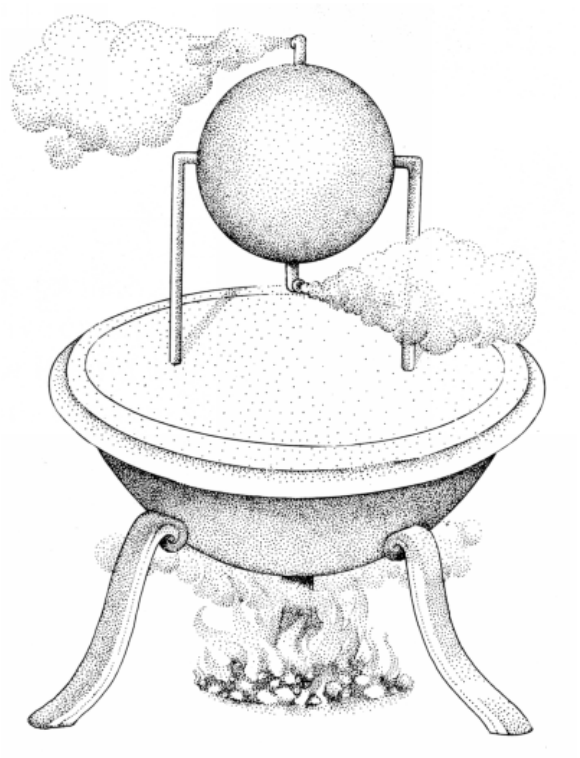


Figure 1: *A manually stippled object (a steam turbine) from a textbook on archeology*[12].

the user to "paint" a scene by manipulating a large number of stipple points interactively. Beneath evenly spaced dots, the user is able to create typical point patterns that are used to represent different kinds of materials; lines and curves can be added to complete the drawings.

2. Related Work

As motivated above, stipple drawings have a strong relation to halftoning as they represent a given tone by dots. On the other hand, they are a special kind of non-photorealistic rendering, a field in computer graphics which has been examined by many authors in the past years.

2.1. Non-Photorealistic Rendering

Salesin, Salisbury, Winkenbach et al. elaborated a variety of methods to create pen-and-ink illustrations on the basis of procedural textures[28, 24, 29, 25]. These textures consist of hatching strokes and allow to represent any given tone by appropriate stroke patterns. Besides representing the tonal value the patterns also must be generated dependent on the desired image resolution[23] and the material to be represented. The methods view the process of generating a hatching as a kind of halftoning[25], which is also done by Deussen et al.[3] for drawing hatching lines generated by intersections between a given 3-d geometry and a set of planes. A similar idea was presented by Ostromoukhov[21] who directed hatching lines by user-defined parameter mates.

In several synthetic line drawings, authors used dots –among other methods– to illustrate objects. Elber[7] gives methods for drawing implicit and parametric surfaces, also Winkenbach et al.[29] used dotted objects in some of their computer generated illustrations.

In all of the above articles dots represent the tonal values of given images or surfaces, but none of them takes into account the second ingredient of a stipple drawing: the geometric relation between the dots. Pen-and-ink artists have developed sophisticated techniques to represent the tonal value of a surface by evenly spaced dots. This enables them to achieve very smooth and clear illustrations which so far could not be generated by computer.

2.2. Halftoning

Like stipple drawings, today's sophisticated halftoning techniques are able to represent grey-scale images by even and visually pleasing pixel sets, but stippling is different: A smaller number of relatively large dots is used which vary in size and sometimes in shape. Dots are often oriented along object borders and are used to enhance details. A variety of different styles is known. Often additional lines are drawn and the stipple points must interact with those lines, either by keeping a predefined distance from the lines or by smoothly adhering to the lines.

Nevertheless, if an initial image of the object is given we can benefit from halftoning techniques as the algorithms are able to provide good initial dot distributions for our stipple points. Ulichney[26] compares several algorithms and introduces so-called blue noise distributions[14] which place dots randomly but with constraints to the nearest neighbour distances. These algorithms use error diffusion[10] that smears the locally induced quantization error to a neighbourhood. Several improvements deal with generating visually pleasant dot patterns in light and dark areas[9] and allow to enhance edges[15].

Another interesting halftoning method is Pulse-Density Modulation, where small areas of the image are integrated in a certain order and the grey tone is approximated by white pixels[8]. The generated patterns also appear visually smooth, but like the above methods, interaction with the image content or additional drawing elements is not given. Nevertheless, the method is a good choice for the generation of initial stipple dot distributions as PDM works especially good for dots of arbitrary position and varying size.

