

Step-By-Step Straight Skeletons

Günther Eder 

Universität Salzburg, FB Computerwissenschaften, Austria
geder@cs.sbg.ac.at

Martin Held 

Universität Salzburg, FB Computerwissenschaften, Austria
held@cs.sbg.ac.at

Peter Palfrader 

Universität Salzburg, FB Computerwissenschaften, Austria
palfrader@cs.sbg.ac.at

Abstract

We present two software packages for computing straight skeletons: MONOS, our implementation of an algorithm by Biedl et al. (2015), computes the straight skeleton of a monotone input polygon, and SURFER2 implements a generalization of an algorithm by Aichholzer and Aurenhammer (1998) to handle multiplicatively-weighted planar straight-line graphs as input.

The graphical user interfaces that ship with our codes support step-by-step computations, where each event can be investigated and studied by the user. This makes them a canonical candidate for educational purposes and detailed event analyses. Both codes are freely available on GitHub.

2012 ACM Subject Classification Theory of computation → Computational geometry

Keywords and phrases weighted straight skeleton, implementation, visualization, graphical user interface, education

Digital Object Identifier 10.4230/LIPIcs.SoCG.2020.76

Category Media Exposition

Funding Work supported by Austrian Science Fund (FWF): Grants ORD 53-VO and P31013-N31.

1 Introduction

The straight skeleton of a polygon P is a skeletal structure similar to the medial axis. It consists of straight-line segments only and was introduced to computational geometry by Aichholzer et al. [2]. Consider a shrinking process of P , called the wavefront propagation, where all edges of P move inwards in a parallel manner at unit speed. The shrinking polygon, called the wavefront, will change as edges collapse to zero length and are removed (*edge event*), and vertices move into non-incident wavefront edges, thereby splitting the wavefront into more components (*split event*). The propagation ends when all wavefront components have collapsed. The straight skeleton is then defined as the set of line segments which cover the traces of vertices of the wavefront polygons during this process; cf. Figure 1.

Straight skeletons can be generalized by using planar straight-line graphs (PSLGs) as input instead of just polygons [1], or by weighting the input edges either multiplicatively or additively which causes their wavefront edges to move either faster [1, 5] or to start moving at different times [6].

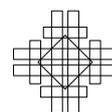
Several algorithms exist to compute straight skeletons. Aichholzer and Aurenhammer [1] describe an algorithm with a worst-case complexity of $\mathcal{O}(n^3 \log n)$ for an n -vertex PSLG which, however, seems to run in near-linear time in practice [4, 7]. Currently, the algorithm with the best known worst-case complexity for general input is due to Eppstein and Erickson [5] and

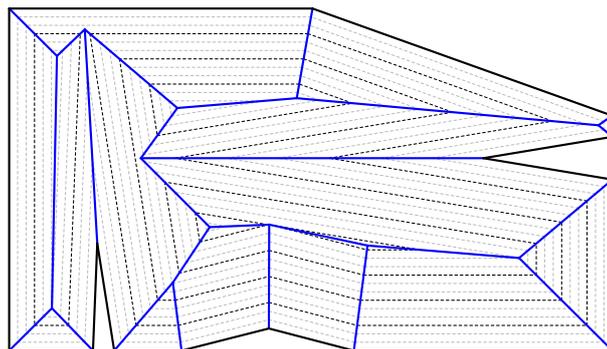


© Günther Eder, Martin Held, and Peter Palfrader;
licensed under Creative Commons License CC-BY
36th International Symposium on Computational Geometry (SoCG 2020).
Editors: Sergio Cabello and Danny Z. Chen; Article No. 76; pp. 76:1–76:4



Leibniz International Proceedings in Informatics
Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany





■ **Figure 1** The straight skeleton (blue) of a polygon P (black) is given by the traces of the vertices of shrinking wavefront polygons (dashed) during the wavefront-propagation process.

runs in $\mathcal{O}(n^{17/11+\varepsilon})$ time and space for any positive ε . We are not aware of any implementation of their algorithm, though. Biedl et al. [3] present an $\mathcal{O}(n \log n)$ time algorithm to construct the straight skeleton of an n -vertex monotone polygon.

2 Contribution

We implemented the algorithms by Aichholzer and Aurenhammer [1] as well as Biedl et al. [3] and will report at SoCG 2020 on the engineering challenges of casting these algorithms into actual software [4]. Our code is freely available on GitHub: SURFER2 computes the weighted straight skeleton of planar straight-line graphs, and MONOS is a special-purpose code to construct straight skeletons of monotone polygons¹.

In the course of implementing the algorithms, we also created graphical user interfaces (GUIs). These GUIs enable the user to “see” the algorithm in action. The interfaces are invaluable during development and debugging, but they also help to gain a deeper understanding of the various events handled during the wavefront propagation and, in the case of MONOS, the merge step of the algorithm. As such they represent useful aids in teaching computational geometry classes. The concepts which they help to understand are not strictly limited to straight skeletons, but also include general event-based algorithms, kinetic data structures, and sweep-plane processes.

