Minkowski Sums and Offsets of Polygons

Seminar Computational Geometry and Geometric Computing

Andreas Bock

Supervisor: Eric Berberich

Overview

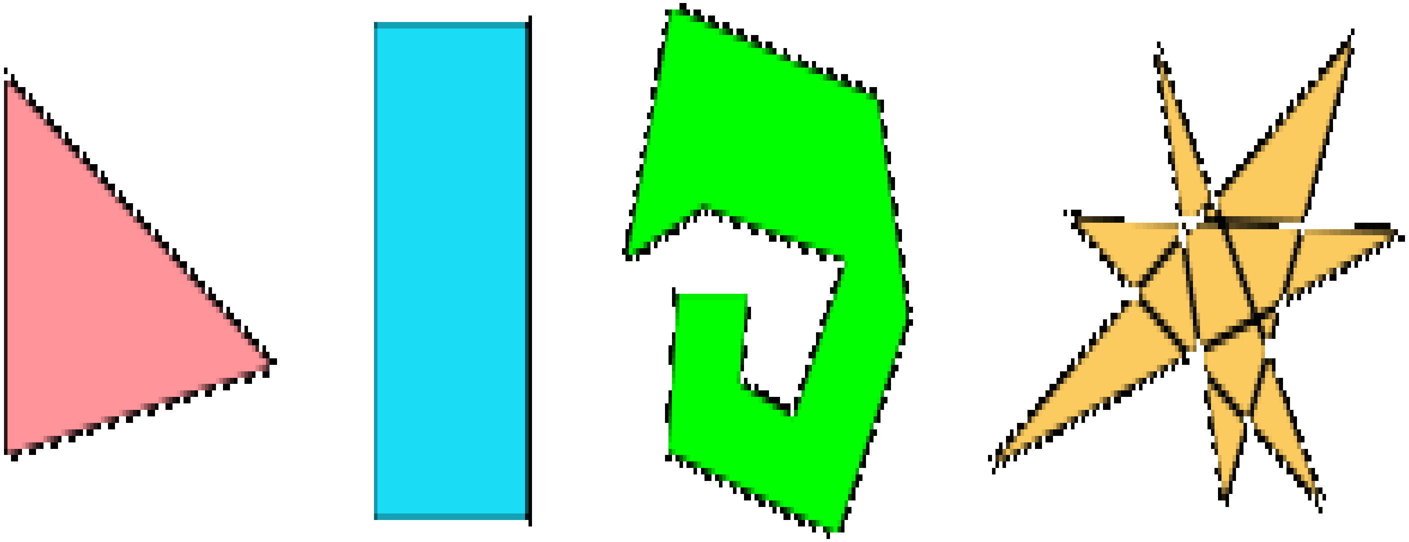
- Polygons
- Minkowski Sums
  - Decomposing into convex sub-polygons
  - Convolution method

- Offsets of polygons
  - Exact representation
  - Approximation

What is a polygon?

- we are talking about geometry
- a polygon is a plane figure with at least 3 points
- bounded by a closed path, composed of a finite sequence of straight line segments
- these segments are called its edges
- the points where two edges meet are the polygon's vertices

What is a polygon?



A few polygons (source: wiki)

Properties

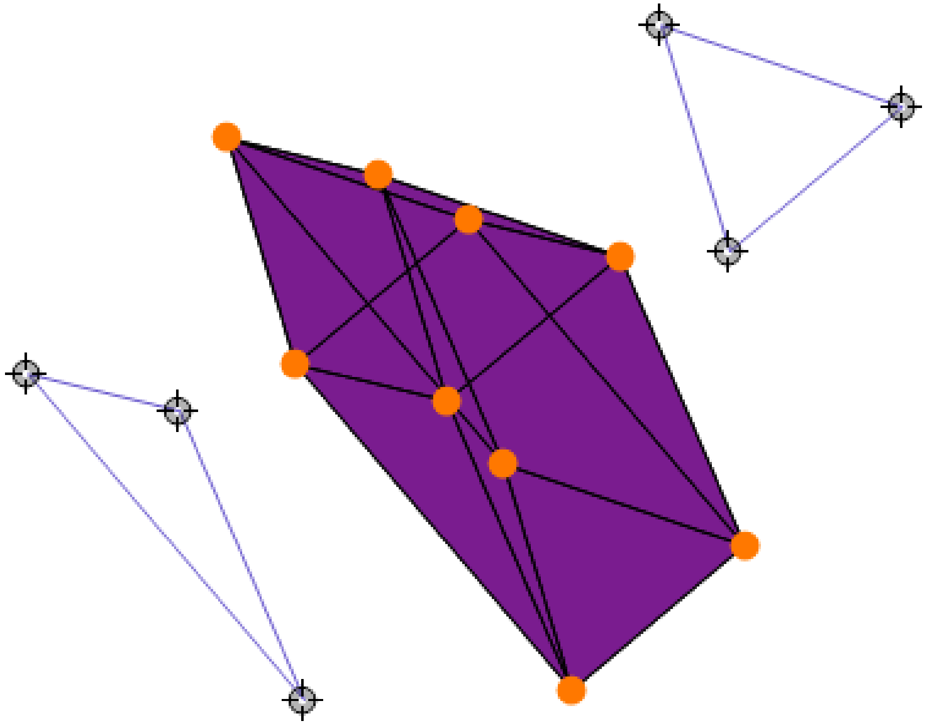
- **Convex**: any line drawn through the polygon (and not tangent to an edge or corner) meets its boundary exactly twice.
- **Non-convex**: a line may be found which meets its boundary more than twice.
- **Simple**: the boundary of the polygon does not cross itself. All convex polygons are simple.
- **Concave**: Non-convex and simple.

Minkowski Sums

Hermann Minkowski (1864-1909) (adapted by wikipedia)

Minkowski Sums

- We have two 2D polygonal sets $A, B \in \mathbb{R}^2$
- The Minkowski sum $A \oplus B$ of this two sets is a set with the sum of all elements from A and all elements of B
- $A \oplus B = \left\{ a + b \mid a \in A, b \in B \right\}$



Minkowski Sum of 2 triangles (created with math.player)

Some properties of Minkowski Sums

- associative
- distributive
- commutative
- Minkowski Sum of convex sets results again in a convex set

Where are Minkowski sums useful?

- Computer aided design
- Robot motion planning
- Computer aided manufacturing
- Mathematical morphology
- etc.

Configuration Space

- Robot *B*, obstacle *A*
- Reference point r attached to *B*
- B' is a copy of *B* rotated by 180°
- $A \oplus B'$ is the locus (Linie) of placements of the point *r* where $A \cap B \neq \emptyset$
- *B* collides with *A* when translated along a path, if *r* – moved along this path – intersects $A \oplus B'$