A method of halftoning that also produces large dots with varying size is clustered dot dither[26]. If the dots are placed randomly, dot shapes similar to stipple drawings arise. Ostromoukhov elaborated two methods for generating dither

matrices with random but smooth dot distributions: in[19], a random point set is connected by springs and the positions are altered iteratively by force relaxation, in[22] the positions are computed by a random space-filling curve. The polygonal shape of the dots is then computed according to the shape of the Voronoi areas of the points.

We propose a similar method to position stipple dots in our drawing. An initial dot distribution is moved by iterative relaxation, but instead of using forces or a pre-defined space filling curve, we move each point to the center of gravity of its Voronoi region. Using this method offers several major advantages: the interaction with a given polygonal border is very simple, no damping has to be used like in the force model, additional constraints can be introduced easily to the relaxation, we do not have to take care about spring topology during interaction and, as our stipple points are limited to a minimal and maximal dot size, the density of the distribution must be modified locally, which is not possible with the proposed space filling curve.

After a closer look on traditional stippling, our method is described in Section 4 where we also present our interactive editor. In Section 5 we show how to achieve different stippling styles in order to represent different types of materials. Section 6 presents selected results.

3. Traditional Stippling

An artist creates a stipple drawing by locally placing dots. He/she has to take care that no macroscopic patterns arise, which is done by using a reduction lens during drawing. The lens allows to observe directly the tonal value of a region by optical reduction.

It is a highly time consuming process until the result is reached: Very small variances of the density lead to the impression of darker or lighter regions. Also, local artifacts like points touching others must be avoided as long as the overall darkness of the region allows this.

Among evenly spaced dots, jittered patterns are used sparsely to render materials like flesh, feathers or fur[13]. These patterns are mostly used in places where a dark tone must be achieved. In general, the following degrees of freedom are used in stipple drawings:

- **Dot spacing**
  Points are usually distributed randomly but nearly evenly spaced; sometimes jittered distributions are found. There is also a method where regular point patterns are used. For several materials, placement of stipple dots has a main direction, sometimes lines of stipple dots are combined to some kind of cross hatching. We will term this as using *stippling styles*.
- **Dot size**
  Point sizes may vary for lighter and darker regions. According to our observation the largest dots are up to about twice as large as the smallest dots. Artists achieve this by varying the pen pressure during drawing.
- **Dot shape**
  In some illustrations the shape of the individual dots varies, while in others very regular shapes are found. By holding the pen perpendicular to the paper artists create regular shapes; by drawing quickly with a smaller angle shape variations are introduced. Sometimes special paper is used for additional variations.
- **Inverse drawing**
  If very dark regions have to be generated, an inverse process is performed. The background is drawn in black, and white stipples are used.

The ability to generate stipple drawings by computer requires solutions to several subproblems: besides the already mentioned creation of evenly spaced but random point sets we need to represent grey-scale tone properly by dot density, dot size and dot shape; plus, we have to find efficient and intuitive manipulation methods for interactively editing dots. These will be described in the next section.

4. The Stipple Editor

Again: stippling is more than generating a set of randomly spaced dots representing a given image. An artist enhances parts of the image, introduces borders; he wants to move, delete, and change points interactively. We provide an interactive editor that supports these operations.

To be intuitive, our editor was designed similar to a conventional painting program. The user operates on the given dots by a set of “brushes” that modify their appearance and allow to add or remove dots. The user should be able to use the system irrespective of whether a reference image is or is not given. Therefore, we divide the overall process into several steps:

First, a segmentation of the drawing area has to be performed to separate areas of different stippling styles. Each area is polygonally bound, each stipple dot later belongs to one area. If an initial grey-scale image is given (case one in the introduction) or the 3-d data is available, segmentation might be done automatically, but our experiences in this direction are not very promising.

Each area is now processed separately. If a certain stippling style should be applied, the area is covered with additionally specified sets of polygons. These are used to constrain the movement of dots during relaxation (see Sect. 5).

The next step is to define an initial dot distribution in each segment. If a reference image is given, the dot positions are obtained by applying a halftoning method (see Section 4.2), if not, the user adds points manually by a special brush that fills in a specified number of points per second.

These dot sets are now processed in order to form an evenly-spaced distribution. Therefore we apply our Voronoi-based relaxation method which is described in Section 4.3.

The algorithm is either used globally on the selected area or locally by a so-called relaxation brush which moves only points under its brush shape. During this step, the points move according to the given constraints. Contouring and the enhancement of details is done by changing the dot sizes and/or dot shapes by applying other brushes. Points might also be moved or deleted, additional drawing lines can be inserted.

