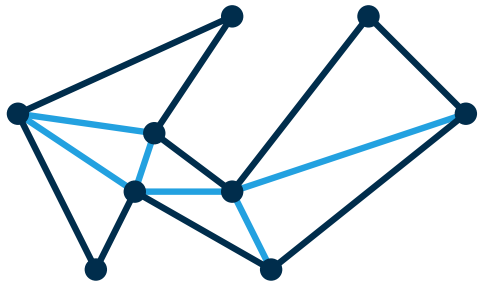
Computational Geometry

Polygon Triangulation

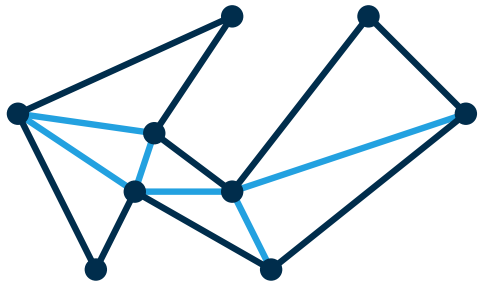
Thomas Bläsius

Polygon Triangulation



Definition
A **triangulation** of a polygon P is a planar subdivision of P such that each face is a triangle.

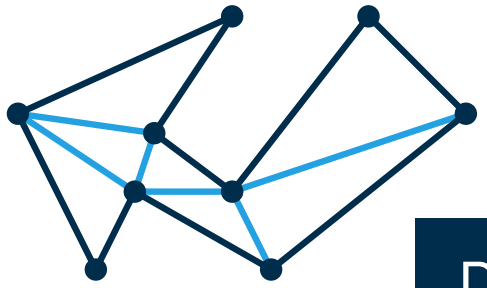
Polygon Triangulation



Definition
A **triangulation** of a polygon P is a planar subdivision of P such that each face is a triangle.

Problem
Given P, find diagonals that triangulate P.

Polygon Triangulation

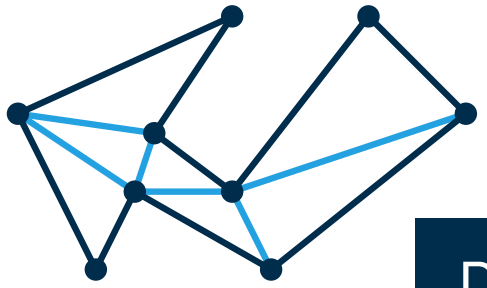


Does this always exist?

Definition
A **triangulation** of a polygon P is a planar subdivision of P such that each face is a triangle.

Problem
Given P, find diagonals that triangulate P.

Polygon Triangulation



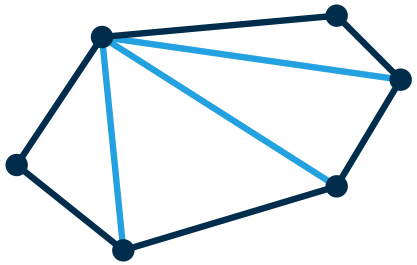
Does this always exist?

Definition
A **triangulation** of a polygon P is a planar subdivision of P such that each face is a triangle.

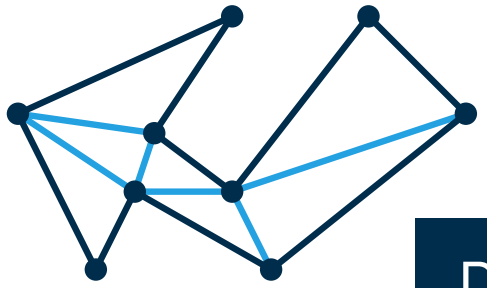
Problem
Given P, find diagonals that triangulate P.

Let's Simplify First

- convex polygons are easy to triangulate



Polygon Triangulation



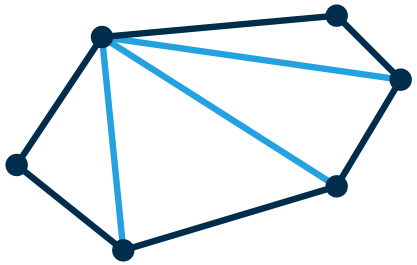
Does this always exist?

Definition
A **triangulation** of a polygon P is a planar subdivision of P such that each face is a triangle.

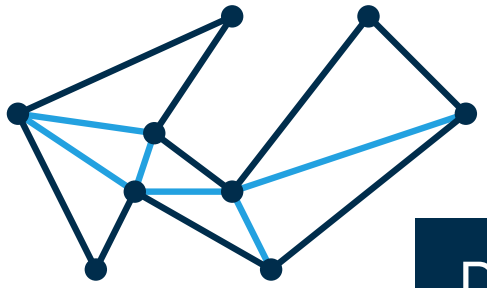
Problem
Given P, find diagonals that triangulate P.

Let's Simplify First

- convex polygons are easy to triangulate
- idea: subdivide P into convex pieces then triangulate those pieces



Polygon Triangulation



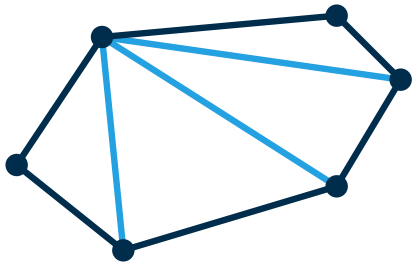
Does this always exist?

Definition
A **triangulation** of a polygon P is a planar subdivision of P such that each face is a triangle.

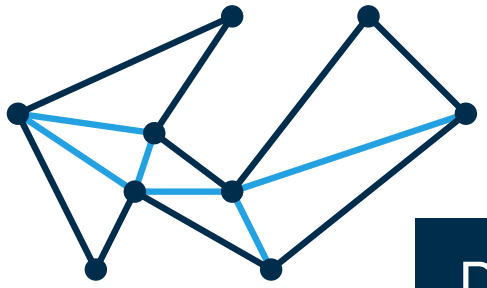
Problem
Given P, find diagonals that triangulate P.

Let's Simplify First

- convex polygons are easy to triangulate
- idea: subdivide P into convex pieces then triangulate those pieces
- problem: finding a convex subdivision is not much easier



Polygon Triangulation



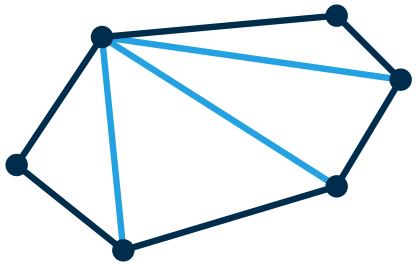
Definition
A **triangulation** of a polygon P is a planar subdivision of P such that each face is a triangle.

Does this always exist?

Problem
Given P, find diagonals that triangulate P.

Let's Simplify First

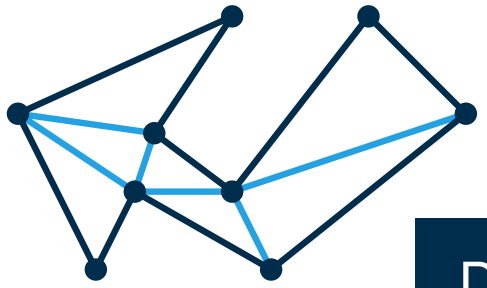
- convex polygons are easy to triangulate
- idea: subdivide P into convex pieces then triangulate those pieces
- problem: finding a convex subdivision is not much easier



Our Plan

- find a weaker condition than convexity

Polygon Triangulation



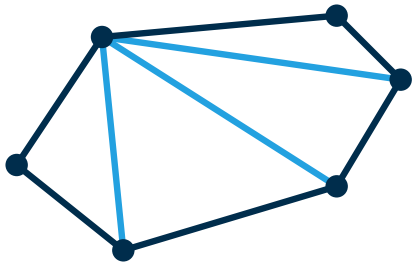
Does this always exist?

Definition
A **triangulation** of a polygon P is a planar subdivision of P such that each face is a triangle.

Problem
Given P, find diagonals that triangulate P.

Let's Simplify First

- convex polygons are easy to triangulate
- idea: subdivide P into convex pieces then triangulate those pieces
- problem: finding a convex subdivision is not much easier



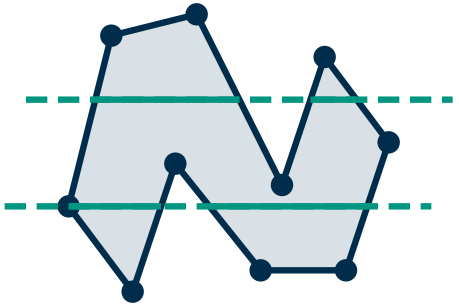
Our Plan

- find a weaker condition than convexity
- subdividing P into pieces with this property becomes easier
- triangulating the pieces becomes more difficult

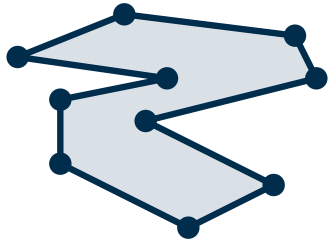
y-Monotone Polygons

Definition
A polygon is ***y*-monotone** if the intersection with every horizontal line is connected.

not *y*-monotone



***y*-monotone**

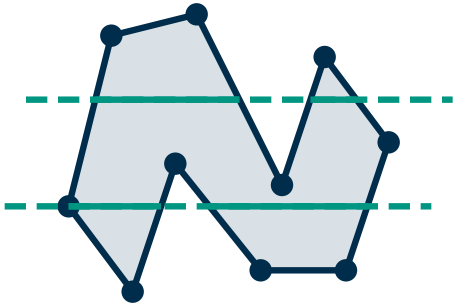


disclaimer: I will not be super consistent whether "polygon" refers to its interior or its boundary; but it will be always clear from the context

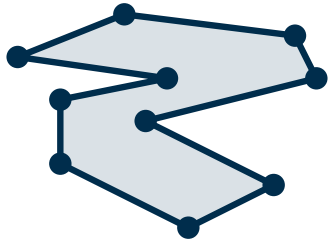
y-Monotone Polygons

Definition
A polygon is ***y*-monotone** if the intersection with every horizontal line is connected.

not *y*-monotone



***y*-monotone**



Remark

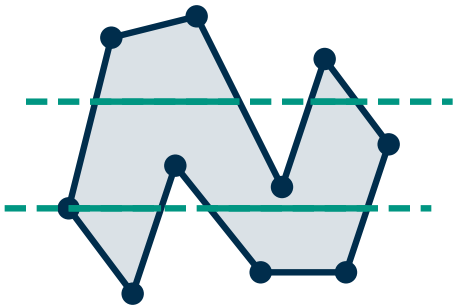
- convex polygons are monotone in every direction

disclaimer: I will not be super consistent whether "polygon" refers to its interior or its boundary; but it will be always clear from the context

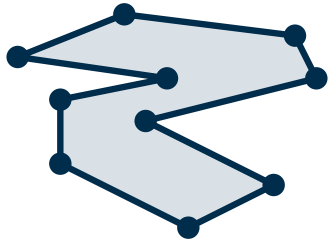
y-Monotone Polygons

Definition
A polygon is ***y*-monotone** if the intersection with every horizontal line is connected.

not y-monotone



y-monotone



disclaimer: I will not be super consistent whether "polygon" refers to its interior or its boundary; but it will be always clear from the context

Remark

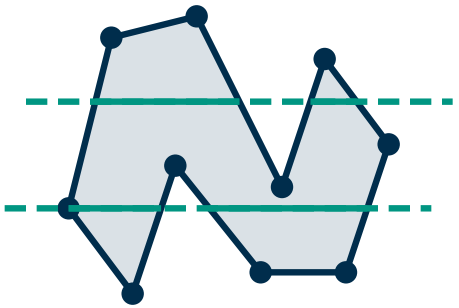
- convex polygons are monotone in every direction

x- and y-monotone $\Rightarrow$ convex?

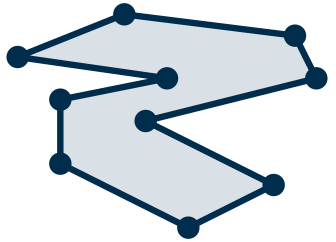
y-Monotone Polygons

Definition
A polygon is ***y*-monotone** if the intersection with every horizontal line is connected.

not y-monotone



y-monotone



disclaimer: I will not be super consistent whether "polygon" refers to its interior or its boundary; but it will be always clear from the context

Remark

- convex polygons are monotone in every direction

x- and y-monotone $\Rightarrow$ convex?

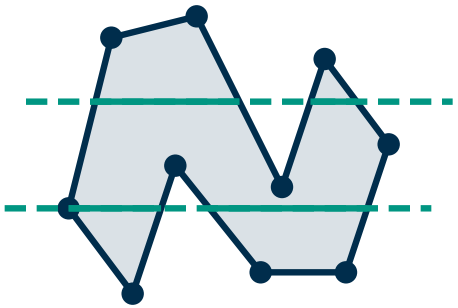
Our Plan

- subdivide arbitrary polygon in $O(n \log n)$ time in y-monotone pieces $\rightarrow$ today

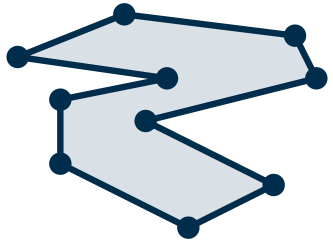
y-Monotone Polygons

Definition
A polygon is ***y*-monotone** if the intersection with every horizontal line is connected.

not y-monotone



y-monotone



disclaimer: I will not be super consistent whether "polygon" refers to its interior or its boundary; but it will be always clear from the context

Remark

- convex polygons are monotone in every direction

x- and y-monotone $\Rightarrow$ convex?

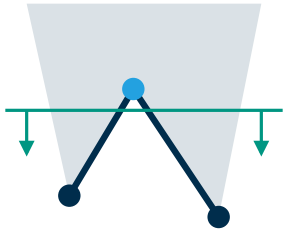
Our Plan

- subdivide arbitrary polygon in $O(n \log n)$ time in y-monotone pieces $\rightarrow$ today
- triangulate a y-monotone polygon in $O(n)$ time $\rightarrow$ exercise sheet

What Makes A Polygon Not y-Monotone?

What Makes A Polygon Not y-Monotone?

Split Vertex

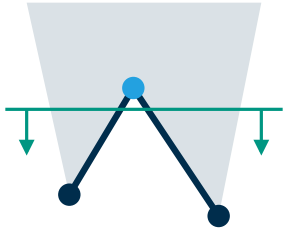


- edges lie below
- polygon lies above
- polygon splits (coming from above)

What Makes A Polygon Not y-Monotone?

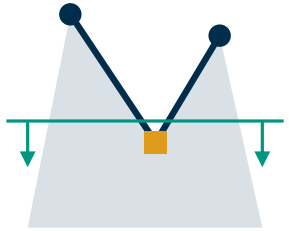
Split Vertex

- edges lie below
- polygon lies above
- polygon splits (coming from above)



Merge Vertex

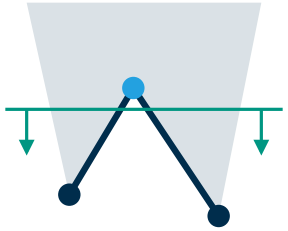
- edges lie above
- polygon lies below
- polygons parts merge



What Makes A Polygon Not y-Monotone?

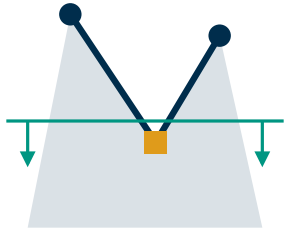
Split Vertex

- edges lie below
- polygon lies above
- polygon splits (coming from above)



Merge Vertex

- edges lie above
- polygon lies below
- polygons parts merge



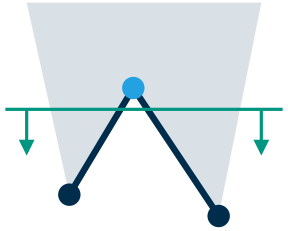
Observation

- a merge or split vertex exists $\Rightarrow$ the polygon is not y-monotone

What Makes A Polygon Not y-Monotone?

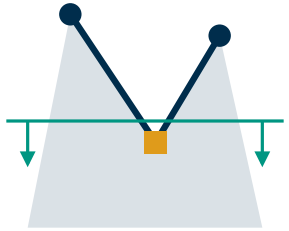
Split Vertex

- edges lie below
- polygon lies above
- polygon splits (coming from above)



Merge Vertex

- edges lie above
- polygon lies below
- polygons parts merge

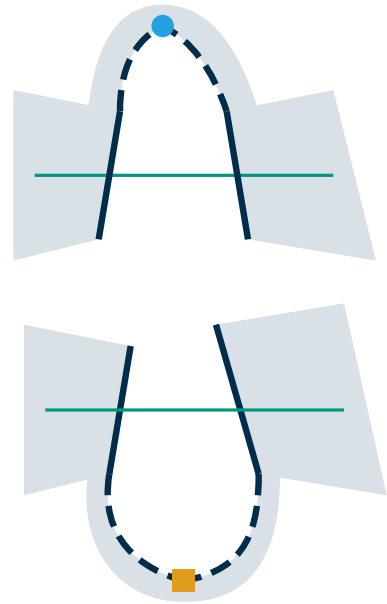


Observation

- a merge or split vertex exists $\Rightarrow$ the polygon is not y-monotone

Lemma **(y-monotonicity)**
A polygon is y-monotone if and only if it has no split or merge vertex.

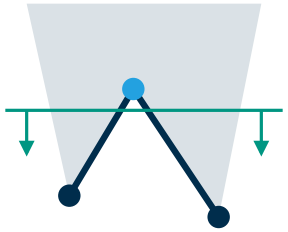
proof by picture



What Makes A Polygon Not y-Monotone?

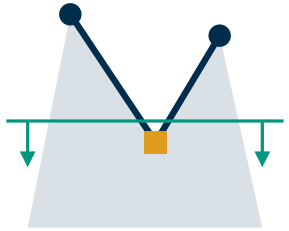
Split Vertex

- edges lie below
- polygon lies above
- polygon splits (coming from above)



Merge Vertex

- edges lie above
- polygon lies below
- polygons parts merge

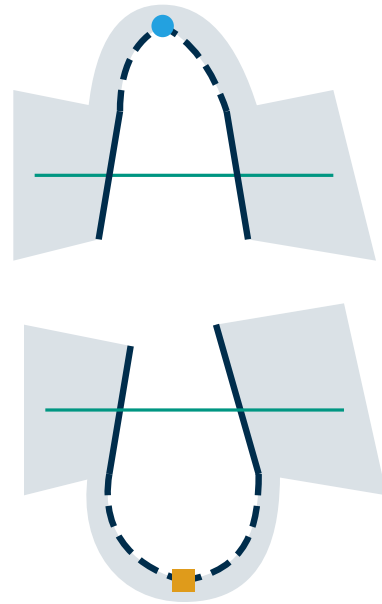


Observation

- a merge or split vertex exists $\Rightarrow$ the polygon is not y-monotone

Lemma **(y-monotonicity)**
A polygon is y-monotone if and only if it has no split or merge vertex.

proof by picture



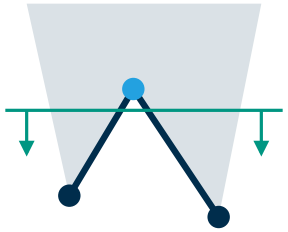
Goal

- eliminate all split and merge vertices by inserting diagonals

What Makes A Polygon Not y-Monotone?

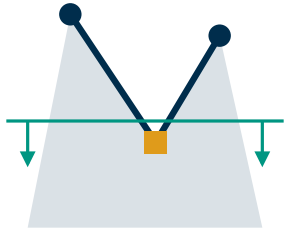
Split Vertex

- edges lie below
- polygon lies above
- polygon splits (coming from above)



Merge Vertex

- edges lie above
- polygon lies below
- polygons parts merge

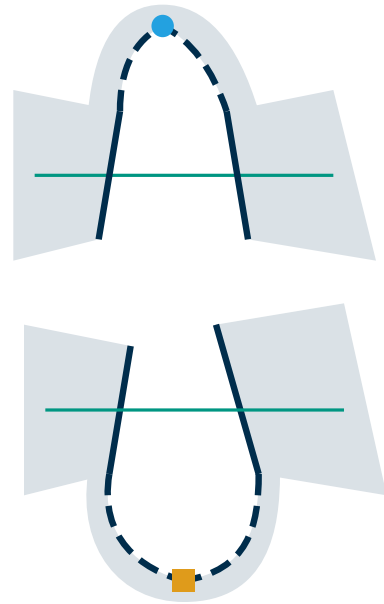


Observation

- a merge or split vertex exists $\Rightarrow$ the polygon is not y-monotone

Lemma **(y-monotonicity)**
A polygon is y-monotone if and only if it has no split or merge vertex.

proof by picture



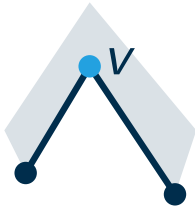
Goal

- eliminate all split and merge vertices by inserting diagonals
- upwards for split vertices and downwards for merge vertices

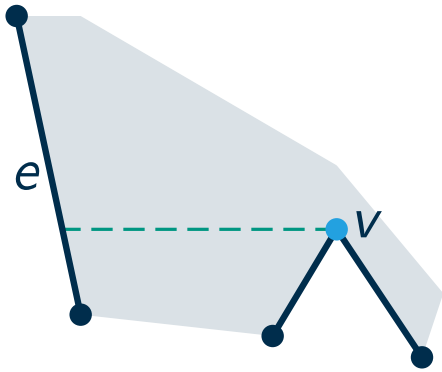
Eliminating Split Vertices

Idea For Split Vertex *v*

- idea: connect *v* to vertex *u* that is above *v* and close to *v*



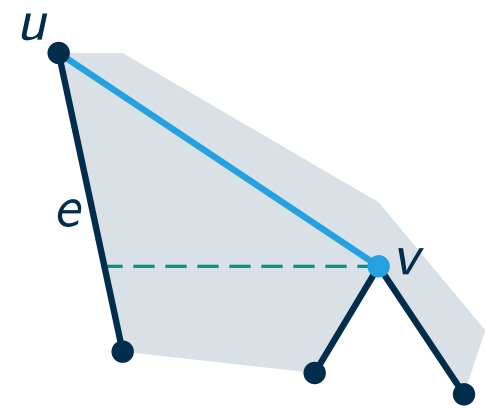
Eliminating Split Vertices



Idea For Split Vertex *v*

- idea: connect *v* to vertex *u* that is above *v* and close to *v*
- *e*: edge to the left of *v* (“to the left of”: the next edge you hit when shooting a ray from *v* to the left)

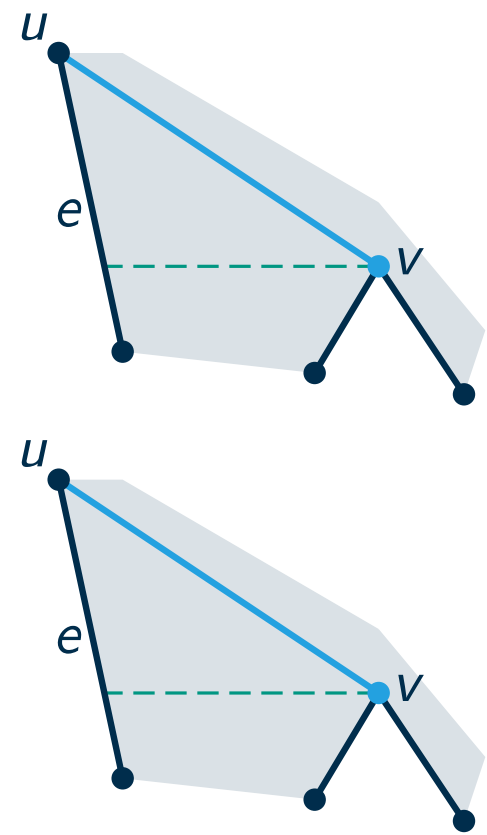
Eliminating Split Vertices



Idea For Split Vertex *v*

- idea: connect *v* to vertex *u* that is above *v* and close to *v*
- *e*: edge to the left of *v* (“to the left of”: the next edge you hit when shooting a ray from *v* to the left)
- choose *u* to be the upper vertex of *e*

Eliminating Split Vertices



Idea For Split Vertex *v*

- idea: connect *v* to vertex *u* that is above *v* and close to *v*
- *e*: edge to the left of *v* (“to the left of”: the next edge you hit when shooting a ray from *v* to the left)
- choose *u* to be the upper vertex of *e*

Issue (And How To Fix It)