3 MonosGUI: Straight skeleton of a monotone polygon

Biedl et al. [3] describe an $\mathcal{O}(n \log n)$ time algorithm to compute the straight skeleton of a simple n -vertex monotone polygon \mathcal{P} . Their algorithm consists of two steps: (i) The polygon \mathcal{P} is split into an upper and lower monotone chain, and the straight skeleton of each chain is computed individually. (ii) The final straight skeleton $\mathcal{S}(\mathcal{P})$ is obtained by merging these two straight skeletons.

We employ a classical wavefront propagation to obtain the straight skeleton of a monotone chain in $\mathcal{O}(n \log n)$ time [3]. In the second step, the skeletons of the upper and lower chains are merged to form $\mathcal{S}(\mathcal{P})$. This merge is based on a left-to-right traversal of the two chains and their respective straight-skeleton faces. Starting at the leftmost vertex of both chains, the angular bisector between the incident edges is constructed. It intersects arcs of both the

¹ Source available at <https://github.com/cgalab/surfer2> and <https://github.com/cgalab/monos>.

top and bottom straight skeleton. We stop at the first intersection reached and modify the respective straight skeleton locally by creating a node. Then the next bisector is constructed between the two edges that induce the faces of the upper and lower skeleton we are currently in. This process is repeated until the rightmost vertex of \mathcal{P} is reached, thus obtaining the final skeleton $\mathcal{S}(\mathcal{P})$.

MONOSGUI, the graphical user interface that comes with MONOS, allows to step through both the skeleton computation of the chains and the merge process; cf. Figure 2. Pressing `n`, MONOSGUI steps to the next event. We can skip the chain computation with `c`, then MONOSGUI stops before the merge process starts. Simply pressing `↩` computes the given input and displays its straight skeleton.

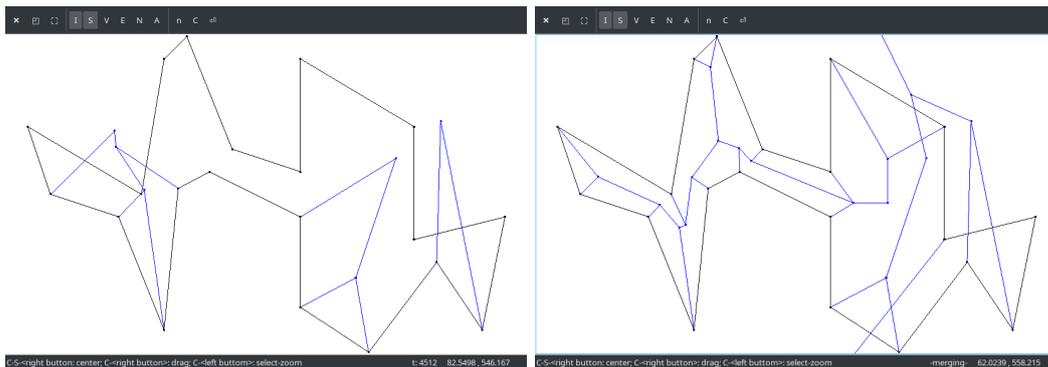


Figure 2 MONOSGUI: A step-by-step construction of the straight skeleton of a single chain (left); The construction of the merge chain between the finished upper and lower chain's skeleton (right).

