

‘votess: A fast, multi-target voronoi tessellator using the SYCL framework’

Samridh Dev Singh¹, Chris Byrohl², and Dylan Nelson²

¹ Grinnell College, 1115 8th Avenue, 50112 Grinnell, United States of America ² Heidelberg University, Institute for Theoretical Astronomy, Albert-Ueberle-Str. 2, 69120 Heidelberg, Germany

DOI: [10.xxxxxx/draft](https://doi.org/10.xxxxxx/draft)

Software

- [Review](#) ↗
- [Repository](#) ↗
- [Archive](#) ↗

Editor: [Open Journals](#) ↗

Reviewers:

- [@openjournals](#)

Submitted: 01 January 1970

Published: unpublished

License

Authors of papers retain copyright and release the work under a Creative Commons Attribution 4.0 International License ([CC BY 4.0](#)).

Statement of need

Voronoi tessellation is a fundamental spatial decomposition technique widely used in scientific domains such as astrophysics ([Springel, 2010](#)), earth sciences [], materials science [], and biochemistry []. It enables the partitioning of space into regions based on proximity to given points, which is essential for analyzing large datasets. Although performing Voronoi tessellations on large datasets has been feasible for some time, the increasing complexity and size of modern data have underscored the need for faster, more efficient computation. With the rise of accelerator architectures like GPUs and FPGAs, the computational power available today has greatly improved the ability to handle these large datasets more effectively.

However, despite these advancements, most existing implementations of Voronoi tessellations are tailored to specific platforms and architectures, limiting their portability. This results in the need for bespoke solutions for different hardware setups, creating inefficiencies and increasing development time.

To address this problem, votess provides a portable solution that can operate across various accelerator architectures, without modification to the source code, enabling developers within various scientific fields to be able to make use of the new computing architectures.

Summary

votess is a library for performing 3D Voronoi tessellations on heterogeneous platforms via the SYCL framework. votess was designed to be portable yet performant, with an easy-to-use interface.

The underlying algorithm is based on a paper (?), which highlights that many applications, such as in astrophysics [] or fluid simulations [], only require the geometry of the Voronoi cells and their neighboring information, rather than a full combinatorial mesh data structure. This observation allows for a simplified algorithm, as presented here, which avoids the need for classical mesh-based approaches like the Bowyer-Watson algorithm.

The core algorithm employed by votess consists of two main steps. First, the input set of points is sorted into a grid, and a k-nearest neighbors search is performed. Once the nearest neighbors are identified for each point, the Voronoi cell is computed by iteratively clipping a bounding box using the perpendicular bisectors between the point and its neighbors. To optimize the process and avoid iterating through all neighbors, a security radius condition is applied. If a Voronoi cell cannot be validated, a CPU fallback mechanism is used to ensure robustness.

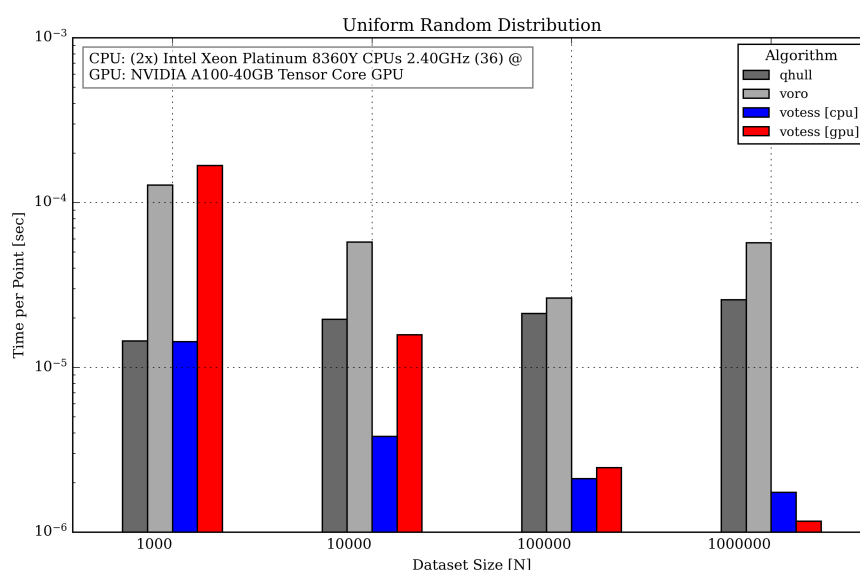
This efficient algorithm allows for independent thread execution, making it highly suitable for GPU parallelism. Unlike previous algorithms that relied on sequential execution due to their

40 mesh insertion methods [], votess leverages the independence of cell computations to achieve
41 significant speedups in parallel environments.