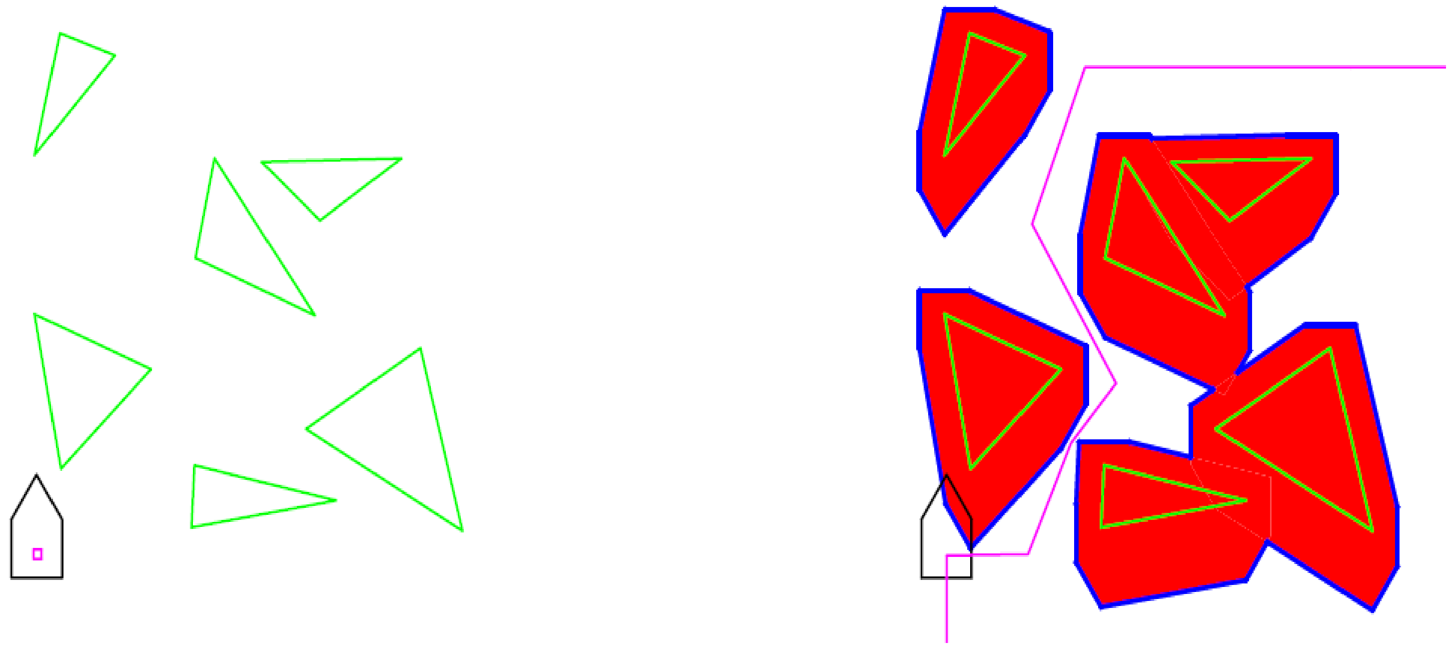


Figure 1: Robot and obstacles: a reference point is rigidly attached to the robot on the left-hand side. The configuration space obstacles and a free translational path for the robot on the right-hand side.

Adapted by Agarwal

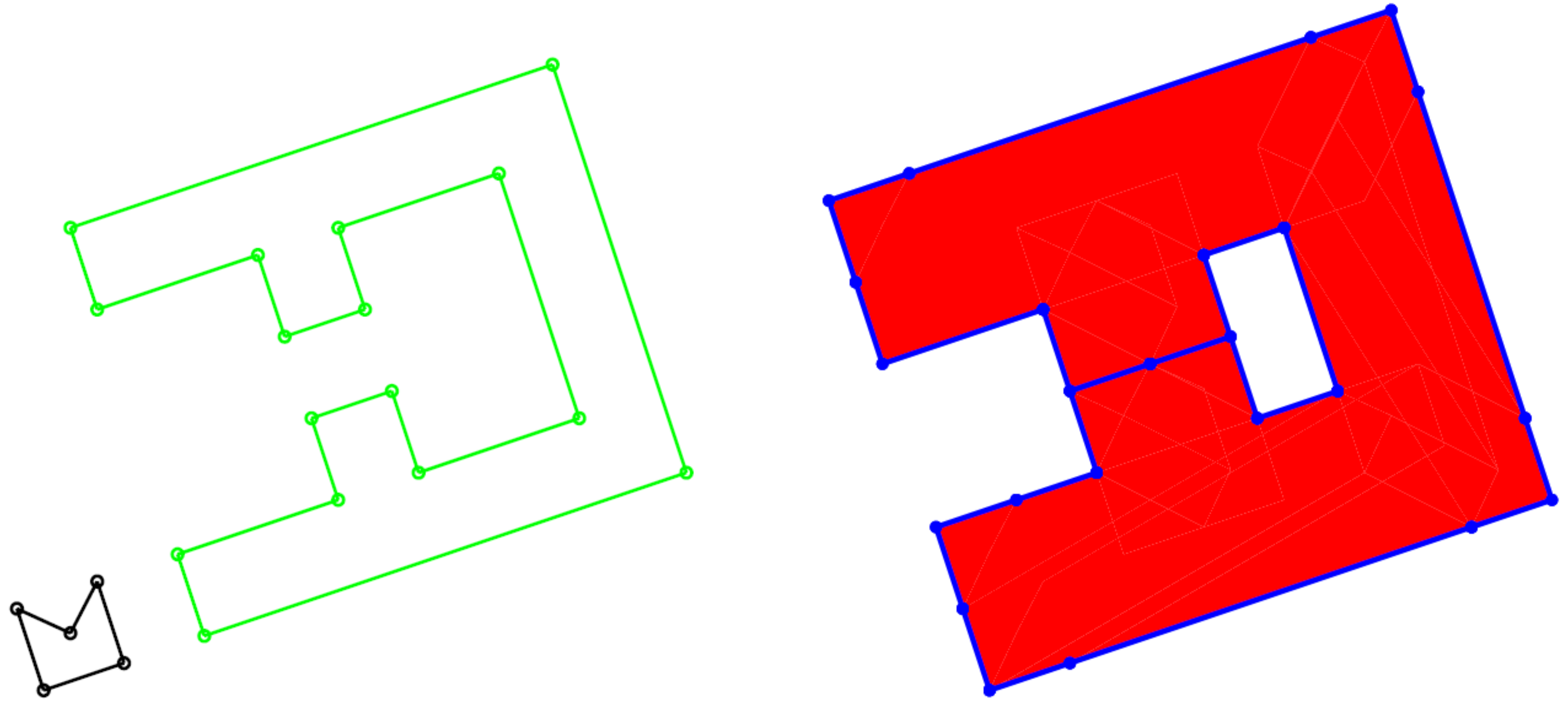


Figure 3: Tight passage: the desired target placement for the small polygon is inside the inner room defined by the larger polygon. In the configuration space the only possible path to achieve this target passes through the line segment emanating from the hole in the sum on the right-hand side.

Adapted by Flato

How much effort Minkowski Sums take?

- Lets say we have different polygonal sets *P, Q* with m, n vertices
- $P \oplus Q$ is a portion of the arrangement of mn segments
- Each segment is the Minkowski sum of a vertex of *P* and an edge of *Q* or the other way around

How much effort Minkowski Sums take?

- Size of $P \oplus Q$ is $O\left(m^2 n^2\right)$, same as computing time worst case
- If both polygons are convex, we have only m+n vertices and with calculation time of *O*(m+n)

How to calculate the Minkowski Sum ?

We will check two methods here

1. Decomposing into convex sub-polygons
2. Convolution method

decomposing into convex sub-polygons

- We decompose *P, Q* into convex sub-polygons $P_{1,} P_{2,\ldots,} P_s$ and $Q_{1,} Q_{2,\ldots,} Q_t$
- Then we calculate $P \oplus Q = U_{i,j} (P_i \oplus Q_j)$
- In theory the choice of decomposition method does not matter, because even in the worst case running time will not be affected.
- In practice this choice has an effect (later).

Minkowski Sum Algorithm

- Step 1: Decompose P into convex sub-polygons $P_1, P_{2,\ldots}, P_s$ and Q into the convex sub-polygons $Q_1, Q_{2,\ldots}, Q_t$
- Step 2: For each $i \in [1..s]$ and for each $j \in [1..t]$, compute the Minkowski sub-sum $P_i \oplus Q_j$ $(O(1))$ which we denote by R_{ij}. We denote by R the set $\{R_{ij} \mid i \in [1..s], j \in [1..t]\} \rightarrow O(m,n)$
- Step 3: Construct the union of all polygons in R, computed in Step 2; the output is represented as a planar map.

