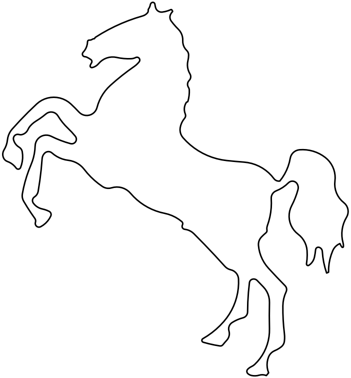
COMPUTING STRAIGHT SKELETONS BY MEANS OF KINETIC TRIANGULATIONS

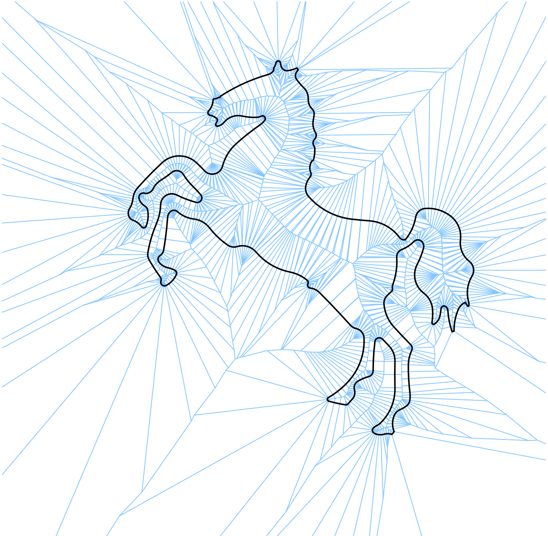
Peter Palfrader

October 2013

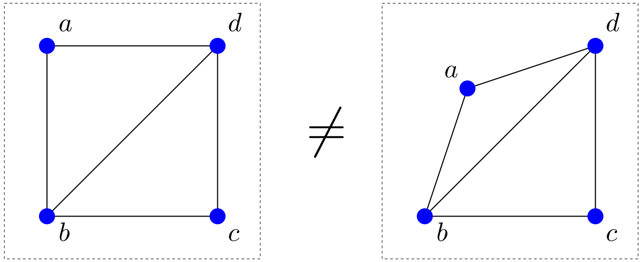
- Aichholzer, Alberts, Aurenhammer, Gärtner 1995.
- Problem: Given input graph, find the *straight skeleton*.



- Aichholzer, Alberts, Aurenhammer, Gärtner 1995.
- Problem: Given input graph, find the *straight skeleton*.

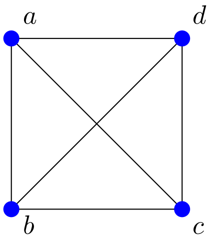


- When we consider a graph $G = (V, E)$ we always consider its *embedding*.



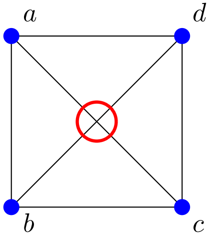
PRELIMINARIES

- When we consider a graph $G = (V, E)$ we always consider its *embedding*.
- We care about planar, straight-line graphs or PSLGs.



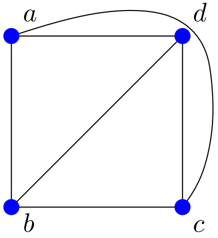
Preliminaries

- When we consider a graph $G = (V, E)$ we always consider its *embedding*.
- We care about planar, straight-line graphs or PSLGs.



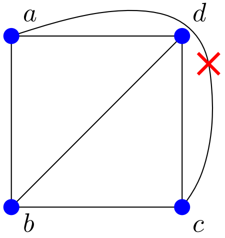
PRELIMINARIES

- When we consider a graph $G = (V, E)$ we always consider its *embedding*.
- We care about planar, straight-line graphs or PSLGs.



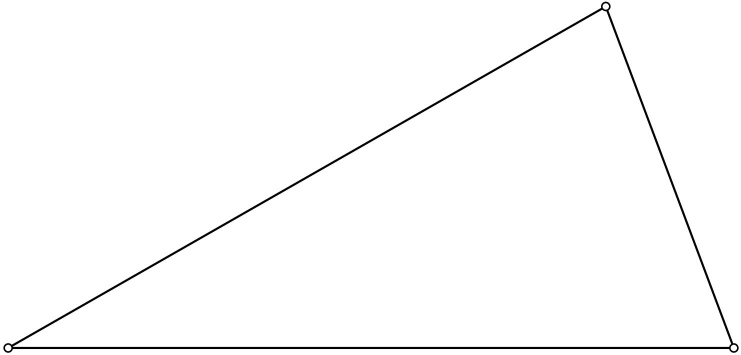
PRELIMINARIES

- When we consider a graph $G = (V, E)$ we always consider its *embedding*.
- We care about planar, straight-line graphs or PSLGs.



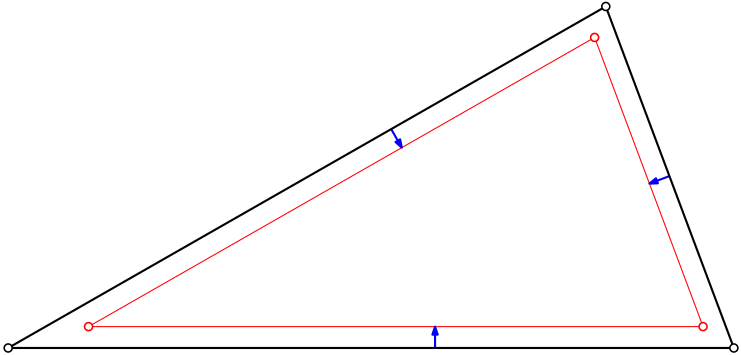
Straight Skeletons – Motivation

- Aichholzer et al. [AAAG95].
- input polygon $\mathcal{P}$ emanates wavefront $\mathcal{WF}(\mathcal{P}, t)$.



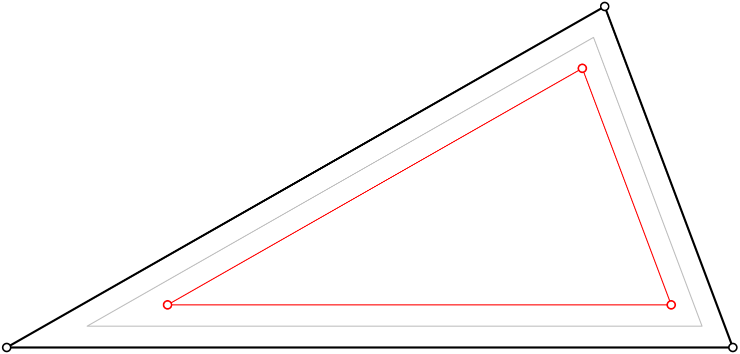
Straight Skeletons – Motivation

- Aichholzer et al. [AAAG95].
- input polygon $\mathcal{P}$ emanates wavefront $\mathcal{WF}(\mathcal{P}, t)$.



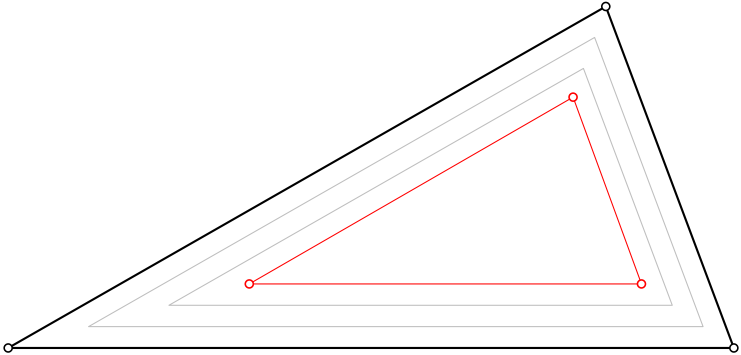
Straight Skeletons – Motivation

- Aichholzer et al. [AAAG95].
- input polygon $\mathcal{P}$ emanates wavefront $\mathcal{WF}(\mathcal{P}, t)$.
- wavefront propagation — shrinking process.



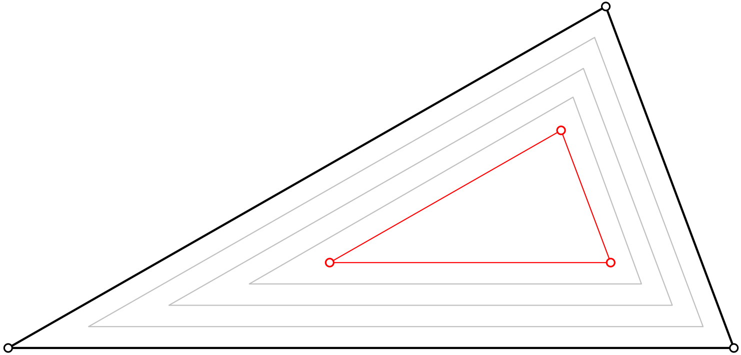
Straight Skeletons – Motivation

- Aichholzer et al. [AAAG95].
- input polygon $\mathcal{P}$ emanates wavefront $\mathcal{WF}(\mathcal{P}, t)$.
- wavefront propagation — shrinking process.



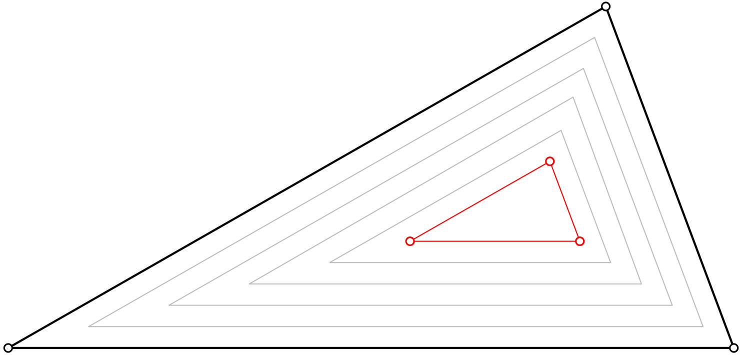
Straight Skeletons – Motivation

- Aichholzer et al. [AAAG95].
- input polygon $\mathcal{P}$ emanates wavefront $\mathcal{WF}(\mathcal{P}, t)$.
- wavefront propagation — shrinking process.

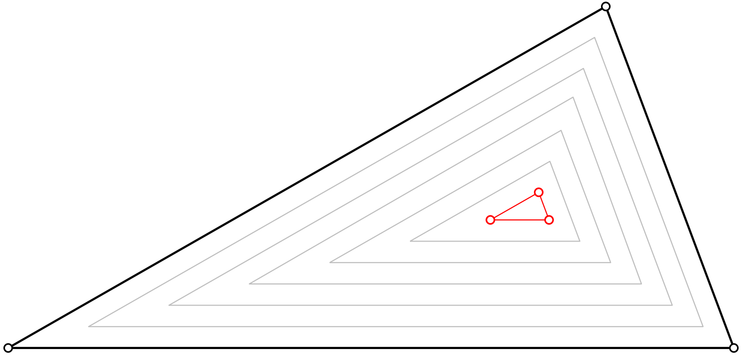


Straight Skeletons – Motivation

- Aichholzer et al. [AAAG95].
- input polygon $\mathcal{P}$ emanates wavefront $\mathcal{WF}(\mathcal{P}, t)$.
- wavefront propagation — shrinking process.

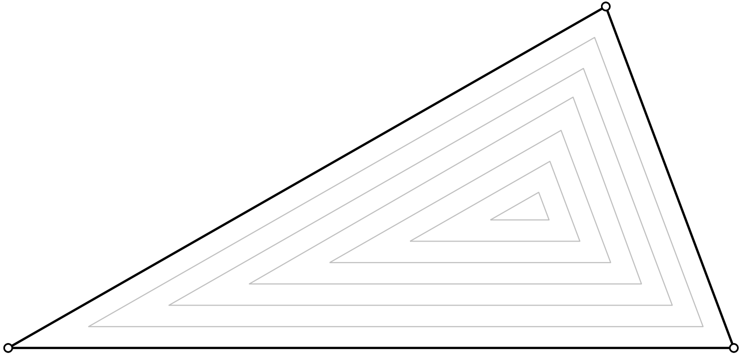


- Aichholzer et al. [AAAG95].
- input polygon $\mathcal{P}$ emanates wavefront $\mathcal{WF}(\mathcal{P}, t)$.
- wavefront propagation — shrinking process.

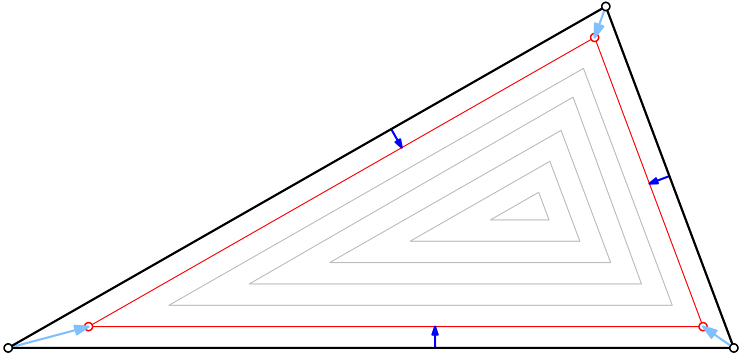


Straight Skeletons – Motivation

- Aichholzer et al. [AAAG95].
- input polygon $\mathcal{P}$ emanates wavefront $\mathcal{WF}(\mathcal{P}, t)$.
- wavefront propagation — shrinking process.

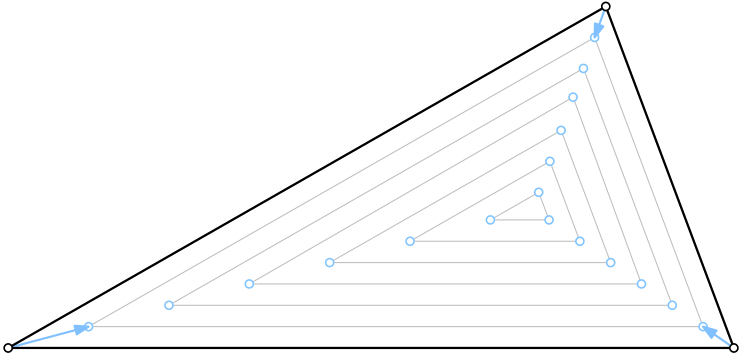


- Aichholzer et al. [AAAG95].
- input polygon $\mathcal{P}$ emanates wavefront $\mathcal{WF}(\mathcal{P}, t)$.
- wavefront propagation — shrinking process.



