ReScience

[Re] Weighted Voronoi Stippling

Nicolas P. Rougier[1,2,3]

1 INRIA Bordeaux Sud-Ouest, Bordeaux, France. **2** LaBRI, Université de Bordeaux, Institut Polytechnique de Bordeaux, Centre National de la Recherche Scientifique, UMR 5800, Talence, France. **3** Institut des Maladies Neurodégénératives, Université de Bordeaux, Centre National de la Recherche Scientifique, UMR 5293, Bordeaux, France.

Nicolas.Rougier@inria.fr

Editor
Name Surname

Reviewers
Name Surname
Name Surname

Received Feb, 1, 2017
Accepted Feb, 1, 2017
Published Feb, 1, 2017

Competing Interests:
The authors have declared that no competing interests exist.

Article repository
Code repository
Data repository

A reference implementation of

→ Weighted Voronoi Stippling, Adrian Secord. In: Proceedings of the 2Nd International Symposium on Non-photorealistic Animation and Rendering. NPAR '02. ACM, 2002, pp. 37– 43.

Introduction

Adrian Secord introduces in [6] a *techniques for generating stipple drawings from grayscale images using weighted centroidal Voronoi diagrams* as in *the traditional artistic technique of stippling that places small dots of ink onto paper such that their density give the impression of tone*. The paper is not accompanied by any code and, even though the webpage of the author points to a code archive, this one is actually nowhere to be found. We decided to replicate the method because it can be used for different purpose, especially in computational biology or neuroscience where the distribution of a cell population can be conveniently described by shades of grey. It is to be noted that since the publication of this paper, a number of related techniques have been proposed [1][2][3][4] that may be more efficient but with lower quality. We thus privileged this implementation that is simple and provides high stippling quality.

Methods

We applied the method proposed in the original paper with some variations. First, we did not use the GPU to generate the Voronoi diagram because this would introduce a dependency on OpenGL that would make the script more fragile and more difficult to run on all systems (Linux/OSX/Windows). However, we provide in the accompanying code an experimental implementation based on the glumpy library. As in the original article, this experimental implementation takes advantage of the fact that a set of cones seen from above using an orthographic projectionactually represents a Voronoi diagram (see [5]). This also gives "for free" a rasterization of each Voronoi region provided each cone has a distinct color.

Consequently, for all the figures in this article, we used instead the QHull library (through the Scipy Python package) for computing the Voronoi diagram together with the Voronoi regions (as a list of segments). We took care of adding extra points such that each cell is contained within a user-defined bounding box that is set to the dimension of the density image. Furthermore, each input image is resized (without

interpolation) such that the mean pixel size of a voronoi cell is 500 (e.g. for 1000 stipples, the input image needs to be resized such it contains at least 500x1000 pixels).

For computing the weighted centroid, we applied the definition proposed in the original paper over the discrete representation of the domain:

$$\mathbf{C}_i = \frac{\int_A \mathbf{x}\rho(\mathbf{x})dA}{\int_A \rho(\mathbf{x})}$$

Each cell is rasterized (as a set of pixels) and the centroid is computed using the optimization proposed by the author that allow to avoid to compute the integrals over the whole set of pixels composing the Voronoi cell. As noted by the author, the precision of the method is directly related to the size of the Voronoi cell. Consequently, if the original density image is too small relatively to the number of stipples, there might be quality issues. We used a fixed number of iteration ($n = 50$) instead of using the difference in the standard deviation of the area of the Voronoi regions as in the original paper since the definition of the rejection criterion was not clear in the original article and quite arbitrary.

Last, we added a threshold parameter that allows to perform a pre-processing of the densty image. Any pixel with a gray level above the threshold is set to the threshold value before normalizing the density image. This was necessary for replicating some of the original images (see results section for further explanation).

Results

We display only the output of our replication to be compared with the ones in the original articles. The climbing shoe (figure 1), the corn plant (figure 2) and the large Peperomia plants (figure 3) are very similar to the images displayed in the original article. It it to be noted that we used different thresholds in order to obtain white areas similmar to the original images. Without such threshold there would be no reason to have white areas as for example in the leaves of figure 3.