Minkowski Sum Algorithm

- Like mentioned before, there are some algorithms for Decomposition
- Triangulation
  - Naive triangulation
  - Optimal triangulation (also different methods) $\rightarrow O(n^3)$
- Convex decomposition with and without Steiner points $\rightarrow O(r^2 n \log n)$
  - Steiner point means additional vertex which is not part of original signal

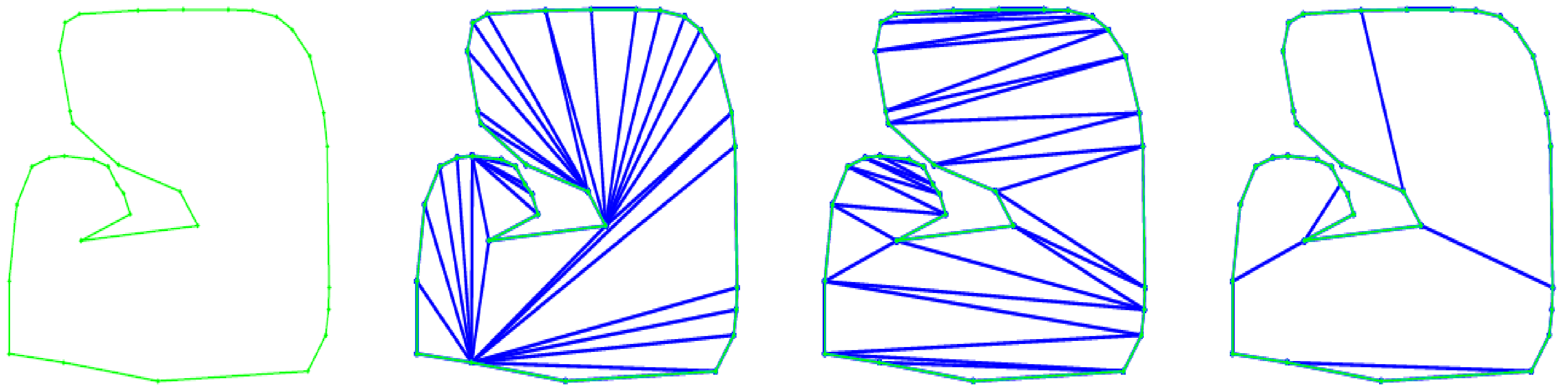


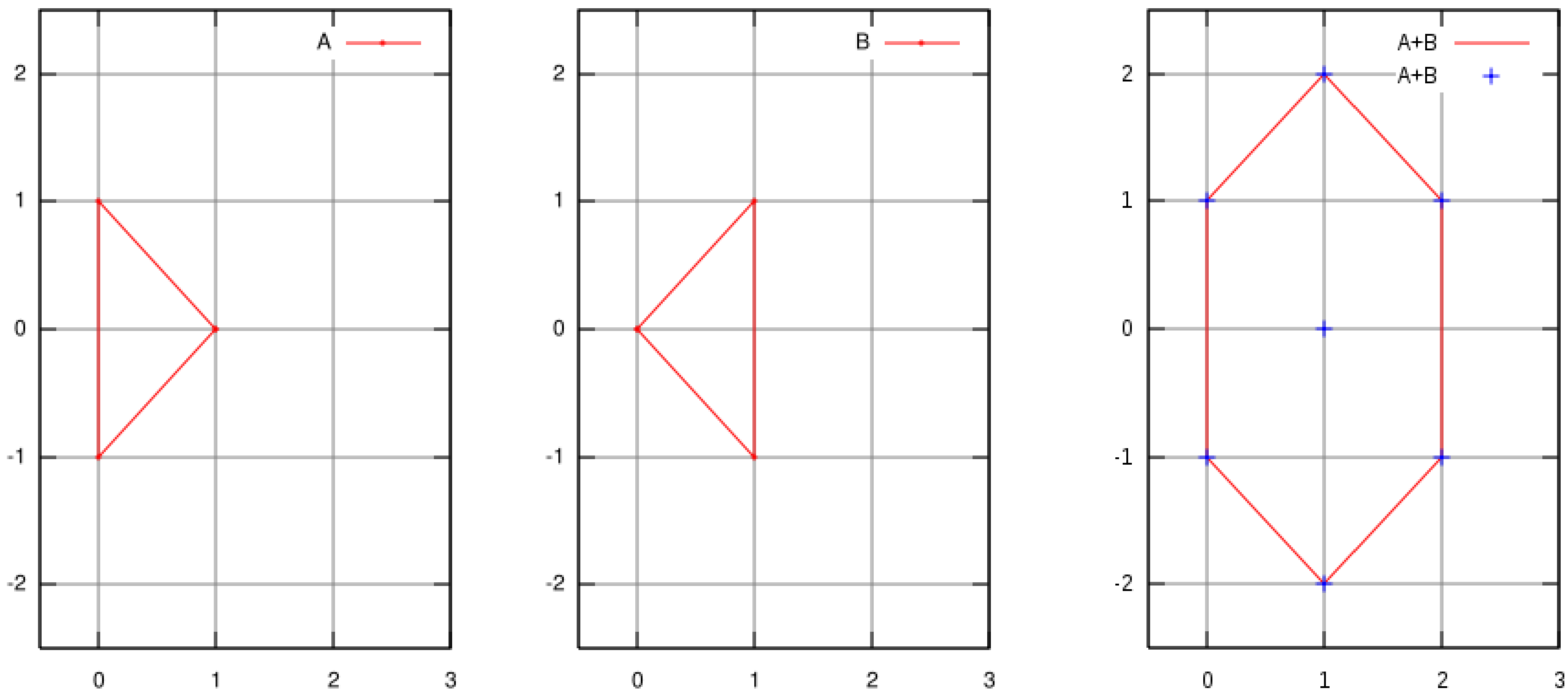
Figure 5: Different decomposition methods applied to the polygon P (leftmost in the figure), from left to right: naïve triangulation, minimum Σd_i^2 triangulation and minimum convex decomposition

Adapted by Agarwal

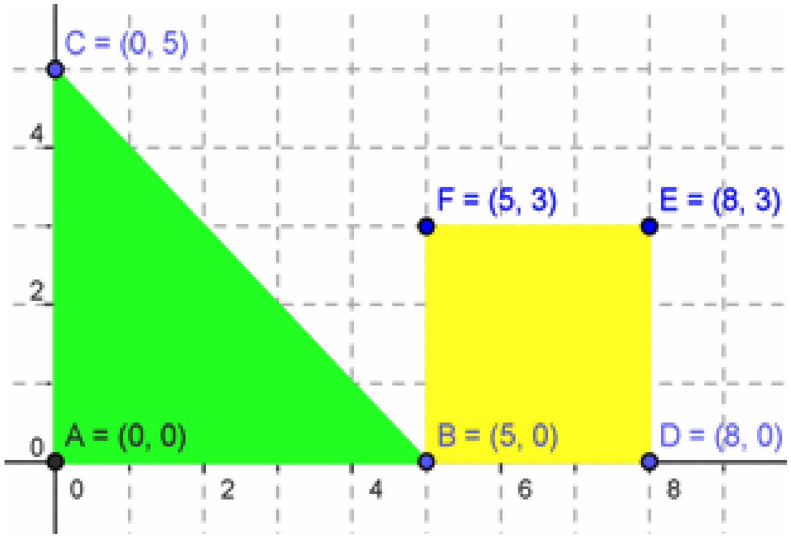
Minkowski Sum Algorithm

- Calculating the Minkowski sub-sum of the convex sub-polygons
- $A \oplus B = \{a + b \mid a \in A, b \in B\}$
- Two triangles:
  - A = { (1, 0), (0, 1), (0, −1)}
  - B = { (0, 0), (1, 1), (1, −1)}
- Result:
  - A + B = { (1, 0), (2, 1), (2, −1), (0, 1), (1, 2), (1, 0), (0, −1), (1, 0), (1, −2)}