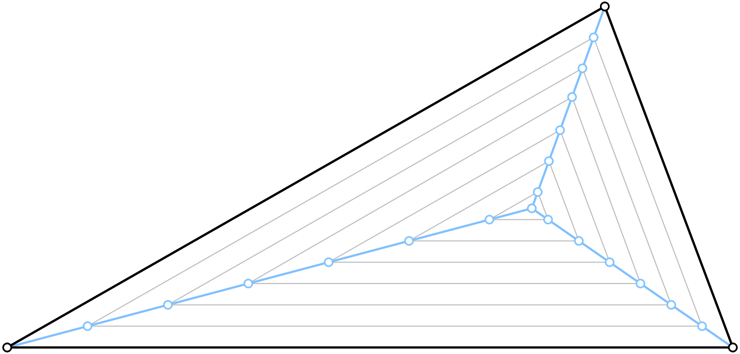
Straight Skeletons – Motivation

- Aichholzer et al. [AAAG95].
- input polygon $\mathcal{P}$ emanates wavefront $\mathcal{WF}(\mathcal{P}, t)$.
- wavefront propagation — shrinking process.



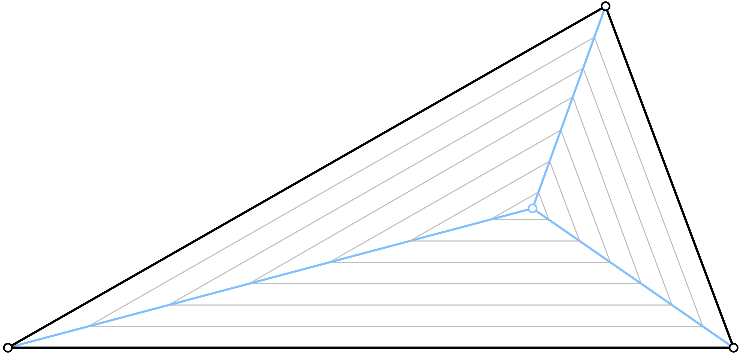
Straight Skeletons – Motivation

- Aichholzer et al. [AAAG95].
- input polygon $\mathcal{P}$ emanates wavefront $\mathcal{WF}(\mathcal{P}, t)$.
- wavefront propagation — shrinking process.
- straight skeleton $\mathcal{SK}(\mathcal{P})$ is traces of wavefront vertices.



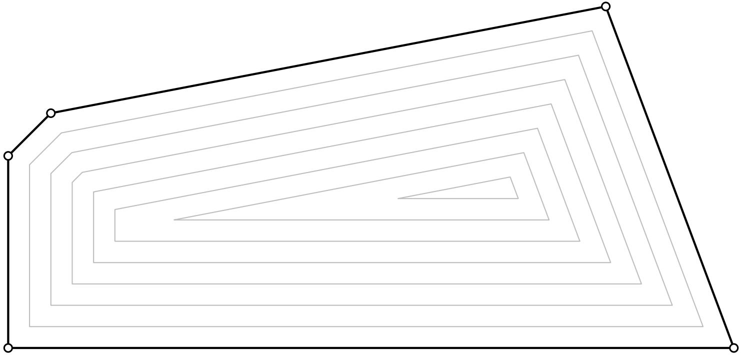
STRAIGHT SKELETONS – MOTIVATION

- Aichholzer et al. [AAAG95].
- input polygon $\mathcal{P}$ emanates wavefront $\mathcal{WF}(\mathcal{P}, t)$.
- wavefront propagation — shrinking process.
- straight skeleton $\mathcal{SK}(\mathcal{P})$ is traces of wavefront vertices.



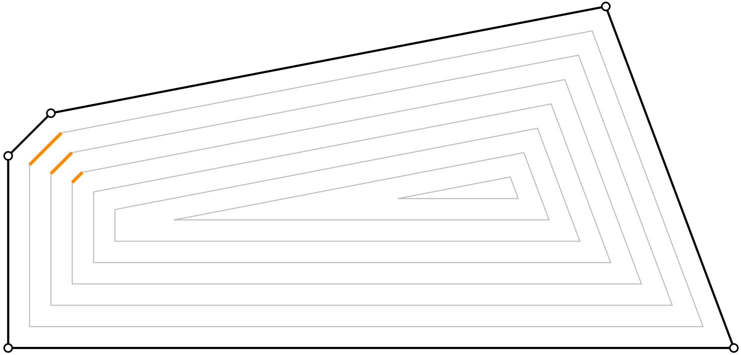
TOPOLGY CHANGES – EDGE EVENTS

- Wavefront topology changes over time.



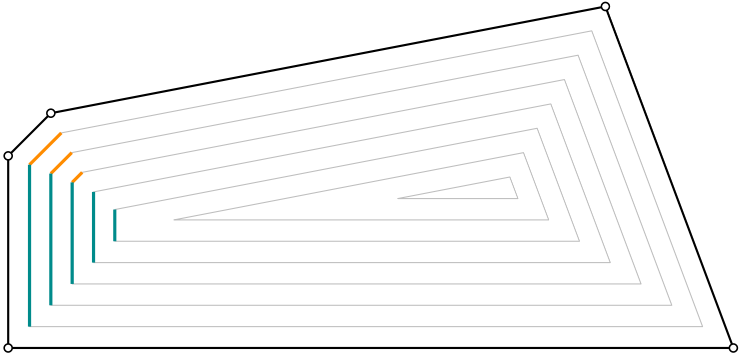
TOPOLGY CHANGES – EDGE EVENTS

- Wavefront topology changes over time.



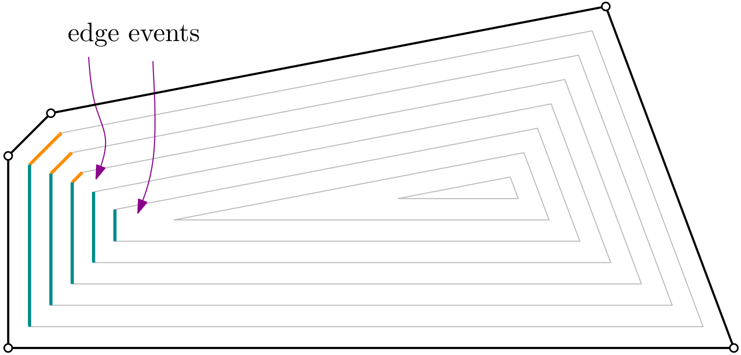
TOPOLGY CHANGES – EDGE EVENTS

- Wavefront topology changes over time.



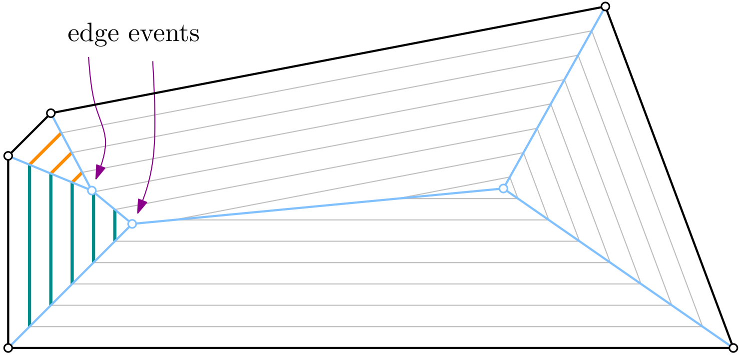
TOPOLGY CHANGES – EDGE EVENTS

- Wavefront topology changes over time.
- *edge event*: an edge of $\mathcal{WF}(\mathcal{P}, t)$ vanishes.



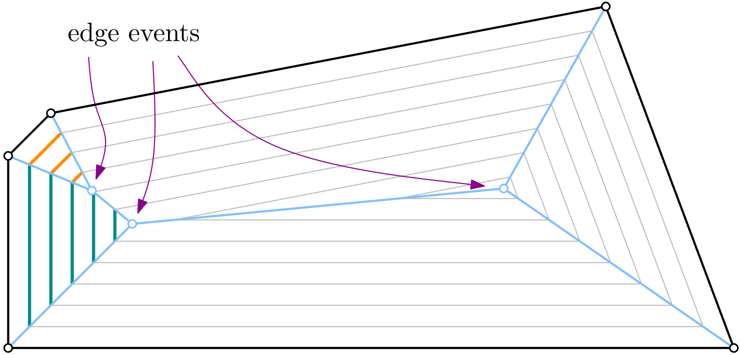
Topolgy Changes – Edge Events

- Wavefront topology changes over time.
- *edge event*: an edge of $\mathcal{WF}(\mathcal{P}, t)$ vanishes.
- In $\mathcal{SK}(\mathcal{P})$, such a topology change is witnessed by *nodes*.



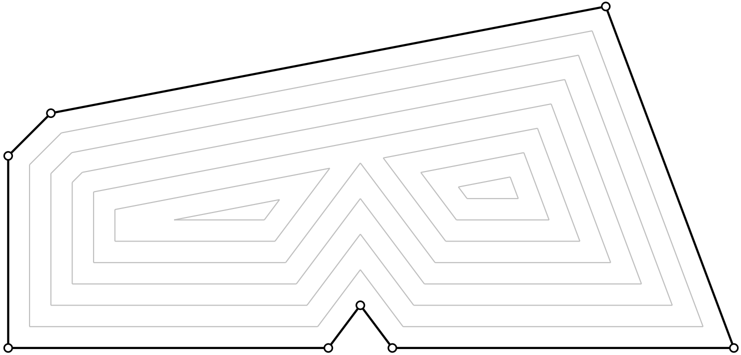
TOPOLGY CHANGES – EDGE EVENTS

- Wavefront topology changes over time.
- *edge event*: an edge of $\mathcal{WF}(\mathcal{P}, t)$ vanishes.
- In $\mathcal{SK}(\mathcal{P})$, such a topology change is witnessed by *nodes*.



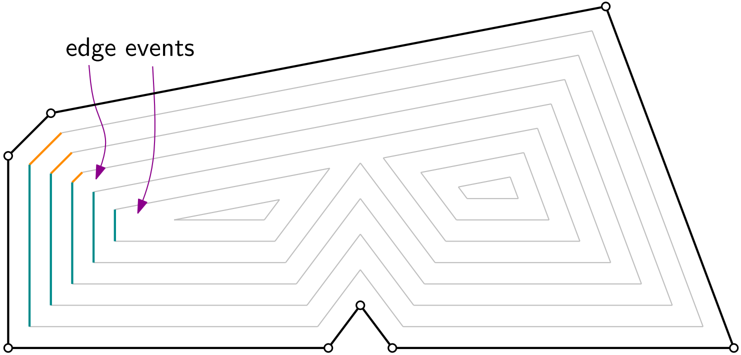
TOPOLGY CHANGES – SPLIT EVENTS

- Wavefront topology changes over time.



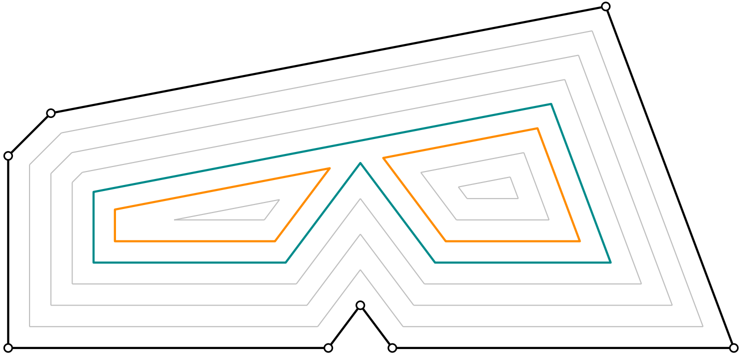
Topolgy Changes – Split Events

- Wavefront topology changes over time.
- *edge event*: an edge of $\mathcal{WF}(\mathcal{P}, t)$ vanishes.



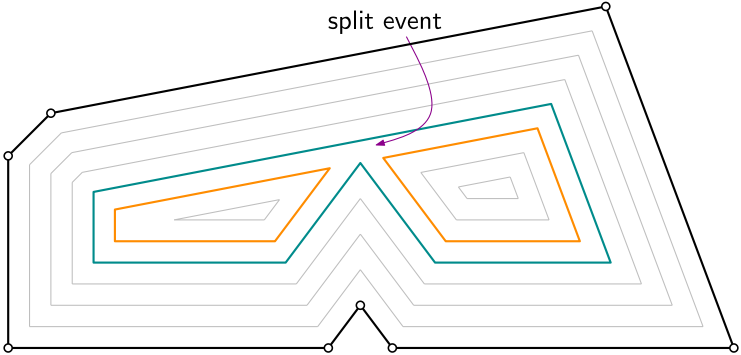
Topolgy Changes – Split Events

- Wavefront topology changes over time.
- *edge event*: an edge of $\mathcal{WF}(\mathcal{P}, t)$ vanishes.



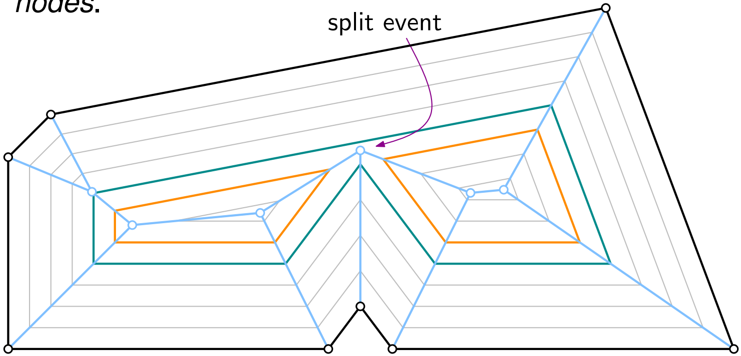
TOPOLGY CHANGES – SPLIT EVENTS

- Wavefront topology changes over time.
- *edge event*: an edge of $\mathcal{WF}(\mathcal{P}, t)$ vanishes.
- *split event*: wavefront splits into two parts.