4.1. Brush types

Manipulating the set of dots is done by the following brush types, each type is used to define one or more different specific brushes. The user selects one of the brushes by opening a toolbox similarly to a conventional paint program. If the brush is selected, the user is able to determine its size. Currently, all brushes have circular shape, future works will include other shapes.

- **Edit brushes**
  Adding or removing points is done either on a point-by-point basis or by a brush which operates on a given number of points per time step. Dots can also be marked as fixed, which prevents them from being affected by subsequent moving operations.
- **Relaxation brush**
  Using this brush the user is able to relax the positions of stipple dots in order to achieve an evenly spaced distribution. After insertion or deletion of points the spacing of the dots is visually disturbed. The brush is moved over the dots and the relaxation is applied to all underlying points.
- **Jitter brush**
  Sometimes points are too regular. For example, the user wants to eliminate a visual border or change the even spacing in a region. Using the jitter brush adds a small random offset to all active dots. This offset value is a percentage of the average point-to-point distance.
- **Shape brushes**
  Shape brushes allow to modify parameters of dots. The user is able to enlarge/reduce points by a given percentage or set their size directly. This is helpful for changing the tone of a region slightly or for generating strong dark borders.
  Another possibility is to affect the shape of the dots by introducing or reducing deformations of the circular shape. The shape variation can be done randomly or with a given main direction to visualize anisotropic materials.

4.2. Obtaining Initial Dot Positions by Halftoning

We investigated several halftoning algorithms like the Floyd-Steinberg method to generate a dot distribution from a given grey-scale image[10], but then we switched to Pulse-Density Modulation (PDM), applied by Eschbach[8] to halftoning, as it generates visually pleasing results and adapts easily to our needs.

Two-dimensional PDM generates a pulse of size S by growing a region A_m of the input image $I(x,y)$ until the measured value

$$I = \int\int_{A_m} I(x,y)\ dxdy \tag{1}$$

reaches the amount $I = SI_0$, where I_0 is a normalization constant. The integration areas A_m have to be chosen in a way that they do not overlap and that constant input functions generate evenly spaced center points for the pulses. Eschbach[8] describes the integration areas by the order in which neighbouring pixels are integrated, circular and triangular areas were used. We developed a modification of PDM which fulfills the needs that arise with stippling. The following modifications were made:

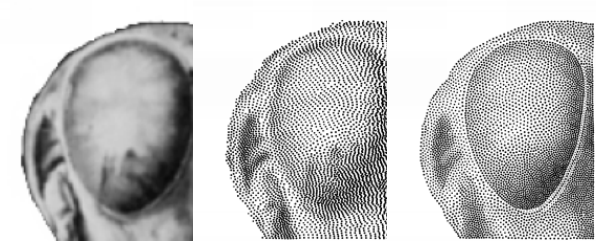
- Most of the problems with PDM arise on a rastered medium when the pulses have to be spatially quantized. In our case this is not necessary as long as the printer resolution is much below the size of the dots. Our PDM uses subpixel positioning for the pulses by super-sampling.
- As mentioned in Section 3, dots may vary in their size and, if center-to-center spaces are too large, dots are omitted. Size variation is implemented by enlarging dots if the integration areas A_m are getting too small and by reducing dot size in case that the covered areas are too big.
  If the size of A_m extends over a second level, the dot is not included in the output list though the covered region is marked as to be processed.

PDM works only well for regions with the grey level above 50%, in darker areas patterns occur. This effect can be seen in Figure 2 where an image of a grasshopper's eye was processed. In Figure 2(b) the result of applying the modified PDM is shown. The resulting dot patterns in the dark areas are usually reduced by combining PDM with an error diffusion process and inverse drawing[8]. In our approach we take them and apply the relaxation as described in the next section. This generates visually pleasing results even in these regions.

4.3. Voronoi-based dot relaxation

After the initial point distribution is obtained, the points are moved by an iteration scheme. This removes all artifacts introduced by halftoning or manual positioning. The iteration in its basic form is very simple and relaxes the nearest neighbour distances. It is known as Lloyd's method[16], and has applications in many areas (see below). The following steps are performed:

1. compute the Voronoi diagram[2] of the points. This assigns a Voronoi region to each point. The boundaries of these regions are (possibly open) polygons;
2. intersect the Voronoi regions with the boundaries of the region to be stippled;
3. move each point to the center of mass of its Voronoi region.



(a) (b) (c)

Figure 2: *Steps in creating a stipple drawing of a grasshopper's eye: a) reference image; b) initial dot distribution by PDM; c) resulting dot set after applying the relaxation to the interior of the eye and to the outer dots separately. The border between the two segments was enhanced manually with the editor.*

In the second step, the Voronoi regions open so far are closed, which prevents points from escaping from the drawing region.

