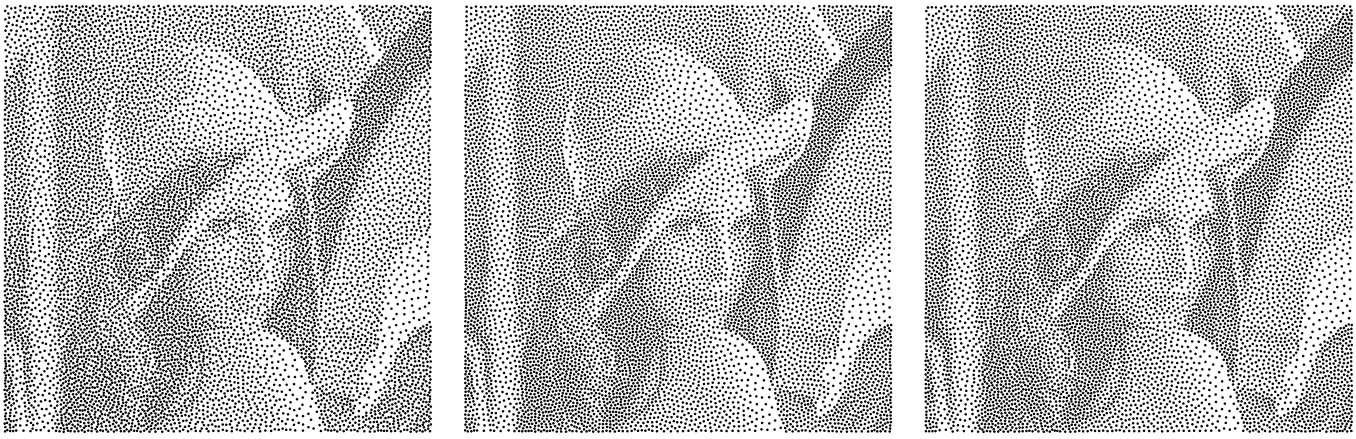
EG UK Computer Graphics & Visual Computing (2017)
Tao Ruan Wan and Franck Vidal (Editors)

Capacity-Constrained Voronoi Tessellation Revisited

Abdalla G. M. Ahmed[†] Oliver Deussen

Department of Computer and Information Science, University of Konstanz, Germany



(a) (b) (c)

Figure 1: *10k-site stippling of Lena. (a) The initial distribution of sites in our implementation (Section 3.4), using recursive division (256 points per site, 1.0 second). (b) After CCVT optimization (total 3.8 seconds, using our implementation in Section 3). (c) BNOT output (52 seconds, using the published code) shown for comparison. All results using a 2.5GHz CORE i5 laptop with 4GB memory.*

Abstract

Capacity Constrained Voronoi Tessellation is an important concept that greatly influenced recent research on point sampling. The original concept was based on discretizing the sampled domain, and the algorithm was prohibitively slow, even with some proposed accelerations. We present a few improvements that make a real difference in the speed of the algorithm, bringing it back into presence.

1. Introduction

Capacity Constrained Voronoi Tessellation (CCVT) [BSD09] is an important milestone in point sampling. It changed the perspective from focusing only on the spacing between samples to also consider the area associated with each sample, which is directly reflected on the density of sample points. The original algorithm by Balzer et al. relies on discretizing the sampled domain and assigning a fixed quota of the high-resolution auxiliary samples, called points, to each of the final samples, called sites. Run-time complexity is quadratic in the number of samples, which induced further research to find faster alternatives such as the continuous-domain BNOT [dGBOD12] and CCDT [XLGG11]. While these alternatives work well, the discrete-space algorithm remains elegant and easy to understand, and may therefore become popular if it can be sped up to a practical range. Li et al. [LNW*10] earlier proposed a few high- and low-level accelerations to CCVT, but they did not really eliminate the quadratic component in the algorithm, which we do below.