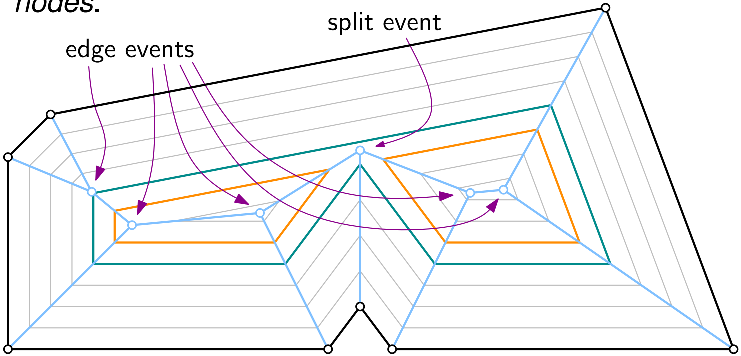
Topolgy Changes – Split Events

- Wavefront topology changes over time.
- *edge event*: an edge of $\mathcal{WF}(\mathcal{P}, t)$ vanishes.
- *split event*: wavefront splits into two parts.
- In $\mathcal{SK}(\mathcal{P})$, events (topology changes) are witnessed by *nodes*.



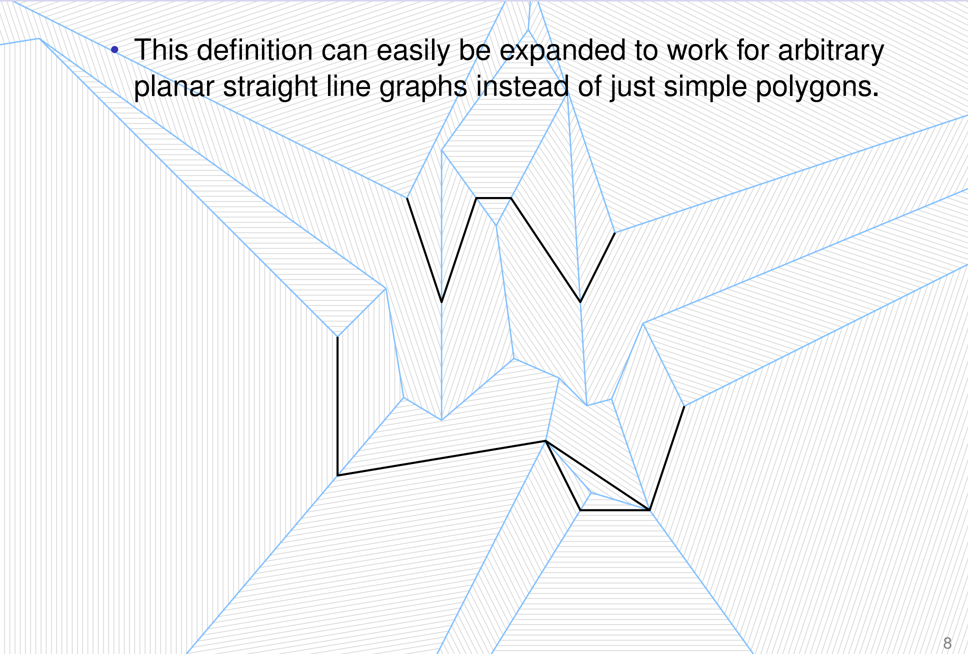
Topolgy Changes – Split Events

- Wavefront topology changes over time.
- *edge event*: an edge of $\mathcal{WF}(\mathcal{P}, t)$ vanishes.
- *split event*: wavefront splits into two parts.
- In $\mathcal{SK}(\mathcal{P})$, events (topology changes) are witnessed by *nodes*.



Straight Skeleton of a PSLG

- This definition can easily be expanded to work for arbitrary planar straight line graphs instead of just simple polygons.



Summary: Straight Skeletons

- The straight skeleton is the union of traces of wavefront vertices over the propagation process.
- The topology of the wavefront changes with time due to edge and split events. These are witnessed in $\mathcal{SK}$ as nodes.

Applications: Roof Modeling

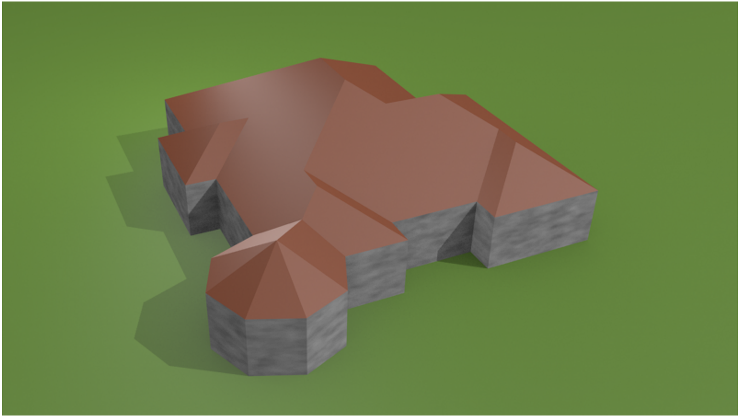


image credit: Stefan Huber

Applications: GIS

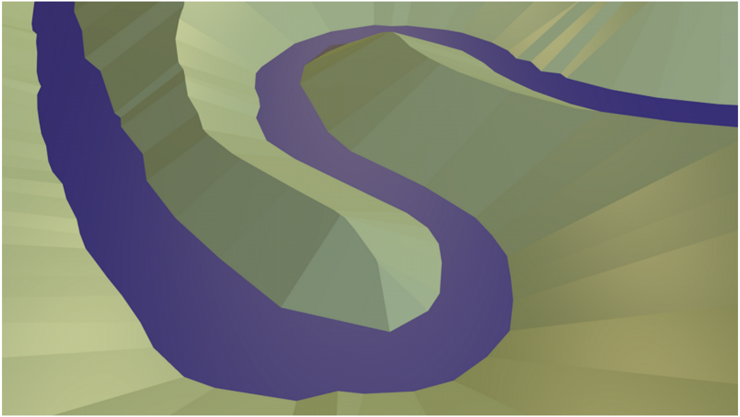
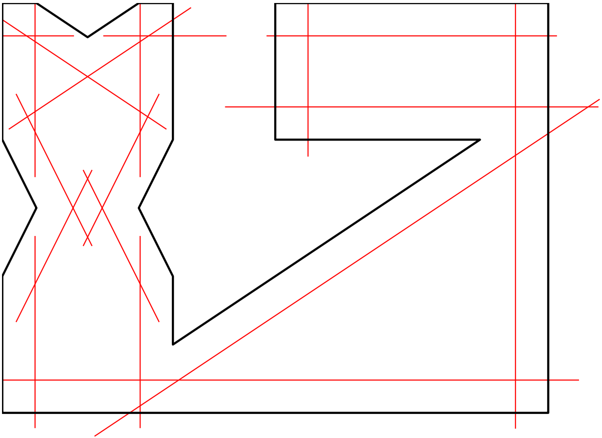
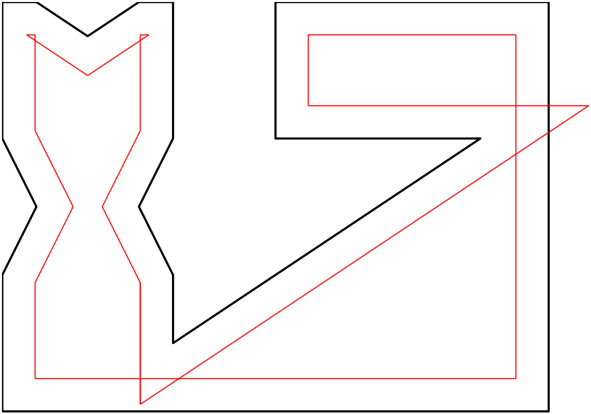
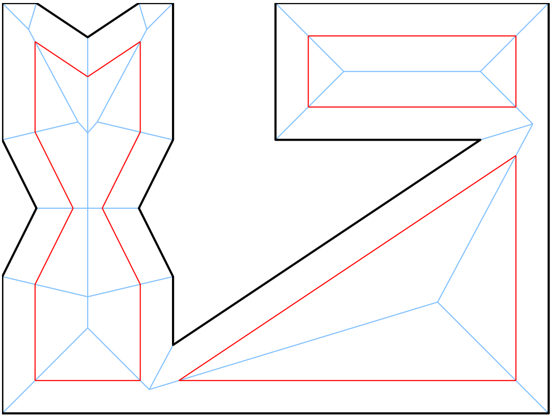


image credit: Stefan Huber

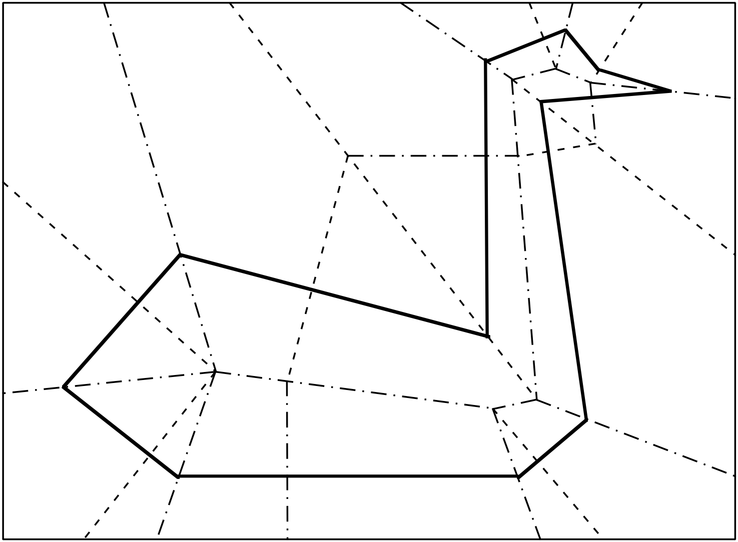




Applications: Offsetting



Applications: Cut and Fold



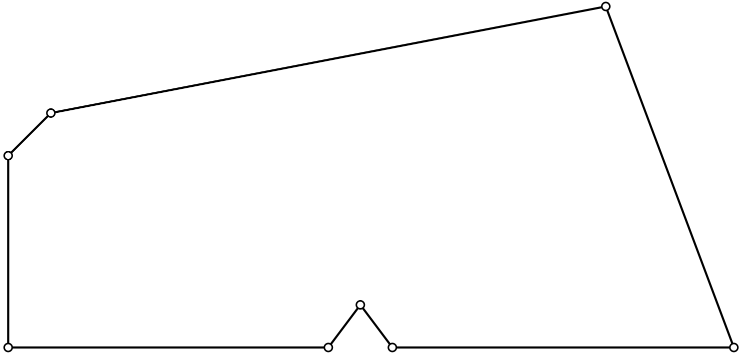
[DDL98]

image credit: Erik Demaine

APPLICATIONS: AND MUCH MORE

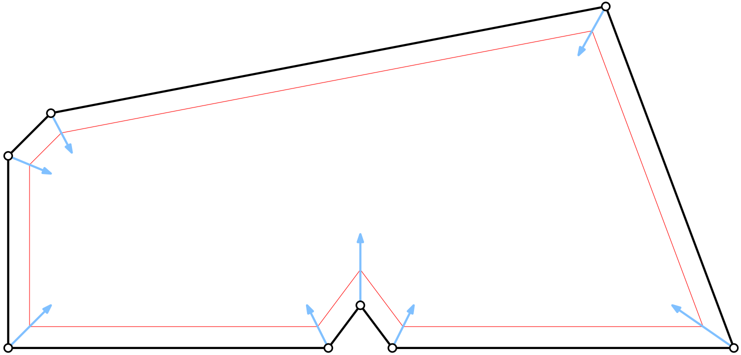
- Design of Pop-Up cards [Sugi13].
- Shape reconstruction and contour interpolation [OPC96].
- Area collapsing in geographic maps and centerlines of roads [HS08].

- Common approach: simulate the wavefront propagation.
- Problem: When will the next event happen, and what is it?
- If we solve this, we can incrementally construct the SK.

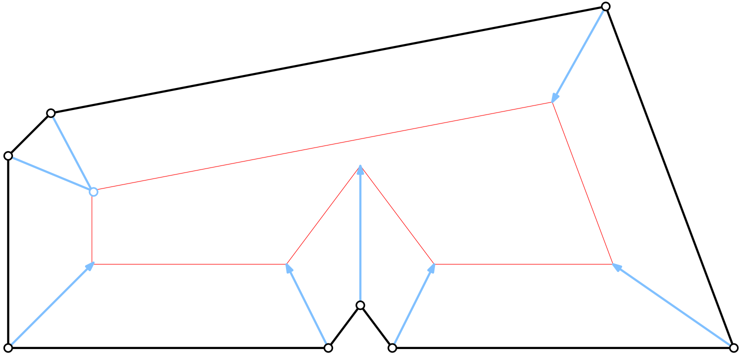


Computing the Straight Skeleton

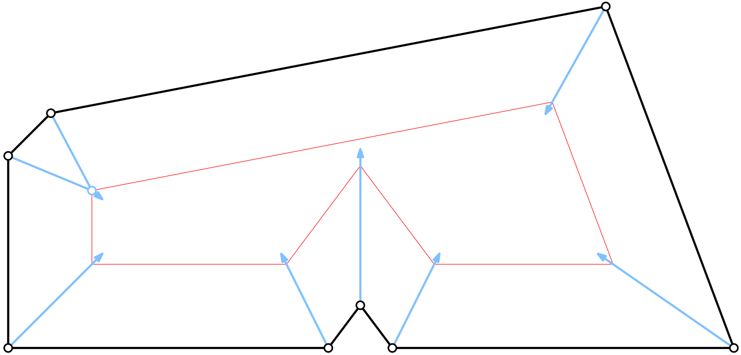
- Common approach: simulate the wavefront propagation.
- Problem: When will the next event happen, and what is it?
- If we solve this, we can incrementally construct the SK.



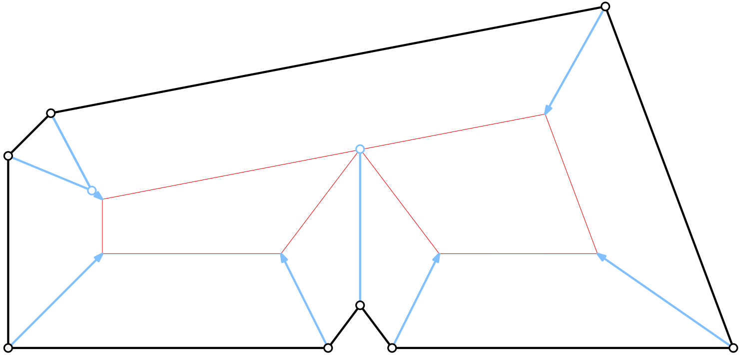
- Common approach: simulate the wavefront propagation.
- Problem: When will the next event happen, and what is it?
- If we solve this, we can incrementally construct the SK.



- Common approach: simulate the wavefront propagation.
- Problem: When will the next event happen, and what is it?
- If we solve this, we can incrementally construct the SK.