In Figure 2(c) the relaxation is applied to the initial dot distribution. The dot set was processed in two separate regions. At first, the eye was enclosed by a polygon and all the dots in the interior were relaxed. Then the outer dots were processed. Later some of them were reduced in their size to enhance the contrast around the eye.

We also tried to move each point to the center point of all Delaunay neighbour points. These points are the ones that are connected to the point by edges of the Delaunay triangulation. The Delaunay triangulation is dual to the Voronoi diagram. For every two Voronoi regions touching each other an edge is drawn between the corresponding points[2].

This kind of iteration offers two main problems: First, this works not for points at the border of the set, second, the iteration is not stable.

The reason is the following: During iteration the number of Delaunay neighbours sometimes changes for a point, the corresponding change to its Voronoi region and its center of gravity is small because the inserted edge is usually very short. Instead, the appearance of the new point has a strong effect on the calculation of the center point of all adjacent edges which causes oscillations.

The small variance in the resulting sets seems to indicate approximate Poisson disc distributions. These distributions provide random points with the points not closer to each other than a given distance[11]. Such distributions are, as stated by Mitchell[18], expensive to create and useful for many purposes, like integrating functions or halftoning. In the appendix we statistically analyze our relaxation results with respect to their nearest neighbour distances and their spatial frequencies.

Several authors have also used Voronoi diagrams for relaxation: General Voronoi-based optimization schemes were already described by Okabe et al.[1]; Deussen et al. use the above iteration for knot placement in finite element mechanics[5] and for the even distribution of plants[4]. Lyons et al. use a similar scheme for graph drawing[17], Walter et al.[27] calculate coat patterns of animals by the help of relaxed Voronoi areas. A survey on centroid Voronoi tessellations with some proofs and numerical results of Lloyd's method was given by Du et al.[6].

As mentioned in the introduction, Ostromoukhov[20] proposes a similar method that modifies a given point distribution by relaxation. He uses springs of uniform rest length between Delaunay neighbors and moves the points by forces. Our relaxation method needs no integration, no damping must be used to prevent oscillation. While the force relaxation was only applied to square fields, in our case the border is a polygon which needs a good treatment of border points. This is easy with our approach but difficult to do with a spring-mass system.

Computational Effort

Calculating the Voronoi diagram for n points is performed in $O(n \log n)$ steps on average by using a sweep line algorithm[2]. The calculation of the center of masses is done in $O(n)$ as it linearly depends on the number of edges in the Voronoi diagram, and only $O(n)$ edges are found in the Voronoi diagram itself.

Therefore, $O(n \log n)$ is the overall time complexity for one step of the iteration. Given a good initial value, in practice it takes no more than 10 to 20 iterations to get a visually pleasing distribution from a random or halftoned point set and, as mentioned above, often the iteration is performed locally on the selected area with only some hundreds of dots. Table 1 shows the duration for one relaxation step for several point sets.

Table 1: *Time needed for one relaxation step (measured on SGI Octane R10000 195 MHz)*

# Dots:	100	1,000	5,000	10,000	50,000
Time (s):	0.07	0.57	2.9	6.16	33.4

After some iterations, the points move slowly and the topology of the Delaunay triangulations changes only at a few positions of the whole net. Instead of recalculating the whole Voronoi diagram for each step, it is sufficient to update only the local changes by edge swapping in the triangulation. This will increase the speed but has not been implemented yet.

5. Stippling Styles

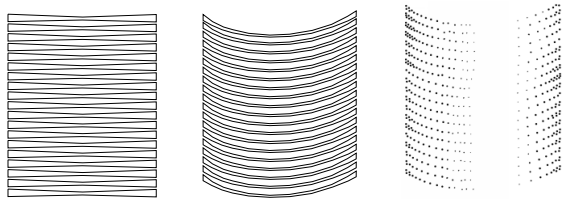
The relaxation tends by its very nature to distribute dots evenly over the given stipple area. This is mostly used but only one distinct way of stippling. By looking at traditional drawings, a number of different stippling styles can be found.

5.1. Introducing Randomness

As mentioned in Section 3, artists often use jittered patterns to stipple materials like flesh or stone. Hereby, randomness is used at different levels. By altering the dot shape, additional high spatial frequencies are introduced to the image. This is used to achieve roughness. If the dot shape is changed by using a main direction for the deformation, anisotropic materials can be simulated.