Minkowski Sum Algorithm

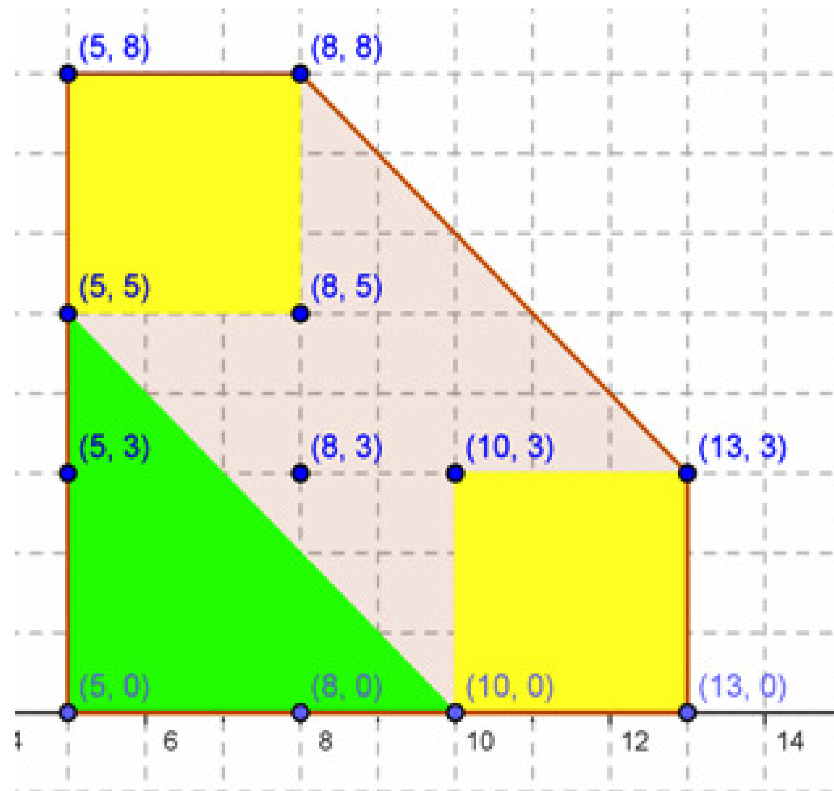


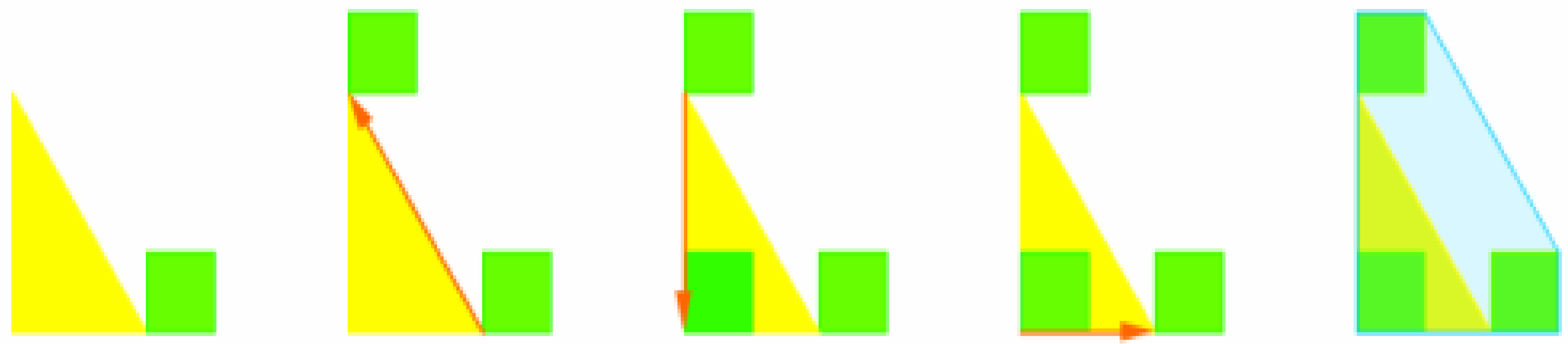
Adapted by wiki



Algebraic: Summing the vertices (+ convex hull)

A+B=(5,0), B+B=(10,0), C+B=(5,5), A+D=(8,0), B+D=(13,0), C+D=(8,5), A+E=(8,3), B+E=(13,3), C+E=(8,8), A+F=(5,3), B+F=(10,3), C+F=(5,8)
adapted by Korcz





Outline the sets (adapted by Korcz)

Minkowski Sum Algorithm

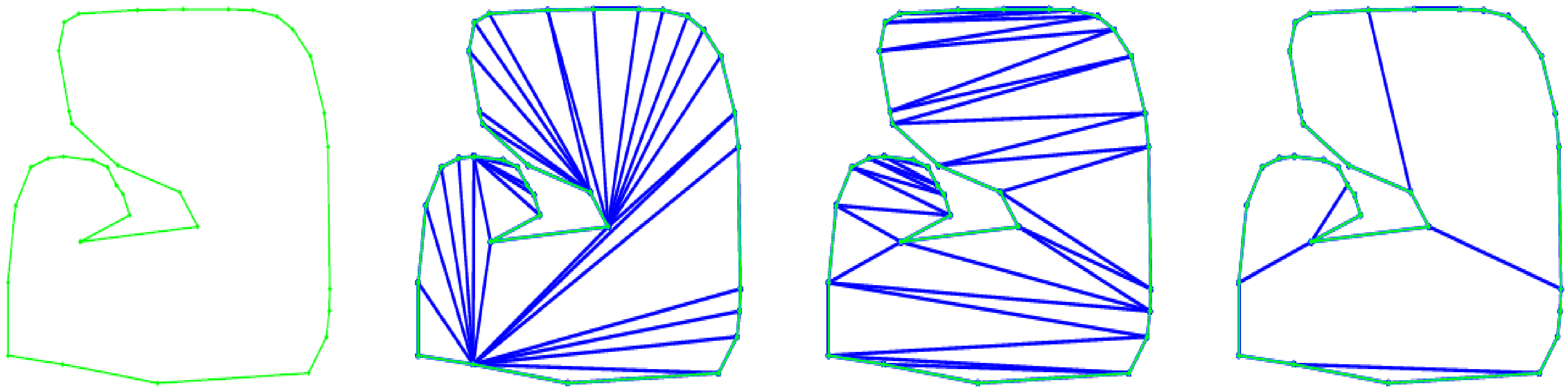
There are several possibilities for step 3:

- Arrangement algorithm
  - Construction of the arrangement takes $O(l + k \log k)$
  - Traversal stage takes $O(l + k)$ time

  k: the overall number of edges of the polygons in *R*
  l: the overall number of intersections between edges of polygons in *R*

- Incremental union algorithm
  - $O(k^2 \log^2 k)$
- Divide and Conquer Algorithm
  - Combination of above algorithms

Running time



	P's decomposition		
	naïve triang.	min Σd_i^2 triang.	min convex
Σd_i^2	754	530	192
# of convex subpolygons in P	33	33	6
time (mSec) to compute $P \oplus Q$	2133	1603	120

Figure 5: Different decomposition methods applied to the polygon P (leftmost in the figure), from left to right: naïve triangulation, minimum Σd_i^2 triangulation and minimum convex decomposition The table illustrates, for each decomposition, the sum of squares of degrees, the number of convex subpolygons, and the time in milliseconds to compute the Minkowski sum of P and a convex polygon, Q, with 4 vertices.

How to calculate the Minkowski Sum ?