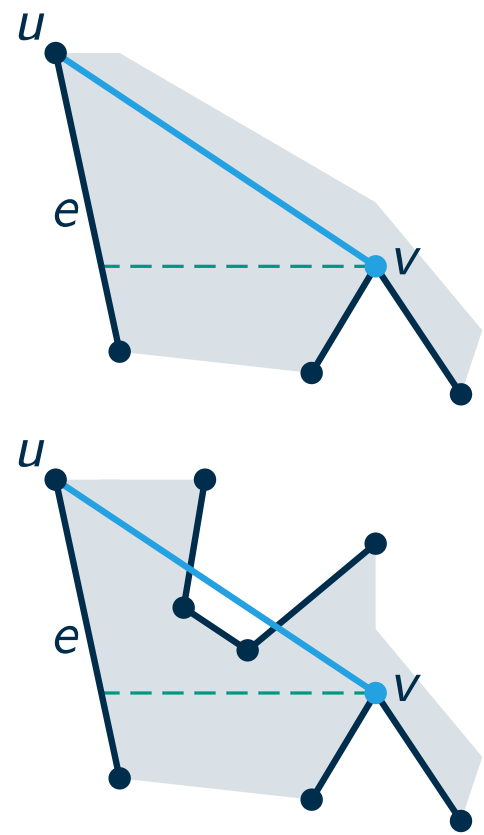
Eliminating Split Vertices

Idea For Split Vertex v

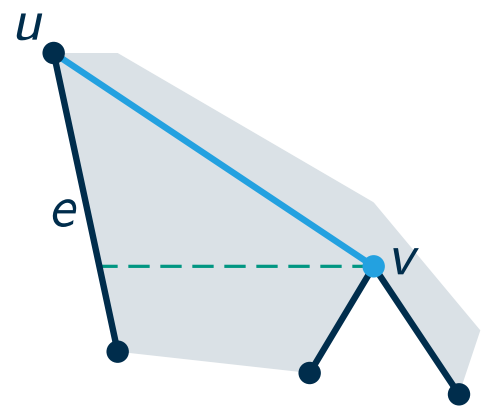
- idea: connect v to vertex u that is above v and close to v
- e: edge to the left of v ("to the left of": the next edge you hit when shooting a ray from v to the left)
- choose u to be the upper vertex of e

Issue (And How To Fix It)

- uv might intersect another edge of the polygon

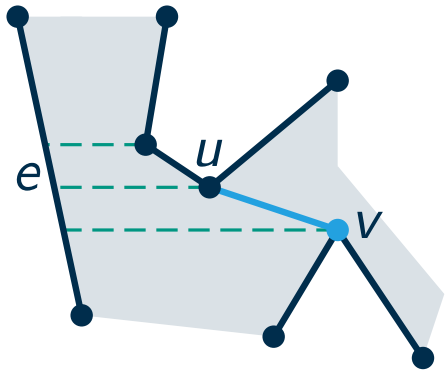


Eliminating Split Vertices



Idea For Split Vertex *v*

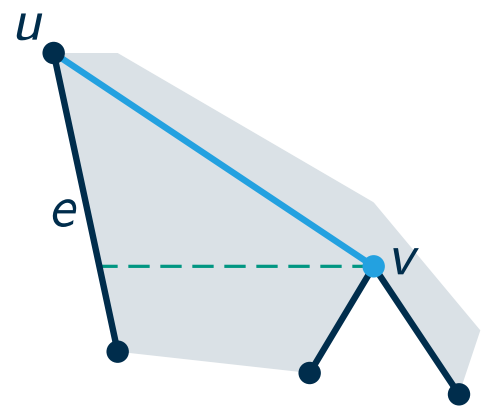
- idea: connect *v* to vertex *u* that is above *v* and close to *v*
- *e*: edge to the left of *v* (“to the left of”: the next edge you hit when shooting a ray from *v* to the left)
- choose *u* to be the upper vertex of *e*



Issue (And How To Fix It)

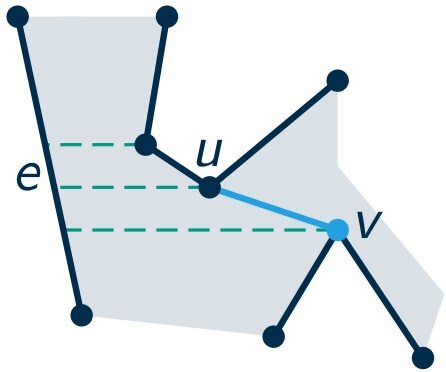
- *uv* might intersect another edge of the polygon
- fix: choose for *u* the lowest vertex above *v* such that *e* is to the left of *u*

Eliminating Split Vertices



Idea For Split Vertex *v*

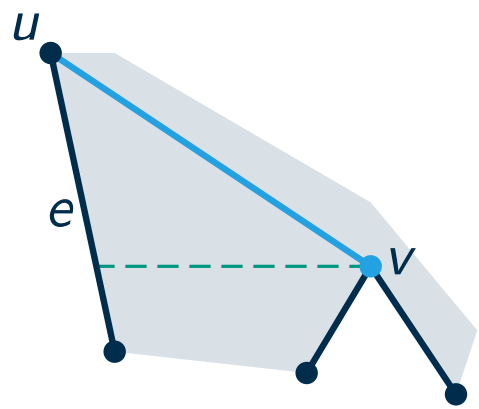
- idea: connect *v* to vertex *u* that is above *v* and close to *v*
- *e*: edge to the left of *v* (“to the left of”: the next edge you hit when shooting a ray from *v* to the left)
- choose *u* to be the upper vertex of *e*



Issue (And How To Fix It)

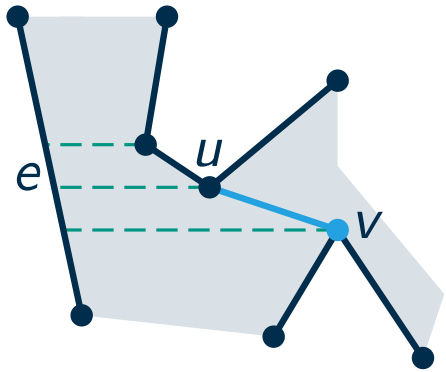
- *uv* might intersect another edge of the polygon
- fix: choose for *u* the lowest vertex above *v* such that *e* is to the left of *u*
- we call *u* the **helper** of *e* (note: it depends on *v*)

Eliminating Split Vertices



Idea For Split Vertex v

- idea: connect v to vertex u that is above v and close to v
- e: edge to the left of v ("to the left of": the next edge you hit when shooting a ray from v to the left)
- choose u to be the upper vertex of e

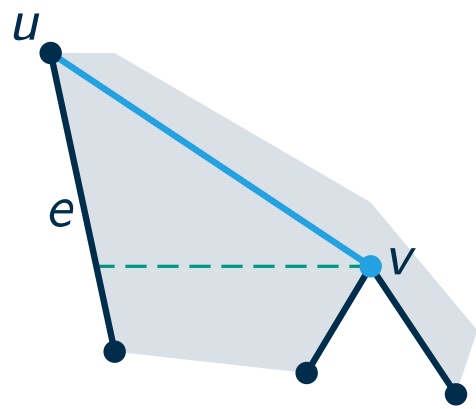


Issue (And How To Fix It)

- uv might intersect another edge of the polygon
- fix: choose for u the lowest vertex above v such that e is to the left of u
- we call u the **helper** of e (note: it depends on v)

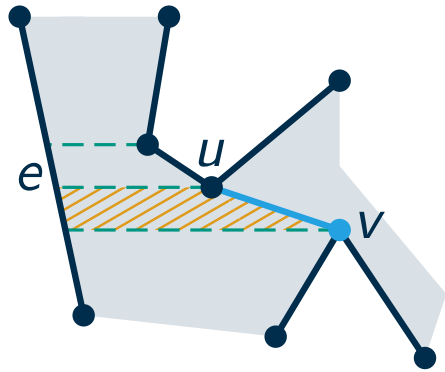
Lemma **(the helper is helpful)**
Let v be a split vertex, e the edge left of v, and u the helper of e (wrt v). Then uv does not intersect an edge of the polygon (except in u and v).

Eliminating Split Vertices



Idea For Split Vertex *v*

- idea: connect *v* to vertex *u* that is above *v* and close to *v*
- *e*: edge to the left of *v* ("to the left of": the next edge you hit when shooting a ray from *v* to the left)
- choose *u* to be the upper vertex of *e*



Issue (And How To Fix It)

- *uv* might intersect another edge of the polygon
- fix: choose for *u* the lowest vertex above *v* such that *e* is to the left of *u*
- we call *u* the **helper** of *e* (note: it depends on *v*)

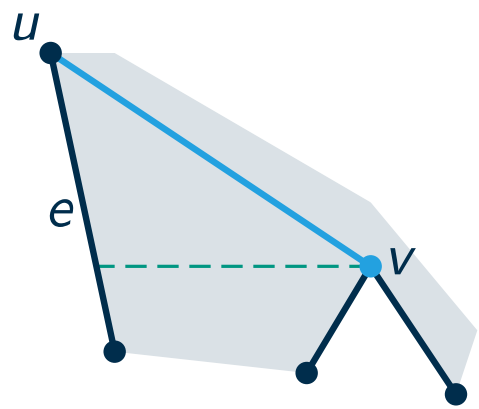
Lemma **(the helper is helpful)**
Let *v* be a split vertex, *e* the edge left of *v*, and *u* the helper of *e* (wrt *v*). Then *uv* does not intersect an edge of the polygon (except in *u* and *v*).

Proof

- the quadrilateral between *uv* and *e* contains no vertex

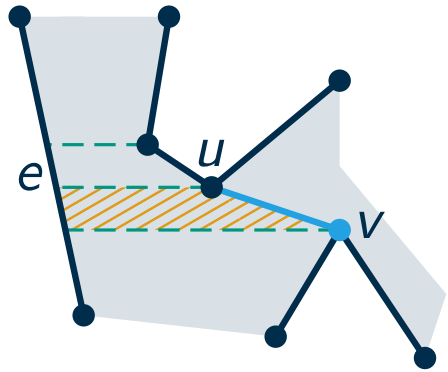
Why?

Eliminating Split Vertices



Idea For Split Vertex *v*

- idea: connect *v* to vertex *u* that is above *v* and close to *v*
- *e*: edge to the left of *v* ("to the left of": the next edge you hit when shooting a ray from *v* to the left)
- choose *u* to be the upper vertex of *e*



Issue (And How To Fix It)

- *uv* might intersect another edge of the polygon
- fix: choose for *u* the lowest vertex above *v* such that *e* is to the left of *u*
- we call *u* the **helper** of *e* (note: it depends on *v*)

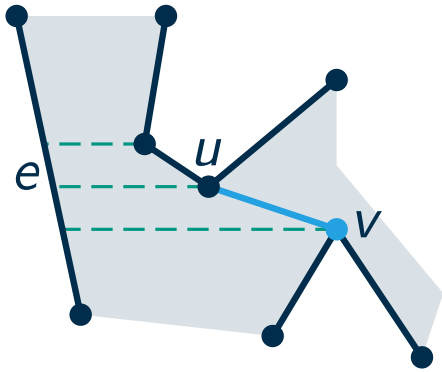
Lemma (the helper is helpful)
Let *v* be a split vertex, *e* the edge left of *v*, and *u* the helper of *e* (wrt *v*). Then *uv* does not intersect an edge of the polygon (except in *u* and *v*).

Proof

- the quadrilateral between *uv* and *e* contains no vertex
- no edge intersects *uv*

Why?

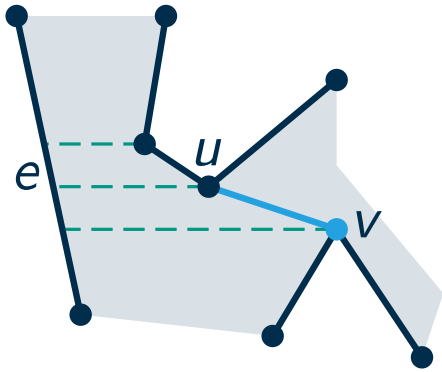
Eliminating Split Vertices



Observations

- goal for split vertex v: find edge e to the left of v and helper of e

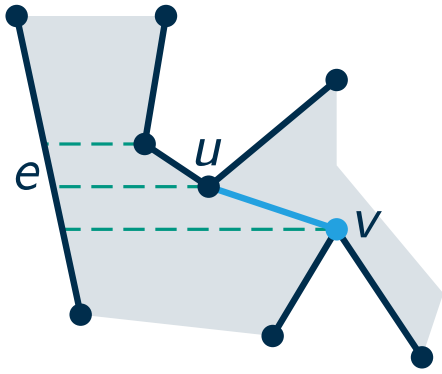
Eliminating Split Vertices



Observations

- goal for split vertex v: find edge e to the left of v and helper of e
- e lies (partially) above v
- the helper of e lies above v

Eliminating Split Vertices

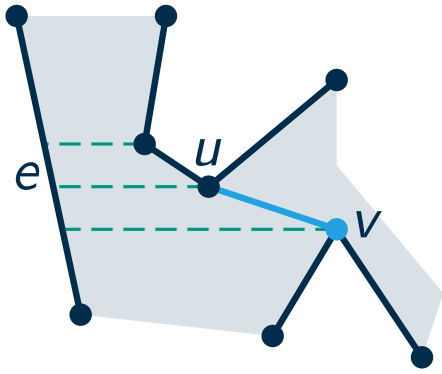


Observations

- goal for split vertex v: find edge e to the left of v and helper of e
- e lies (partially) above v
- the helper of e lies above v

$\Rightarrow$ sweep line seems to be a good idea
(horizontal sweep line ℓ from top to bottom)

Eliminating Split Vertices



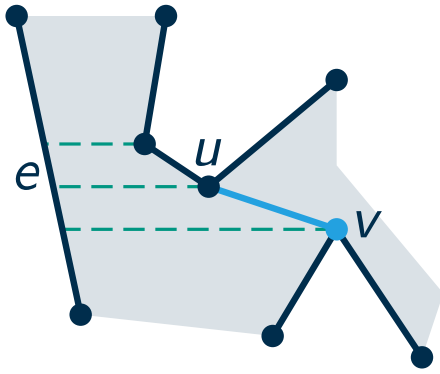
Observations

- goal for split vertex v: find edge e to the left of v and helper of e
- e lies (partially) above v
- the helper of e lies above v

$\Rightarrow$ sweep line seems to be a good idea
(horizontal sweep line ℓ from top to bottom)

Event Queue

Eliminating Split Vertices



Observations

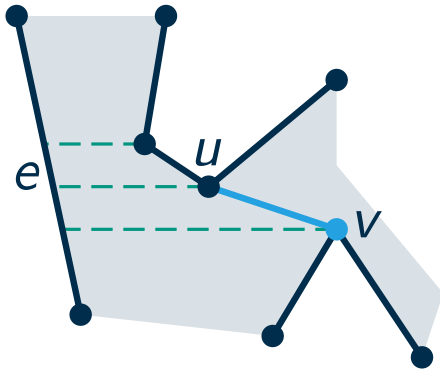
- goal for split vertex v: find edge e to the left of v and helper of e
- e lies (partially) above v
- the helper of e lies above v

$\Rightarrow$ sweep line seems to be a good idea
(horizontal sweep line ℓ from top to bottom)

Event Queue

- vertices of the polygon
- sorted by y-coordinate (or lexicographic by yx)

Eliminating Split Vertices



Observations

- goal for split vertex v: find edge e to the left of v and helper of e
- e lies (partially) above v
- the helper of e lies above v

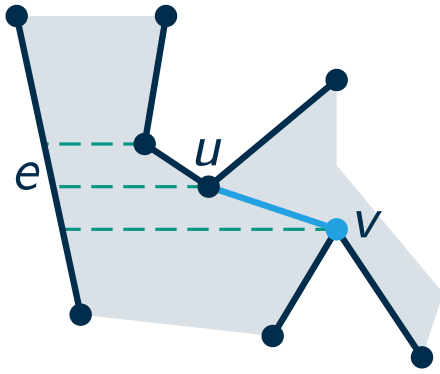
$\Rightarrow$ sweep line seems to be a good idea
(horizontal sweep line ℓ from top to bottom)

Event Queue

- vertices of the polygon
- sorted by y-coordinate (or lexicographic by yx)

Sweep Line Status

Eliminating Split Vertices



Observations

- goal for split vertex v: find edge e to the left of v and helper of e
- e lies (partially) above v
- the helper of e lies above v

$\Rightarrow$ sweep line seems to be a good idea
(horizontal sweep line ℓ from top to bottom)

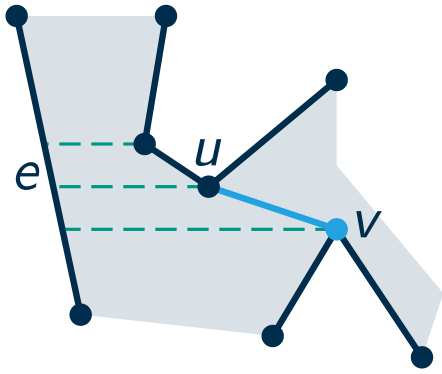
Event Queue

- vertices of the polygon
- sorted by y-coordinate (or lexicographic by yx)

Sweep Line Status

- edges that intersect ℓ sorted by x-coordinate

Eliminating Split Vertices



Observations

- goal for split vertex v: find edge e to the left of v and helper of e
- e lies (partially) above v
- the helper of e lies above v

$\Rightarrow$ sweep line seems to be a good idea
(horizontal sweep line ℓ from top to bottom)

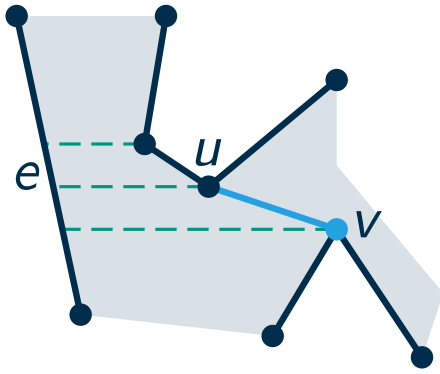
Event Queue

- vertices of the polygon
- sorted by y-coordinate (or lexicographic by yx)

Sweep Line Status

- edges that intersect ℓ sorted by x-coordinate
- edges that have the polygon to their right suffice

Eliminating Split Vertices



Observations

- goal for split vertex v: find edge e to the left of v and helper of e
- e lies (partially) above v
- the helper of e lies above v

$\Rightarrow$ sweep line seems to be a good idea
(horizontal sweep line ℓ from top to bottom)

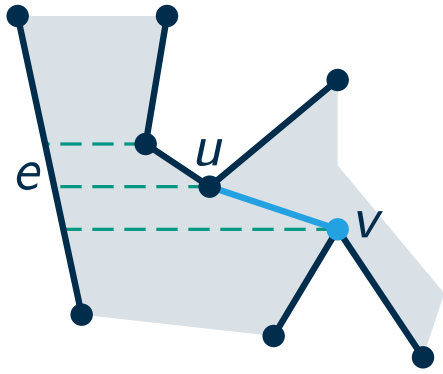
Event Queue

- vertices of the polygon
- sorted by y-coordinate (or lexicographic by yx)

Sweep Line Status

- edges that intersect ℓ sorted by x-coordinate
- edges that have the polygon to their right suffice
- current helper for every edge

Eliminating Split Vertices



Observations

- goal for split vertex v: find edge e to the left of v and helper of e
- e lies (partially) above v
- the helper of e lies above v

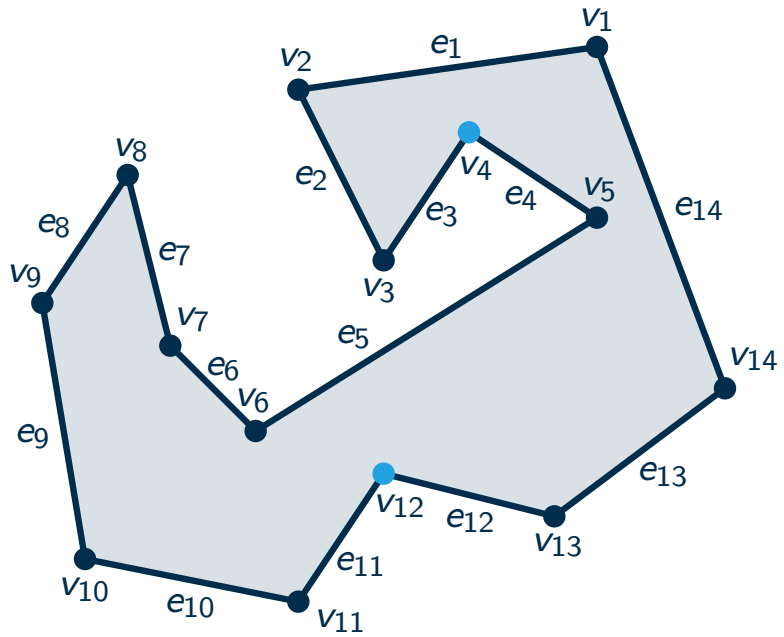
$\Rightarrow$ sweep line seems to be a good idea
(horizontal sweep line ℓ from top to bottom)

Event Queue

- vertices of the polygon
- sorted by y-coordinate (or lexicographic by yx)

Sweep Line Status

- edges that intersect ℓ sorted by x-coordinate
- edges that have the polygon to their right suffice
- current helper for every edge



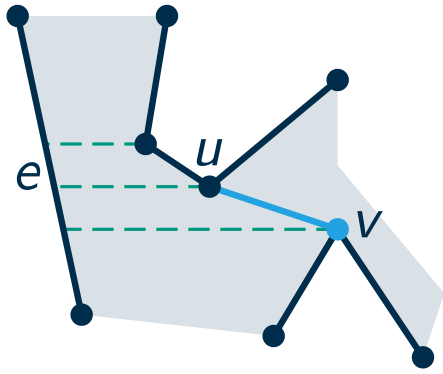
Queue

v_1 v_2 v_4 v_8 v_5 v_3 v_9 v_7 v_{14} v_6 v_{12} v_{13} v_{10} v_{11}

Status

Helper

Eliminating Split Vertices



Observations

- goal for split vertex v: find edge e to the left of v and helper of e
- e lies (partially) above v
- the helper of e lies above v

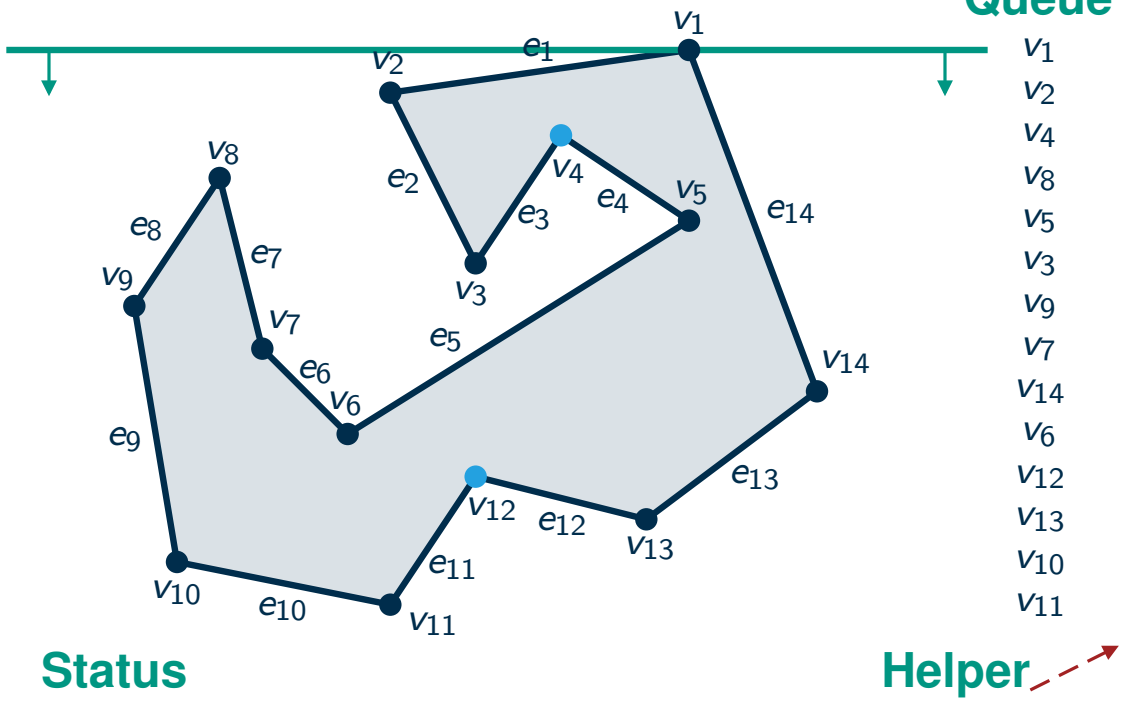
$\Rightarrow$ sweep line seems to be a good idea
(horizontal sweep line ℓ from top to bottom)