The second way of jittering is to move dots away from even positioning. This forms micro-clusters which introduce lower spatial frequencies to the image. Clustering is used to visualize inhomogeneities of medium size. For large inhomogeneities, such as a wall of natural stone, dot sizes are also altered over larger areas.

The first task is implemented by deforming the dot shape for the dots to be printed, the second by applying a jitter vector to each dot in dependency to the size of its Voronoi area. This moves dots in dark areas less than those in lighter areas and helps to avoid too much irregularity.



(a) (b) (c)

Figure 3: *Generating dot patterns (a) a given polygon set; (b) deformed polygons; (c) result after dot relaxation in polygons.*

5.2. Patterns

For a variety of anisotropic materials and also for the illustration of geometric shape, dots are positioned along lines or in thin areas. While in these areas even spacing is used, the whole dot distribution is highly inhomogeneous.

According to our idea of polygonal separation of stippling areas, these patterns are generated by adapting pre-defined sets of polygons to the drawing. The inspiration was taken from Ostromoukhov[21] who used parameter mates that are deformed to direct strokes. After deforming a given set, the polygons are used to constrain the movement of the dots.

In Figure 3 the process is shown. A given polygon set is deformed and placed on the drawing. Points are filled in and relaxation is performed simultaneously on the points in all polygons.

Future works will include a mechanism to relax dots along lines. This is helpful if dots are to be distributed strictly in a row. In this case, the relaxation is performed in only one dimension.

6. Results

In Figure 4(a) a grasshopper was stippled. The initial distribution was determined by halftoning, stipple regions were introduced manually. It took our system about eight hours to generate the grasshopper. Most of the time was spent by defining stipple regions and fine tuning of dots. Figure 4(b) shows a part of a machine cover, here stippling was used very much like halftoning.

In Figure 4(c) a natural stone bridge was shaded by 12,000 points of different size and shape. It took approximately two hours to generate the image with our editor.

Figure 4(d) shows a stipple drawing of a small ancient statue. The reference image was a photograph taken from an archeological atlas. 6,000 dots were used and 4 hours were needed to generate the final drawing. In this case the dots are taken relatively large to demonstrate that good drawings can be achieved by a small number of dots.

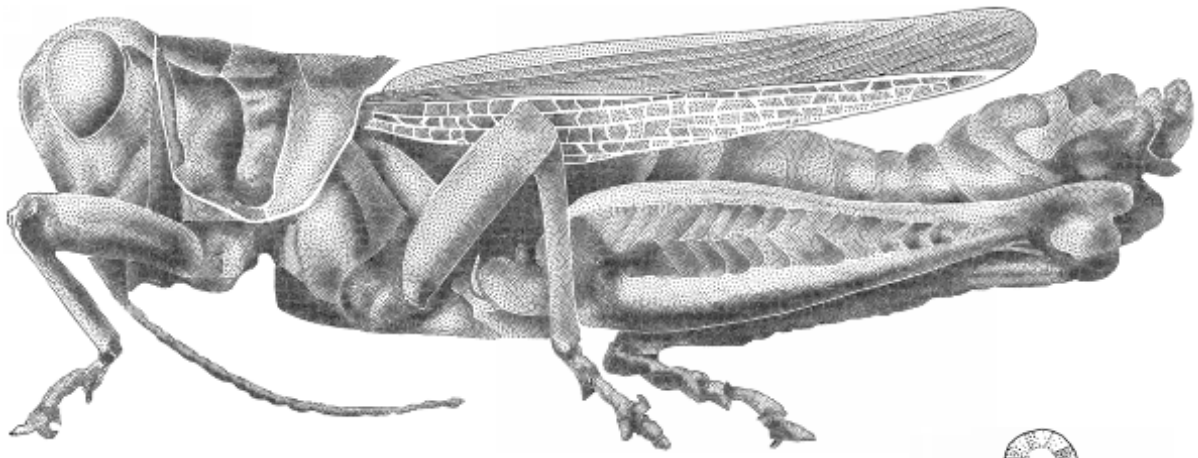
7. Conclusion and Future Work

In this paper we presented a method for computing stipple drawings. We introduced an interactive stipple editor similar to painting systems that was used to generate such drawings. The time needed is between minutes and some hours for complex models.

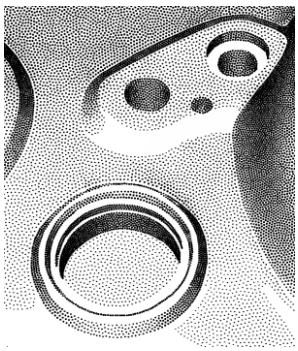
The approach computes an initial dot distribution by a specialized halftoning technique if a reference image is given. Otherwise the user creates an initial distribution manually. This dot set is modified automatically or semi-automatically to generate a final distribution similar to a stipple drawing. We used points of different size and shape to resemble manual drawing, in the Appendix we motivate statistically our point distributions to be similar to manual ones.