2. Background

The discretized CCVT algorithm uses a set of points to make a high-resolution sampling of the domain, assigns a fixed quota of points to the sites, and then iteratively alternates between positioning the sites at the centroids of their assigned points, and exchanging points between sites using the squared point-to-site distance as

† Email:abdalla_gafar@hotmail.com

DOI: 10.2312/cgvc.20171285

an energy. The latter is the time-consuming step, since it considers all pairs of sites in the original algorithm, and a wide neighborhood in the accelerated algorithm. Computing the energy itself is wasteful in the original algorithm, but Li et al proposed a median-cut approach for acceleration.

3. Our Improvements

3.1. Energy Function

Centroidal capacity-constrained cells are power cells, not Voronoi cells [dGBOD12], hence the slicing edges are straight line segments perpendicular to the lines between sites. This has been known for long, but neither the original [BSD09] nor the accelerated [LNW*10] implementation took full advantage of it. Indeed, from the energy used for keys by Li et al. we can derive:

$$\begin{aligned}
\Delta E &= |p-s_j|^2 - |p-s_i|^2 \\
&= \left((x-x_j)^2+(y-y_j)^2\right) - \left((x-x_i)^2+(y-y_i)^2\right) \\
&= \left((x-x_j)^2-(x-x_i)^2\right) + \left((y-y_j)^2-(y-y_i)^2\right) \\
&= 2(x_i-x_j)x + \left(x_j^2-x_i^2\right) + 2(y_i-y_j)y + \left(y_j^2-y_i^2\right) \\
&= 2(x_i-x_j)x + 2(y_i-y_j)y + \left(x_j^2+y_j^2-x_i^2-y_i^2\right) \\
&= ax+by+c\,.
\end{aligned}$$

Thus, the sorting keys for median-cut are actually distances along the line between the two sites. This reduces calculating the keys to a single multiplication and a single addition, which gives a substantial performance boost, observing that key generation accounts for most of the cost in CCVT; but only after implementing the following improvements.

3.2. Site Neighborhood

Once all the points assigned to each site make a contiguous set, the fact that slicing always uses a straight edge between sites implies that a site has to interact only with its immediate neighbors (in the Voronoi sense). In fact, even when a site's points are fragmented, immediate neighbors may act as proxies for exchanging the points with remote sites. We may therefore maintain a Delaunay triangulation to keep track of neighbors. This saves the cost of determining a cell radius (cf. [LNW*10]). More importantly, it saves the cost of querying all sites, which accounts for the quadratic complexity of CCVT.

Consider an example of n sites, and $m = 256n$ points. In a single (early) iteration, querying occurs $n(n-1)$ times, but exchanges occur only about $3 \times n$ times, and keys are generated about $3m$ times. For large n, say 10^5, querying occurs 10^{10} times, more than 100 times key generations ($\sim 7.68 \times 10^7$)! Therefore, CCVT with large numbers of sites (say 1M) is not common; to the best of our knowledge.

Thus, this is the most important improvement in terms of run-time complexity. In each iteration sites make only small shifts, which leads to linear-time updates in the triangulation.

3.3. Points Distribution

Li et al. proposed a random (white noise) distribution of the points, since a regular distribution creeps into the final distribution of the sites. Instead, we propose using a blue-noise distribution of points generated by one of the ultra-fast look-up samplers, such as AA Patterns [AHD15] for uniform sampling and ART [ANHD17] or Polyhexes [WPC*14] for stippling.

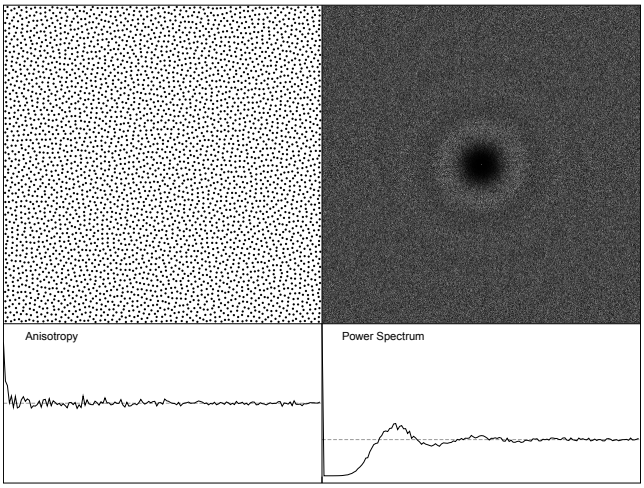


Figure 2: *Distribution and spectrum of centroids after a recursive-division slicing of an underlying blue-noise set of points.*