We will check two methods here:

1. Decomposing into convex sub-polygons
2. Convolution method

Convolution method

- German word for convolution: Faltung
- geometric convolution

Main Idea:

- Calculating the convolution of the boundaries of *P* and *Q*

Convolution

Concept of convolutions of general planar tracings by Guibas:

- Polygonal tracings by interleaved *moves* and *turns*
  - *Move:* translation in a fixed direction
  - *Turn:* rotation at a fixed location

Convolution

- *P, Q* with vertices $(p_{0,\ldots}, p_{m-1})$ and $(q_{0,\ldots}, q_{n-1})$
- Move: traverse a polygon-edge $\overrightarrow{p_{i_o} p_{i_o+1}}$
- Turn: rotate a polygon vertex p_i from $\overrightarrow{p_{i-1} p_i}$ to $\overrightarrow{p_i p_{i+1}}$

- The polygons are counter-clockwise oriented in this assumption

Convolution

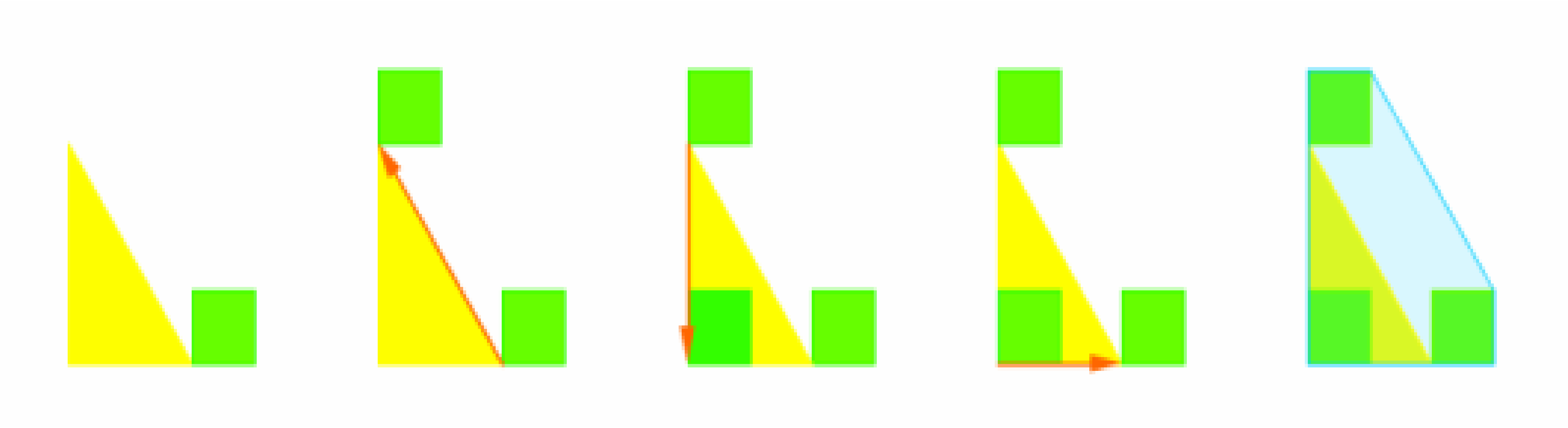
Convolution $P*Q$

- Collection of line segments $\overrightarrow{(p_i+q_j)(p_{i+1}+q_j)}$ who's vector $\overrightarrow{p_i p_{i+1}}$ lies between $\overrightarrow{q_{j-1} q_j}$ and $\overrightarrow{q_j q_{j+1}}$

and

- Collection of line segments $\overrightarrow{(p_i+q_j)(p_i+q_{j+1})}$ who's vector $\overrightarrow{q_j q_{j+1}}$ lies between $\overrightarrow{p_{i-1} p_i}$ and $\overrightarrow{p_i p_{i+1}}$

- $P*Q$ contains at most O(mn) line segments



Outline the sets (adapted by Korcz)

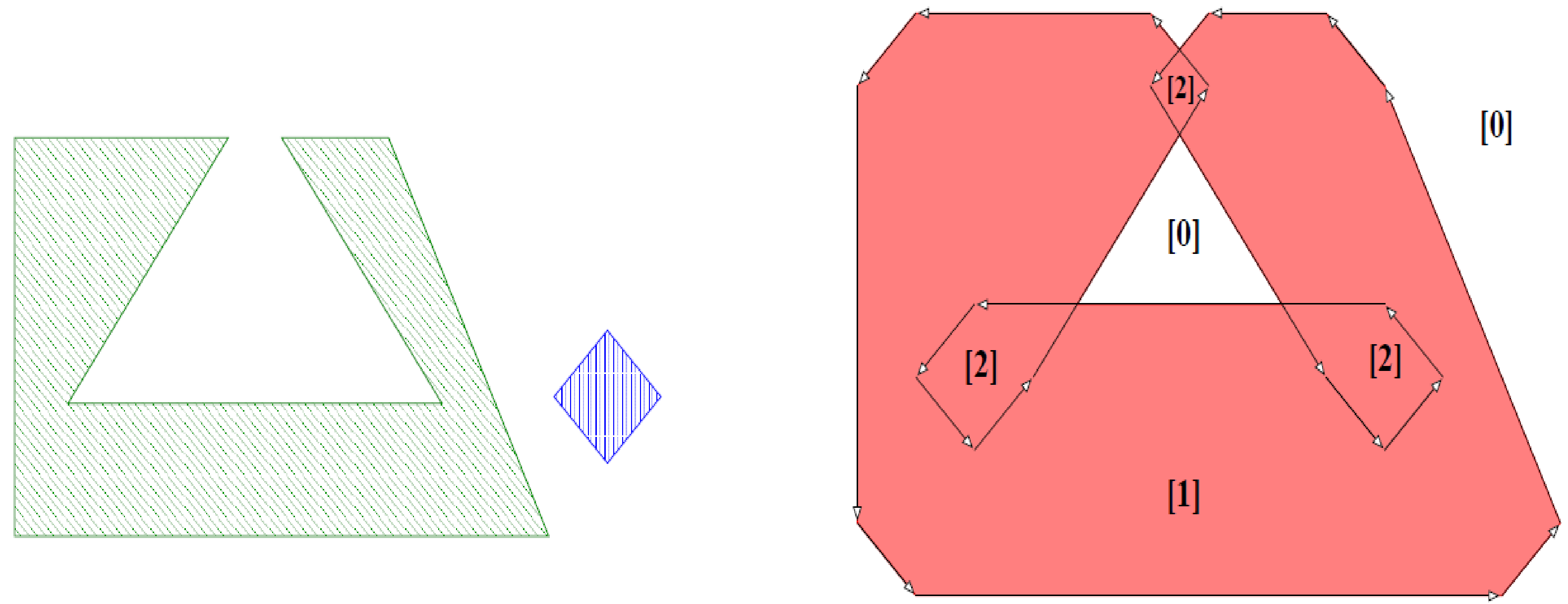


Figure 1: Computing the convolution of a convex polygon and a non-convex polygon (left). The convolution consists of a single self-intersecting cycle, drawn as a sequence of arrows (right). The winding number associated with each face of the arrangement induced by the segments forming the cycle appears in brackets. The Minkowski sum of the two polygons is shaded.

Adapted by Wein (!)