From the methods developed to date we can derive some areas of further research:

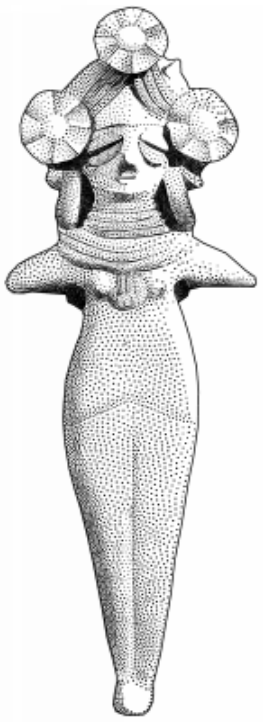
- **Automated segmentation**
  The most time-consuming operation when generating a stipple drawing is to manually define the areas which have to be relaxed locally in order to enhance the visual expression of the drawing. An illustration like Figure 4(a) has dozens of such areas. Automated image segmentation could help finding such areas by computer. Even better opportunities exist when a 3-d geometry is the source of a stipple drawing.



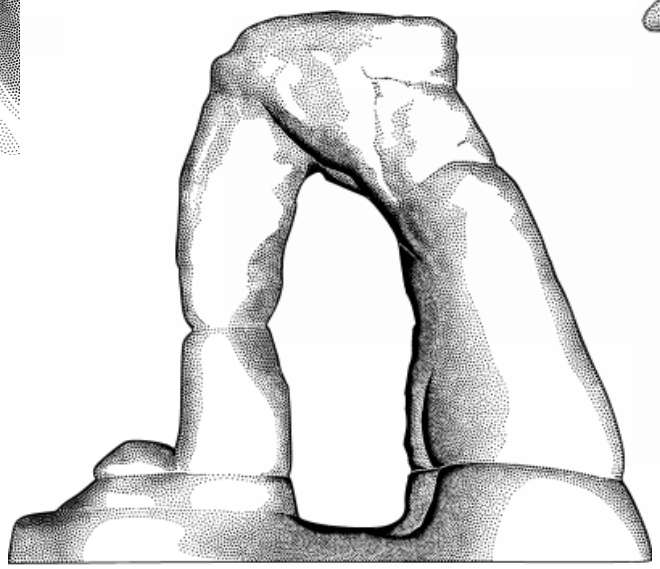
(a)



(b)



(c)



(d)

Figure 4: *Computer-generated stipple drawings: (a) Grasshopper, generated from 60,000 dots. The initial distribution was determined by halftoning, stipple regions were introduced manually. The generation time was about eight hours; (b) A part of a metal cover - stippling is used as halftoning; (c) Natural stone bridge, rendered by 12,000 dots of varying size and shape; (d) Terra cotta statue, created from 6,000 dots of varying size. Generation time was four hours.*

- **Improved Halftoning**
  A segment of the drawing to be stippled is usually bound by a polygon. The outermost points should be placed along the border. A future version of our PDM method will start along the border and place pulses there at first. This will enable us to get improved initial dot distributions. In combination with automated segmentation we hope to achieve automatic stipple drawings efficiently.
- **Level-of-detail**
  So far, the drawings are generated for a specific scale. An automatic level-of-detail mechanism based on inserting or deleting points combined with relaxation steps could help here. If a point was deleted or inserted, its neighborhood is relaxed to re-establish the even spacing. This will help us to adapt the drawings to various resolutions and output devices.

8. Acknowledgments

Thanks to the unknown reviewers for many helpful comments and to Sylvia Zabel (University of Magdeburg) for proofreading this article.