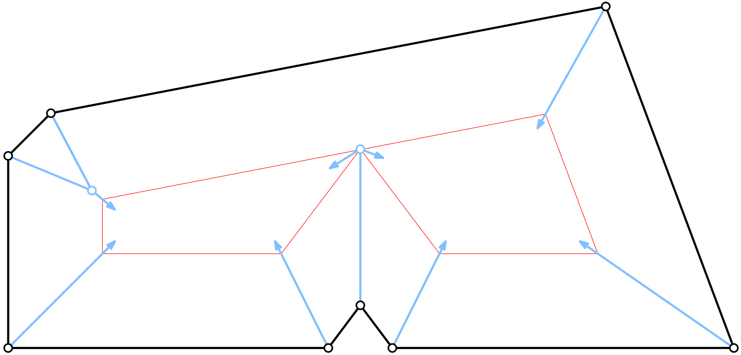


- Common approach: simulate the wavefront propagation.
- Problem: When will the next event happen, and what is it?
- If we solve this, we can incrementally construct the SK.



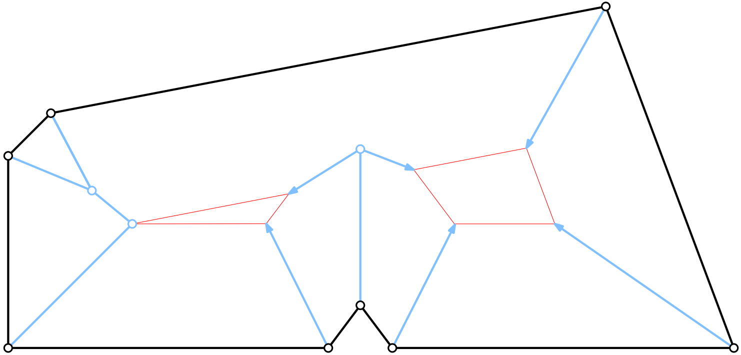
Computing the Straight Skeleton

- Common approach: simulate the wavefront propagation.
- Problem: When will the next event happen, and what is it?
- If we solve this, we can incrementally construct the SK.



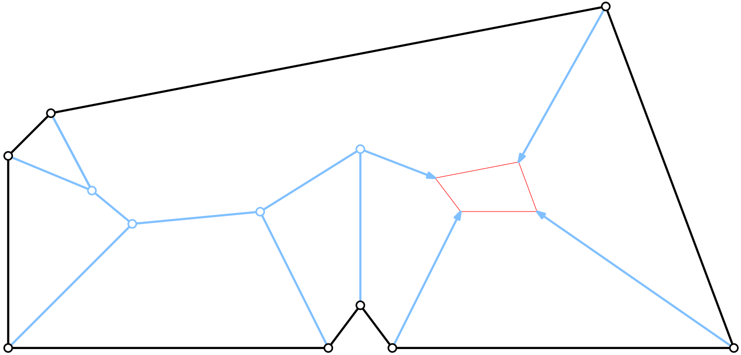
Computing the Straight Skeleton

- Common approach: simulate the wavefront propagation.
- Problem: When will the next event happen, and what is it?
- If we solve this, we can incrementally construct the SK.



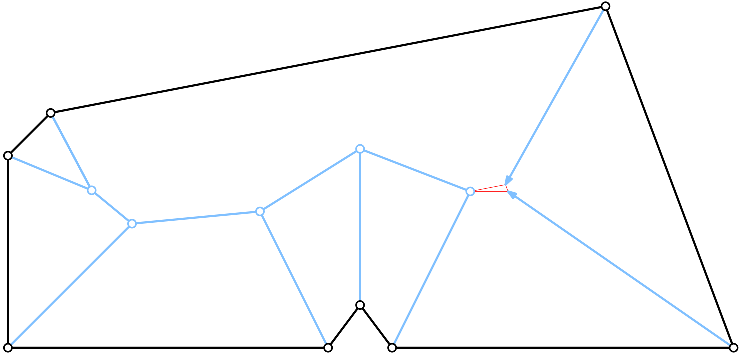
Computing the Straight Skeleton

- Common approach: simulate the wavefront propagation.
- Problem: When will the next event happen, and what is it?
- If we solve this, we can incrementally construct the SK.



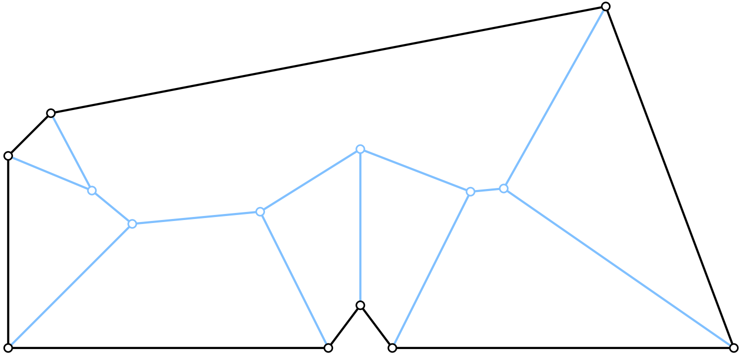
COMPUTING THE STRAIGHT SKELETON

- Common approach: simulate the wavefront propagation.
- Problem: When will the next event happen, and what is it?
- If we solve this, we can incrementally construct the SK.



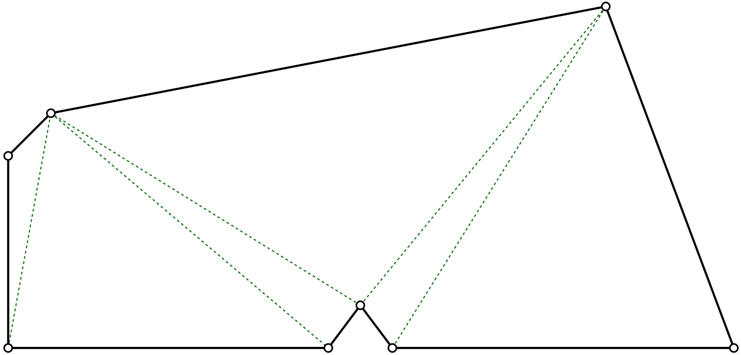
Computing the Straight Skeleton

- Common approach: simulate the wavefront propagation.
- Problem: When will the next event happen, and what is it?
- If we solve this, we can incrementally construct the SK.



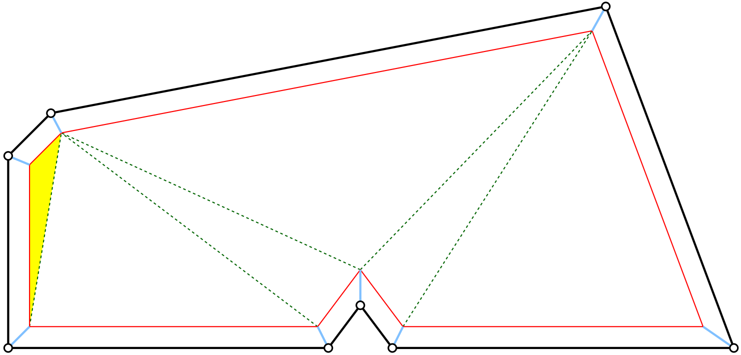
Triangulation-based Algorithm

- Aichholzer, Aurenhammer [AA98].
- Maintain a kinetic triangulation of the points of the plane not yet visited.



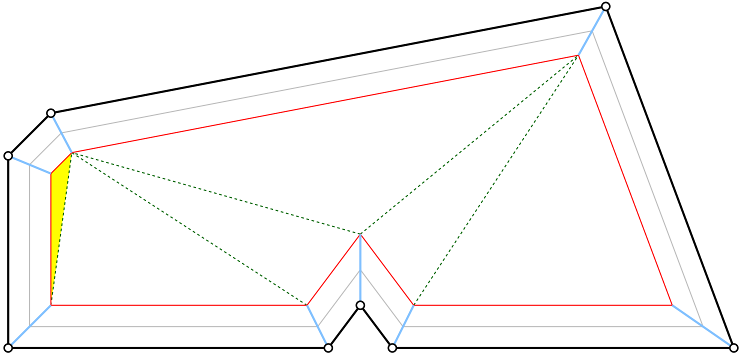
Triangulation-based Algorithm

- Aichholzer, Aurenhammer [AA98].
- Maintain a kinetic triangulation of the points of the plane not yet visited.

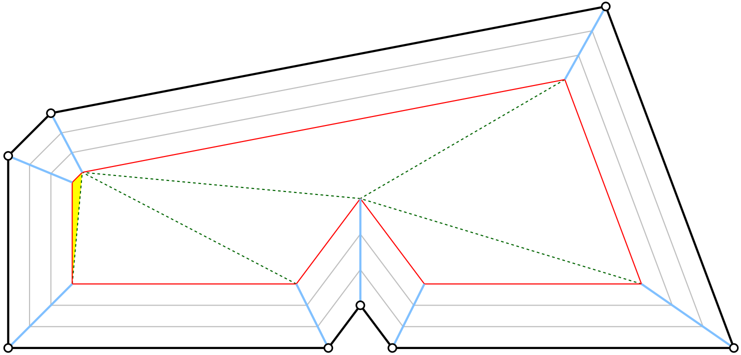


TRIANGULATION-BASED ALGORITHM

- Aichholzer, Aurenhammer [AA98].
- Maintain a kinetic triangulation of the points of the plane not yet visited.

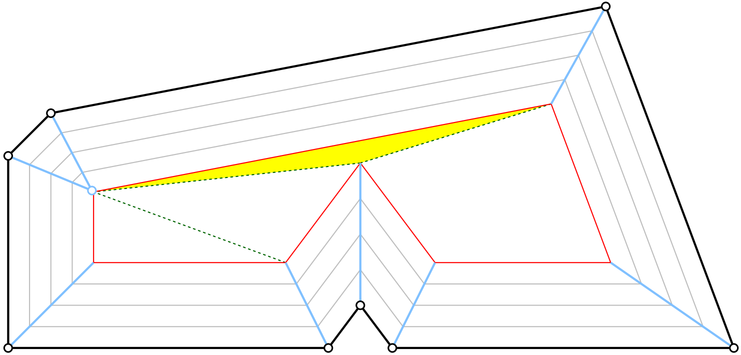


- Aichholzer, Aurenhammer [AA98].
- Maintain a kinetic triangulation of the points of the plane not yet visited.



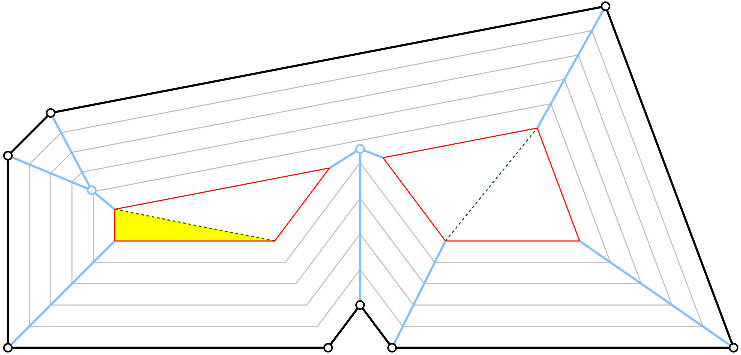
TRIANGULATION-BASED ALGORITHM

- Aichholzer, Aurenhammer [AA98].
- Maintain a kinetic triangulation of the points of the plane not yet visited.
- Collapsing triangles witness edge and split events.



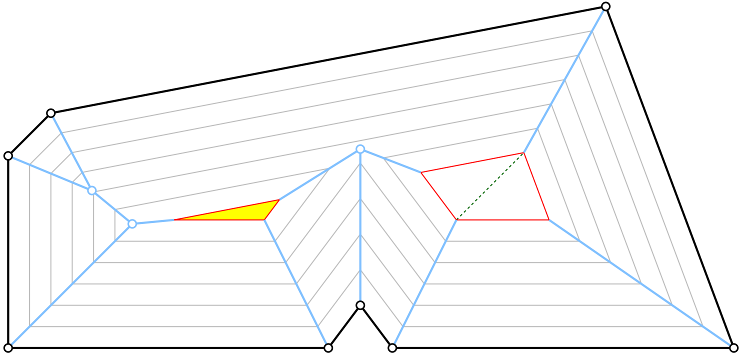
TRIANGULATION-BASED ALGORITHM

- Aichholzer, Aurenhammer [AA98].
- Maintain a kinetic triangulation of the points of the plane not yet visited.
- Collapsing triangles witness edge and split events.

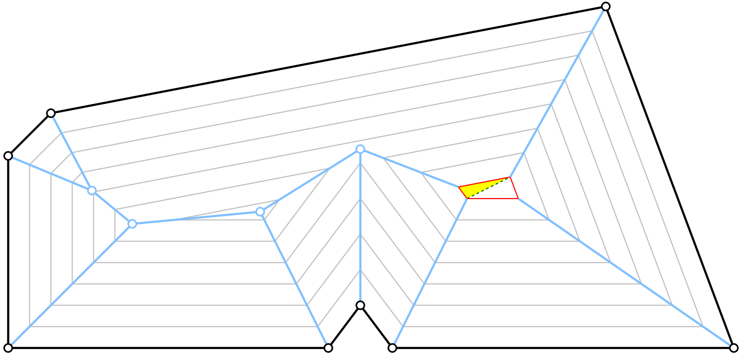


Triangulation-based Algorithm

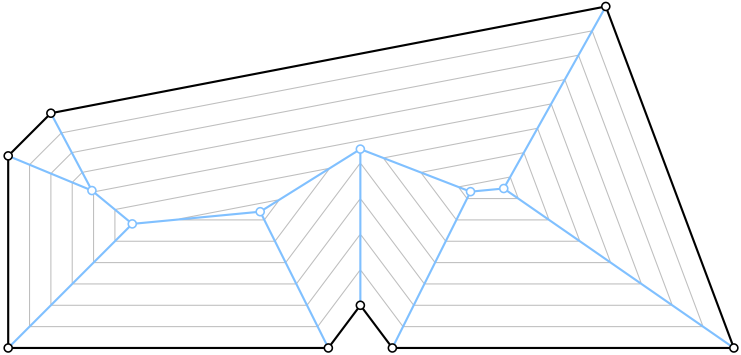
- Aichholzer, Aurenhammer [AA98].
- Maintain a kinetic triangulation of the points of the plane not yet visited.
- Collapsing triangles witness edge and split events.