Convolution cycles

- The segments of the convolutions form a number of closed polygonal curves [Wein]

  → convolution cycles
- Three cases:
  - Both polygons where convex → convolution is a polygonal tracing → one cycle, non-intersection
  - One were not convex → convolution still contains a single cycle (maybe not simple) -> one cycle + intersection
  - Both are not convex → convolution could be comprised of several cycles → n cycles + x

Winding number

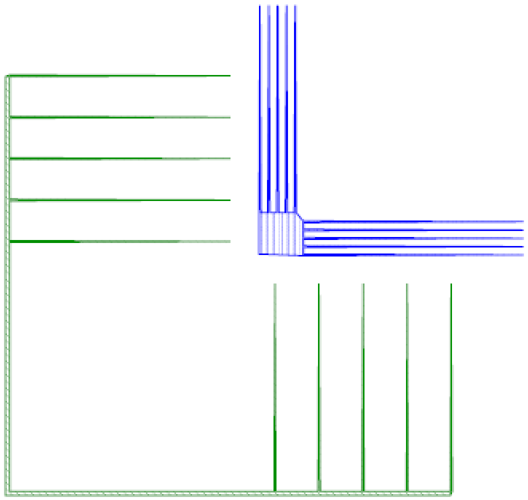
- non-negative
- Counting how often the convolution curve winds in a counter-clockwise direction around the geometrical face

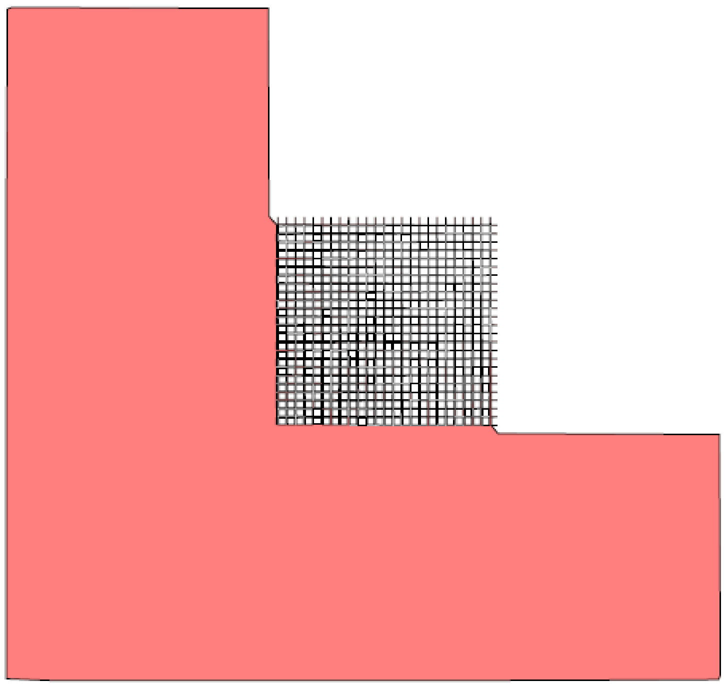
minus

- Counting how often the convolution curve winds in a clockwise direction around the geometrical face
- Maximum {above difference | 0}

Convolution method

- The Minkowski sum $P \oplus Q$ is the set of points having a non-zero winding number with respect to the convolution cycles [Wein]
- Experiments showed, that the convolution method is superior to decomposition on almost cases
- Running times improved by a factor 2-5





Fork example (adapted by Wein)

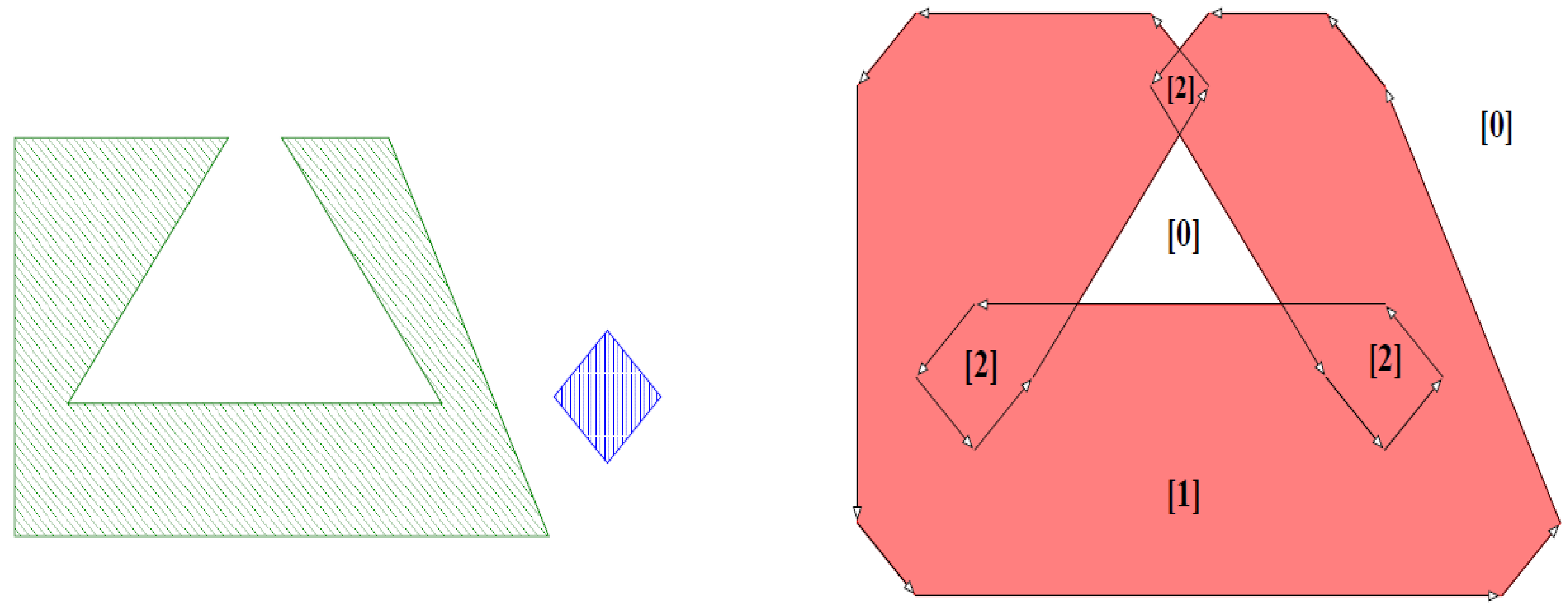


Figure 1: Computing the convolution of a convex polygon and a non-convex polygon (left). The convolution consists of a single self-intersecting cycle, drawn as a sequence of arrows (right). The winding number associated with each face of the arrangement induced by the segments forming the cycle appears in brackets. The Minkowski sum of the two polygons is shaded.