References

1. B. Boots, A. Okabe, and K. Sugihara. *Spatial Tessellations: Concepts and Applications of Voronoi Diagrams*. John Wiley, 1992.
2. M. de Berg, M. van Kreveld, M. Overmars, and O. Schwarzkopf. *Computational Geometry: Algorithms and Applications*. Springer-Verlag, 1997.
3. O. Deussen, J. Hamel, A. Raab, S. Schlechtweg, and T. Strothotte. An illustration technique using hardware-based intersections and skeletons. In *Proceedings of Graphics Interface 99*, pages 175–182. Canadian Human-Computer Communications Society, 1999.
4. O. Deussen, P. Hanrahan, M. Pharr, B. Lintermann, R. Měch, and P. Prusinkiewicz. Realistic modeling and rendering of plant ecosystems. In *Proceedings of SIGGRAPH 98*, pages 275–286, August 1998.
5. O. Deussen, L. Kobbelt, and P. Tücke. Using simulated annealing to obtain good approximations of deformable bodies. In D. Terzopoulos and D. Thalmann, editors, *Computer Animation and Simulation '95*, pages 30–43. Springer-Verlag, 1995.
6. Q. Du, V. Faber, and M. Gunzburger. Centroidal voronoi tesselations. *Siam Review*, 41(4):637–676, 1999.
7. G. Elber. Line art illustrations of parametric and implicit forms. *IEEE Transactions on Visualization and Computer Graphics*, 4(1):71–81, 1998.
8. R. Eschbach. Pulse-density modulation on rastered media: combining pulse-density modulation and error diffusion. *Journal of the Optical Society of America*, 7(4):708–716, April 1990.
9. R. Eschbach. Error diffusion algorithm with homogenous response in highlight and shadow areas. *Journal of Electronic Imaging*, 6(3):348–356, July 1997.
10. R. W. Floyd and L. Steinberg. An adaptive algorithm for spatial grey scale. *Proc. Soc. Inf. Display*, 17:75–77, 1976.
11. A. Glassner. *Principles of Digital Image Synthesis*. Morgan Kaufmann Publishers, 1995.
12. J. Hawkes. *The Atlas of Early Man*. St. Martin's Press Inc., 1976.
13. E. Hodges. *The Guild Handbook of Scientific Illustration*. Guild of Natural Science Illustrators, 1989.
14. K. Knox. Evolution in error diffusion. *SPIE*, 3648 (IS&T/SPIE Conference on Color Imaging: Device-Independent Color, Color Hardcopy and Graphics Arts IV):448–458, 1999.
15. K.T. Knox and R. Eschbach. Threshold modulation in error diffusion. *Journal of Electronic Imaging*, 2(3):185–192, July 1993.
16. S. Lloyd. Least square quantization in PCM. *IEEE Transactions on Information Theory*, 28:129–137, 1982.
17. K. Lyons, H. Meijer, and D. Rappaport. Algorithms for cluster busting in anchored graph drawing. *Journal of Graph Algorithms and Applications*, 2(1):1–24, 1998.
18. D. Mitchell. Generating antialiased images at low sampling densities. In *SIGGRAPH '87 Conference Proceedings*, pages 65–72, 1994.
19. V. Ostromoukhov. Pseudo-random halftone screening for color and black&white printing. In *Proc. 9th congress on advances in non-impact printing technologies, Yokohama*, pages 579–582, 1993.
20. V. Ostromoukhov. Pseudo-random halftone screening for color and black and white printing. In R. Eschbach, editor, *Recent Progress in Digital Halftoning IS&T*, pages 130–134, 1994.
21. V. Ostromoukhov. Digital facial engraving. In *SIGGRAPH 99 Conference Proceedings*, pages 417–424. ACM SIGGRAPH, Addison Wesley, August 1999.
22. V. Ostromoukhov and R.D. Hersch. Stochastic clustered-dot dithering. *Journal of Electronic Imaging (Special issue on Color Imaging: Device-Independent Color, Color Hardcopy, and Graphic Arts)*, 8(5), October 1999.

23. M. Salisbury, C. Anderson, D. Lischinski, and D. Salesin. Scale-dependent reproduction of pen-and-ink illustrations. In *SIGGRAPH 96 Conference Proceedings*, pages 461–468, 1996.

24. M. Salisbury, S. Anderson, R. Barzel, and D. Salesin. Interactive pen–and–ink illustration. In *SIGGRAPH '94 Conference Proceedings*, pages 101–108, 1994.

25. M. Salisbury, M. Wong, J. F. Hughes, and D. Salesin. Orientable textures for image-based pen-and-ink illustration. In *SIGGRAPH 97 Conference Proceedings*, 1997.

26. R. Ulichney. *Digital Halftoning*. The MIT Press, 1987.

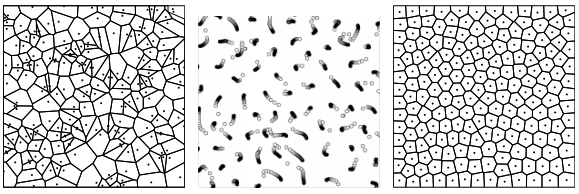
27. M. Walter, A. Fournier, and M. Reimers. Clonal mosaic model for the synthesis of mammalian coat patterns. In *Graphics Interface '98*, pages 82–91, June 1998.