42 Performance

43 With a working implementation of votess, it can be seen that it outperforms several single-
44 threaded applications:

45 In Figure 1, we show the performance of votess compared to two other single-threaded Voronoi
46 tessellation libraries: QHULL and VORO++. QHULL is a well-known computational geometry library
47 that constructs convex hulls and Voronoi diagrams using an indirect projection method (Barber
48 et al., 1996), while Voro++ is a C++ library specifically designed for three-dimensional Voronoi
49 tessellations, utilizing a cell-based computation approach that is well-suited for physical
50 applications (Rycroft, 2009).



51
52 It can be seen that performance is best on larger datasets. The CPU implementation outperforms
53 other applications of atleast tenfold, and at most a hundred fold on large datasets. It must be
54 noted, that the benchmarks were taken before either the CPU and GPU implementations have
55 recieved optimizations.

56 There also exists other multithreaded Voronoi tessellation codes, such as ParVoro++ (Wu et
57 al., 2023), CGAL (The CGAL Project, 2018), and GEOGRAM (Inria, 2018), which are also widely
58 used in large-scale computational geometry applications.

59 Features

60 votess provides a versatile and efficient tool for computing Voronoi tessellations, supporting
61 multiple output formats including neighbor information for each Voronoi cell. It has been
62 tested on various CPU and GPU architectures, delivering high performance on both platforms.

63 Users can leverage votess in three ways: through the C++ library, a command-line interface
64 clvotess, and a Python interface pyvotess. The C++ library offers a simple interface with a
65 primary function, tessellate, that computes the tessellation. Additionally, users can select the
66 target device to run said tessellation. The Python wrapper, pyvotess, mirrors the functionality
67 of the C++ version, providing the same ease of use for Python-based workflows.

68 To fine-tune the behavior of `votess`, the class `vtargs` is provided, allowing users to adjust
69 parameters much like `std::unordered_map` from the STL. These parameters can be used to
70 optimize runtime performance if needed. The `tessellate` function outputs a templated class
71 `dnn`, representing a 2D jagged array of neighbors contributing to each particle's Voronoi cell of
72 the sorted input dataset, as managed via `vtargs`.

73 Acknowledgements

74 CB and DN acknowledge funding from the Deutsche Forschungsgemeinschaft (DFG) through
75 an Emmy Noether Research Group (grant number NE 2441/1-1).

76 References

- 77 Barber, C. B., Dobkin, D. P., & Huhdanpaa, H. (1996). The quickhull algorithm for convex hulls.
78 *ACM Trans. Math. Softw.*, 22(4), 469–483. <https://doi.org/10.1145/235815.235821>
- 79 Inria, P. A.-G. (2018). *Geogram: A programming library of geometric algorithms*. [http://](http://alice.loria.fr/software/geogram/doc/html/index.html)
80 alice.loria.fr/software/geogram/doc/html/index.html
- 81 Rycroft, C. (2009). *VORO++: A three-dimensional voronoi cell library in c++*.
- 82 Springel, V. (2010). Moving-mesh hydrodynamics with the AREPO code. *Proceedings*
83 *of the International Astronomical Union*, 6(S270), 203–206. [https://doi.org/10.1017/](https://doi.org/10.1017/S1743921311000378)
84 [S1743921311000378](https://doi.org/10.1017/S1743921311000378)
- 85 The CGAL Project. (2018). *CGAL user and reference manual* (4.12.1 ed.). CGAL Editorial
86 Board. <https://doc.cgal.org/4.12.1/Manual/packages.html#PkgSpatialSortingSummary>
- 87 Wu, G., Tian, H., Lu, G., & Wang, W. (2023). ParVoro++: A scalable parallel algorithm for
88 constructing 3D voronoi tessellations based on kd-tree decomposition. *Parallel Computing*,
89 115, 102995. <https://doi.org/10.1016/j.parco.2023.102995>