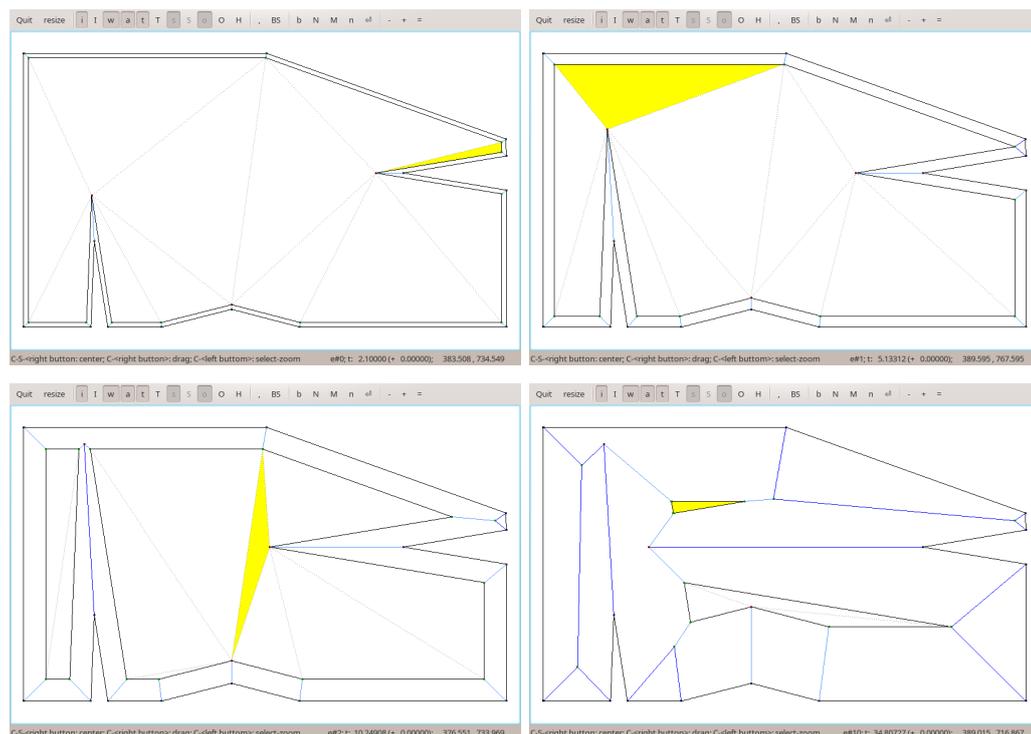
4 SurfGUI: Straight Skeleton of a PSLG

The algorithm by Aichholzer and Aurenhammer [1] constructs the straight skeleton by simulating the wavefront propagation. As the wavefront sweeps the plane, a kinetic triangulation of that part of the plane which has not yet been swept is maintained.

This triangulation is a kinetic data structure where every change in the structure of the wavefront, i.e., every edge event and every split event, is witnessed by a triangle collapse. However, not all triangle collapses witness a corresponding event of the wavefront. Some triangle collapses correspond to internal events only, where the triangulation changes as a wavefront vertex moves over a triangulation diagonal (*flip event*).

Our implementation of this algorithm, SURFER2, comes with a graphical user interface, SURFGUI, which shows the wavefront propagation step by step. By default, SURFGUI shows the input planar straight-line graph, the current kinetic triangulation and all straight skeleton arcs traced out so far. The next triangle to witness an event is highlighted.

The user can interactively control time, moving forwards or backwards. When moving forwards in time, event handling can optionally be skipped, which will result in the display of an invalid configuration after the missed event (time will reset to the first missed event once event-handling is re-enabled). Moving back will not undo events already processed. Shortcuts are provided to jump forward to (but not yet process) the next event or to jump backwards to the event just handled. See Figure 3 for some screenshots.



■ **Figure 3** SURFGUI processing the input from Figure 1. The triangle witnessing the next event is always highlighted (in yellow). In reading order, the highlighted triangle witnesses an edge event, a split event, a flip event, and the collapse of an entire wavefront component.

References

- 1 Oswin Aichholzer and Franz Aurenhammer. Straight Skeletons for General Polygonal Figures in the Plane. In *Voronoi's Impact on Modern Sciences II*, volume 21, pages 7–21. Institute of Mathematics of the National Academy of Sciences of Ukraine, 1998. doi:10.1007/3-540-61332-3_144.
- 2 Oswin Aichholzer, Franz Aurenhammer, David Alberts, and Bernd Gärtner. A Novel Type of Skeleton for Polygons. *Journal of Universal Computer Science*, 1(12):752–761, 1995. doi:10.1007/978-3-642-80350-5_65.
- 3 Therese Biedl, Martin Held, Stefan Huber, Dominik Kaaser, and Peter Palfrader. A Simple Algorithm for Computing Positively Weighted Straight Skeletons of Monotone Polygons. *Information Processing Letters*, 115(2):243–247, 2015. doi:10.1016/j.ipl.2014.09.021.
- 4 Günther Eder, Martin Held, and Peter Palfrader. On Implementing Straight Skeletons: Challenges and Experiences. In *36th International Symposium on Computational Geometry, SoCG 2020*, volume 164 of *LIPICs*, pages 38:1–38:16, Zürich, Switzerland, 2020.
- 5 David Eppstein and Jeff Erickson. Raising Roofs, Crashing Cycles, and Playing Pool: Applications of a Data Structure for Finding Pairwise Interactions. *Discrete & Computational Geometry*, 22(4):569–592, 1999. doi:10.1145/276884.276891.
- 6 Martin Held and Peter Palfrader. Straight Skeletons with Additive and Multiplicative Weights and Their Application to the Algorithmic Generation of Roofs and Terrains. *Computer-Aided Design*, 92(1):33–41, 2017. doi:10.1016/j.cad.2017.07.003.
- 7 Peter Palfrader, Martin Held, and Stefan Huber. On Computing Straight Skeletons by Means of Kinetic Triangulations. In *Proceedings of the 20th Annual European Symposium on Algorithms (ESA)*, pages 766–777, 2012. doi:10.1007/978-3-642-33090-2_66.