Event Queue

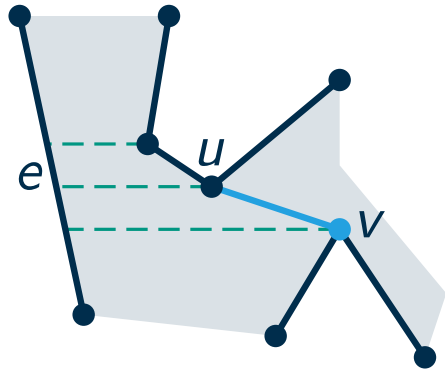
- vertices of the polygon
- sorted by y-coordinate (or lexicographic by yx)

Sweep Line Status

- edges that intersect ℓ sorted by x-coordinate
- edges that have the polygon to their right suffice
- current helper for every edge



Eliminating Split Vertices



Observations

- goal for split vertex v: find edge e to the left of v and helper of e
- e lies (partially) above v
- the helper of e lies above v

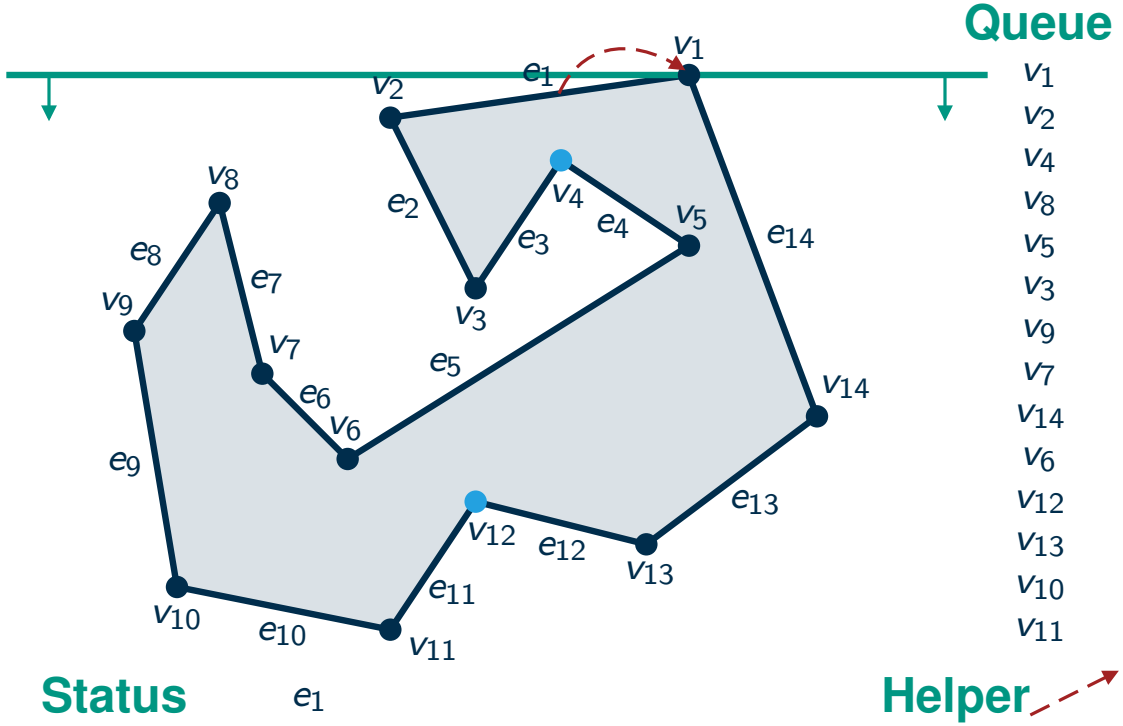
$\Rightarrow$ sweep line seems to be a good idea
(horizontal sweep line ℓ from top to bottom)

Event Queue

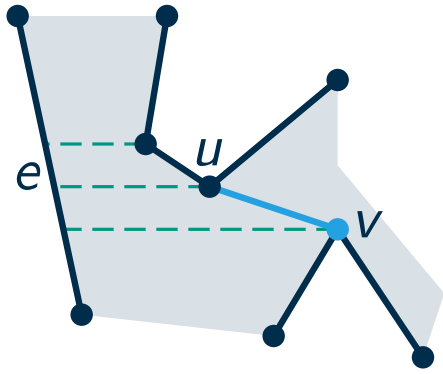
- vertices of the polygon
- sorted by y-coordinate (or lexicographic by yx)

Sweep Line Status

- edges that intersect ℓ sorted by x-coordinate
- edges that have the polygon to their right suffice
- current helper for every edge



Eliminating Split Vertices



Observations

- goal for split vertex v: find edge e to the left of v and helper of e
- e lies (partially) above v
- the helper of e lies above v

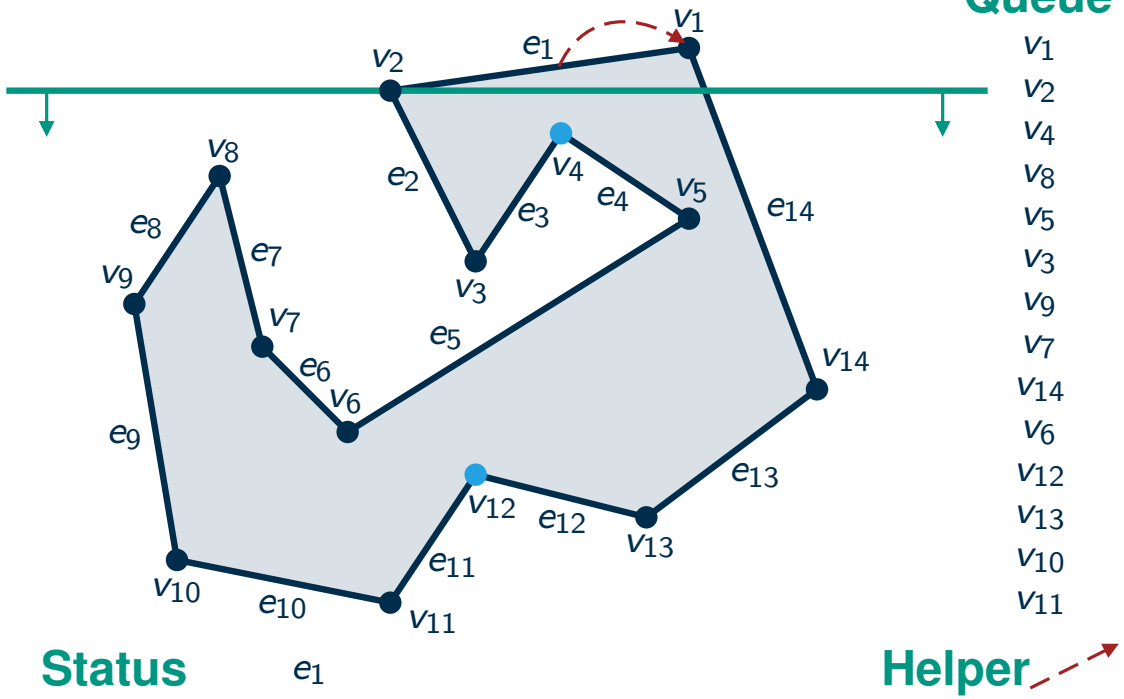
$\Rightarrow$ sweep line seems to be a good idea
(horizontal sweep line ℓ from top to bottom)

Event Queue

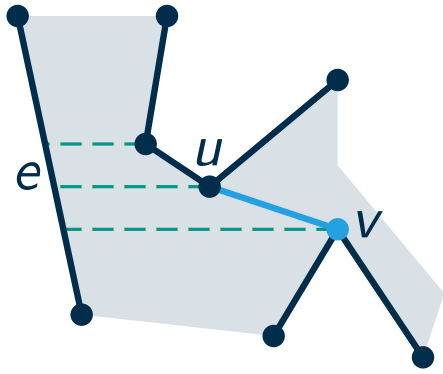
- vertices of the polygon
- sorted by y-coordinate (or lexicographic by yx)

Sweep Line Status

- edges that intersect ℓ sorted by x-coordinate
- edges that have the polygon to their right suffice
- current helper for every edge



Eliminating Split Vertices



Observations

- goal for split vertex v: find edge e to the left of v and helper of e
- e lies (partially) above v
- the helper of e lies above v

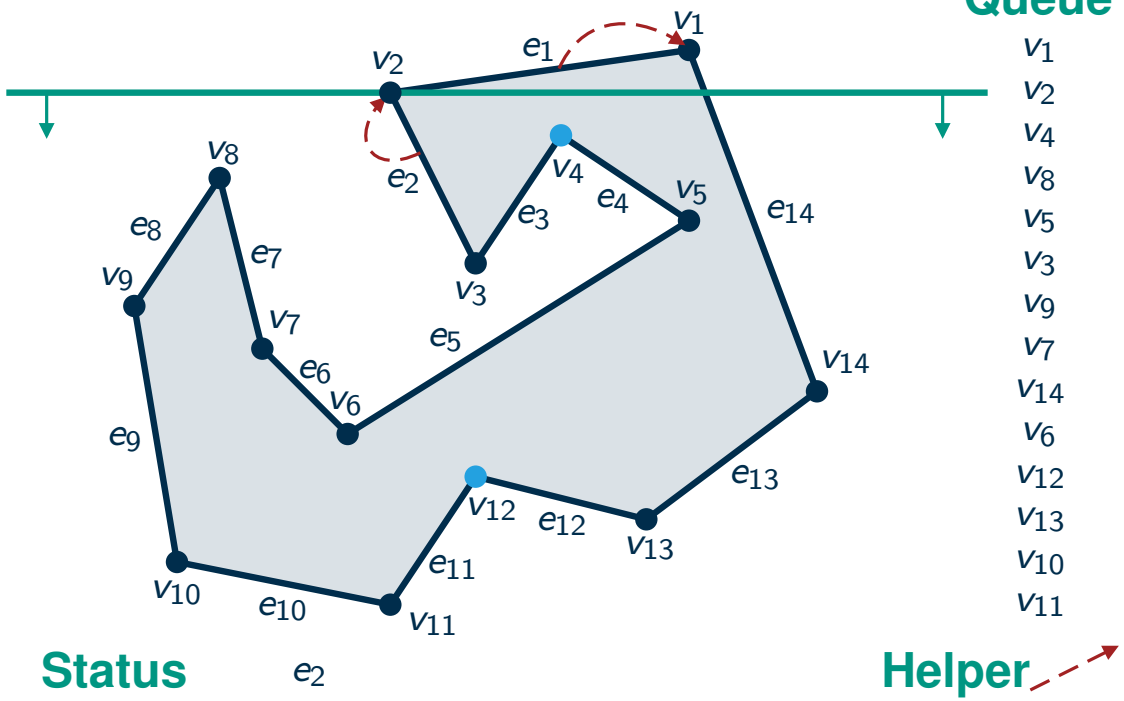
$\Rightarrow$ sweep line seems to be a good idea
(horizontal sweep line ℓ from top to bottom)

Event Queue

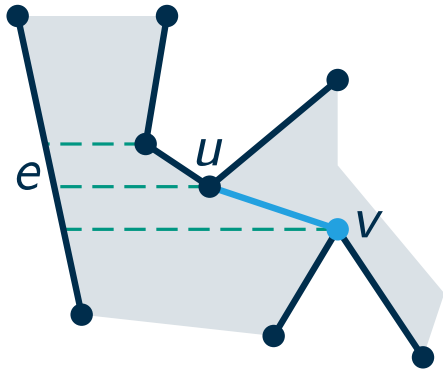
- vertices of the polygon
- sorted by y-coordinate (or lexicographic by yx)

Sweep Line Status

- edges that intersect ℓ sorted by x-coordinate
- edges that have the polygon to their right suffice
- current helper for every edge



Eliminating Split Vertices



Observations

- goal for split vertex v: find edge e to the left of v and helper of e
- e lies (partially) above v
- the helper of e lies above v

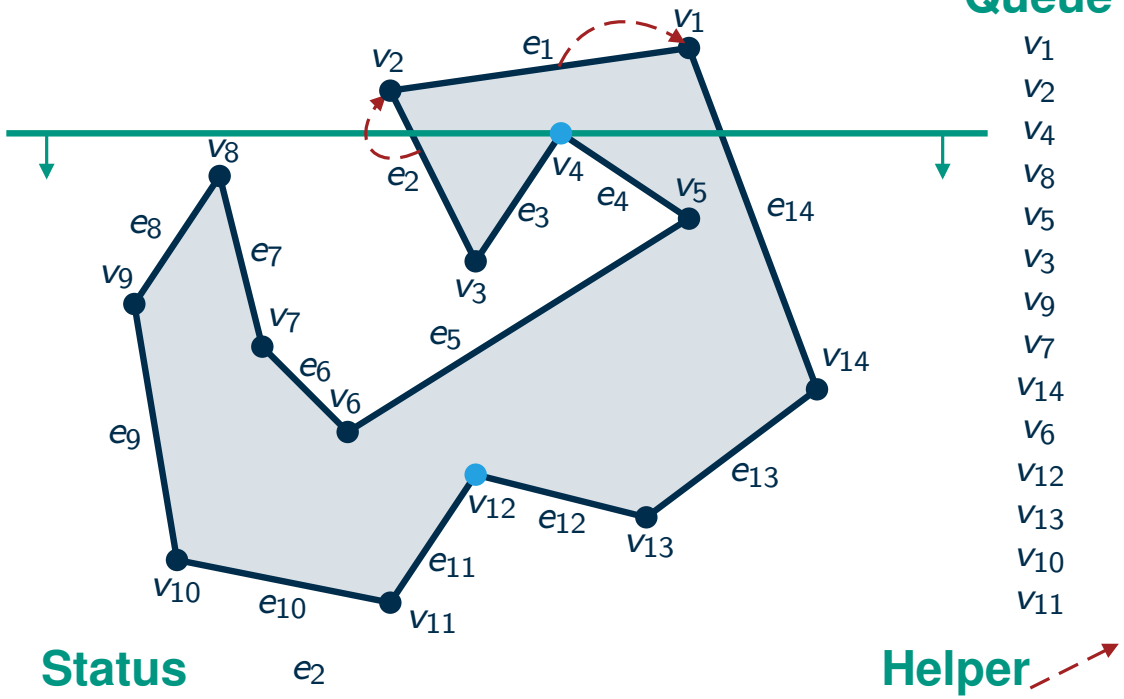
$\Rightarrow$ sweep line seems to be a good idea
(horizontal sweep line ℓ from top to bottom)

Event Queue

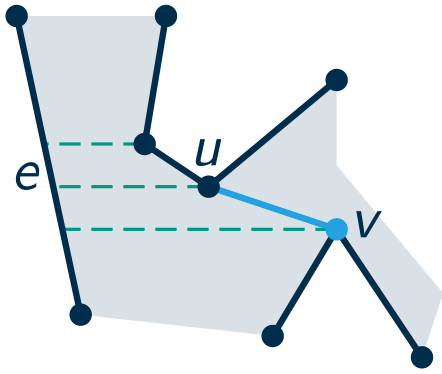
- vertices of the polygon
- sorted by y-coordinate (or lexicographic by yx)

Sweep Line Status

- edges that intersect ℓ sorted by x-coordinate
- edges that have the polygon to their right suffice
- current helper for every edge



Eliminating Split Vertices



Observations

- goal for split vertex v: find edge e to the left of v and helper of e
- e lies (partially) above v
- the helper of e lies above v

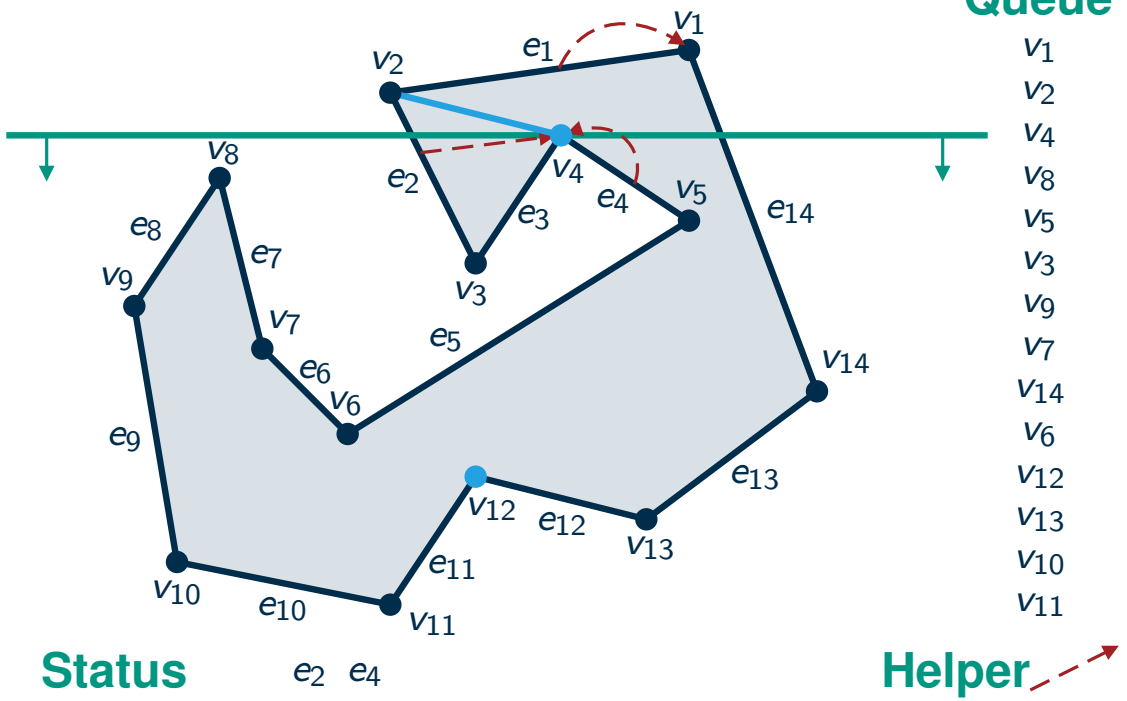
$\Rightarrow$ sweep line seems to be a good idea
(horizontal sweep line ℓ from top to bottom)

Event Queue

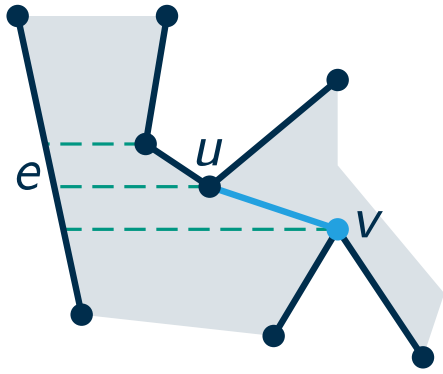
- vertices of the polygon
- sorted by y-coordinate (or lexicographic by yx)

Sweep Line Status

- edges that intersect ℓ sorted by x-coordinate
- edges that have the polygon to their right suffice
- current helper for every edge



Eliminating Split Vertices



Observations

- goal for split vertex v: find edge e to the left of v and helper of e
- e lies (partially) above v
- the helper of e lies above v

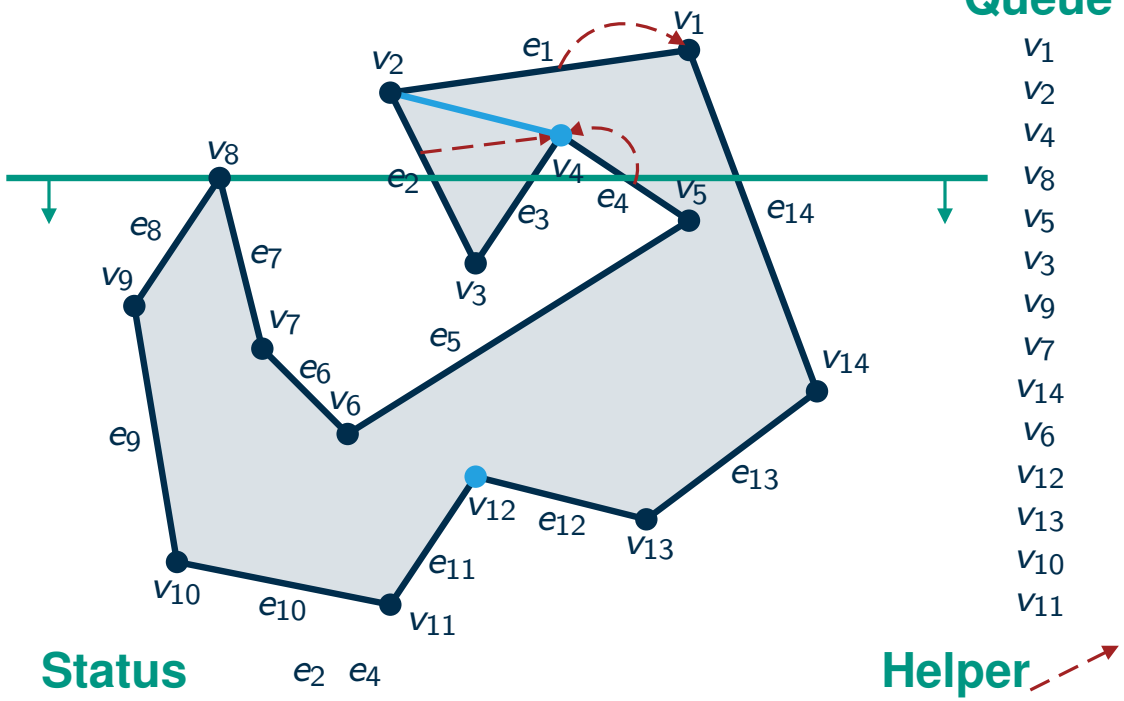
$\Rightarrow$ sweep line seems to be a good idea
(horizontal sweep line ℓ from top to bottom)

Event Queue

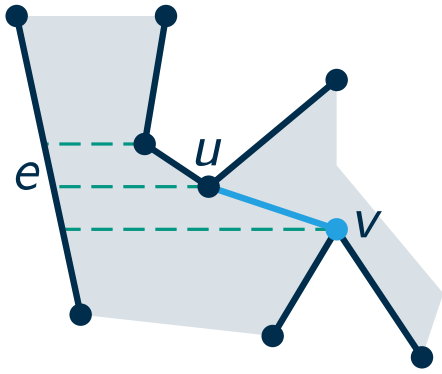
- vertices of the polygon
- sorted by y-coordinate (or lexicographic by yx)

Sweep Line Status

- edges that intersect ℓ sorted by x-coordinate
- edges that have the polygon to their right suffice
- current helper for every edge



Eliminating Split Vertices



Observations

- goal for split vertex v: find edge e to the left of v and helper of e
- e lies (partially) above v
- the helper of e lies above v

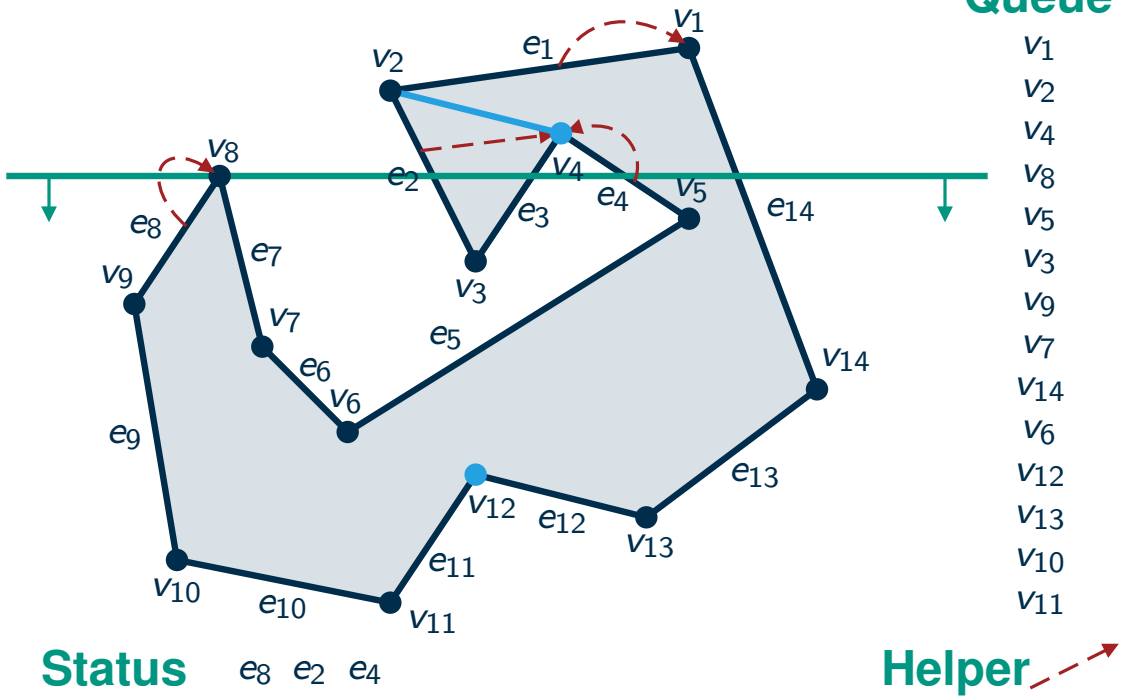
$\Rightarrow$ sweep line seems to be a good idea
(horizontal sweep line ℓ from top to bottom)