- Aichholzer, Aurenhammer [AA98].
- Maintain a kinetic triangulation of the points of the plane not yet visited.
- Collapsing triangles witness edge and split events.

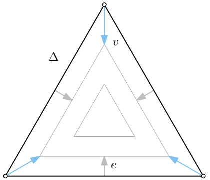


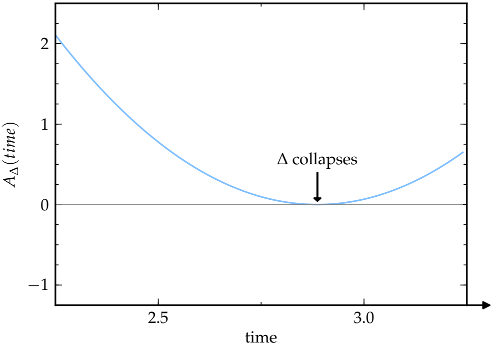
- Aichholzer, Aurenhammer [AA98].
- Maintain a kinetic triangulation of the points of the plane not yet visited.
- Collapsing triangles witness edge and split events.



Triangulation-based Algorithm

- Aichholzer, Aurenhammer [AA98].
- Maintain a kinetic triangulation of the points of the plane not yet visited.
- *Collapsing triangles witness edge and split events.*
- Compute collapse times of triangles.





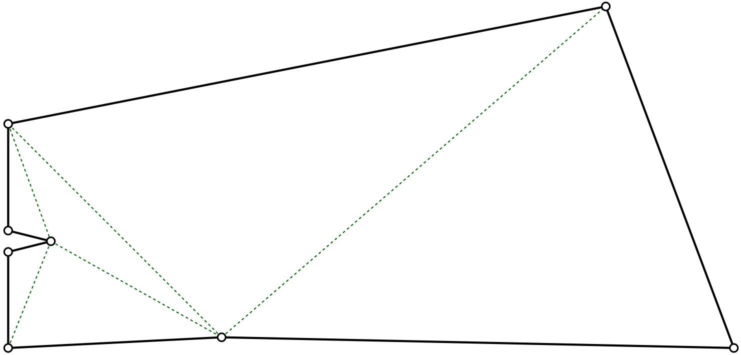
Triangulation-based Algorithm

- Aichholzer, Aurenhammer [AA98].
- Maintain a kinetic triangulation of the points of the plane not yet visited.
- *Collapsing triangles witness edge and split events.*

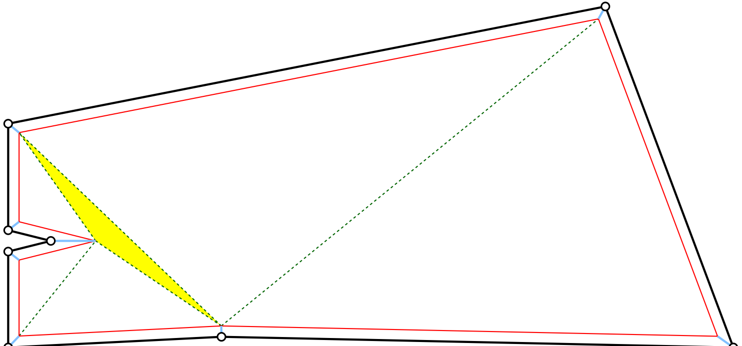
- Compute collapse times of triangles.
- Maintain a priority queue of collapses.
- On events, update triangulation and priority queue as required.

- *We can always easily find the next event, and thus compute the straight skeleton.*

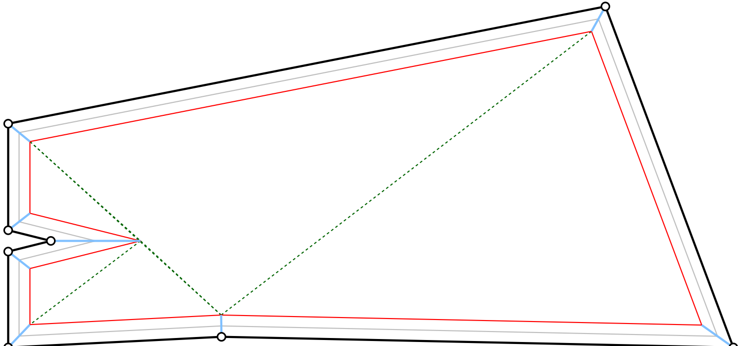
- Caveat: Not all collapses witness changes in the wavefront topology.



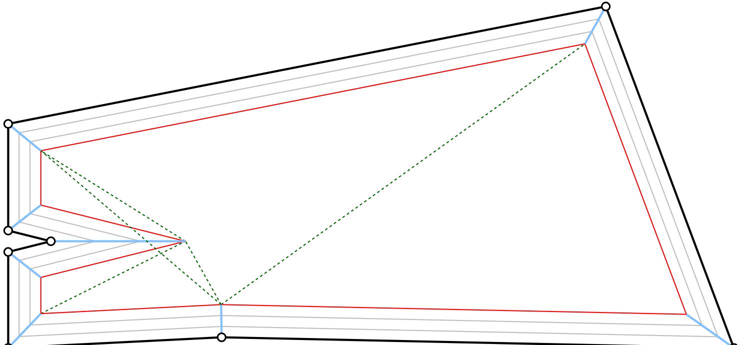
- Caveat: Not all collapses witness changes in the wavefront topology.



- Caveat: Not all collapses witness changes in the wavefront topology.

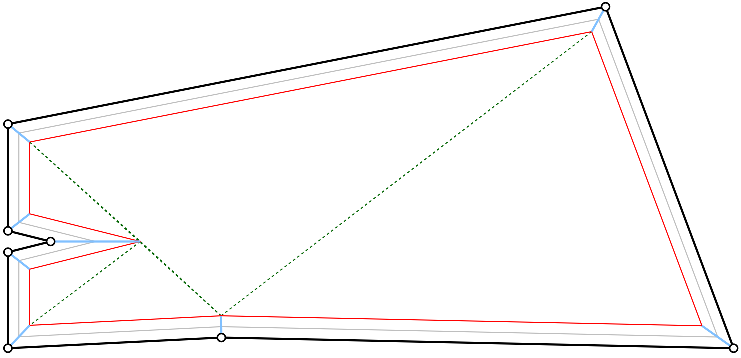


- Caveat: Not all collapses witness changes in the wavefront topology.
- Such collapses cannot be ignored.



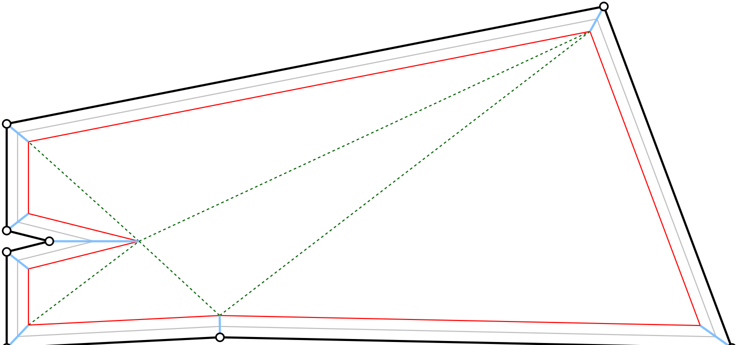
Triangulation-based Algorithm

- Caveat: Not all collapses witness changes in the wavefront topology.
- Such collapses cannot be ignored.
- Instead they need special processing: *flip events*.

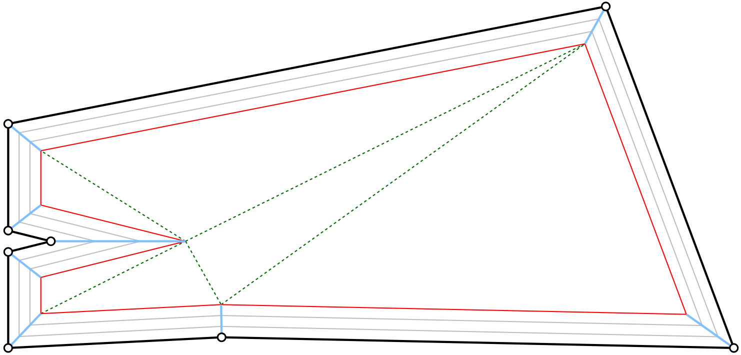


Triangulation-based Algorithm

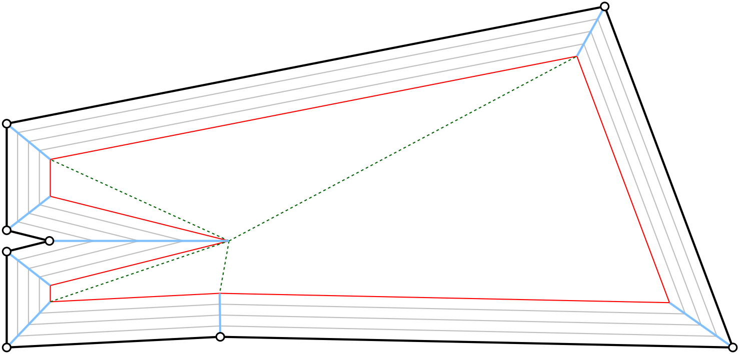
- Caveat: Not all collapses witness changes in the wavefront topology.
- Such collapses cannot be ignored.
- Instead they need special processing: *flip events*.



- Caveat: Not all collapses witness changes in the wavefront topology.
- Such collapses cannot be ignored.
- Instead they need special processing: *flip events*.

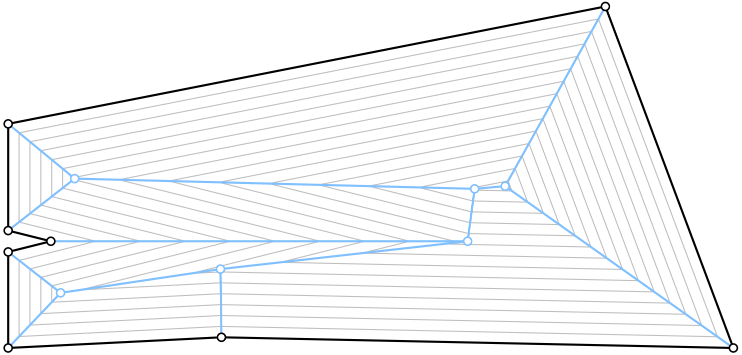


- Caveat: Not all collapses witness changes in the wavefront topology.
- Such collapses cannot be ignored.
- Instead they need special processing: *flip events*.



Triangulation-based Algorithm

- Caveat: Not all collapses witness changes in the wavefront topology.
- Such collapses cannot be ignored.
- Instead they need special processing: *flip events*.

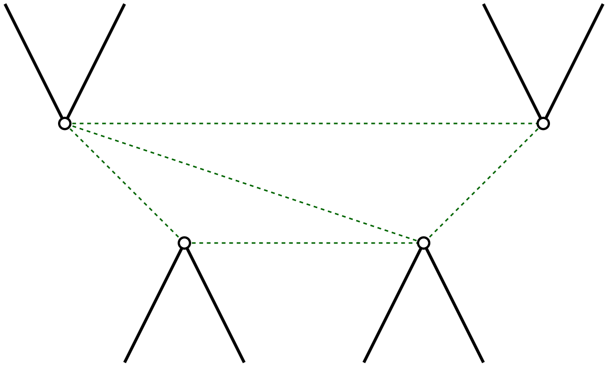


CONTRIBUTION [PHH12]

- We have implemented this algorithm.
- We filled in gaps in the description of the algorithm.
- The algorithm does not always work when input is not in general position. We have identified and corrected these flaws.
- We have run extensive tests using this code.

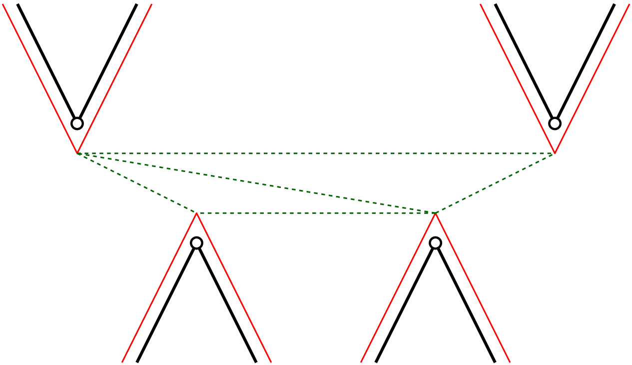
FLIP-EVENT LOOPS

- Without general position, this algorithm can end up in infinite loops.



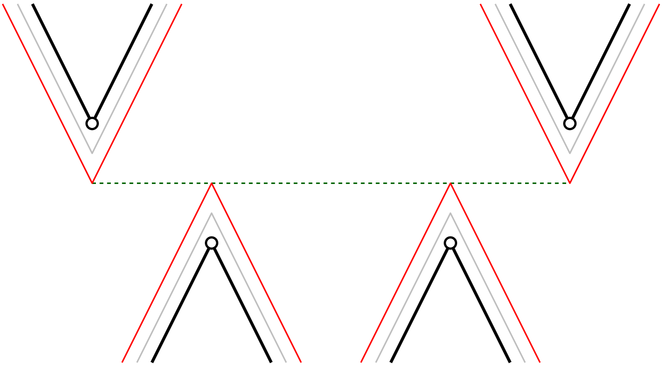
Flip-Event Loops

- Without general position, this algorithm can end up in infinite loops.



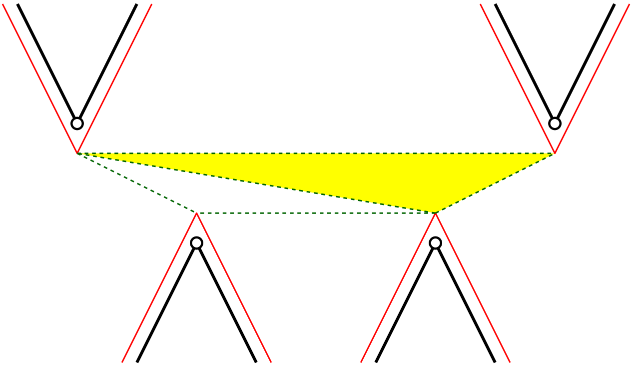
Flip-Event Loops

- Without general position, this algorithm can end up in infinite loops.