For the small Peperomia (figure 4), the output of our replication may be considered to be not exactly similar. If the hard edges are maintained by the stipple drawing, the differences in shading in our replication is not obvious compared to the original image. Most probably, the limited resolution of the input image may be a limiting factor but it is not clear if the author used these small resolution versions or if he used higher resolutions. We tested this hypothesis using a bigger image showing leaf with different shading (figure {#fig:leafs}) and in such a case, shading is visible in the output.

We also provide a new set of data that is freely usable for future comparison (CC0 licence).

Conclusion

Most of the results have been replicated even though some slight discrepancies remain in the final output for one image. Without further contact with the original author, it is difficult to identify the precise cause but most likely, the problem occurs because of the limited resolution of the input picture. We therefore think most of the results have been replicated.

References

[1] I Ascencio-Lopez, O Meruvia-Pastor, and Hugo Hidalho-Silva. "Adaptive incremental stippling using the Poisson-disk distribution". In: *Journal of Graphics* 15.1 (2010), pp. 29–47.

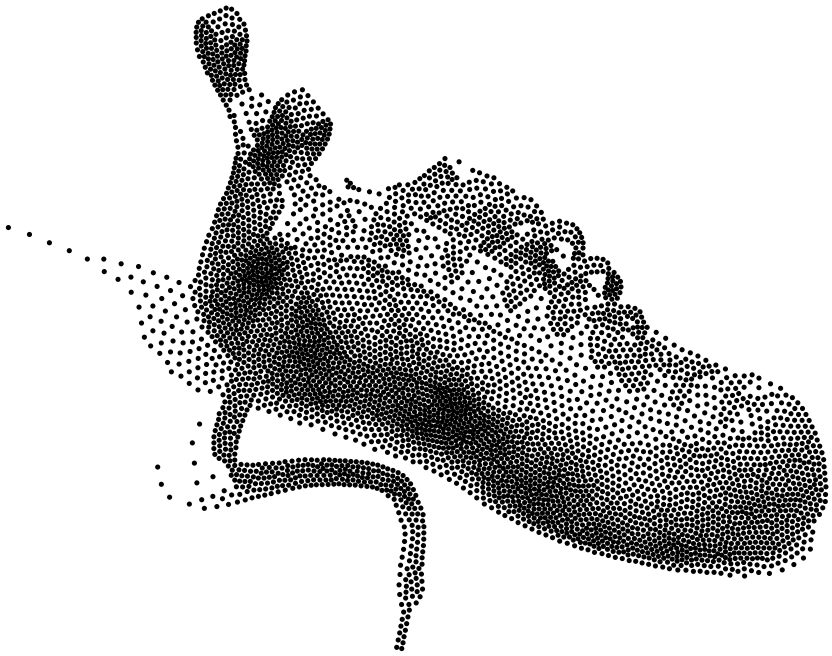

Figure 1: Climbing shoe (1300x1300), 5 000 dots, 50 iterations, fixed point size

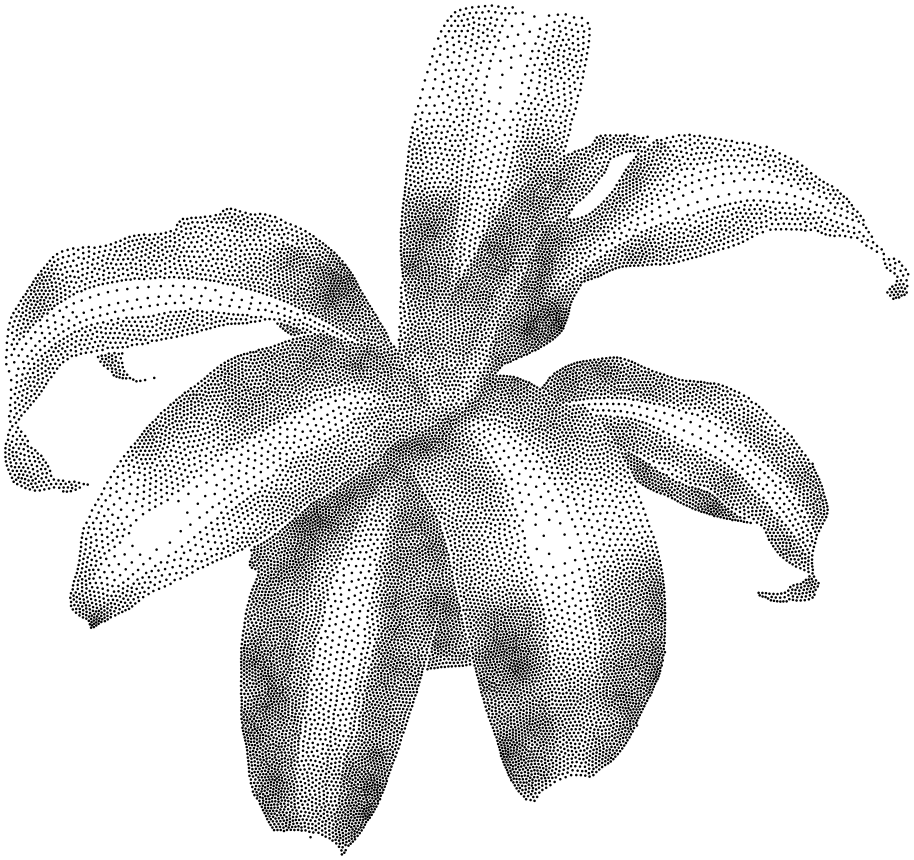

Figure 2: Corn plant (991x934), 20 000 dots, 50 iterations, fixed point size

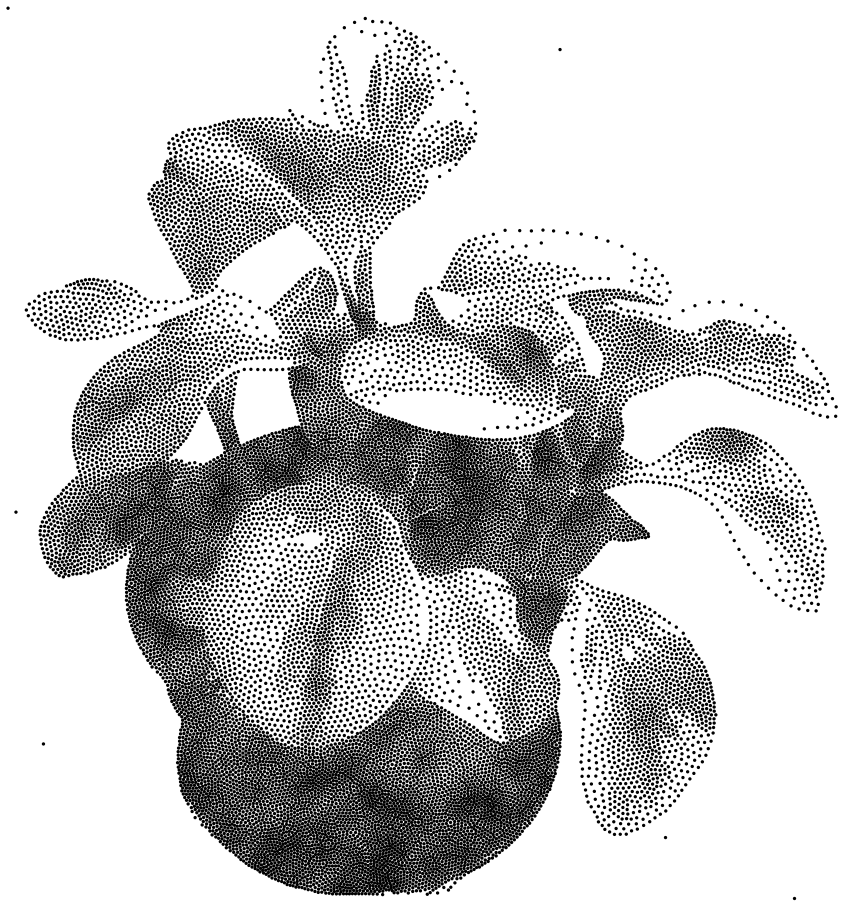

Figure 3: Large Peperomia plant (700x700), 20 000 dots, 50 iterations, fixed point size