Event Queue

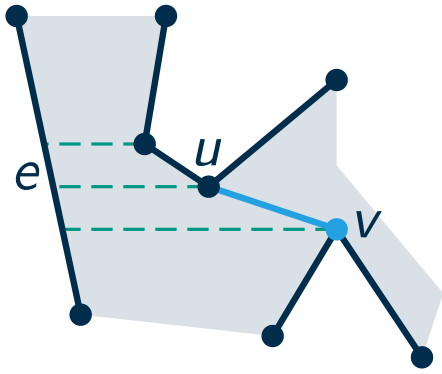
- vertices of the polygon
- sorted by y-coordinate (or lexicographic by yx)

Sweep Line Status

- edges that intersect ℓ sorted by x-coordinate
- edges that have the polygon to their right suffice
- current helper for every edge



Eliminating Split Vertices



Observations

- goal for split vertex v: find edge e to the left of v and helper of e
- e lies (partially) above v
- the helper of e lies above v

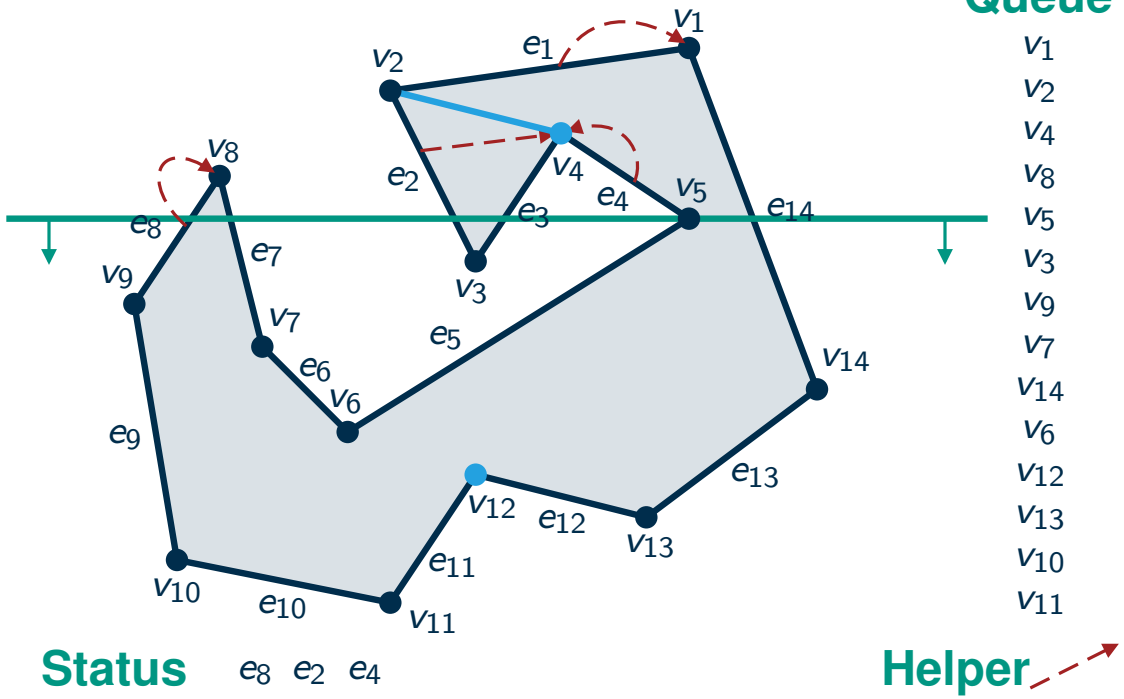
$\Rightarrow$ sweep line seems to be a good idea
(horizontal sweep line ℓ from top to bottom)

Event Queue

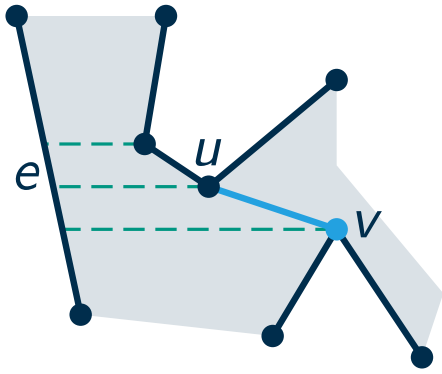
- vertices of the polygon
- sorted by y-coordinate (or lexicographic by yx)

Sweep Line Status

- edges that intersect ℓ sorted by x-coordinate
- edges that have the polygon to their right suffice
- current helper for every edge



Eliminating Split Vertices



Observations

- goal for split vertex v: find edge e to the left of v and helper of e
- e lies (partially) above v
- the helper of e lies above v

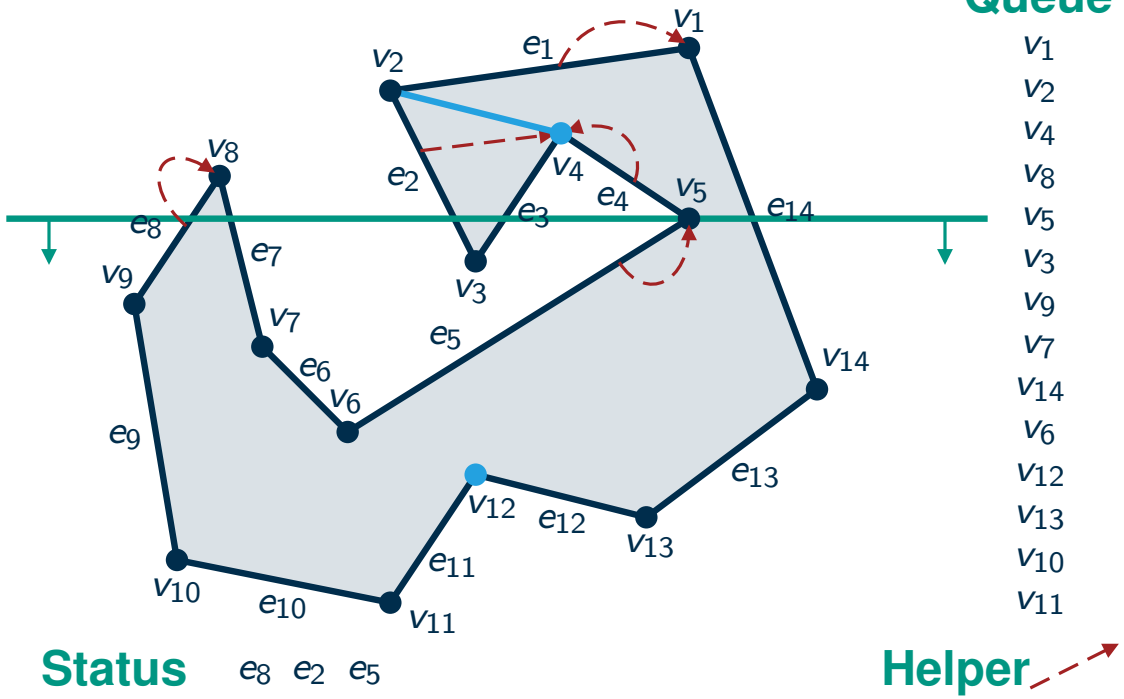
$\Rightarrow$ sweep line seems to be a good idea
(horizontal sweep line ℓ from top to bottom)

Event Queue

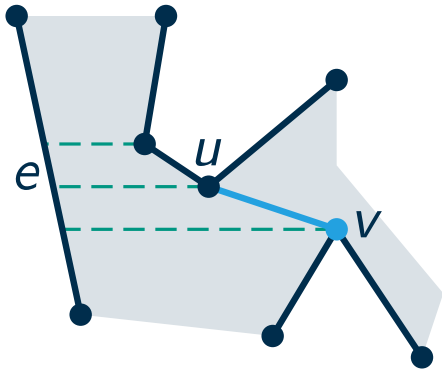
- vertices of the polygon
- sorted by y-coordinate (or lexicographic by yx)

Sweep Line Status

- edges that intersect ℓ sorted by x-coordinate
- edges that have the polygon to their right suffice
- current helper for every edge



Eliminating Split Vertices



Observations

- goal for split vertex v: find edge e to the left of v and helper of e
- e lies (partially) above v
- the helper of e lies above v

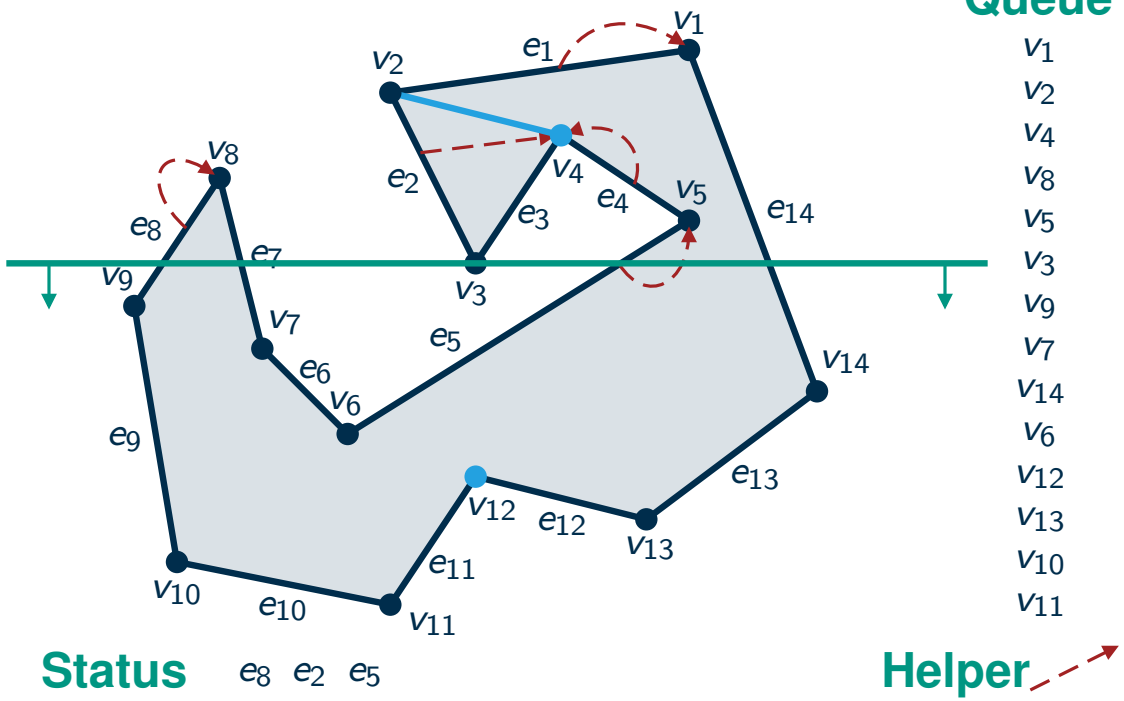
$\Rightarrow$ sweep line seems to be a good idea
(horizontal sweep line ℓ from top to bottom)

Event Queue

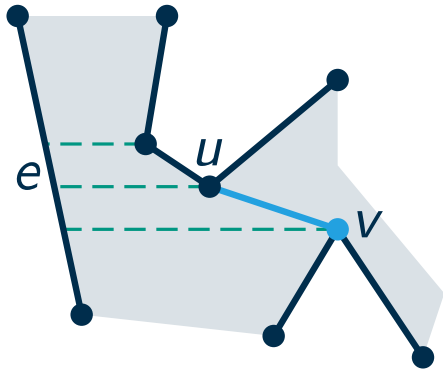
- vertices of the polygon
- sorted by y-coordinate (or lexicographic by yx)

Sweep Line Status

- edges that intersect ℓ sorted by x-coordinate
- edges that have the polygon to their right suffice
- current helper for every edge



Eliminating Split Vertices



Observations

- goal for split vertex v: find edge e to the left of v and helper of e
- e lies (partially) above v
- the helper of e lies above v

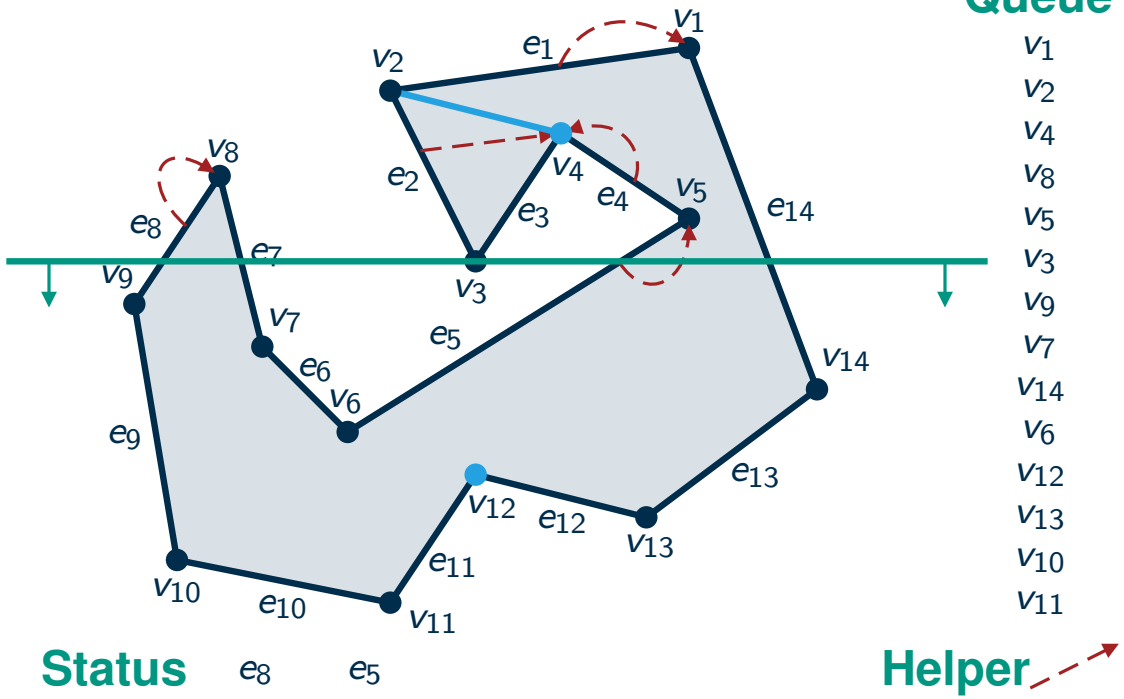
$\Rightarrow$ sweep line seems to be a good idea
(horizontal sweep line ℓ from top to bottom)

Event Queue

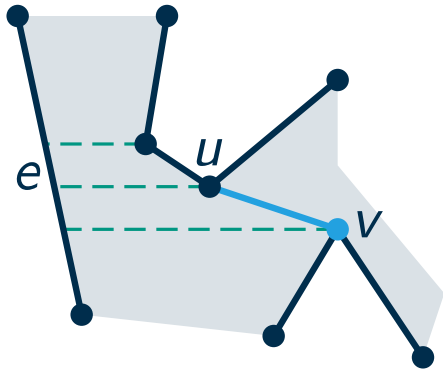
- vertices of the polygon
- sorted by y-coordinate (or lexicographic by yx)

Sweep Line Status

- edges that intersect ℓ sorted by x-coordinate
- edges that have the polygon to their right suffice
- current helper for every edge



Eliminating Split Vertices



Observations

- goal for split vertex v: find edge e to the left of v and helper of e
- e lies (partially) above v
- the helper of e lies above v

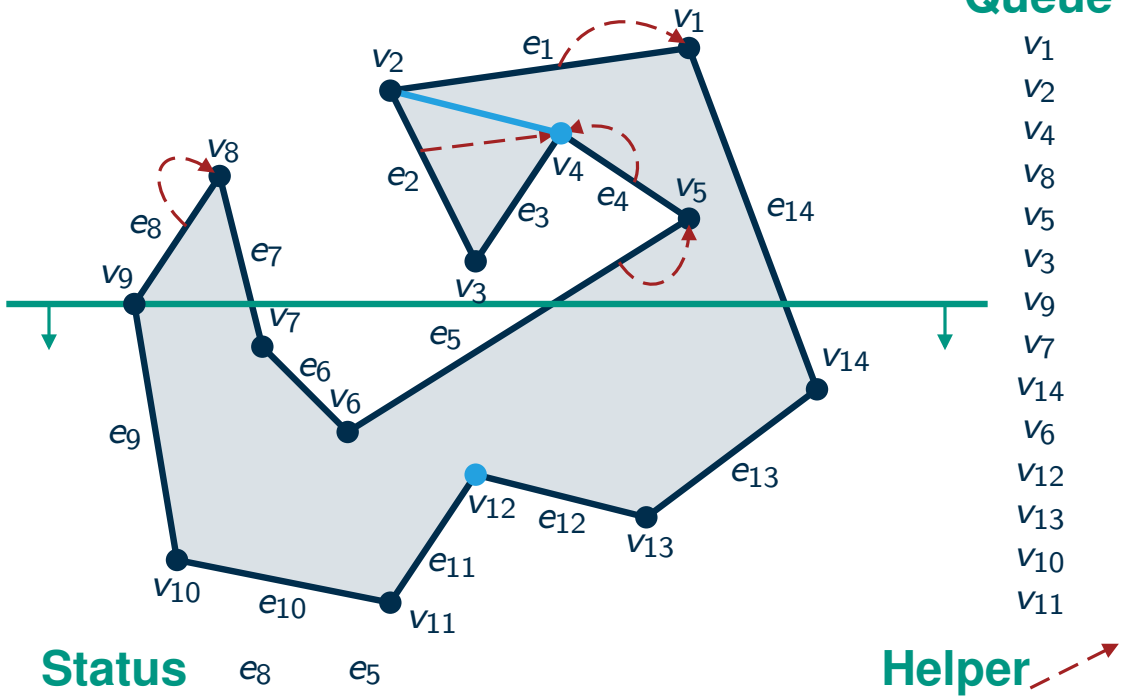
$\Rightarrow$ sweep line seems to be a good idea
(horizontal sweep line ℓ from top to bottom)

Event Queue

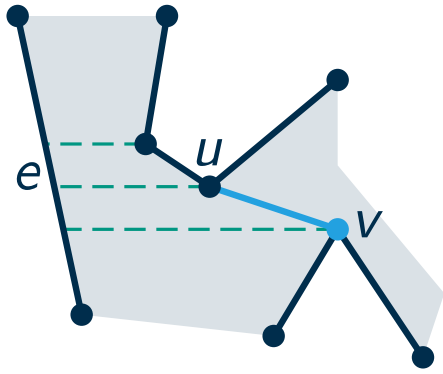
- vertices of the polygon
- sorted by y-coordinate (or lexicographic by yx)

Sweep Line Status

- edges that intersect ℓ sorted by x-coordinate
- edges that have the polygon to their right suffice
- current helper for every edge



Eliminating Split Vertices



Observations

- goal for split vertex v: find edge e to the left of v and helper of e
- e lies (partially) above v
- the helper of e lies above v

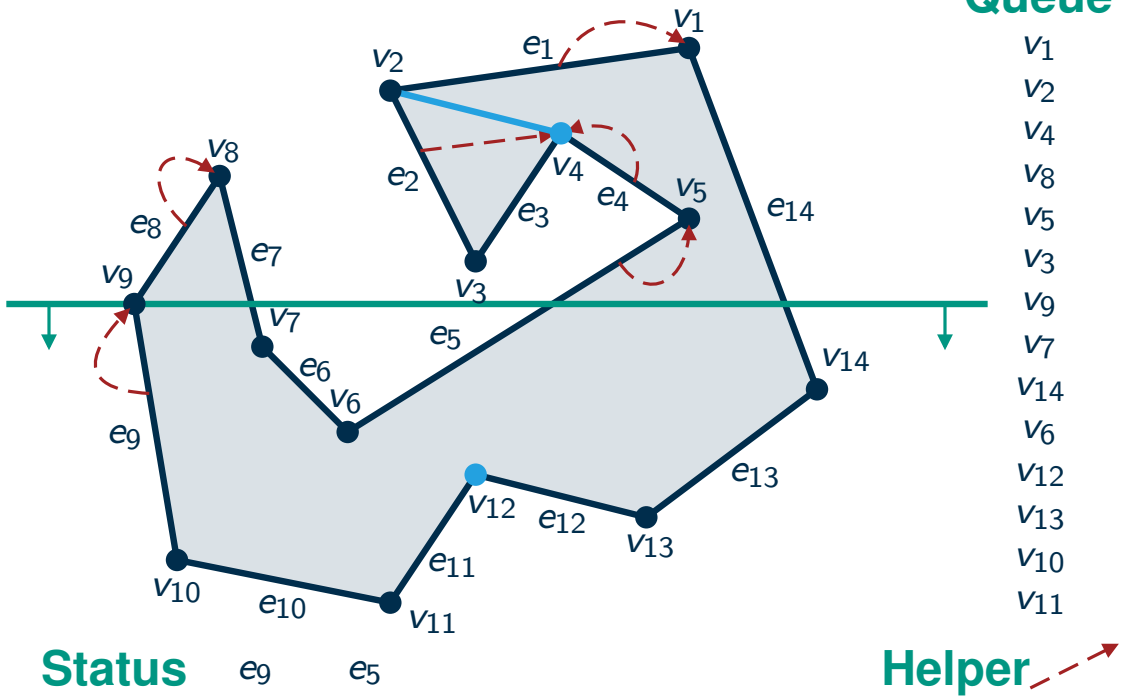
$\Rightarrow$ sweep line seems to be a good idea
(horizontal sweep line ℓ from top to bottom)

Event Queue

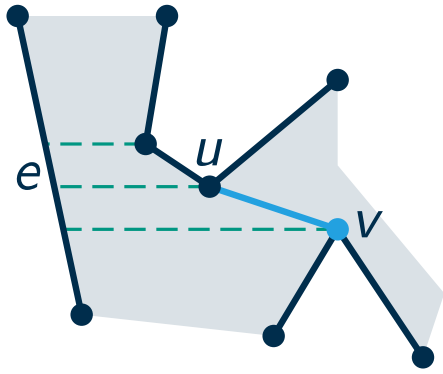
- vertices of the polygon
- sorted by y-coordinate (or lexicographic by yx)

Sweep Line Status

- edges that intersect ℓ sorted by x-coordinate
- edges that have the polygon to their right suffice
- current helper for every edge



Eliminating Split Vertices



Observations

- goal for split vertex v: find edge e to the left of v and helper of e
- e lies (partially) above v
- the helper of e lies above v

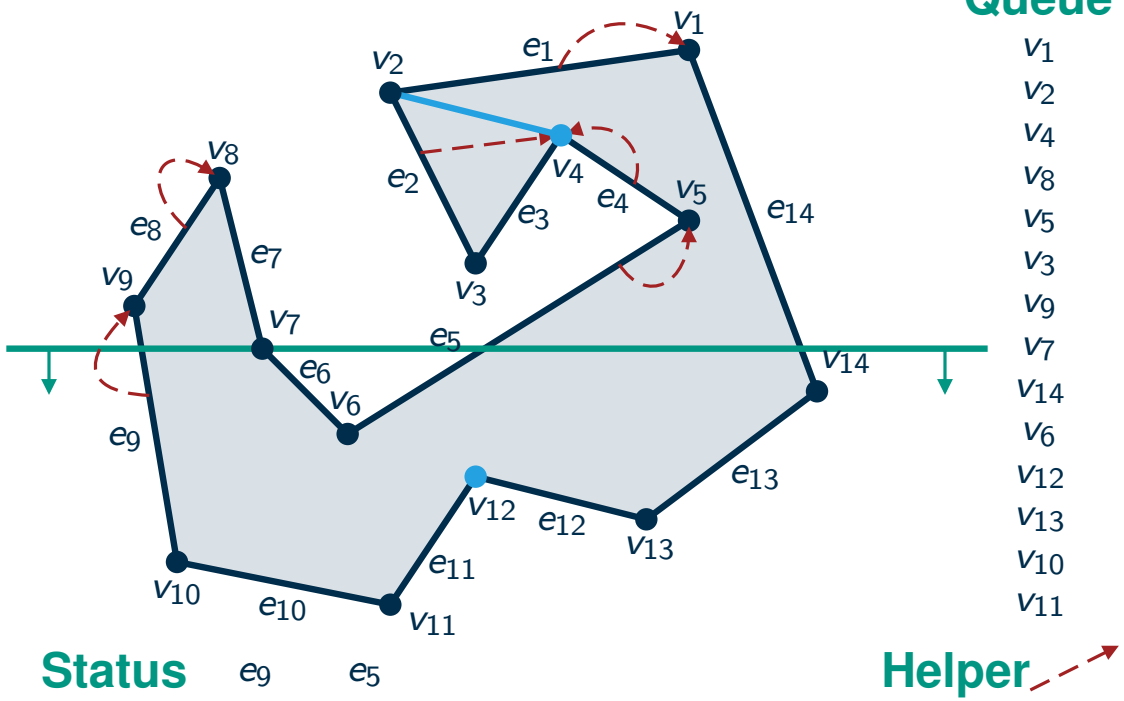
$\Rightarrow$ sweep line seems to be a good idea
(horizontal sweep line ℓ from top to bottom)