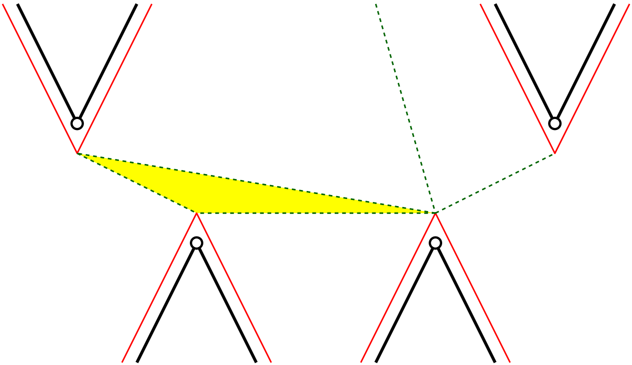
Flip-Event Loops

- Without general position, this algorithm can end up in infinite loops.



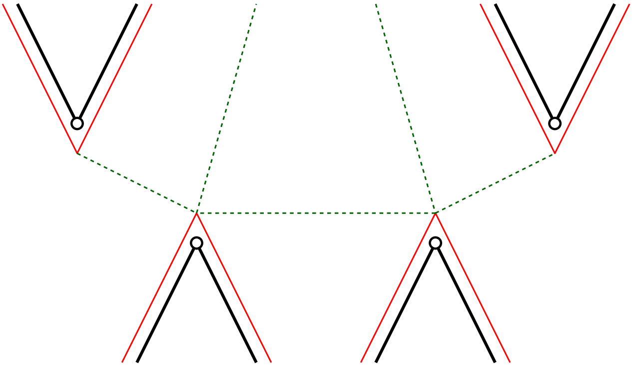
FLIP-EVENT LOOPS

- Without general position, this algorithm can end up in infinite loops.



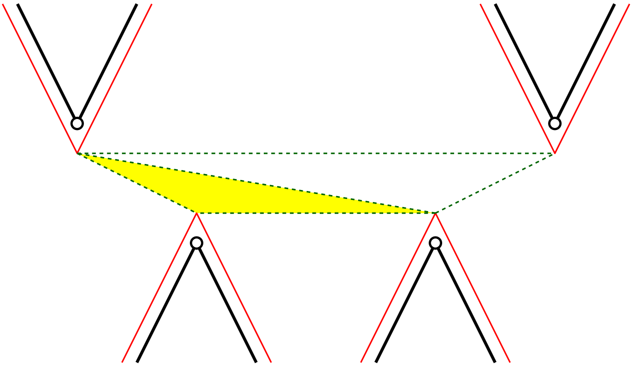
FLIP-EVENT LOOPS

- Without general position, this algorithm can end up in infinite loops.



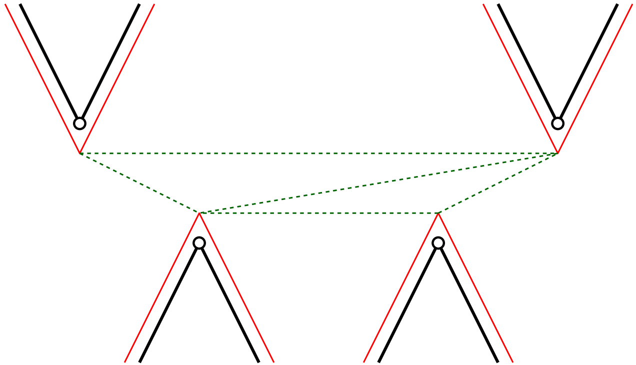
Flip-Event Loops

- Without general position, this algorithm can end up in infinite loops.



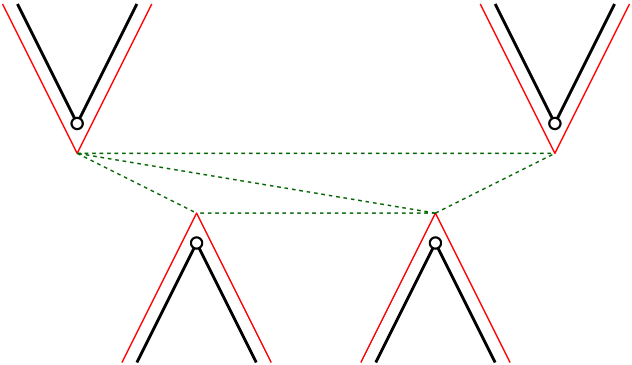
Flip-Event Loops

- Without general position, this algorithm can end up in infinite loops.



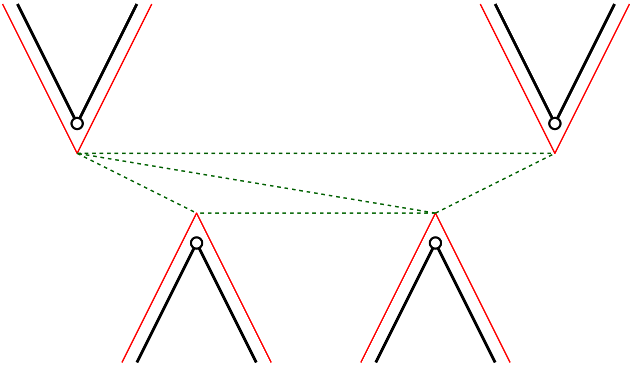
Flip-Event Loops

- Without general position, this algorithm can end up in infinite loops.



FLIP-EVENT LOOPS

- Without general position, this algorithm can end up in infinite loops.



- This is not a result of inexact floating point operations. The same can happen with exact arithmetic!

AVOIDING FLIP-EVENT LOOPS WITH EGC

If we had exact arithmetic operations, the following would work:

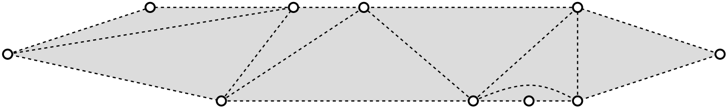
- First, pick the non-flip event $\rightarrow$ reduces triangles
- If only flip events are left, pick the one with the longest edge to flip $\rightarrow$ reduces longest edge (count or length)

Detecting flip-event loops

- Keep a history of flip events $\langle e_1, e_2, \ldots \rangle$ where each $e_i = (t_i, \Delta_i)$.
- This history can be cleared when we encounter an edge or split event.
- If we encounter a flip event a second time, we may be in a flip-event loop.

Brief outline:

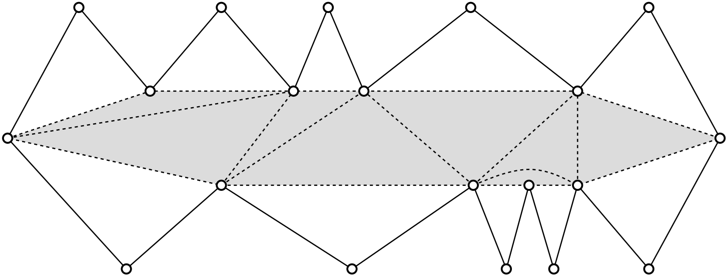
- Identify the polygon P which has collapsed to a straight line.
- Retriangulate P and its neighborhood.



HANDLING FLIP-EVENT LOOPS

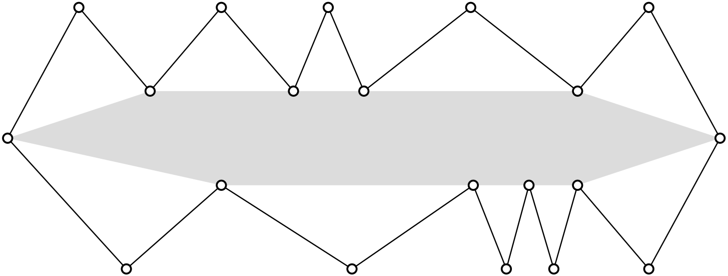
Brief outline:

- Identify the polygon P which has collapsed to a straight line.
- Retriangulate P and its neighborhood.



Brief outline:

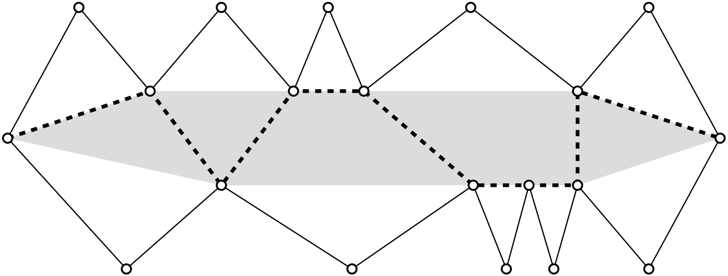
- Identify the polygon P which has collapsed to a straight line.
- Retriangulate P and its neighborhood.



HANDLING FLIP-EVENT LOOPS

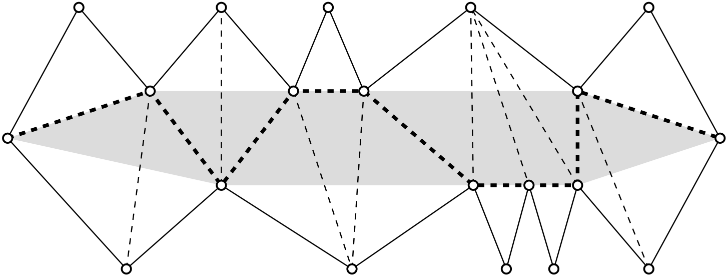
Brief outline:

- Identify the polygon P which has collapsed to a straight line.
- Retriangulate P and its neighborhood.



Brief outline:

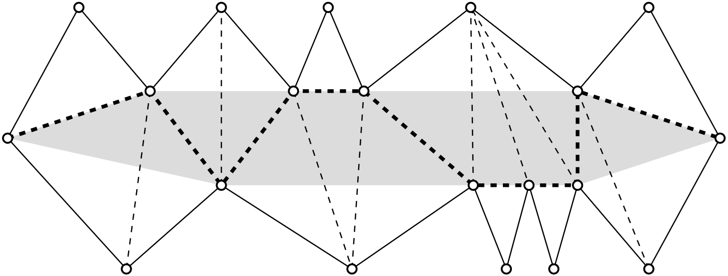
- Identify the polygon *P* which has collapsed to a straight line.
- Retriangulate *P* and its neighborhood.



HANDLING FLIP-EVENT LOOPS

Brief outline:

- Identify the polygon P which has collapsed to a straight line.
- Retriangulate P and its neighborhood.

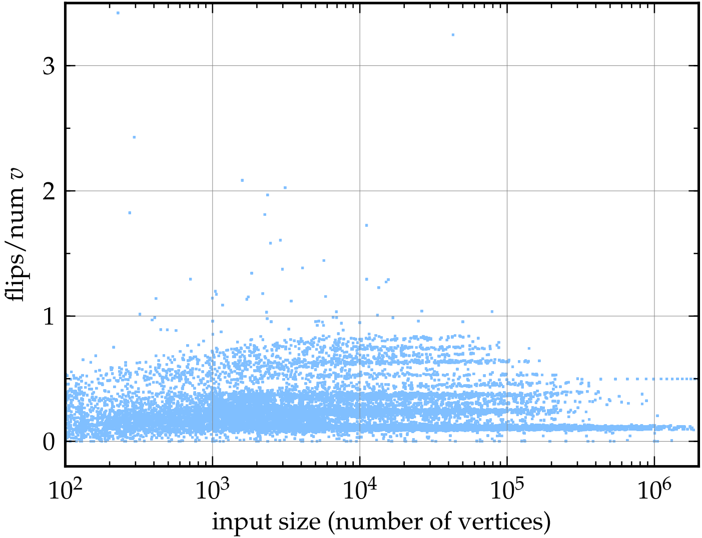


- This approach is also applicable to kinetic triangulations in other algorithms [MHH12].

NUMBER OF FLIP EVENTS

- $\mathcal{O}(n^3)$ is the best known upper bound on the number of flip events,
- No input is known that results in more than quadratically many flip events.
- It turns out that for *practical data* the number of flip events is very linear.

Number of Flip Events, II



Performance observations

	theoretical worst case		practical	
	runtime	space	runtime	space
E&E[1]	$\mathcal{O}(n^{17/11+\epsilon})$	$\mathcal{O}(n^{17/11+\epsilon})$	N/A	
CGAL[2]	$\mathcal{O}(n^2 \log n)$	$\mathcal{O}(n^2)$	$\mathcal{O}(n^2 \log n)$	$\mathcal{O}(n^2)$
Bone[3]	$\mathcal{O}(n^2 \log n)$	$\mathcal{O}(n)$	$\mathcal{O}(n \log n)$	$\mathcal{O}(n)$
Surfer[4]	$\mathcal{O}(n^3 \log n)$	$\mathcal{O}(n)$	$\mathcal{O}(n \log n)$	$\mathcal{O}(n)$

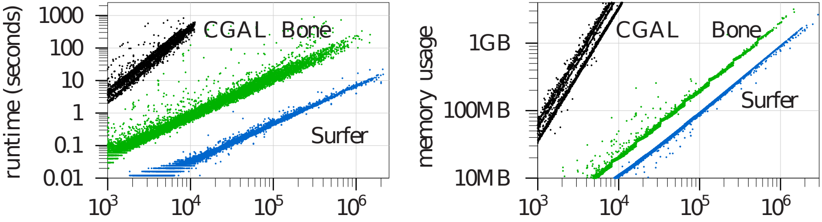
[1] Eppstein and Erickson [EE99]

[2] F. Cacciola, submission to CGAL, 2004

[3] Huber and Held [HH10]

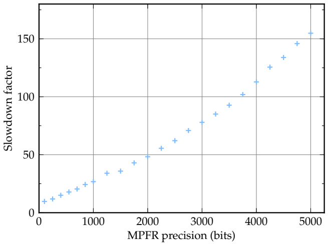
[4] Palfrader et al. [PHH12], based on Aichholzer and Aurenhammer [AA98]

Runtime tests

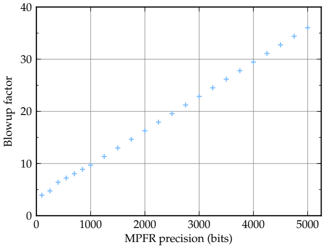


Runtime and memory usage behavior of CGAL, Bone, and Surfer for inputs of different sizes.
Bone and Surfer use their IEEE 754 double precision backend.