3.4. Initial Assignment

We may enforce the contiguous allocation of points outright by using recursive division to assign the points to sites. That is, we start with the full list of points, and recursively slice it, also using a median cut. A good strategy is to guide the slope of slicing by the moments of the set of points being sliced, so as to reduce the aspect ratios of the slices. Finally, sites are added at the centroid of each slice. Figure 2 demonstrates that this initialization strategy produces a decent distribution of sites even before optimization.

3.5. Reduced Overheads

We use simple arrays of points (no need to allocate vectors or heaps each time), and the partitioning routine can easily be adapted to deal with two disjoint array slices. Further, we swap 4-byte pointers/indexes rather than the actual 16-byte points (assuming double precision).

4. Results

Table 1 shows typical execution times. Using the proposed improvements, we managed to eliminate the quadratic element of CCVT and bring it to a very competent range of speeds, even compared to continuous-space algorithms, as demonstrated in Figure 1. We provide source code in the supplementary material and in our website.

Acknowledgments

This work was partially funded by Deutsche Forschungsgemeinschaft Grant (DE-620/22-1).

Number of sites	Time (s)
100	0.027
1,000	0.289
10,000	2.915
100,000	30.945

Table 1: *Typical execution times of our implementation in a 2.5GHz CORE i5 laptop with 4GB memory, using 256 points per site. Points are taken from a high-resolution stippling of Lena using ART [ANHD17].*

References

[AHD15] AHMED A. G. M., HUANG H., DEUSSEN O.: AA Patterns for Point Sets with Controlled Spectral Properties. *ACM Trans. Graph. 34*, 6 (2015), 212:1–212:8. 2

[ANHD17] AHMED A. G. M., NIESE T., HUANG H., DEUSSEN O.: An Adaptive Point Sampler on a Regular Lattice. *ACM Trans. Graph. 36*, 4 (July 2017), 138:1–138:13. 2, 3

[BSD09] BALZER M., SCHLÖMER T., DEUSSEN O.: Capacity-constrained point distributions: A variant of lloyd's method. In *ACM SIGGRAPH 2009 Papers* (New York, NY, USA, 2009), SIGGRAPH '09, ACM, pp. 86:1–86:8. 1, 2

[dGBOD12] DE GOES F., BREEDEN K., OSTROMOUKHOV V., DESBRUN M.: Blue noise through optimal transport. *ACM Trans. Graph. 31*, 6 (Nov. 2012), 171:1–171:11. 1, 2

[LNW*10] LI H., NEHAB D., WEI L.-Y., SANDER P. V., FU C.-W.: Fast capacity constrained voronoi tessellation. In *Proceedings of the 2010 ACM SIGGRAPH symposium on Interactive 3D Graphics and Games* (2010), ACM, p. 13. 1, 2

[WPC*14] WACHTEL F., PILLEBOUE A., COEURJOLLY D., BREEDEN K., SINGH G., CATHELIN G., DE GOES F., DESBRUN M., OSTROMOUKHOV V.: Fast Tile-based Adaptive Sampling with User-specified Fourier Spectra. *ACM Trans. Graph. 33*, 4 (July 2014), 56:1–56:11. 2

[XLGG11] XU Y., LIU L., GOTSMAN C., GORTLER S. J.: Capacity-constrained delaunay triangulation for point distributions. *Computers & Graphics 35*, 3 (2011), 510 – 516. Shape Modeling International (SMI) Conference 2011. 1