Event Queue

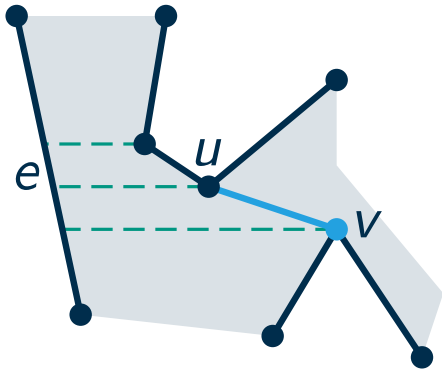
- vertices of the polygon
- sorted by y-coordinate (or lexicographic by yx)

Sweep Line Status

- edges that intersect ℓ sorted by x-coordinate
- edges that have the polygon to their right suffice
- current helper for every edge



Eliminating Split Vertices



Observations

- goal for split vertex v: find edge e to the left of v and helper of e
- e lies (partially) above v
- the helper of e lies above v

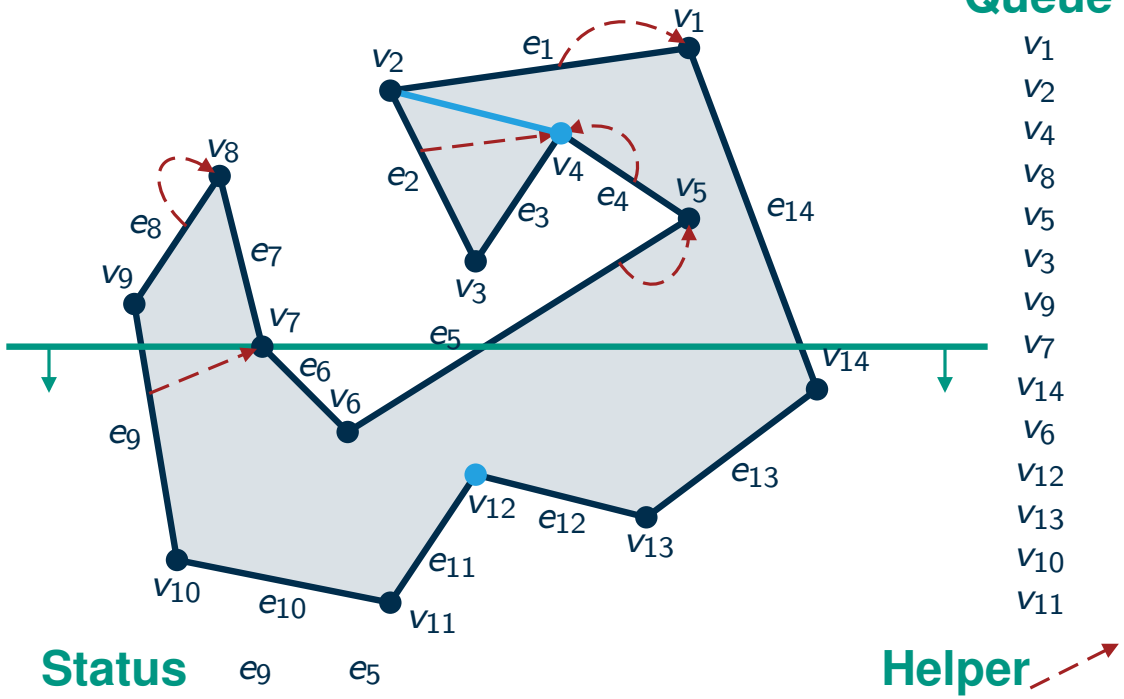
$\Rightarrow$ sweep line seems to be a good idea
(horizontal sweep line ℓ from top to bottom)

Event Queue

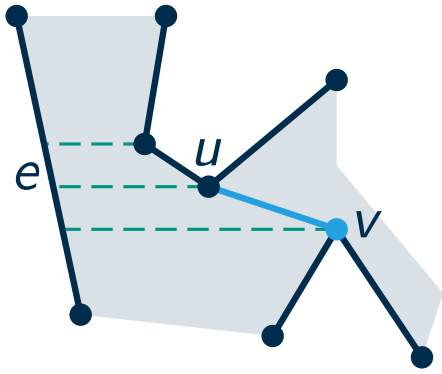
- vertices of the polygon
- sorted by y-coordinate (or lexicographic by yx)

Sweep Line Status

- edges that intersect ℓ sorted by x-coordinate
- edges that have the polygon to their right suffice
- current helper for every edge



Eliminating Split Vertices



Observations

- goal for split vertex v: find edge e to the left of v and helper of e
- e lies (partially) above v
- the helper of e lies above v

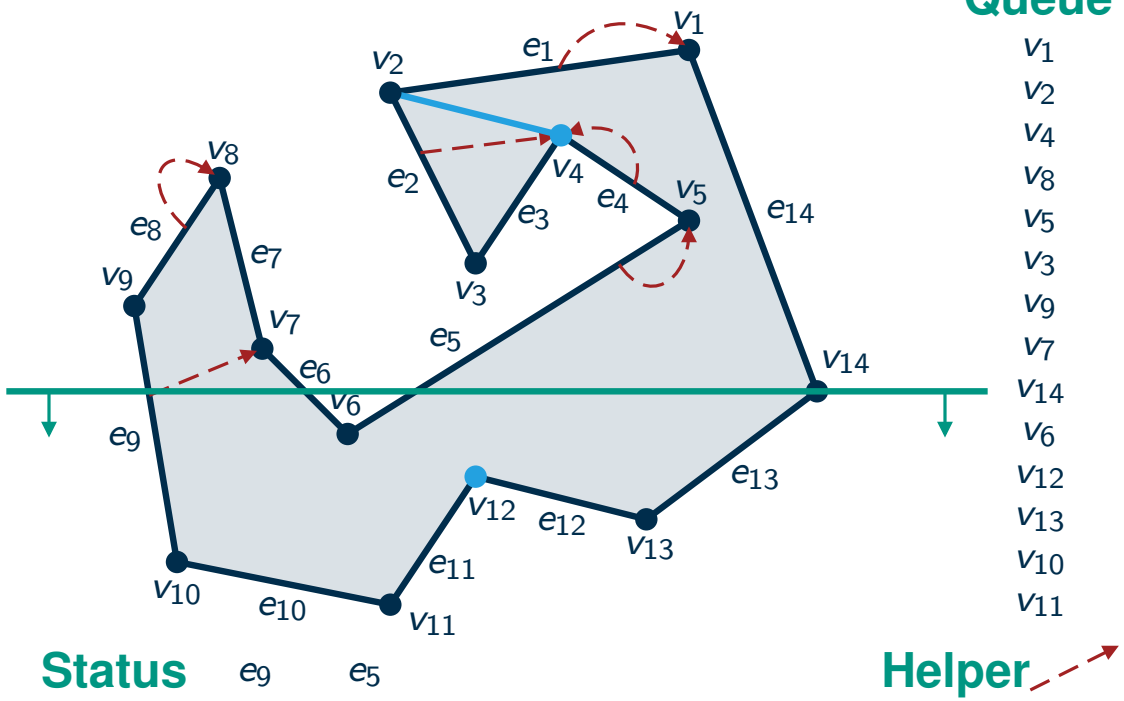
$\Rightarrow$ sweep line seems to be a good idea
(horizontal sweep line ℓ from top to bottom)

Event Queue

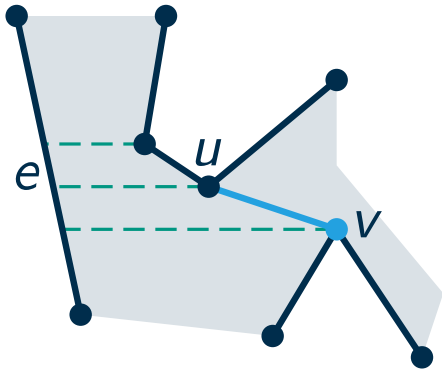
- vertices of the polygon
- sorted by y-coordinate (or lexicographic by yx)

Sweep Line Status

- edges that intersect ℓ sorted by x-coordinate
- edges that have the polygon to their right suffice
- current helper for every edge



Eliminating Split Vertices



Observations

- goal for split vertex v: find edge e to the left of v and helper of e
- e lies (partially) above v
- the helper of e lies above v

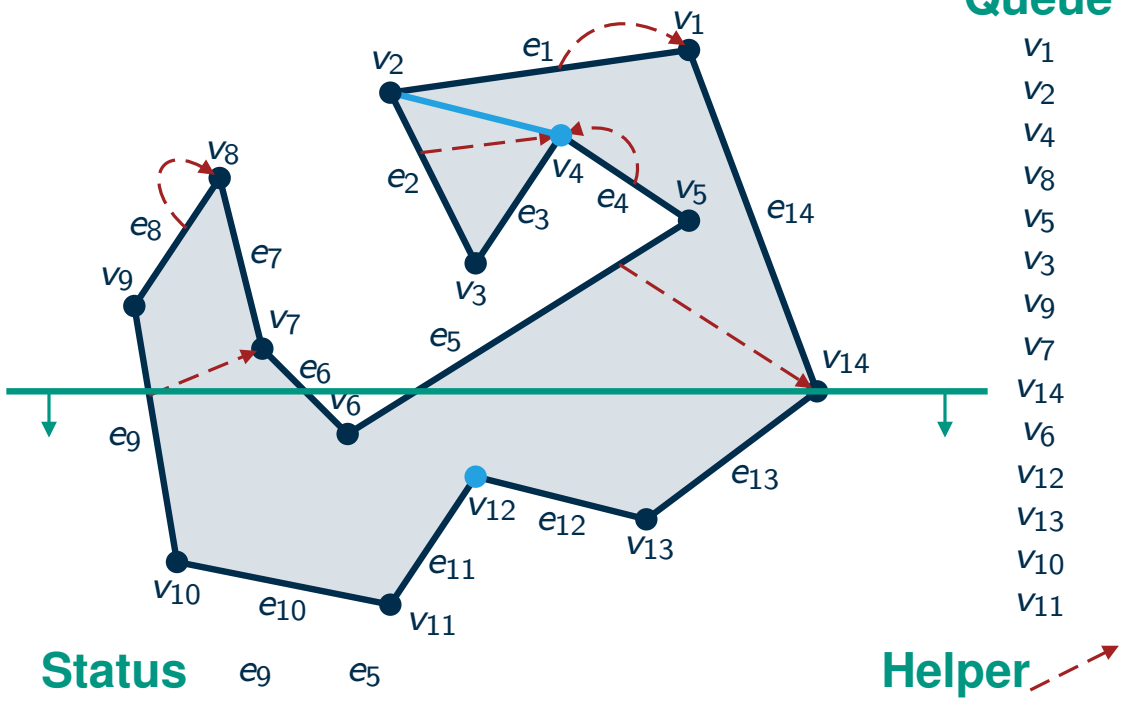
$\Rightarrow$ sweep line seems to be a good idea
(horizontal sweep line ℓ from top to bottom)

Event Queue

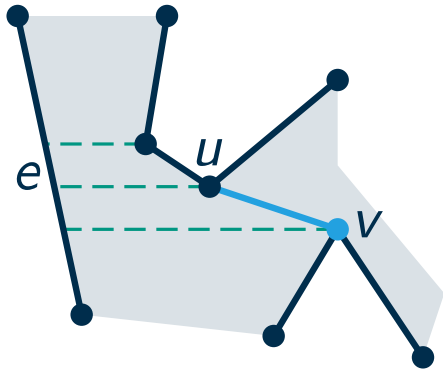
- vertices of the polygon
- sorted by y-coordinate (or lexicographic by yx)

Sweep Line Status

- edges that intersect ℓ sorted by x-coordinate
- edges that have the polygon to their right suffice
- current helper for every edge



Eliminating Split Vertices



Observations

- goal for split vertex v: find edge e to the left of v and helper of e
- e lies (partially) above v
- the helper of e lies above v

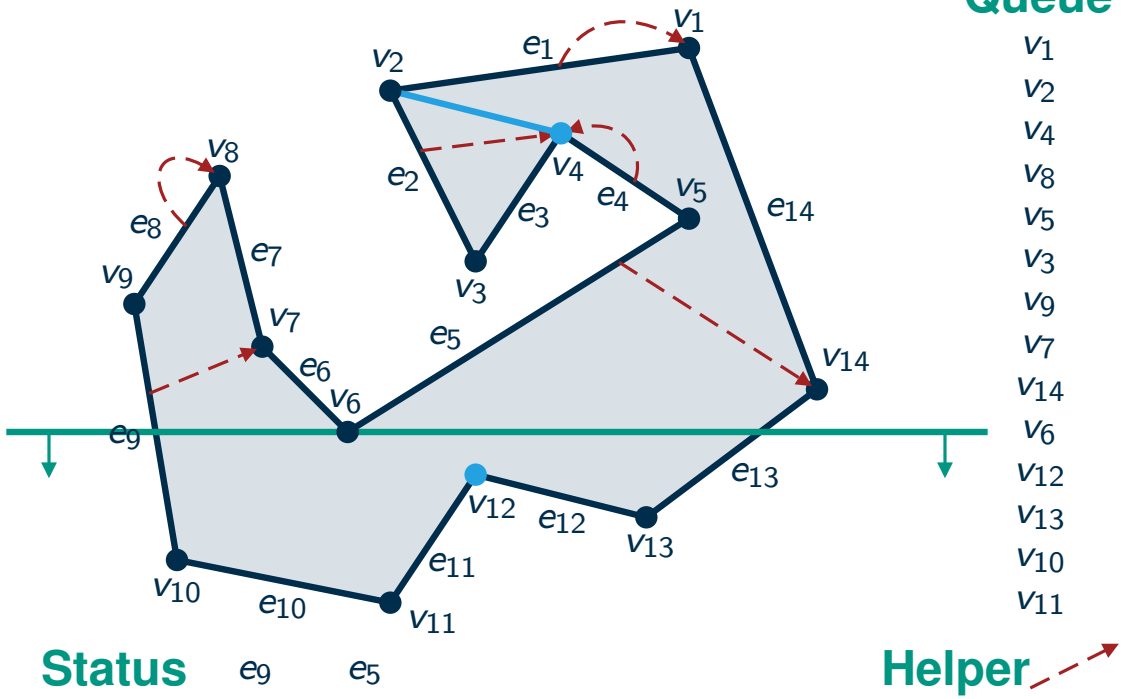
$\Rightarrow$ sweep line seems to be a good idea
(horizontal sweep line ℓ from top to bottom)

Event Queue

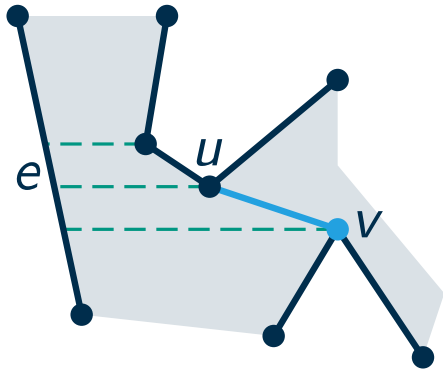
- vertices of the polygon
- sorted by y-coordinate (or lexicographic by yx)

Sweep Line Status

- edges that intersect ℓ sorted by x-coordinate
- edges that have the polygon to their right suffice
- current helper for every edge



Eliminating Split Vertices



Observations

- goal for split vertex v: find edge e to the left of v and helper of e
- e lies (partially) above v
- the helper of e lies above v

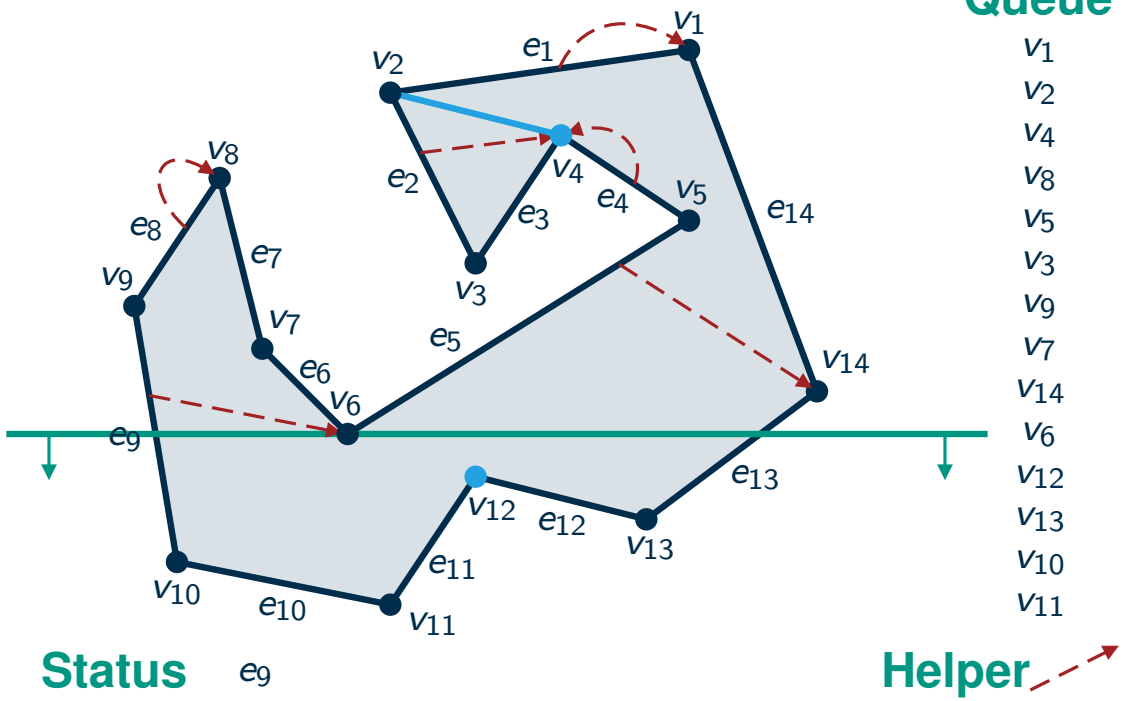
$\Rightarrow$ sweep line seems to be a good idea
(horizontal sweep line ℓ from top to bottom)

Event Queue

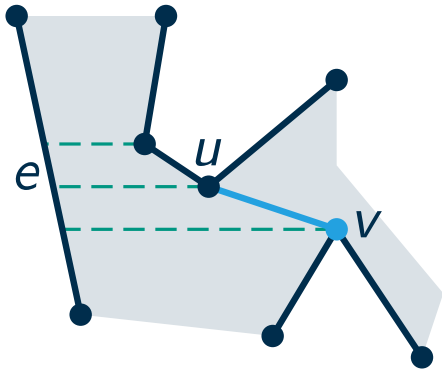
- vertices of the polygon
- sorted by y-coordinate (or lexicographic by yx)

Sweep Line Status

- edges that intersect ℓ sorted by x-coordinate
- edges that have the polygon to their right suffice
- current helper for every edge



Eliminating Split Vertices



Observations

- goal for split vertex v: find edge e to the left of v and helper of e
- e lies (partially) above v
- the helper of e lies above v

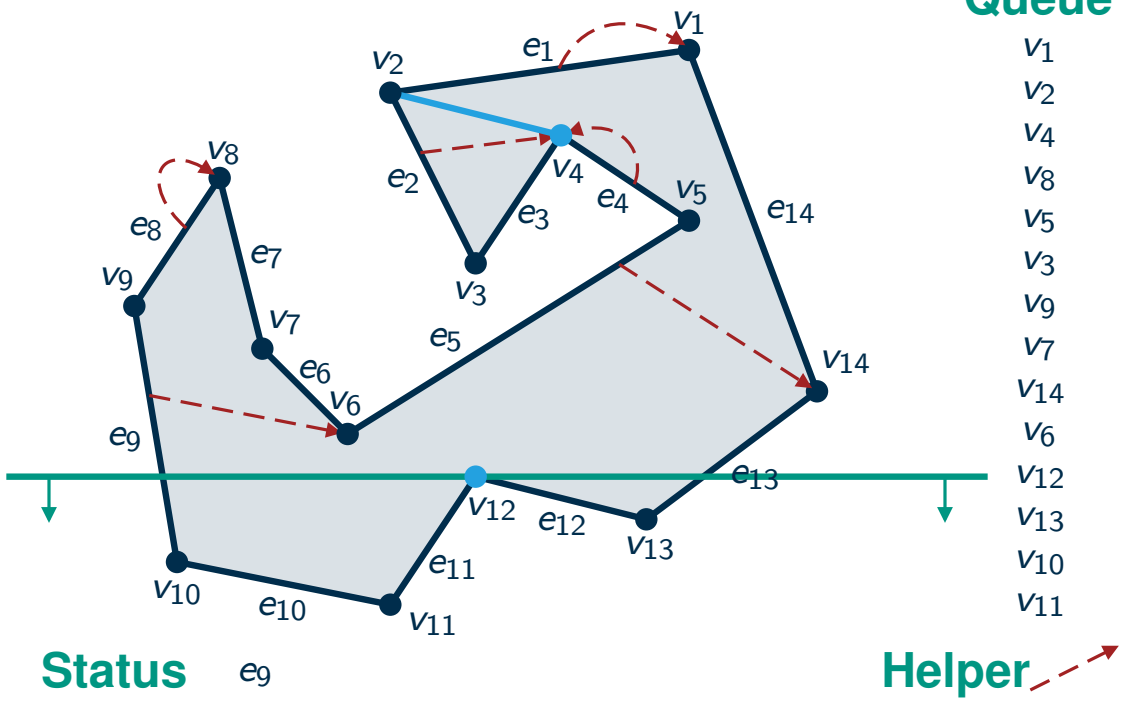
$\Rightarrow$ sweep line seems to be a good idea
(horizontal sweep line ℓ from top to bottom)

Event Queue

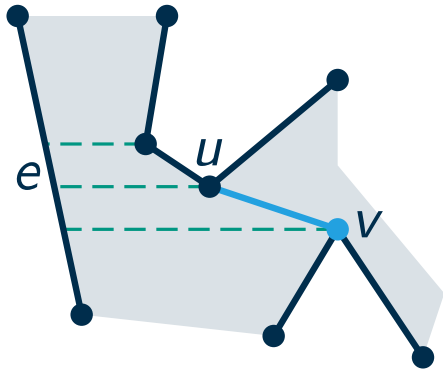
- vertices of the polygon
- sorted by y-coordinate (or lexicographic by yx)

Sweep Line Status

- edges that intersect ℓ sorted by x-coordinate
- edges that have the polygon to their right suffice
- current helper for every edge



Eliminating Split Vertices



Observations

- goal for split vertex v: find edge e to the left of v and helper of e
- e lies (partially) above v
- the helper of e lies above v

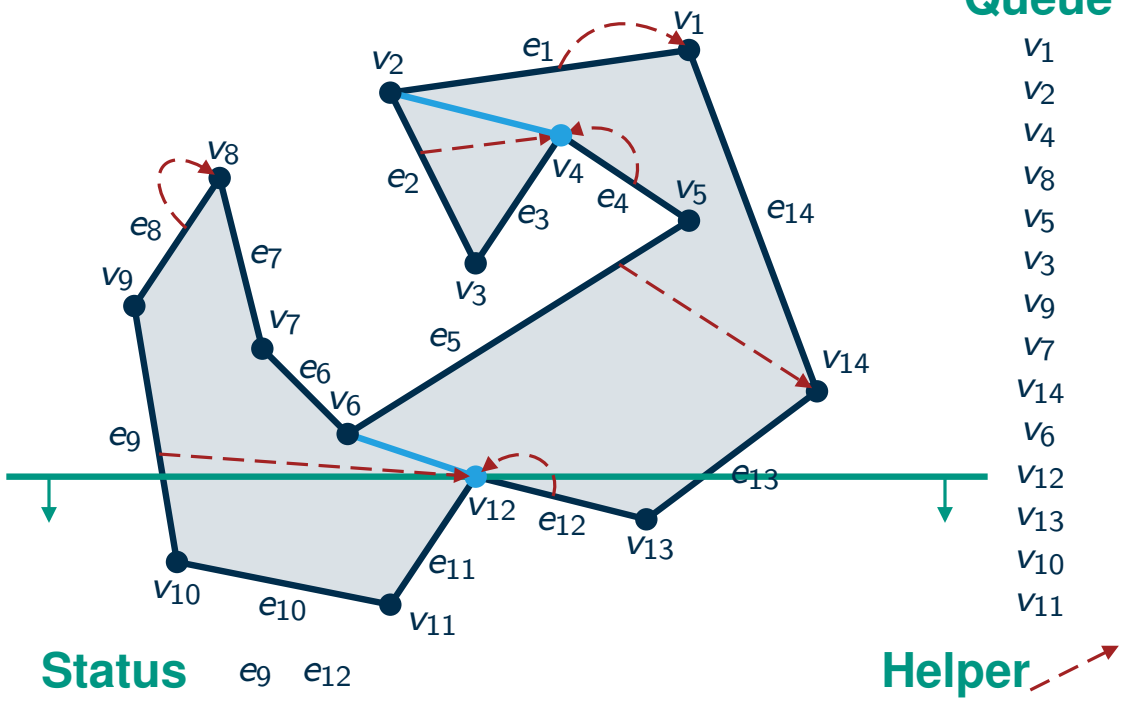
$\Rightarrow$ sweep line seems to be a good idea
(horizontal sweep line ℓ from top to bottom)