MPFR



slowdown



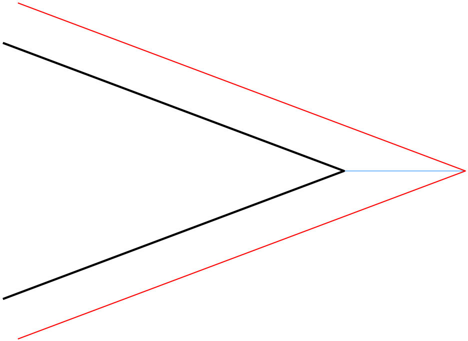
blowup

SUMMARY

- We have implemented Aichholzer and Aurenhammer's algorithm from 1998, filling in details in the algorithm description.
- We fixed real problems that arise in the absence of general position.
- Our approach to handling flip events has wider applications.
- The implementation runs in $\mathcal{O}(n \log n)$ time for *real-world data*. The number of flip events is linear in practice.
- It is industrial-strength, having been tested on tens of thousands of inputs.
- It is the fastest straight skeleton construction code to date, handling millions of vertices in mere seconds.

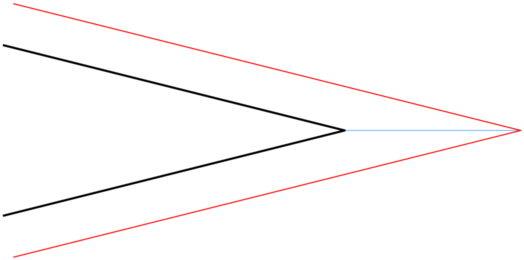
Future Work: Mitered Reflex Vertices

- The current definition causes fast moving vertices for angles approaching $2 \cdot \pi$.



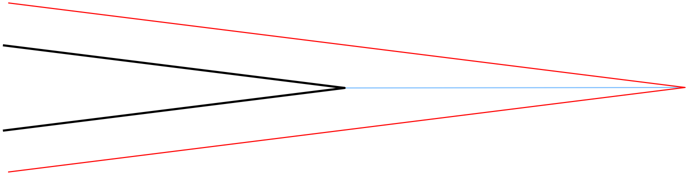
Future Work: Mitered Reflex Vertices

- The current definition causes fast moving vertices for angles approaching $2 \cdot \pi$.



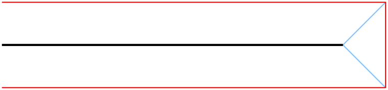
Future Work: Mitered Reflex Vertices

- The current definition causes fast moving vertices for angles approaching $2 \cdot \pi$.



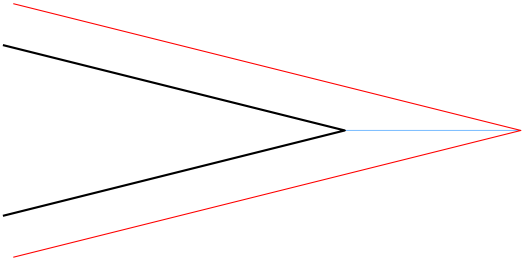
Future Work: Mitered Reflex Vertices

- The current definition causes fast moving vertices for angles approaching $2 \cdot \pi$.



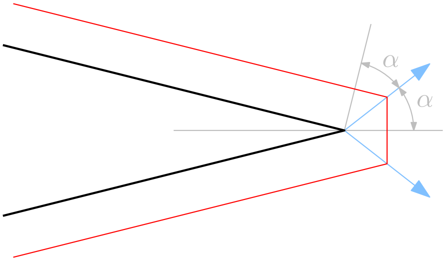
Future Work: Mitered Reflex Vertices

- The current definition causes fast moving vertices for angles approaching $2 \cdot \pi$.
- Investigate and implement some kind of restricted miters.



Future Work: Mitered Reflex Vertices

- The current definition causes fast moving vertices for angles approaching $2 \cdot \pi$.
- Investigate and implement some kind of restricted miters.



FUTURE WORK: UPPER BOUND

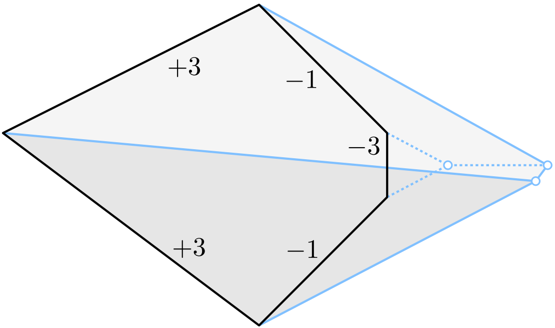
- $\mathcal{O}(n^3)$ is the best known upper bound on the number of flip events.

Future Work: Upper bound

- $\mathcal{O}(n^3)$ is the best known upper bound on the number of flip events.
- But Rubin showed $\mathcal{O}(n^{2+\epsilon})$ for kinetic Delaunay Triangulations where vertices move at unit speed [Rubin13].
- Can we transfer this result?

FUTURE WORK: WEIGHTED STRAIGHT SKELETON

- Weighted $\mathcal{SK}$: Edges move at different speeds, maybe even negative speeds.
- Which of the properties of the straight skeleton (planarity, tree structure, faces are monotone) carry over to weighted straight skeletons [BHHKP13]?



Thank you for your attention.

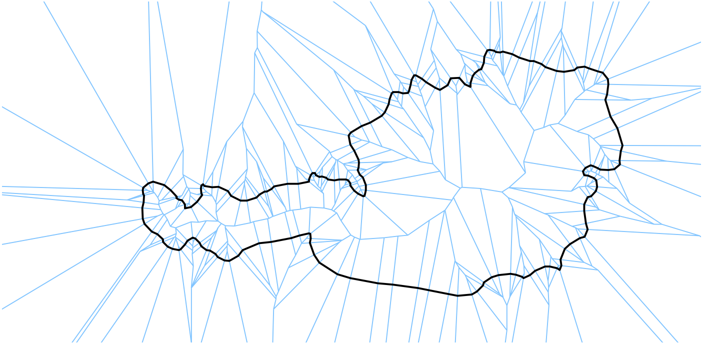
REFERENCES I

- Oswin Aichholzer, Franz Aurenhammer, David Alberts, Bernd Gärtner, ”A Novel Type of Skeleton for Polygons”, Journal of Universal Computer Science, Volume 1, Issue 12, pages 752–761, 1995
- J.M. Oliva, M. Perrin, S. Coquillart, ”3D Reconstruction of Complex Polyhedral Shapes from Contours Using a Simplified Generalized Voronoi Diagram”, Computer Graphics Forum. Volume 15, Issue 3, pages 397–408, 1996
- Oswin Aichholzer, Franz Aurenhammer, ”Straight Skeletons for General Polygonal Figures in the Plane”, Voronoi's Impact on Modern Sciences II, pages 7–21, 1998
- Erik Demaine, Martin Demaine, Anna Lubiw ”Folding and Cutting Paper”, Revised Papers from the Japan Conference on Discrete and Computational Geometry (JCDCG'98)
- David Eppstein, Jeff Erickson, ”Raising Roofs, Crashing Cycles, and Playing Pool Applications of a Data Structure for Finding Pairwise Interactions”, Discrete & Computational Geometry, Volume 22, pages 569–592, 1999
- Kokichi Sugihara, ”Design of Pop-Up Cards Based on Weighted Straight Skeletons”, Proceedings of the 10^{th} International Symposium on Voronoi Diagrams in Science and Engineering (ISVD'13)
- J.-H. Haunert, M. Sester, ”Area Collapse and Road Centerlines Based on Straight Skeletons”, GeoInformatica, Volume 12, pages 169–191, 2008
- Stefan Huber, Martin Held, ”Computing Straight Skeletons of Planar Straight-Line Graphs Based on Motorcycle Graphs”, Proceedings of the 22^{th} Canadian Conference on Computational Geometry (CCCG 2010)
- Peter Palfrader, Martin Held, Stefan Huber, ”On Computing Straight Skeletons by Means of Kinetic Triangulations”, Proceedings of the 20^{th} Annual European Symposium on Algorithms (ESA 2012)
- Willi Mann, Martin Held, Stefan Huber, ”Computing Motorcycle Graphs Based on Kinetic Triangulations”, Proceedings of the 24^{th} Canadian Conference on Computational Geometry (CCCG 2012)
- Therese Biedl, Martin Held, Stefan Huber, Dominik Kaaser, Peter Palfrader, ”Weighted Straight Skeletons In the Plane”, Proceedings of the 25^{th} Canadian Conference on Computational Geometry (CCCG 2013)

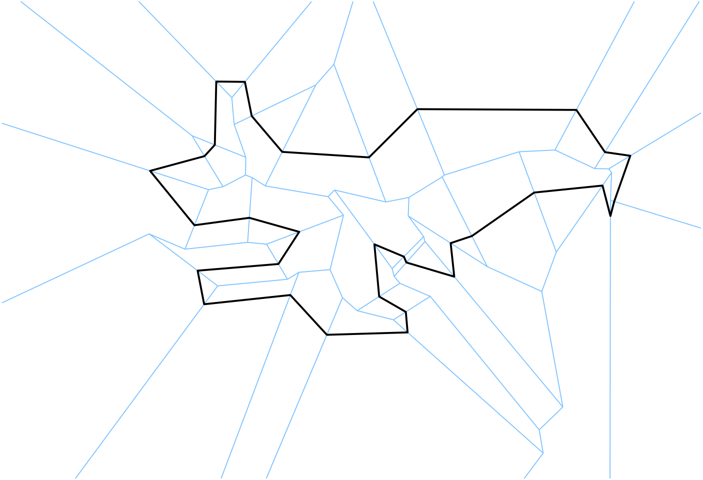
References II

- Natan Rubin, ”On Kinetic Delaunay Triangulations; A Near Quadratic Bound for Unit Speed Motions” Accepted to 54^{th} Annual IEEE Symposium on Foundations of Computer Science (FOCS 2013)