Adapted by Wein

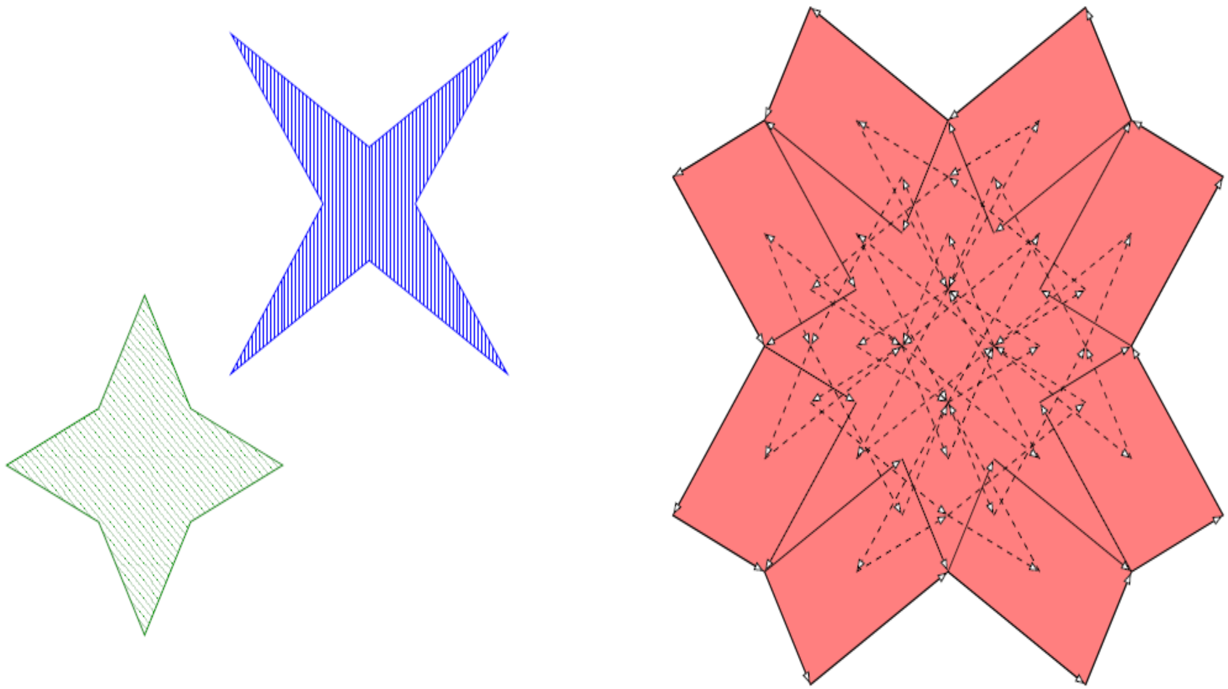


Figure 2: Computing the convolution of two non-convex octagons (left). The convolution consists of two cycles (right), one (solid arrows) is comprised of 32 line segments while the other (dashed arrows) contains 48 line segments, non of which lies on the boundary of the Minkowski sum (shaded).

Adapted by Wein

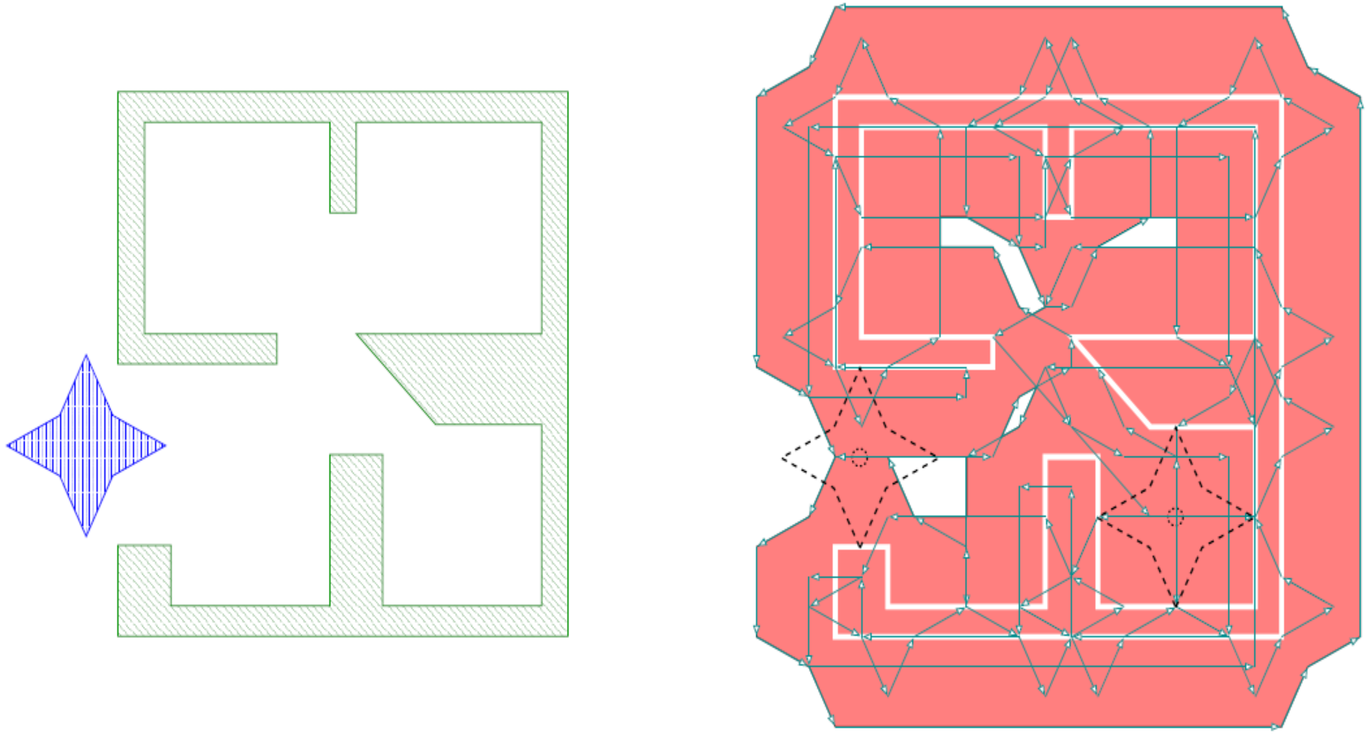


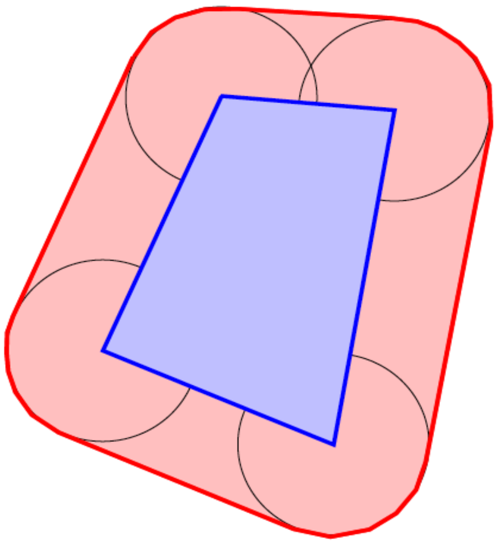
Figure 3: A house plan and a star-shaped polygon (left). The Minkowski sum of the two polygons (right) consists of an antenna and an isolated vertex. For clarity, two copies of the star are drawn using a dashed line with their center positioned on these features.

Adapted by Wein

Offsets of polygons

What is an offset?

- Given a set $A \subseteq \mathbb{R}^2$ the *r*-offset is a super-set of *A:* $\text{offset}(A, r) = \{p \in \mathbb{R}^2 \mid d(p, A) \leqslant r\} = A \oplus D_r$
  with Minkowski sum $A \oplus B = \{a + b \mid a \in A, b \in B\}$
  and disk $D_r = \{p \in \mathbb{R}^2 \mid d(\vec{O}, p) \leqslant r\}$