Event Queue

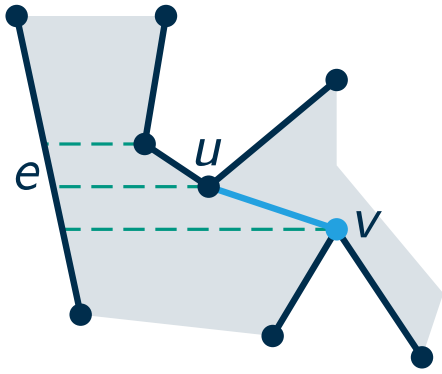
- vertices of the polygon
- sorted by y-coordinate (or lexicographic by yx)

Sweep Line Status

- edges that intersect ℓ sorted by x-coordinate
- edges that have the polygon to their right suffice
- current helper for every edge



Eliminating Split Vertices



Observations

- goal for split vertex v: find edge e to the left of v and helper of e
- e lies (partially) above v
- the helper of e lies above v

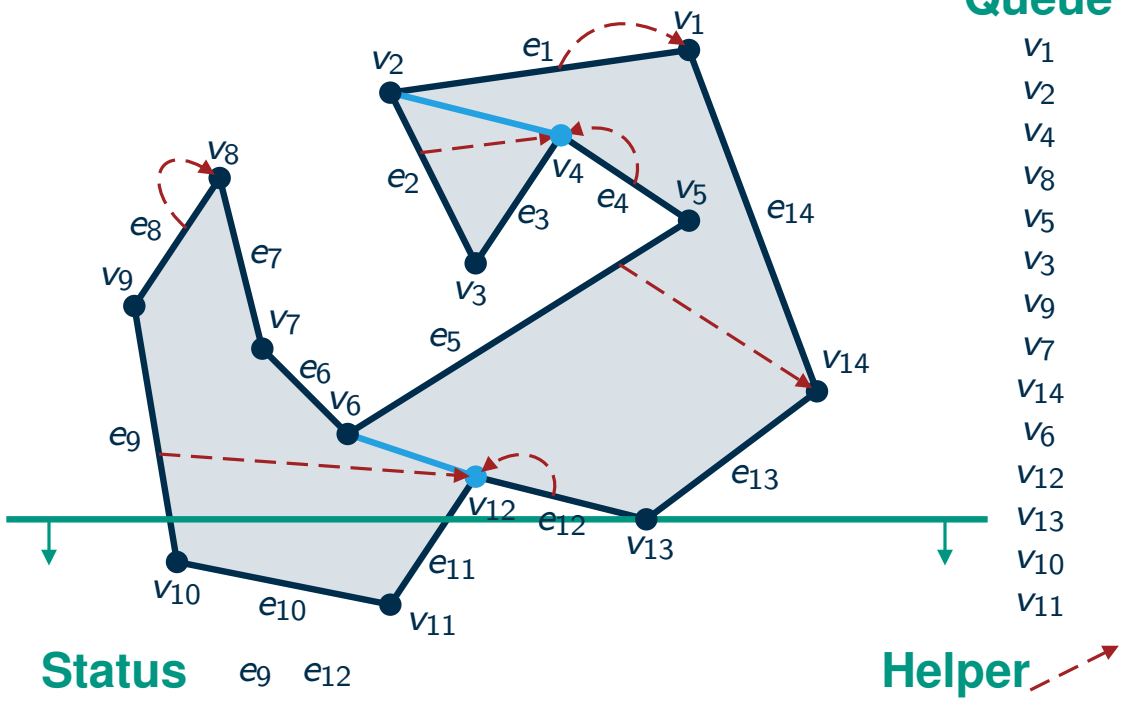
$\Rightarrow$ sweep line seems to be a good idea
(horizontal sweep line ℓ from top to bottom)

Event Queue

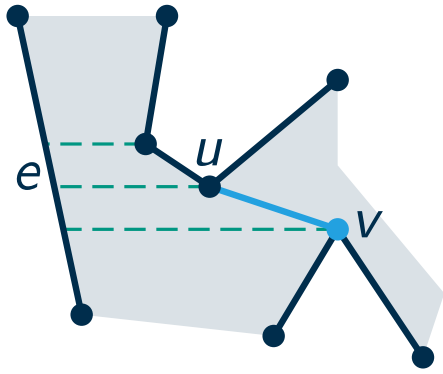
- vertices of the polygon
- sorted by y-coordinate (or lexicographic by yx)

Sweep Line Status

- edges that intersect ℓ sorted by x-coordinate
- edges that have the polygon to their right suffice
- current helper for every edge



Eliminating Split Vertices



Observations

- goal for split vertex v: find edge e to the left of v and helper of e
- e lies (partially) above v
- the helper of e lies above v

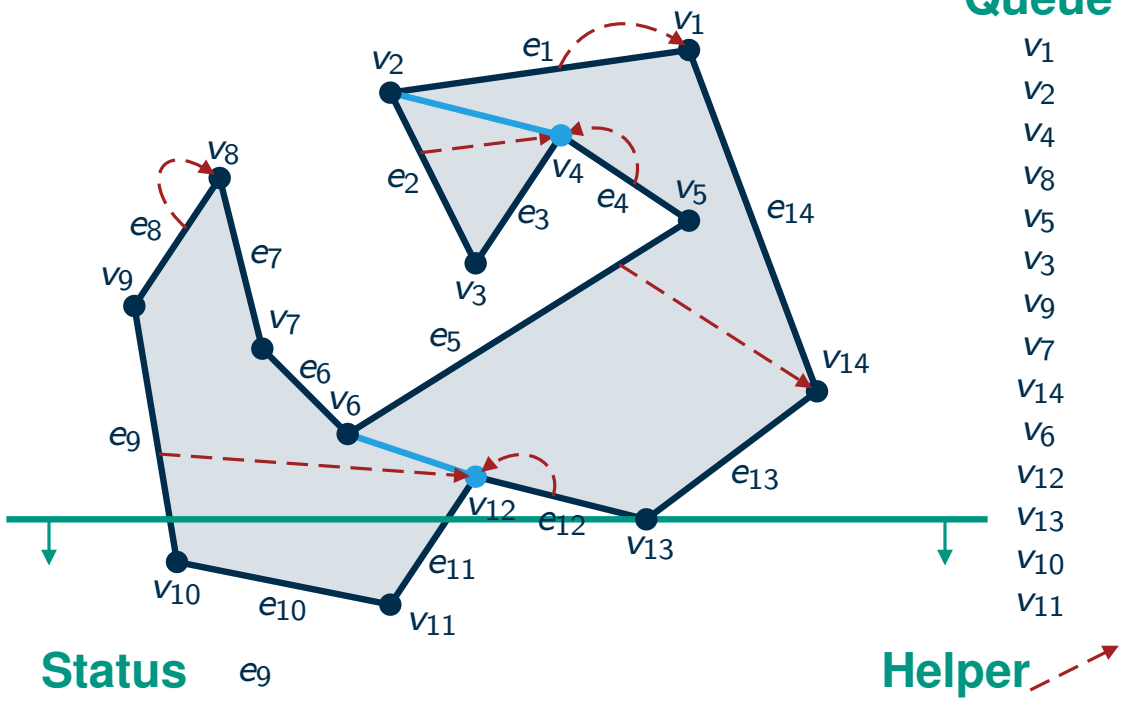
$\Rightarrow$ sweep line seems to be a good idea
(horizontal sweep line ℓ from top to bottom)

Event Queue

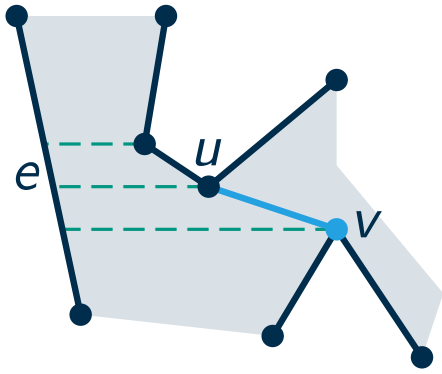
- vertices of the polygon
- sorted by y-coordinate (or lexicographic by yx)

Sweep Line Status

- edges that intersect ℓ sorted by x-coordinate
- edges that have the polygon to their right suffice
- current helper for every edge



Eliminating Split Vertices



Observations

- goal for split vertex v: find edge e to the left of v and helper of e
- e lies (partially) above v
- the helper of e lies above v

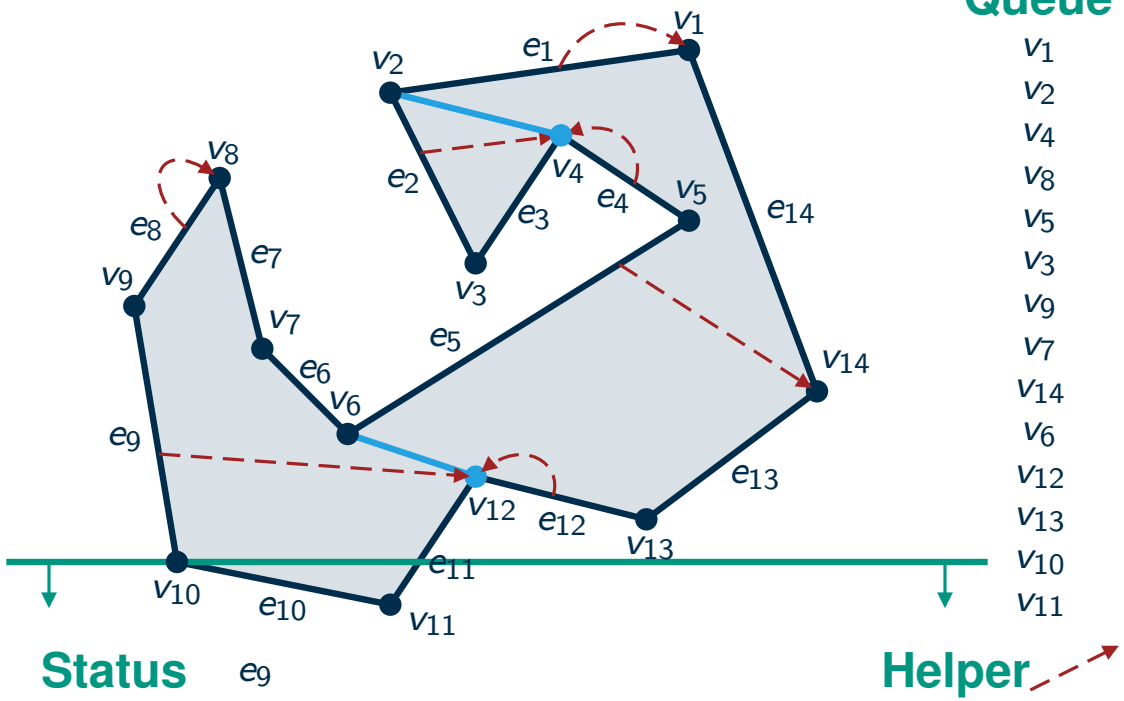
$\Rightarrow$ sweep line seems to be a good idea
(horizontal sweep line ℓ from top to bottom)

Event Queue

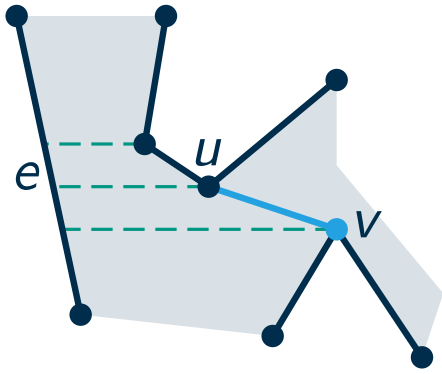
- vertices of the polygon
- sorted by y-coordinate (or lexicographic by yx)

Sweep Line Status

- edges that intersect ℓ sorted by x-coordinate
- edges that have the polygon to their right suffice
- current helper for every edge



Eliminating Split Vertices



Observations

- goal for split vertex v: find edge e to the left of v and helper of e
- e lies (partially) above v
- the helper of e lies above v

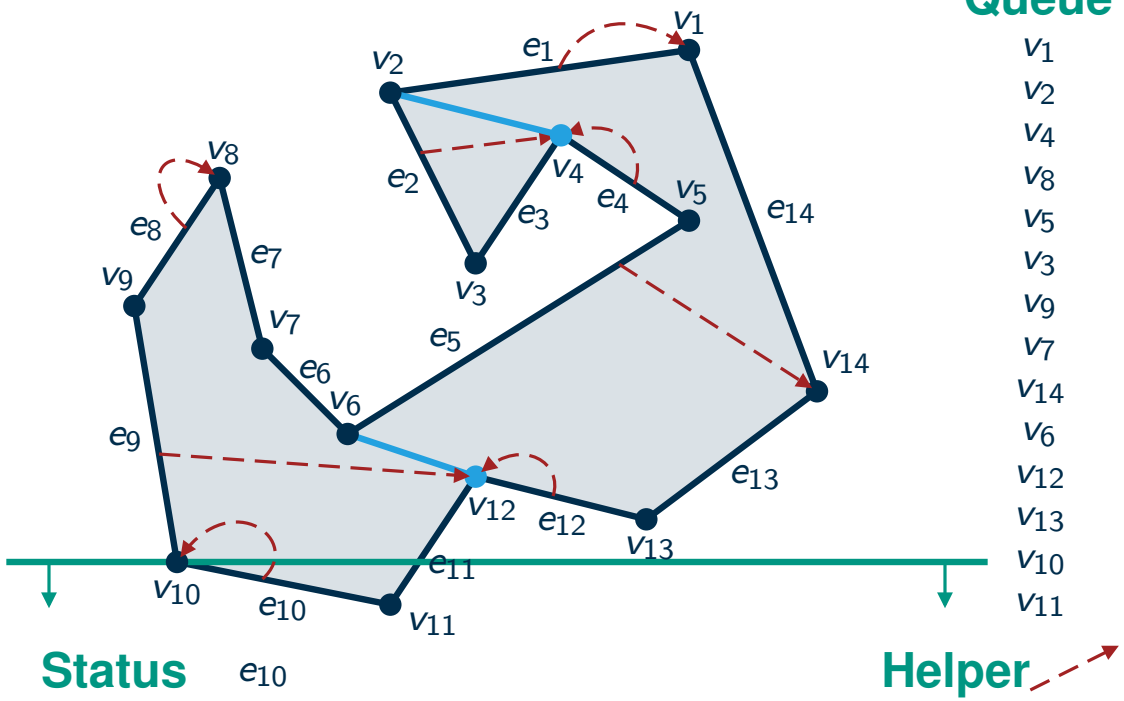
$\Rightarrow$ sweep line seems to be a good idea
(horizontal sweep line ℓ from top to bottom)

Event Queue

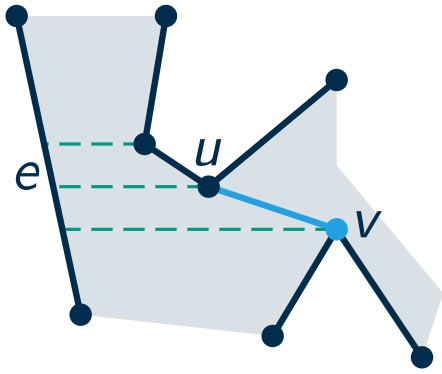
- vertices of the polygon
- sorted by y-coordinate (or lexicographic by yx)

Sweep Line Status

- edges that intersect ℓ sorted by x-coordinate
- edges that have the polygon to their right suffice
- current helper for every edge



Eliminating Split Vertices



Observations

- goal for split vertex v: find edge e to the left of v and helper of e
- e lies (partially) above v
- the helper of e lies above v

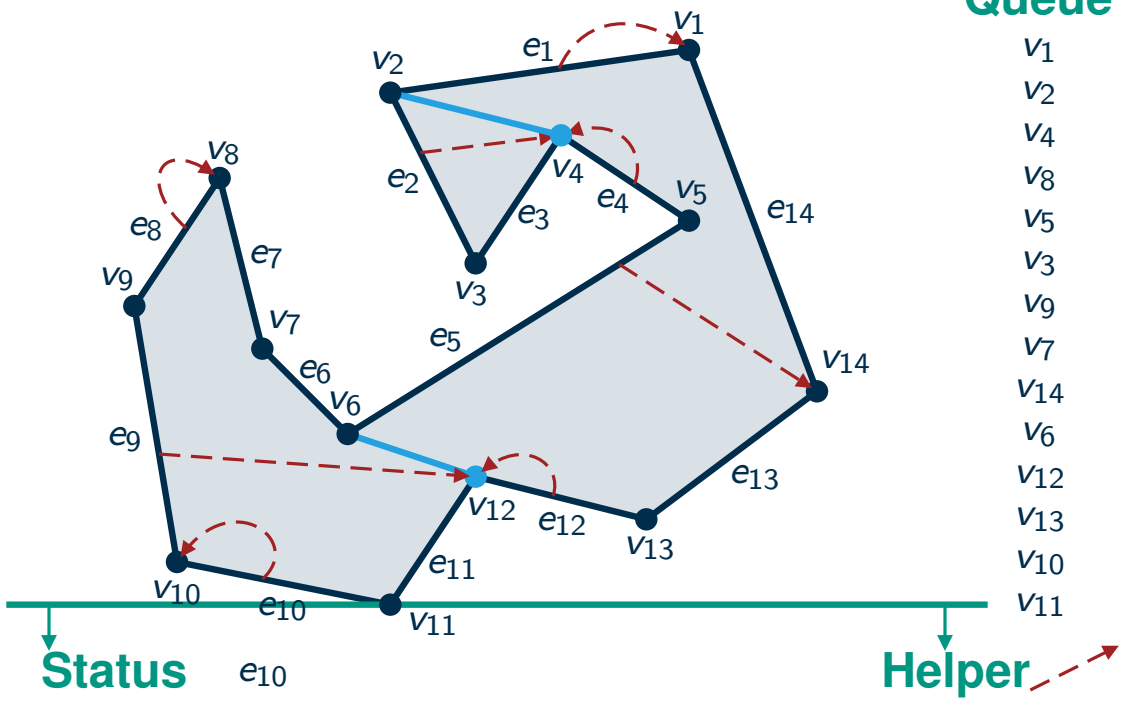
$\Rightarrow$ sweep line seems to be a good idea
(horizontal sweep line ℓ from top to bottom)

Event Queue

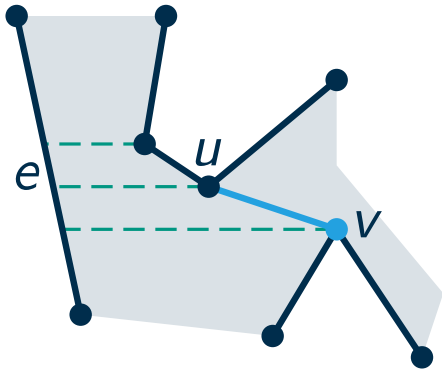
- vertices of the polygon
- sorted by y-coordinate (or lexicographic by yx)

Sweep Line Status

- edges that intersect ℓ sorted by x-coordinate
- edges that have the polygon to their right suffice
- current helper for every edge



Eliminating Split Vertices



Observations

- goal for split vertex v: find edge e to the left of v and helper of e
- e lies (partially) above v
- the helper of e lies above v

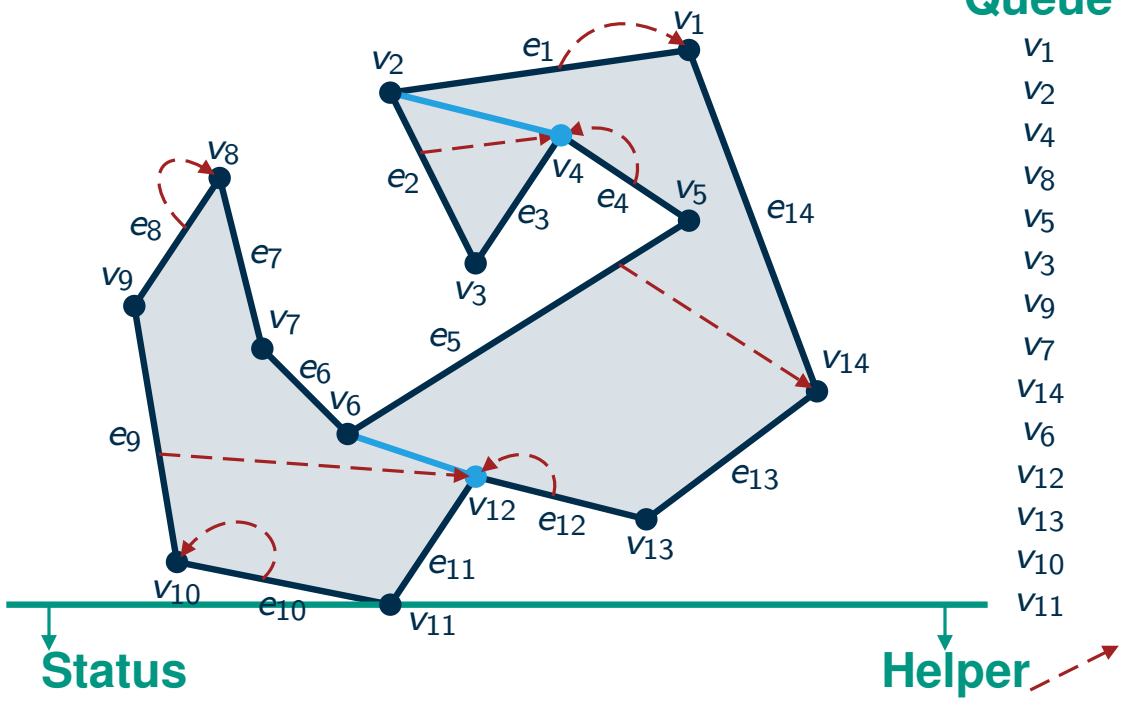
$\Rightarrow$ sweep line seems to be a good idea
(horizontal sweep line ℓ from top to bottom)

Event Queue

- vertices of the polygon
- sorted by y-coordinate (or lexicographic by yx)

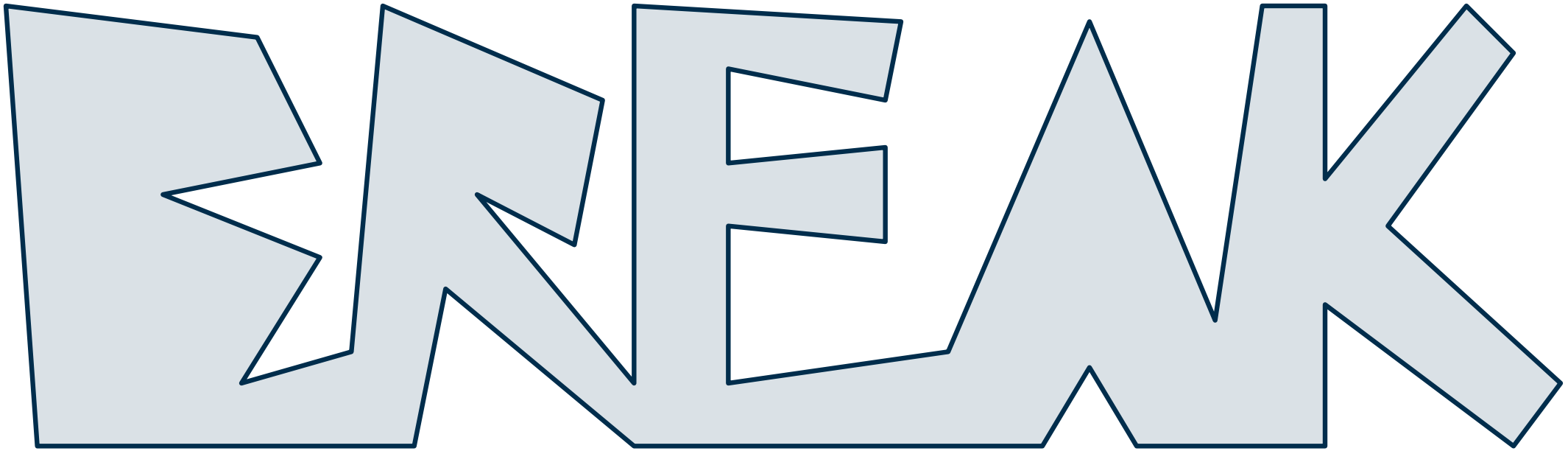
Sweep Line Status

- edges that intersect ℓ sorted by x-coordinate
- edges that have the polygon to their right suffice
- current helper for every edge



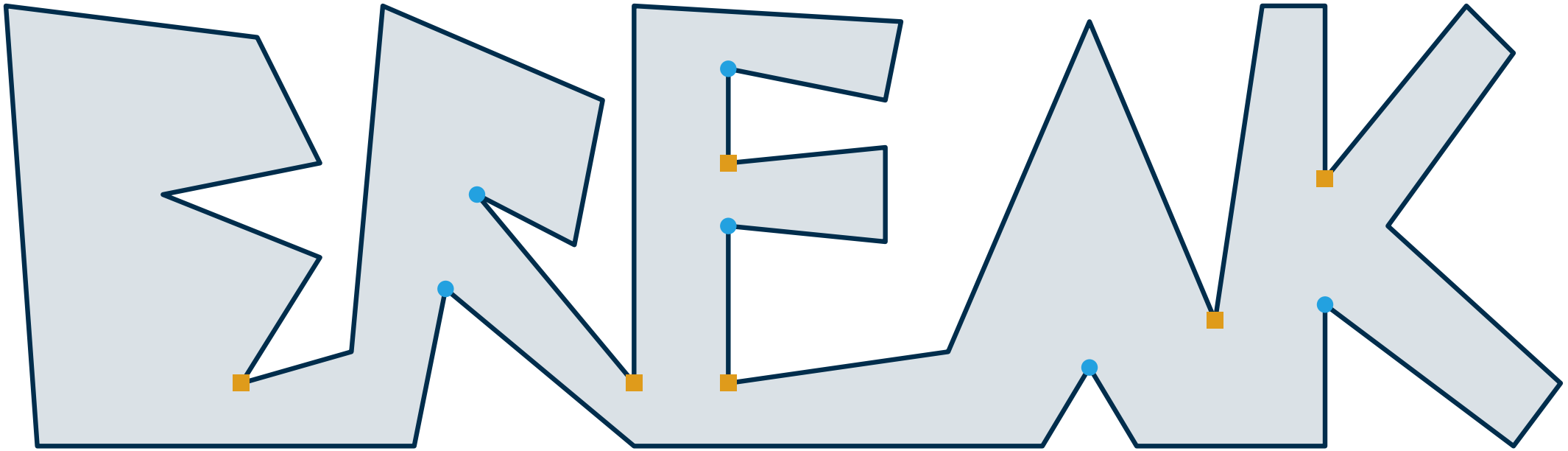
Short Break

How many diagonals do we need at least to get *y*-monotone polygons?