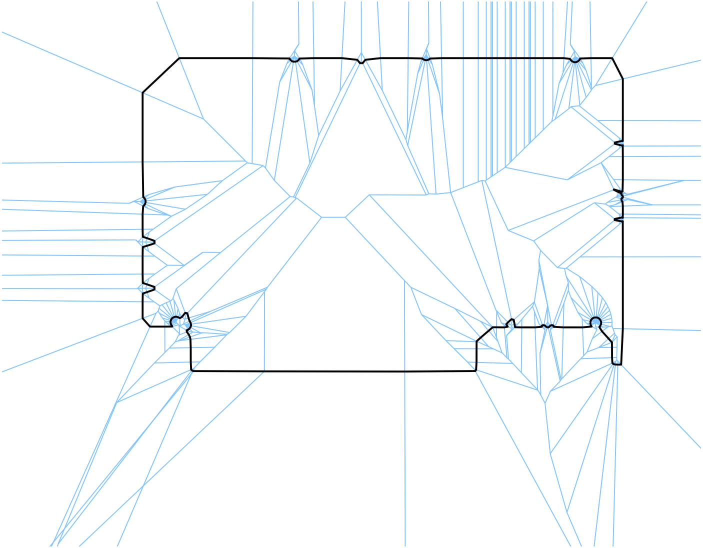
GALLERY: BORDERS OF AUSTRIA



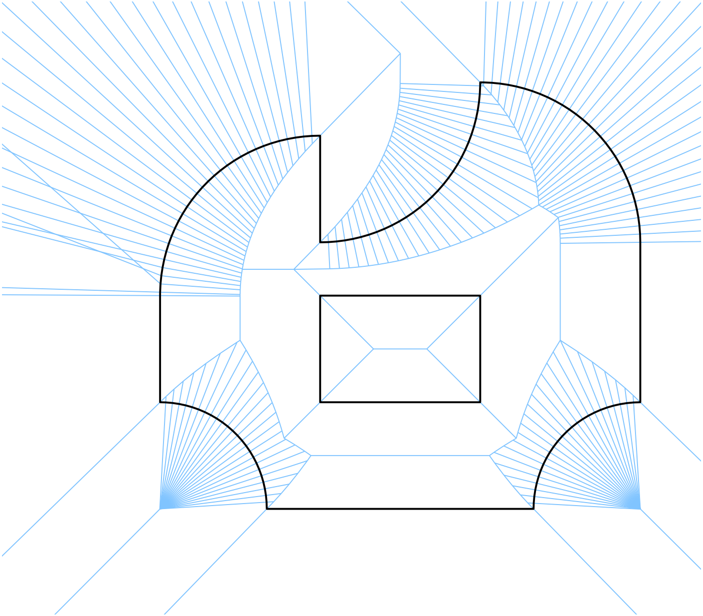
GALLERY: RANDOM POLYGON



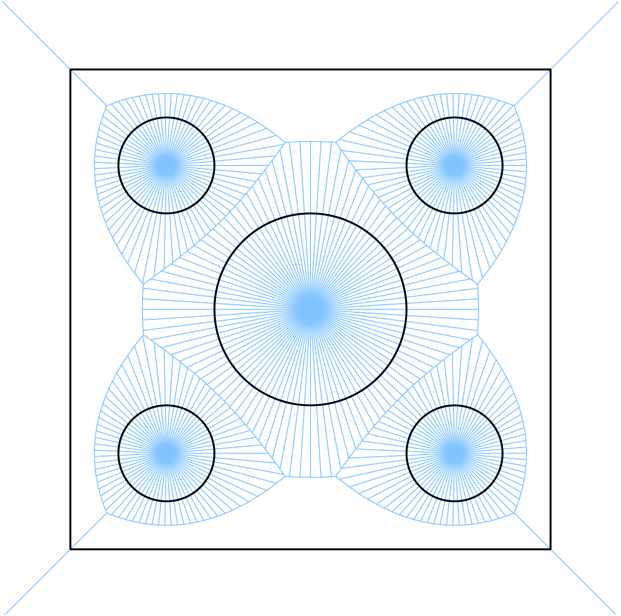
GALLERY: PCB



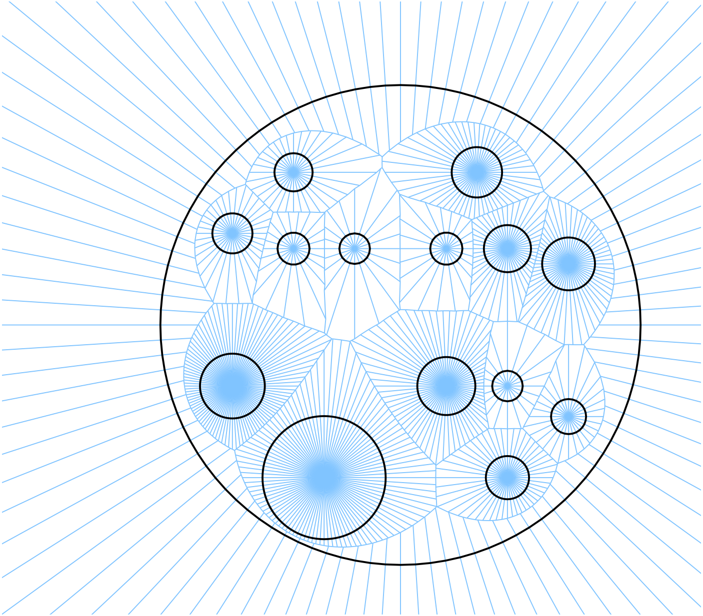
GALLERY: POLYGON WITH HOLE



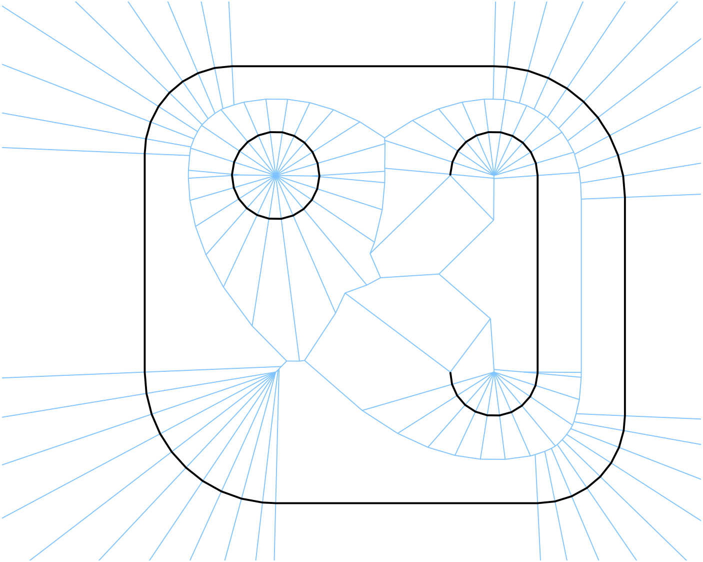
Gallery: Circular Holes



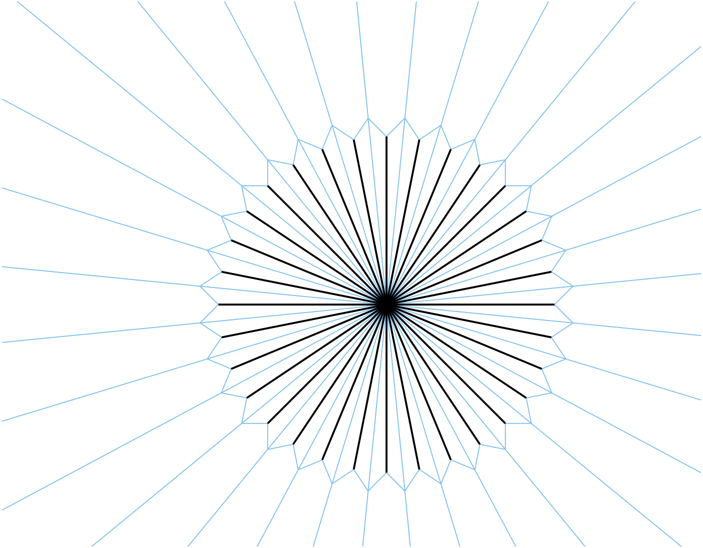
Gallery: More Holes



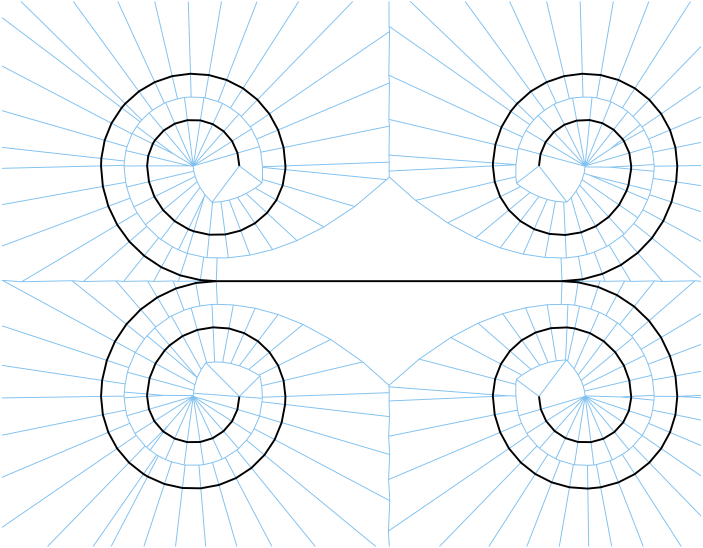
Gallery: Almost Polygon



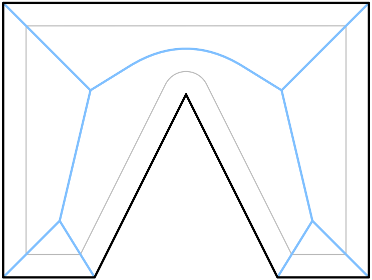
GALLERY: STAR

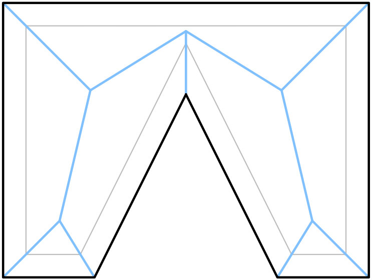


Gallery: Spirals

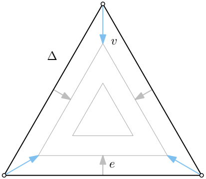


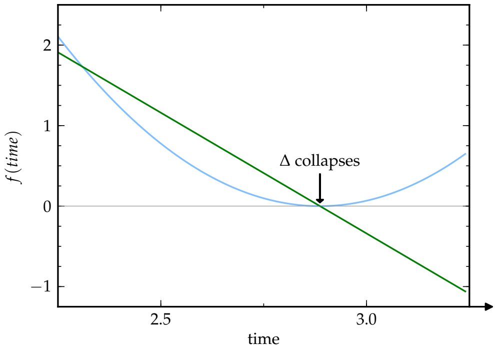
MEDIAL AXIS VS. SK


VD-based MA

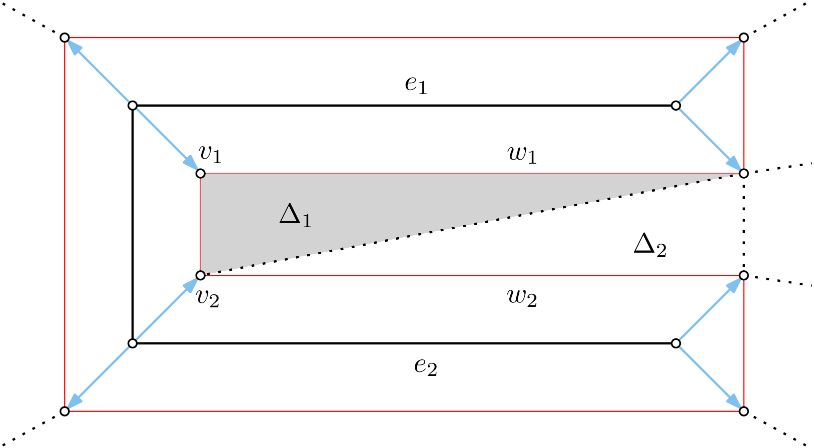

SK

Alternate Computation



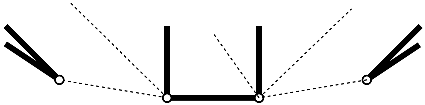


Infinitely fast vertices



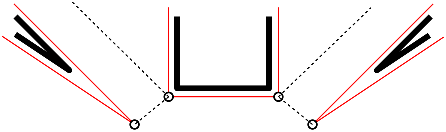
TRIANGULATING

- Triangulate the convex hull.
- Unfortunately the convex hull changes with time, and it matters.

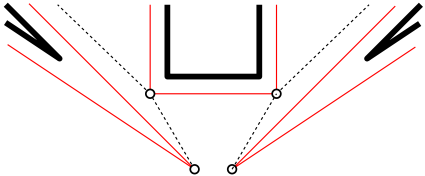


TRIANGULATING

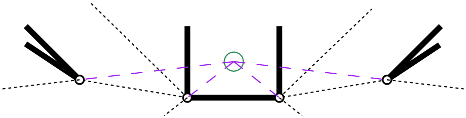
- Triangulate the convex hull.
- Unfortunately the convex hull changes with time, and it matters.



- Triangulate the convex hull.
- Unfortunately the convex hull changes with time, and it matters.



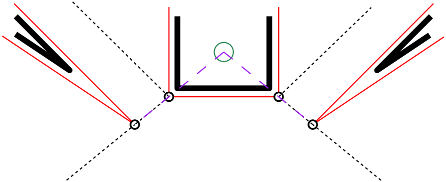
- Triangulate the convex hull.
- Unfortunately the convex hull changes with time, and it matters.



- We need to update the triangulation at some point before this happens, but how?

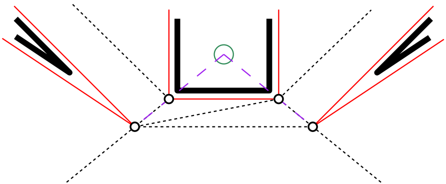
TRIANGULATING

- Triangulate the convex hull.
- Unfortunately the convex hull changes with time, and it matters.



- We need to update the triangulation at some point before this happens, but how?

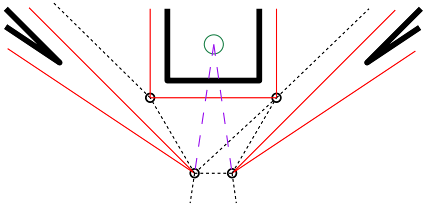
- Triangulate the convex hull.
- Unfortunately the convex hull changes with time, and it matters.



- We need to update the triangulation at some point before this happens, but how?

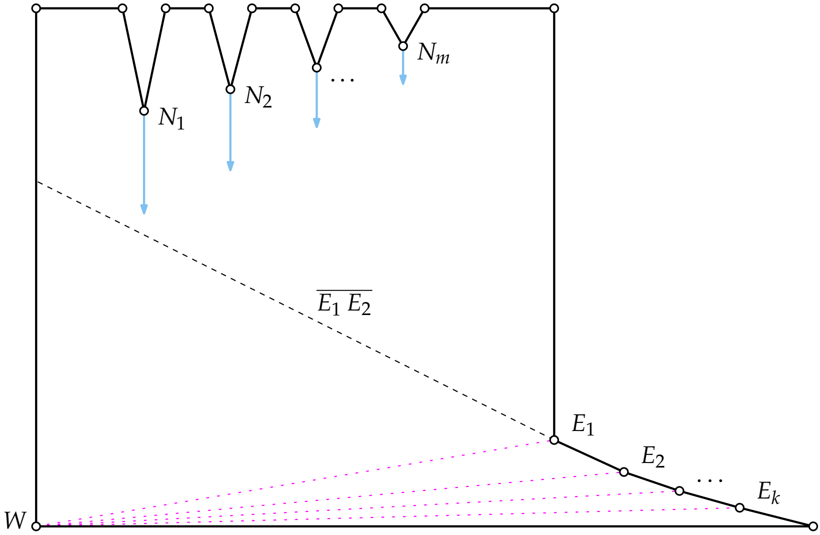
TRIANGULATING

- Triangulate the convex hull.
- Unfortunately the convex hull changes with time, and it matters.

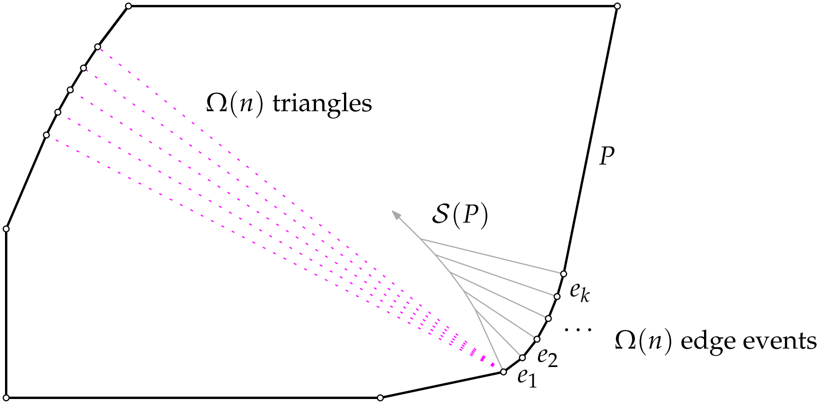


- We need to update the triangulation at some point before this happens, but how?

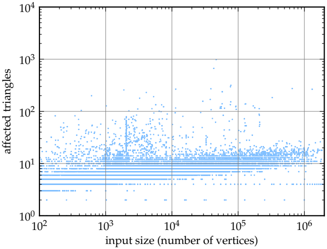
Ω FOR FLIP EVENTS



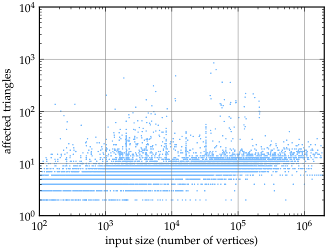
Ω FOR NON-FLIP EVENTS



Affected Triangles, Max

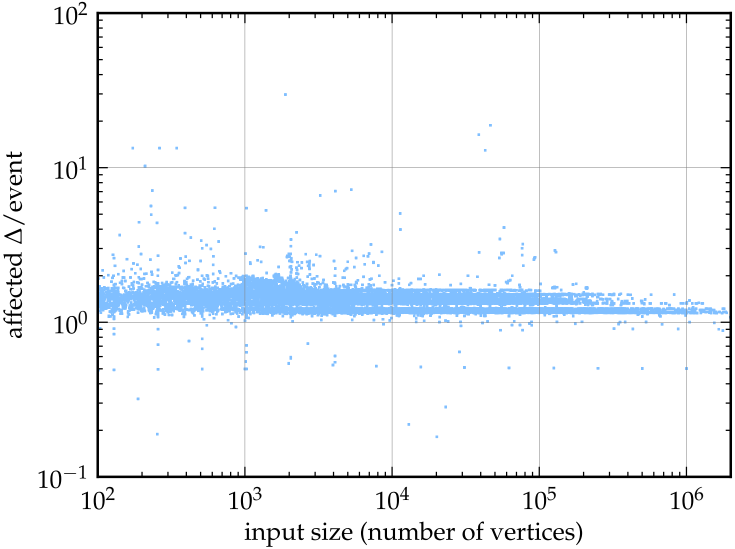


in edge events



in split events

Affected Triangles, Avg



Time spent, phases