Figure 4: Small Peperomia plant (400x400), 20 000 dots, 50 iterations, fixed point size

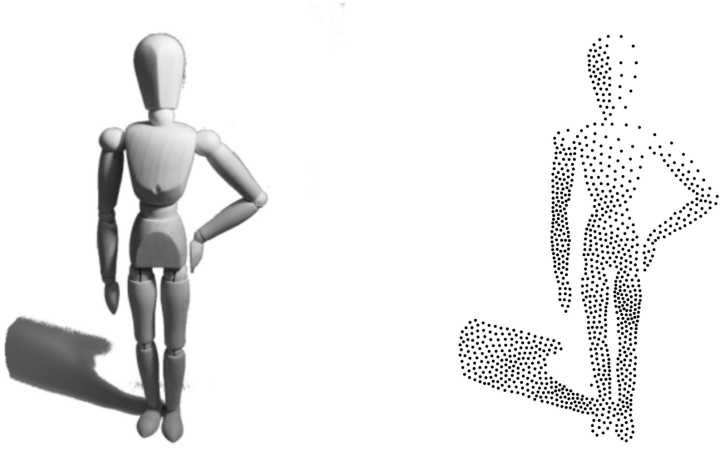

Figure 5: Figure (512x512), 1000 stipples, 50 iterations, fixed point size

Figure 6: Close-up of large Peperomia leaves (1024x1024), 20000 stipples, 50 iterations, fixed point size

Figure 7: Boots (1280x853), 20 000 dots, 50 iterations, variable point size

Figure 8: Pot plant (1195x1024), 20 000 dots, 50 iterations, variable point size

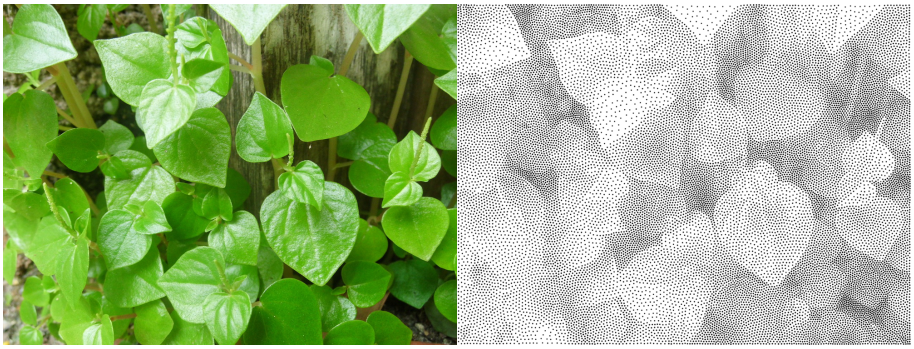

Figure 9: Leafs (1195x1024), 20 000 dots, 50 iterations, variable point size

[2] Michael Balzer. "Capacity-Constrained Voronoi Diagrams in Continuous Spaces". In: *2009 Sixth International Symposium on Voronoi Diagrams*. IEEE, 2009, pp. 79–88.

[3] Michael Balzer, Thomas Schlömer, and Oliver Deussen. *Capacity-constrained point distributions: a variant of Lloyd's method*. Vol. 28. a variant of Lloyd's method. ACM, July 2009.

[4] R Bridson. "Fast Poisson disk sampling in arbitrary dimensions." In: *SIGGRAPH sketches* (2007).

[5] Kenneth E. Hoff III et al. "Fast Computation of Generalized Voronoi Diagrams Using Graphics Hardware". In: *Proceedings of the 26th Annual Conference on Computer Graphics and Interactive Techniques*. SIGGRAPH '99. ACM Press/Addison-Wesley Publishing Co., 1999, pp. 277–286.

[6] Adrian Secord. "Weighted Voronoi Stippling". In: *Proceedings of the 2Nd International Symposium on Non-photorealistic Animation and Rendering*. NPAR '02. ACM, 2002, pp. 37–43.