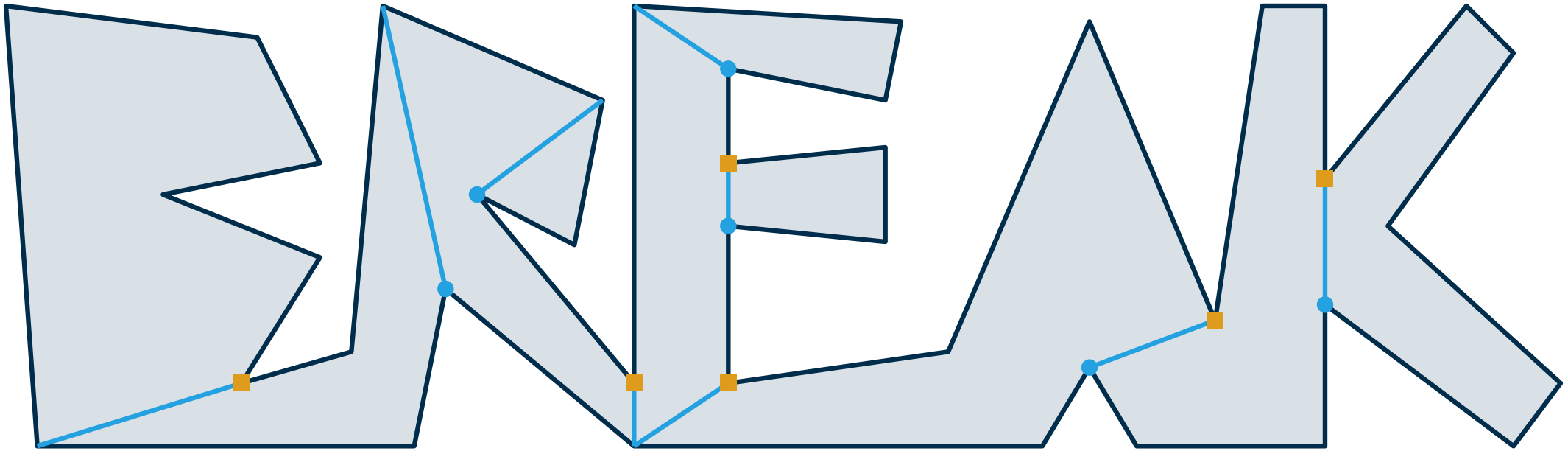
Short Break

How many diagonals do we need at least to get *y*-monotone polygons?



Short Break

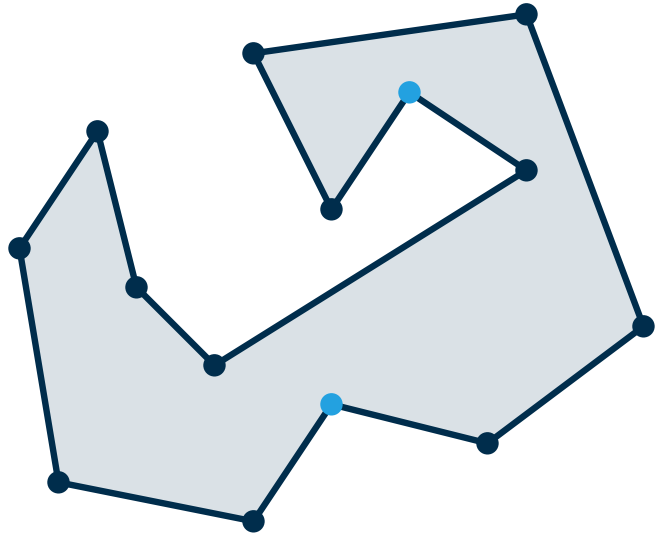
How many diagonals do we need at least to get *y*-monotone polygons?



Sweep-Line With Different Vertex Types

Different Vertex Types

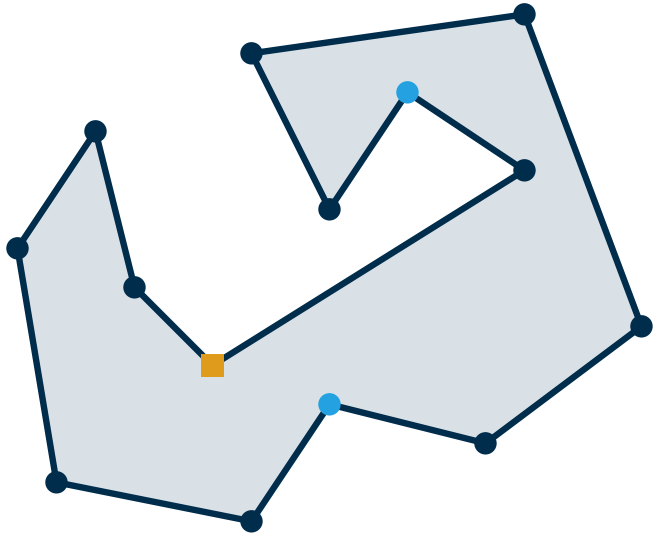
- split: edges below, polygon above



Sweep-Line With Different Vertex Types

Different Vertex Types

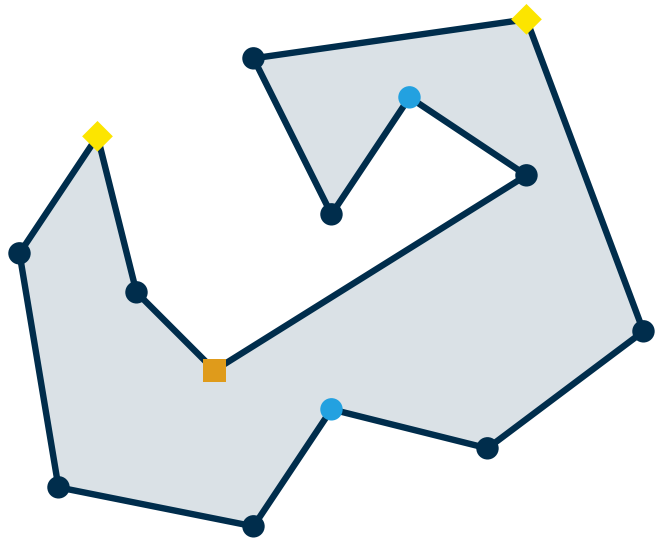
- split: edges below, polygon above
- merge: edges above, polygon below



Sweep-Line With Different Vertex Types

Different Vertex Types

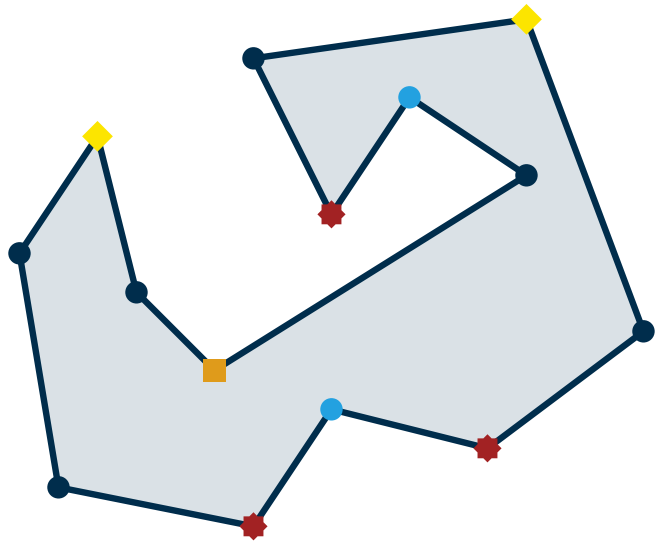
- split: edges below, polygon above
- merge: edges above, polygon below
- start: edges below, polygon below



Sweep-Line With Different Vertex Types

Different Vertex Types

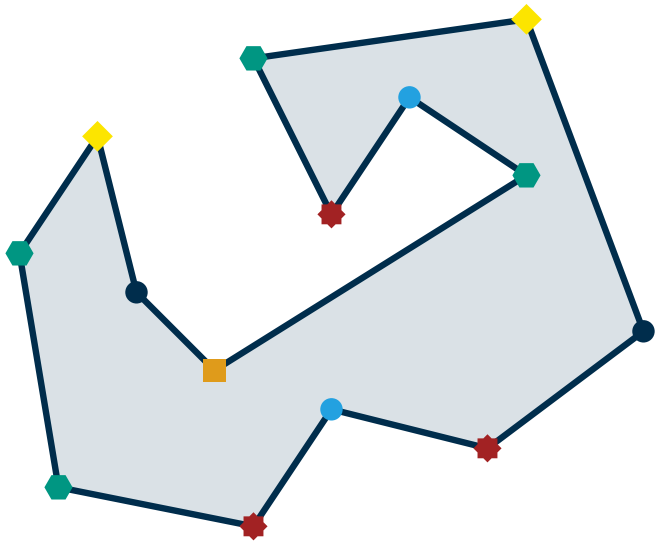
- split: edges below, polygon above
- merge: edges above, polygon below
- start: edges below, polygon below
- end: edges above, polygon above



Sweep-Line With Different Vertex Types

Different Vertex Types

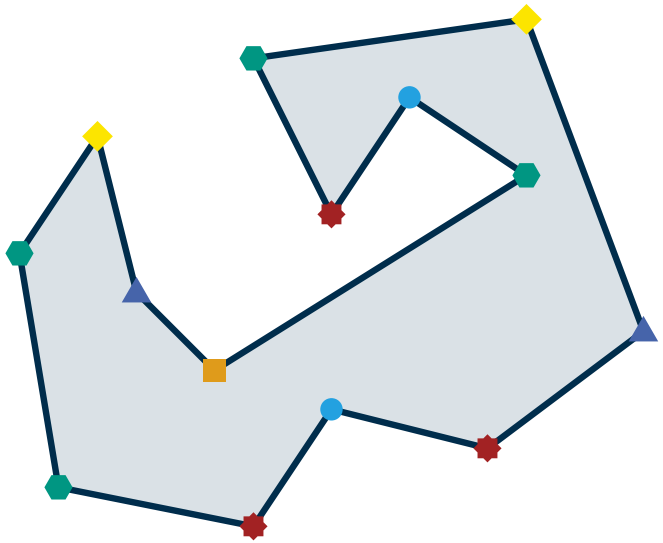
- split: edges below, polygon above
- merge: edges above, polygon below
- start: edges below, polygon below
- end: edges above, polygon above
- left: y-monoton, polygon right



Sweep-Line With Different Vertex Types

Different Vertex Types

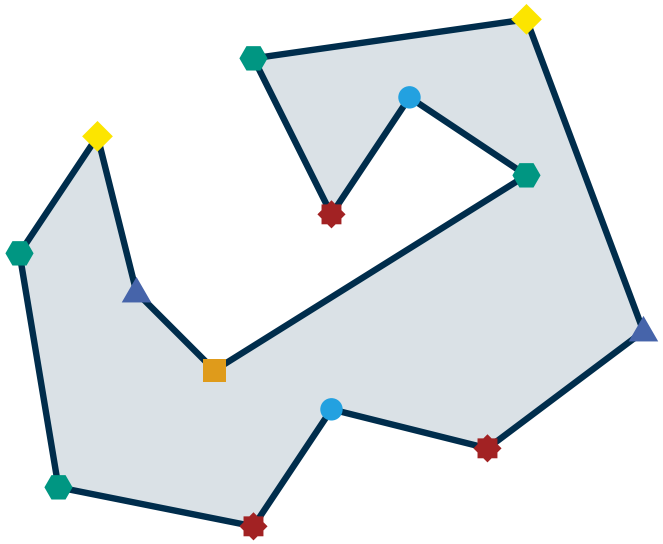
- split: edges below, polygon above
- merge: edges above, polygon below
- start: edges below, polygon below
- end: edges above, polygon above
- left: y-monoton, polygon right
- right: y-monoton, polygon left



Sweep-Line With Different Vertex Types

function MAKEMONOTONE(P)
Input: polygon P (counterclockwise)
Output: diagonals, that make P y-monotone

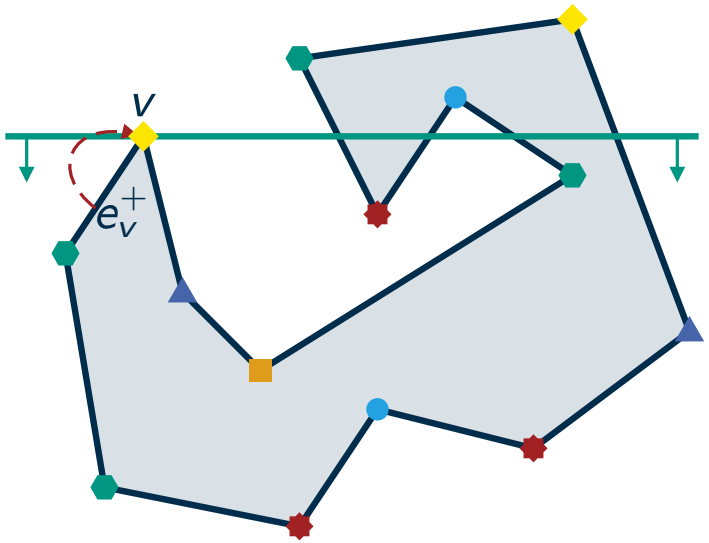
$Q =$ vertices of P sorted by y-coordinate
$T =$ search tree
while $Q \neq \emptyset$
  $v = \min\{Q\}$ and $Q = Q - v$
  HANDLEVERTEX(v)



Sweep-Line With Different Vertex Types

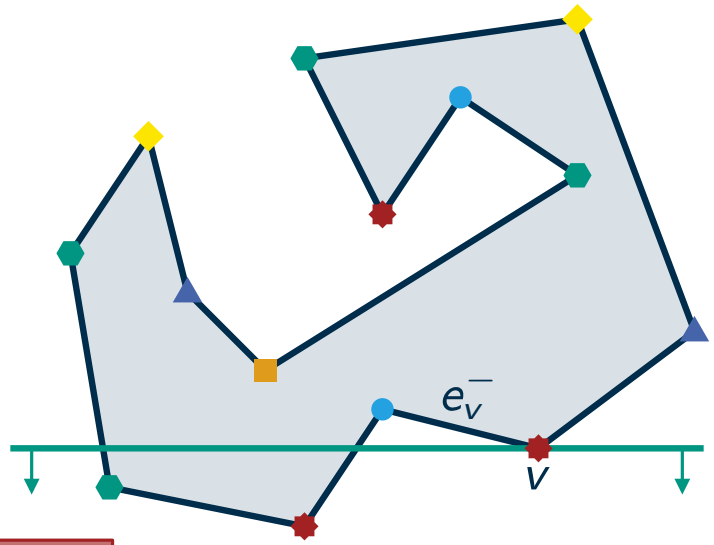
function MAKEMONOTONE(P)
Input: polygon P (counterclockwise)
Output: diagonals, that make P y-monotone
  Q = vertices of P sorted by y-coordinate
  T = search tree
  while $Q \neq \emptyset$
    $v = \min\{Q\}$ and $Q = Q - v$
    HANDLEVERTEX(v)

HANDLESTARTVERTEX(v)
  e_v^+ = edge after v in P
  insert e_v^+ into T
  helper(e_v^+) = v ◇



Sweep-Line With Different Vertex Types

function MAKEMONOTONE(P)
Input: polygon P (counterclockwise)
Output: diagonals, that make P y-monotone
Q = vertices of P sorted by y-coordinate
T = search tree
while $Q \neq \emptyset$
  $v = \min\{Q\}$ and $Q = Q - v$
  HANDLEVERTEX(v)

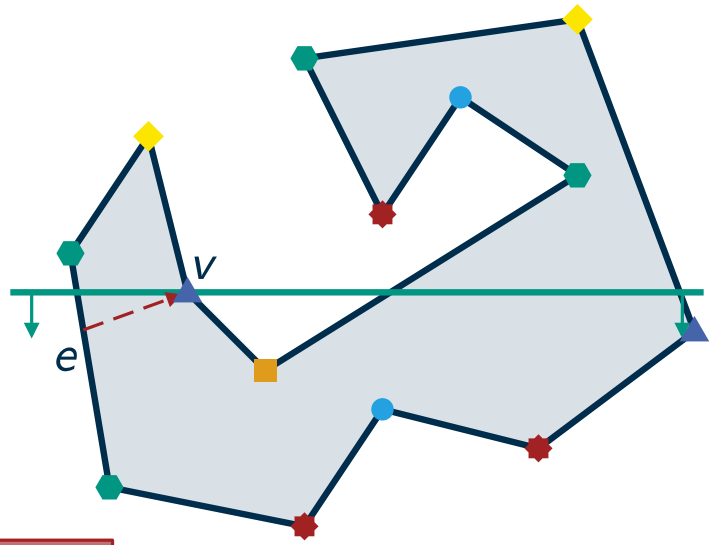


HANDLESTARTVERTEX(v)
e_v^+ = edge after v in P
insert e_v^+ into T
helper(e_v^+) = v ◇

HANDLEENDVERTEX(v)
e_v^- = edge before v in P
remove e_v^- from T ⬡

Sweep-Line With Different Vertex Types

function MAKEMONOTONE(P)
Input: polygon P (counterclockwise)
Output: diagonals, that make P y-monotone
- Q = vertices of P sorted by y-coordinate
- T = search tree
- **while** $Q \neq \emptyset$
  - $v = \min\{Q\}$ and $Q = Q - v$
  - HANDLEVERTEX(v)



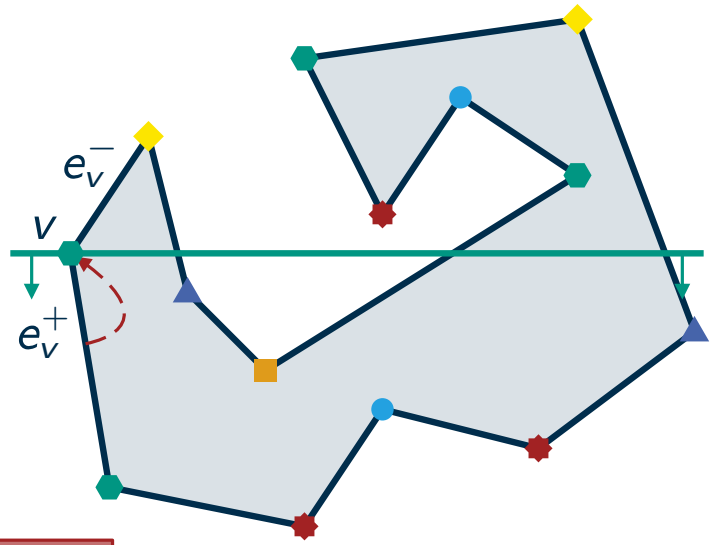
HANDLESTARTVERTEX(v)
- e_v^+ = edge after v in P
- insert e_v^+ into T
- helper(e_v^+) = v ◇

HANDLEENDVERTEX(v)
- e_v^- = edge before v in P
- remove e_v^- from T ⬡

HANDLERIGHTVERTEX(v)
- e = edge left of v in T
- helper(e) = v △

Sweep-Line With Different Vertex Types

function MAKEMONOTONE(P)
Input: polygon P (counterclockwise)
Output: diagonals, that make P y-monotone
- Q = vertices of P sorted by y-coordinate
- T = search tree
- **while** $Q \neq \emptyset$
  - $v = \min\{Q\}$ and $Q = Q - v$
  - HANDLEVERTEX(v)



HANDLESTARTVERTEX(v)
- e_v^+ = edge after v in P
- insert e_v^+ into T
- helper(e_v^+) = v

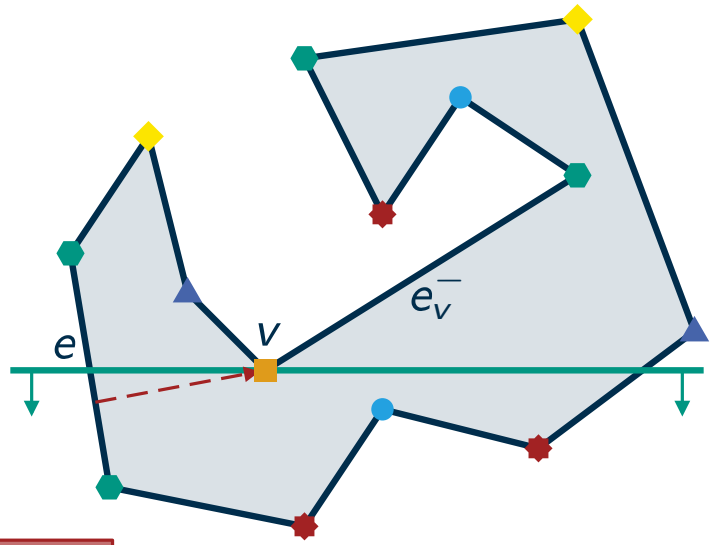
HANDLEENDVERTEX(v)
- e_v^- = edge before v in P
- remove e_v^- from T

HANDLERIGHTVERTEX(v)
- e = edge left of v in T
- helper(e) = v

HANDLELEFTVERTEX(v)
- e_v^- = edge before v in P
- remove e_v^- from T
- e_v^+ = edge after v in P
- insert e_v^+ into T
- helper(e_v^+) = v

Sweep-Line With Different Vertex Types

function MAKEMONOTONE(P)
Input: polygon P (counterclockwise)
Output: diagonals, that make P y-monotone
- Q = vertices of P sorted by y-coordinate
- T = search tree
- **while** $Q \neq \emptyset$
  - $v = \min\{Q\}$ and $Q = Q - v$
  - HANDLEVERTEX(v)



HANDLEMERGEVERTEX(v)
- e = edge left of v in T
- helper(e) = v
- e_v^- = edge before v in P
- remove e_v^- from T

HANDLESTARTVERTEX(v)
- e_v^+ = edge after v in P
- insert e_v^+ into T
- helper(e_v^+) = v