28. G. Winkenbach and D. Salesin. Computer–generated pen–and–ink illustration. In *SIGGRAPH '94 Conference Proceedings*, pages 91–100, 1994.

29. G. Winkenbach and D. Salesin. Rendering parametric surfaces in pen and ink. In *SIGGRAPH '96 Conference Proceedings*, pages 469–476, 1996.

Appendix A: Statistics of Relaxed Dot Sets

Figure 5(a) displays an initial random point distribution together with its Voronoi diagram. In Figure 5(b) the movement of the points is shown, while Figure 5(c) provides the final points and their Voronoi diagram.



(a) (b) (c)

Figure 5: *Voronoi iteration: (a) initial random point set (200 points) with its Voronoi diagram; (b) movement of the points during iteration; (c) result after 20 iterations.*

Applying the iteration to the random set minimizes the variance of nearest neighbor distance as shown in Figure 6. For three random distributed point sets the variance was determined during the iteration. The values are given as percentage of the inital value.

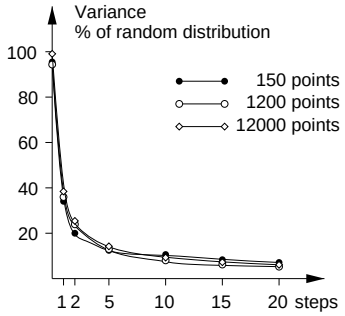


Figure 6: *Variances of Delaunay edges during the iteration. Three randomly distributed point sets with 150, 1,200 and 12,000 points were observed during 20 iteration steps.*

For all random point sets a reduction to less than 10% was achieved after ten iterations, after twenty iterations 6% percent of the original variance is achieved. In Du et al.[6] Lloyd's Method is proved to have linear convergency in the one dimensional case. The curves in Figure 6 indicate this behaviour also for two dimensions.

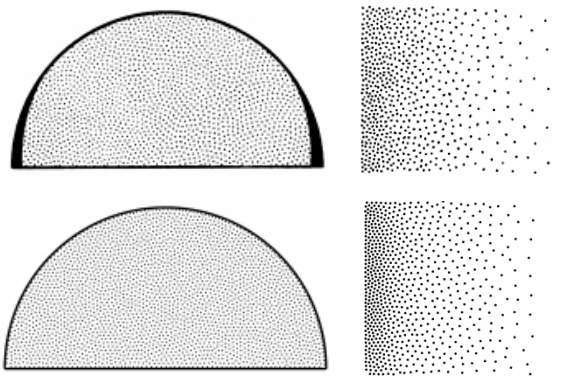
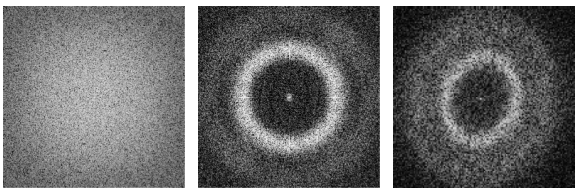


Figure 7: *Upper row: Hand-made stipple drawings, taken from*[13]*. On the left a uniform region, on the right a grey scale ramp. Lower row: Results of Voronoi iteration used for creating similar stipple drawings.*

To compare our relaxation results with hand-made stipple drawings, we scanned a uniformly stippled region and a grey scale ramp from a textbook[13]. In the upper row of Figure 7 they are shown, in the lower row similar stipple results by the iteration are given. The patterns were calculated by using PDM to generate the initial distribution and ten steps of the Voronoi iteration.

To indicate that our iteration provides good approximations of Poisson disc distributions, we have to show that the point-to-point distances are mainly above a given threshold and points are randomly distributed. The first is automatically achieved by the small variance of the Delaunay trian-

gulation after iteration, the latter is indicated by applying a Fourier transform to the images.



(a) (b) (c)

Figure 8: *(a) Magnitude Fourier spectrum of evenly distributed points. (b) Spectrum of points after 20 iterations. (c) Spectrum obtained by manually drawn points taken from Figure 7.*

In Figure 8(a) the magnitude Fourier spectrum of evenly distributed points is shown. It is nearly uniform and contains all frequencies. The spectrum of an iterated point set after 20 iterations (cf. Figure 8(b)) contains mainly frequencies within a small band without dominant directions.

The noise function of the points in this case is called “Blue Noise”[26] and indicates points to be spaced in excess of some minimal distance, and, what is also interesting, below a second threshold. The absence of dominant directions indicates that no patterns appear within the generated point sets. Figure 8(c) shows the Fourier spectrum of the manually printed points from Figure 7 which is similar to that of the iterated points.