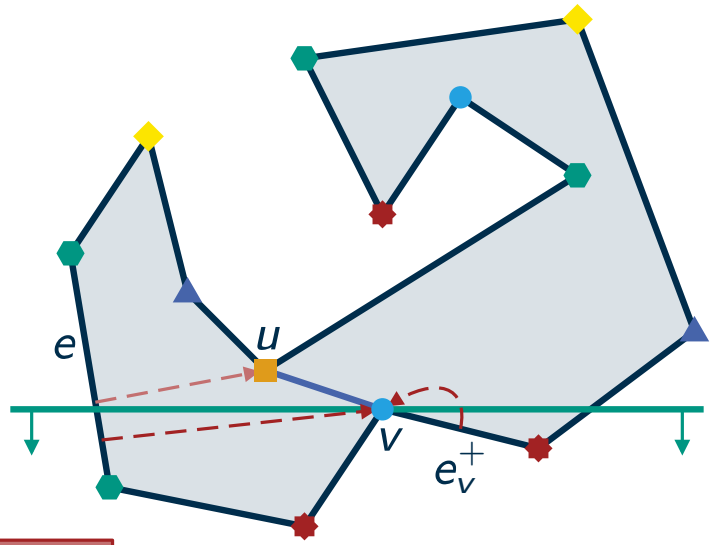
HANDLEENDVERTEX(v)
- e_v^- = edge before v in P
- remove e_v^- from T

HANDLERIGHTVERTEX(v)
- e = edge left of v in T
- helper(e) = v

HANDLELEFTVERTEX(v)
- e_v^- = edge before v in P
- remove e_v^- from T
- e_v^+ = edge after v in P
- insert e_v^+ into T
- helper(e_v^+) = v

Sweep-Line With Different Vertex Types

function MAKEMONOTONE(P)
Input: polygon P (counterclockwise)
Output: diagonals, that make P y-monotone
- Q = vertices of P sorted by y-coordinate
- T = search tree
- **while** $Q \neq \emptyset$
  - $v = \min\{Q\}$ and $Q = Q - v$
  - HANDLEVERTEX(v)



HANDLEMERGEVERTEX(v)
- e = edge left of v in T
- helper(e) = v
- e_v^- = edge before v in P
- remove e_v^- from T

HANDLESTARTVERTEX(v)
- e_v^+ = edge after v in P
- insert e_v^+ into T
- helper(e_v^+) = v

HANDLEENDVERTEX(v)
- e_v^- = edge before v in P
- remove e_v^- from T

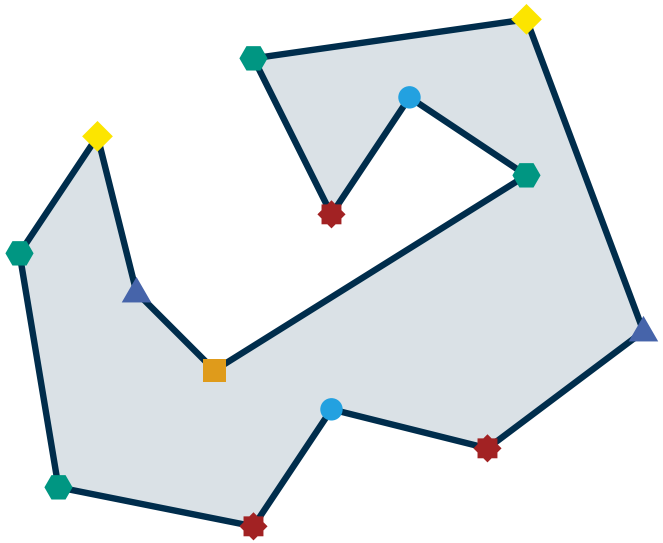
HANDLERIGHTVERTEX(v)
- e = edge left of v in T
- helper(e) = v

HANDLELEFTVERTEX(v)
- e_v^- = edge before v in P
- remove e_v^- from T
- e_v^+ = edge after v in P
- insert e_v^+ into T
- helper(e_v^+) = v

HANDLESPLITVERTEX(v)
- e = edge left of v in T
- u = helper(e)
- **output** diagonal uv
- helper(e) = v
- e_v^+ = edge after v in P
- insert e_v^+ into T
- helper(e_v^+) = v

Sweep-Line With Different Vertex Types

function MAKEMONOTONE(P)
Input: polygon P (counterclockwise)
Output: diagonals, that make P y-monotone
- Q = vertices of P sorted by y-coordinate
- T = search tree
- **while** $Q \neq \emptyset$
  - $v = \min\{Q\}$ and $Q = Q - v$
  - HANDLEVERTEX(v)



HANDLESTARTVERTEX(v)
- e_v^+ = edge after v in P
- insert e_v^+ into T
- helper(e_v^+) = v

HANDLEENDVERTEX(v)
- e_v^- = edge before v in P
- remove e_v^- from T

HANDLERIGHTVERTEX(v)
- e = edge left of v in T
- helper(e) = v

HANDLELEFTVERTEX(v)
- e_v^- = edge before v in P
- remove e_v^- from T
- e_v^+ = edge after v in P
- insert e_v^+ into T
- helper(e_v^+) = v

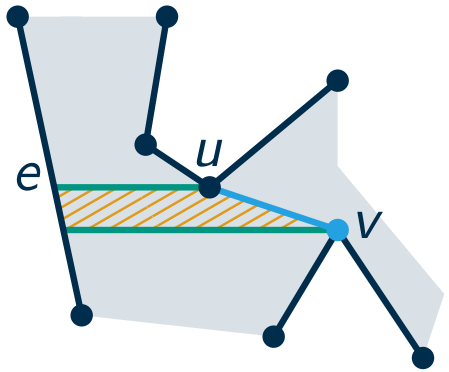
HANDLEMERGEVERTEX(v)
- e = edge left of v in T
- helper(e) = v
- e_v^- = edge before v in P
- remove e_v^- from T

HANDLESPLITVERTEX(v)
- e = edge left of v in T
- u = helper(e)
- **output** diagonal uv
- helper(e) = v
- e_v^+ = edge after v in P
- insert e_v^+ into T
- helper(e_v^+) = v

We Create No Intersections

Recall

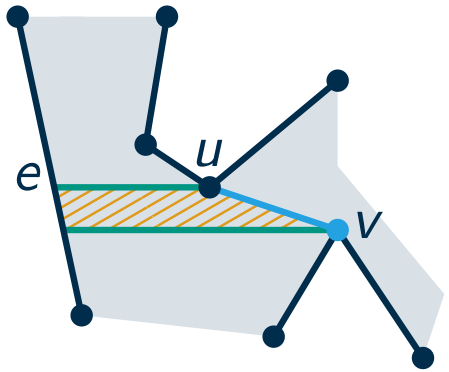
- the inserted diagonals do not intersect the polygon
- core argument: the quadrilateral between uv and e contains no vertex



We Create No Intersections

Recall

- the inserted diagonals do not intersect the polygon
- core argument: the quadrilateral between uv and e contains no vertex

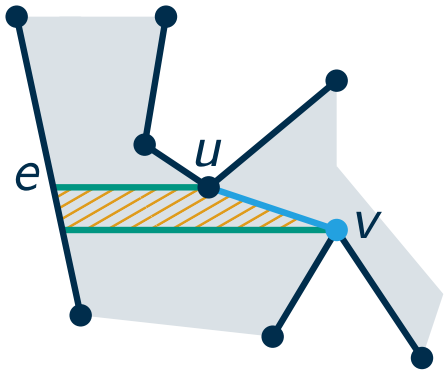


Can We Get An Intersection With A Previously Inserted Diagonal?

We Create No Intersections

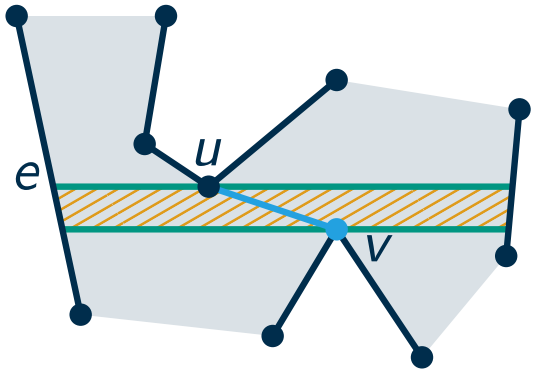
Recall

- the inserted diagonals do not intersect the polygon
- core argument: the quadrilateral between uv and e contains no vertex



Can We Get An Intersection With A Previously Inserted Diagonal?

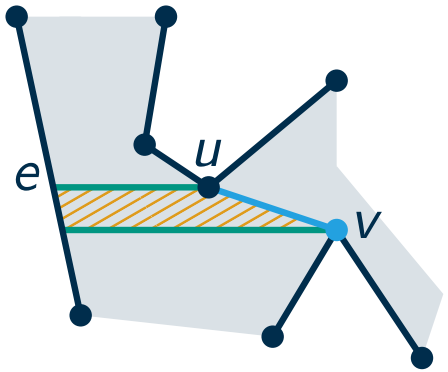
- extend the quadrilateral to the right
- same argument: the extended quadrilateral also contains no vertex



We Create No Intersections

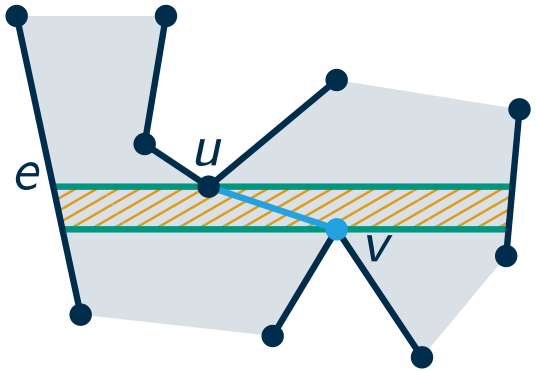
Recall

- the inserted diagonals do not intersect the polygon
- core argument: the quadrilateral between uv and e contains no vertex



Can We Get An Intersection With A Previously Inserted Diagonal?

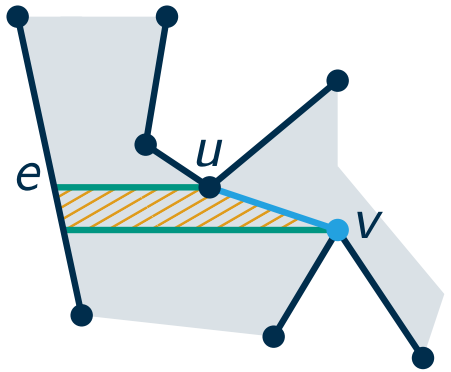
- extend the quadrilateral to the right
- same argument: the extended quadrilateral also contains no vertex
- both vertices of a previously inserted diagonal lie above v



We Create No Intersections

Recall

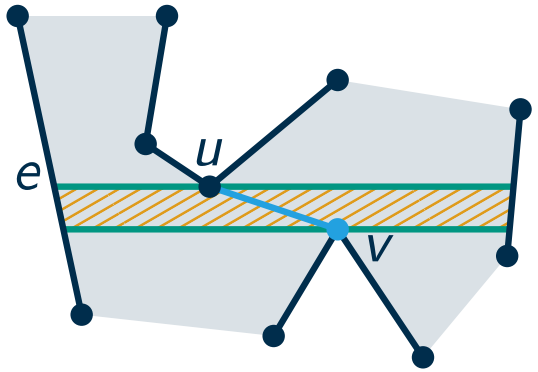
- the inserted diagonals do not intersect the polygon
- core argument: the quadrilateral between uv and e contains no vertex



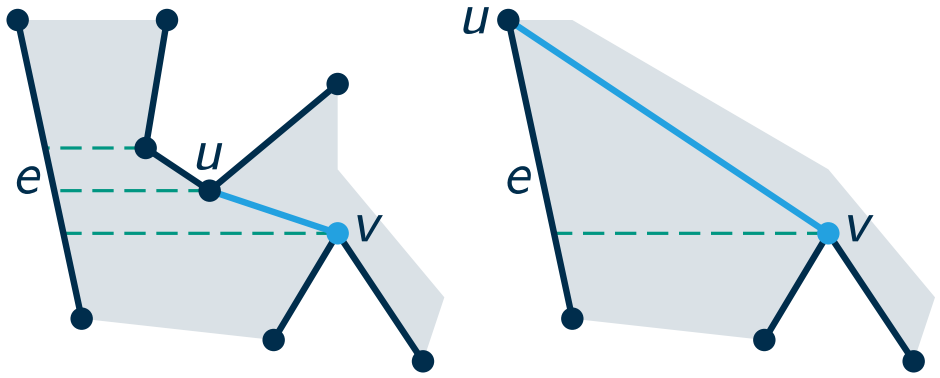
Can We Get An Intersection With A Previously Inserted Diagonal?

- extend the quadrilateral to the right
- same argument: the extended quadrilateral also contains no vertex
- both vertices of a previously inserted diagonal lie above v

$\Rightarrow$ uv does not intersect a previous diagonal



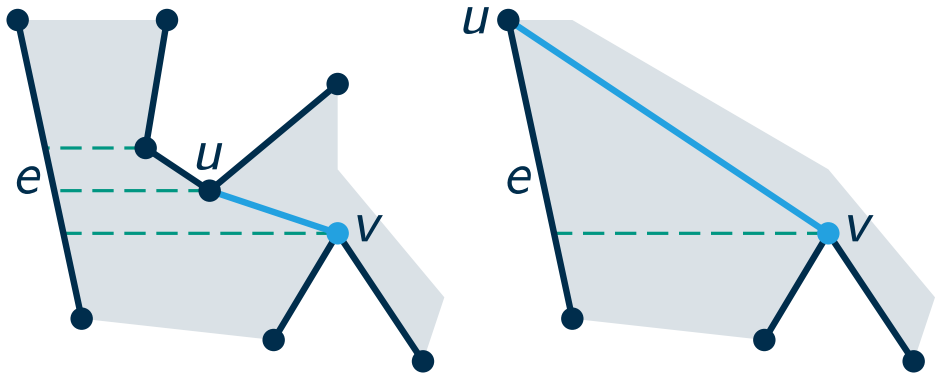
And What About Merge Vertices?



Recall: Split Vertex *v*

- e: edge to the left of v
- u: lowest vertex above v that has e to its left (helper of e)
- if it doesn't exist: choose u as the upper vertex of e
- connect v with u

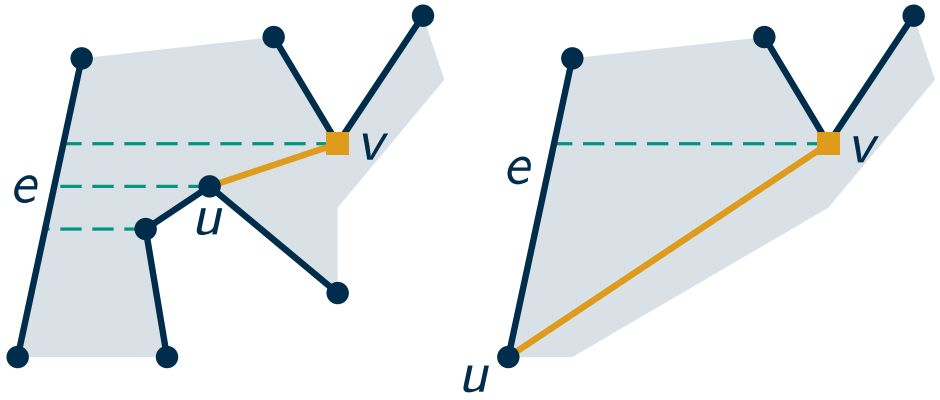
And What About Merge Vertices?



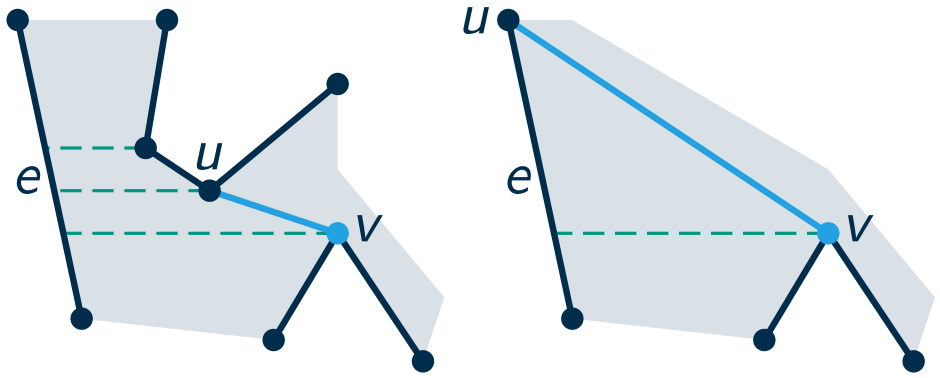
Recall: Split Vertex *v*

- *e*: edge to the left of *v*
- *u*: lowest vertex above *v* that has *e* to its left (helper of *e*)
- if it doesn't exist: choose *u* as the upper vertex of *e*
- connect *v* with *u*

mirroring translates *v* into a merge vertex



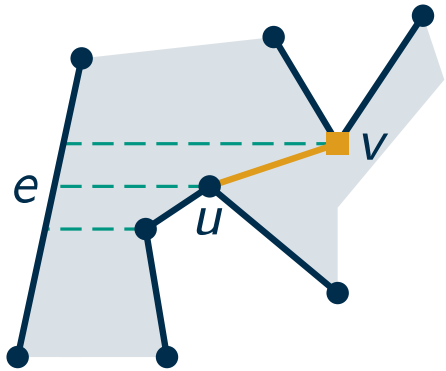
And What About Merge Vertices?

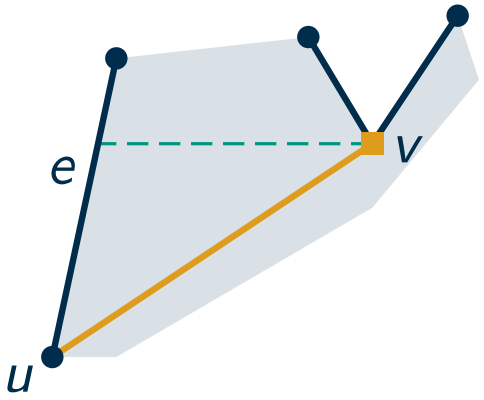


Recall: Split Vertex *v*

- *e*: edge to the left of *v*
- *u*: lowest vertex above *v* that has *e* to its left (helper of *e*)
- if it doesn't exist: choose *u* as the upper vertex of *e*
- connect *v* with *u*

mirroring translates *v* into a merge vertex

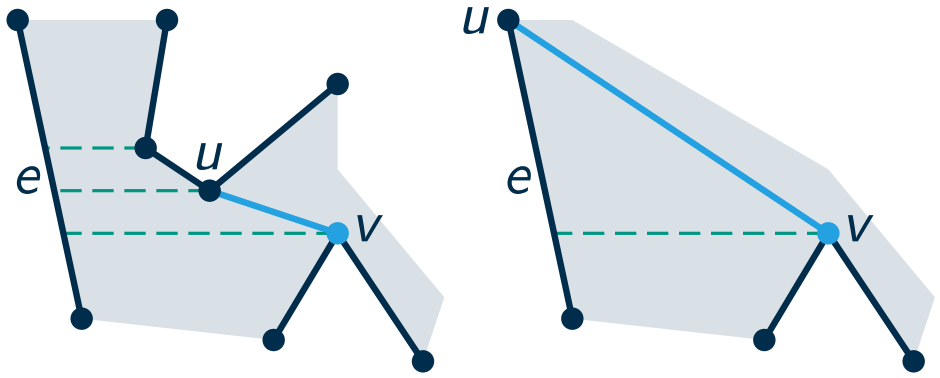




Handling A Merge Vertex *v*

- solution for the lazy theoretician:
  - just run it again in the opposite direction

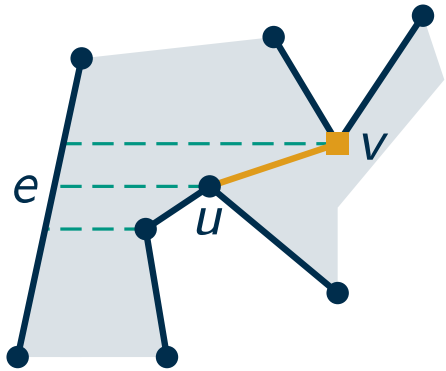
And What About Merge Vertices?

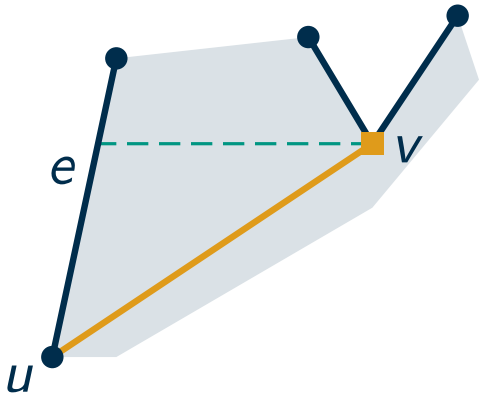


Recall: Split Vertex *v*

- *e*: edge to the left of *v*
- *u*: lowest vertex above *v* that has *e* to its left (helper of *e*)
- if it doesn't exist: choose *u* as the upper vertex of *e*
- connect *v* with *u*

mirroring translates *v* into a merge vertex





Handling A Merge Vertex *v*

- solution for the lazy theoretician:
  - just run it again in the opposite direction
- alternative:
  - observe: *v* is the helper of *e* when we process *u*
  - when *e* ends at *u* or gets a new helper *u*: current helper *v* is merge vertex → insert *uv*

Wrap-Up

Theorem **(subdivision into y-monotone pieces)**
A polygon with n vertices can be subdivided into y-monotone pieces in $O(n \log n)$ time.

Wrap-Up

Theorem **(subdivision into y-monotone pieces)**
A polygon with n vertices can be subdivided into y-monotone pieces in $O(n \log n)$ time.

What Else Have We Learned Today?

- additional application of the sweep line technique
- concept of monotonicity
- splitting a complicated problem into two simpler subproblems

Wrap-Up

Theorem **(subdivision into y-monotone pieces)**
A polygon with n vertices can be subdivided into y-monotone pieces in $O(n \log n)$ time.

What Else Have We Learned Today?

- additional application of the sweep line technique
- concept of monotonicity
- splitting a complicated problem into two simpler subproblems

What Else Is There?

- lower bound of $\Omega(n \log n)$ if the polygon can have holes

Wrap-Up

Theorem **(subdivision into y-monotone pieces)**
A polygon with n vertices can be subdivided into y-monotone pieces in $O(n \log n)$ time.

What Else Have We Learned Today?

- additional application of the sweep line technique
- concept of monotonicity
- splitting a complicated problem into two simpler subproblems

What Else Is There?

- lower bound of $\Omega(n \log n)$ if the polygon can have holes
- $O(n \log \log n)$, $O(n \log^* n)$, and even $O(n)$ is possible, if the polygon has no holes

Wrap-Up

Theorem (subdivision into y-monotone pieces)
A polygon with n vertices can be subdivided into y-monotone pieces in $O(n \log n)$ time.

What Else Have We Learned Today?

- additional application of the sweep line technique
- concept of monotonicity
- splitting a complicated problem into two simpler subproblems

What Else Is There?

- lower bound of $\Omega(n \log n)$ if the polygon can have holes
- $O(n \log \log n)$, $O(n \log^* n)$, and even $O(n)$ is possible, if the polygon has no holes
- corresponding 3-dimensional problem is NP-hard

References

$O(n \log \log n)$

- **An $O(n \log \log n)$-Time Algorithm for Triangulating a Simple Polygon** https://doi.org/10.1137/0217010 (1988)
  Robert E. Tarjan, Christopher J. Van Wyk
- **Polygon triangulation in $O(n \log \log n)$ time with simple data structures** https://doi.org/10.1007/BF02187846 (1992)
  David G. Kirkpatrick, Maria M. Klawe, Robert E. Tarjan

$O(n \log^* n)$

- **A fast Las Vegas algorithm for triangulating a simple polygon** https://doi.org/10.1007/BF02187741 (1989)
  Kenneth L. Clarkson, Robert E. Tarjan, Christopher J. Van Wyk
- **Randomization Yields Simple $O(n \log^* n)$ Algorithms for Difficult $\Omega(n)$ Problems** https://doi.org/10.1142/S021819599200007X (1992)
  Olivier Devillers
- **A simple and fast incremental randomized algorithm for computing trapezoidal decompositions and for triangulating polygons** (1991)
  Raimund Seidel https://doi.org/10.1016/0925-7721(91)90012-4

$O(n)$

- **Triangulating a simple polygon in linear time** https://doi.org/10.1007/BF02574703 (1991)
  Bernard Chazelle
- **A Randomized Algorithm for Triangulating a Simple Polygon in Linear Time** https://doi.org/10.1007/s00454-001-0027-x (2001)
  N. M. Amato, M. T. Goodrich, E. A. Ramos