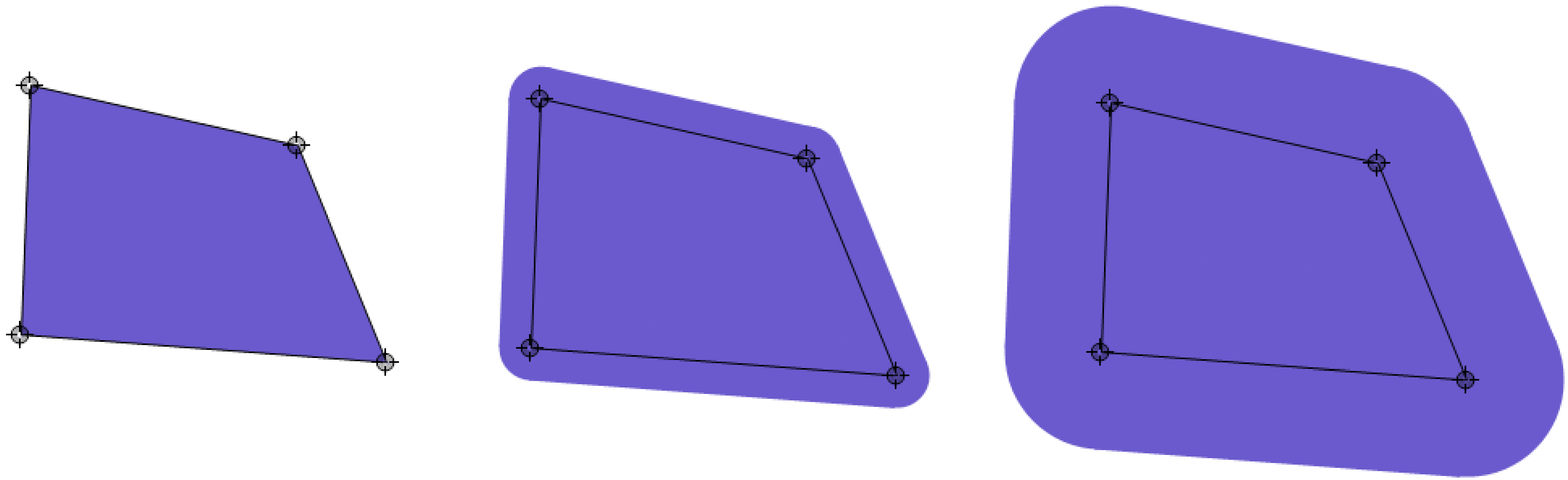
Offset polygons

- Fundamental task in CAM/CAD

  Idea:

- Construction of the Minkowski sum of a polygon with a disc
- For calculating the Minkowski sums one could use both seen methods; Wein chooses the convolution method

Offset polygons



Construction of the Minkowski sum of a polygon with a disc with different radii (created with math.player)

Complexity

- Minkowski sum of two polygonal sets could be combinatorially complex
- Complexity of the Minkowski sum of a polygon with n vertices with a disc is always O(n).
  - Circles are always convex
  - Complexity is caused by polygon
- Difficulty in offsetting polygons is not combinatorial, it is numerical, therefore
  - Doing it exactly or
  - Doing it with an approximation (better)

Offsetting a polygon

- polygon P with n vertices $(p_0, \ldots, p_{n-1})$
- Ordered counter-clockwise around P's interior
- All vertices of P have rational coordinates
- Goal: computing the offset polygon P_r, the Minkowski sum of P with a disc of radius r, r is rational
- Can be done for example by arrangement package in CGAL

Offsetting a polygon

P is a convex polygon:

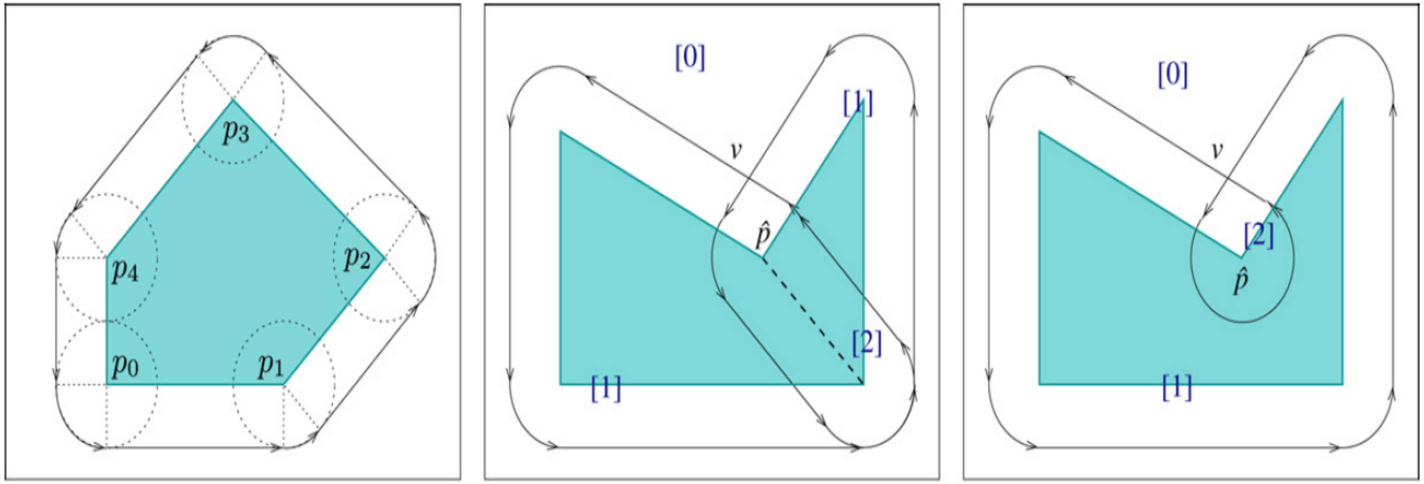
1. Computing the offset by shifting each polygonal edge by r away from the polygon
2. Results in a collection of n disconnected offset edges, each pair of adjacent offset edges is connected by circular arc of radius *r*, whose supporting circle is centred at p_i

- Running time linear in the size of the polygon

Offsetting a polygon

P is a non-convex polygon:

- Done by decomposing into convex sub-polygons $P_1, \ldots, P_m$
- Computing offset of each sub-polygonal
- Calculating the union of these offsets
- <u>Better</u>: using convolution, only one convolution cycle is needed there → segments + arcs



(a) (b) (c)

Fig. 1. (a) Offsetting a convex polygon. (b) Computing the offset of a non-convex polygon by decomposing it into convex sub-polygons by adding the dashed diagonal; $\hat{p}$ is a reflex vertex. The boundary curves of the two sub-offsets induce an arrangement with four faces, whose *cover numbers* are shown in brackets. (c) Offsetting a non-convex polygon by computing its convolution with a disc. The convolution cycle induces an arrangement with three faces, whose *winding numbers* are shown in brackets.

Adapted by Wein2

Exact representation of the offset edges

- Convolution cycle formed by line segments and circular arcs
- All circular arcs are supported by rational circles, as their centre points (polygonal vertices) always have rational coordinates and their radii equal $r \in \mathbb{Q}$ [Wein2]
- Problem: the coordinates of the vertices of the offset of a rational polygonal set by a rational radius r are in general irrational [Wein2]

Problem

- To get the coordinates of the new segment points quadratic equations with rational coefficients are solved
- But the new segment between these points is supported by a line of irrational coefficients
- If the supporting line of points $p_1 p_2$ is $ax+by+c=0$ where $a,b,c \in \mathbb{Q}$, then the line supporting $p_1 p_2$ is $ax+by+(c+lr)=0$ where l is an irrational number

Problem

- Offset edges can not be realised as segments of lines with rational coefficients
- Not representable by rational circles and segments

Conic curve

- Another more simple representation of offset edges
- Based on the fact that the locus of all points lying at distance *r* from the line ax+by+c = 0

$$\frac{(ax+by+c)^2}{a^2+b^2}=r^2$$

the offset polygon

Problem:

- Exact computation leads to computational overhead

Remedy:

- Staying in exact rational arithmetic with rational lines, circles and arcs by using one-root numbers
- Using an algorithm which only uses rational arithmetic

one-root number

The solution of $ax^2+bx+c=0$, with $a, b, c \in \mathbb{Q}, c \geq 0$ is now a one-root number
$\alpha+\beta\sqrt{\gamma}$, with $\alpha, \beta, \gamma \in \mathbb{Q}, \gamma \geq 0$

- Ability to compare two such numbers in an exact manner
- Important property: operations of evaluating the sign of a one-root number and comparing two one-root numbers can be carried out precisely using only exact rational arithmetic [Wein2]

one-root number

With properties of one-root numbers:

- Robust implementation possible
- Geometric predicates and constructions needed for the arrangement construction and maintenance are using only exactly rational arithmetic

Remedy

- Approximation algorithm that avoids using expensive computation with algebraic numbers

Approximation scheme

- for a horizontal edge $(y_1 = y_2)$ or a vertical edge $(x_1 = x_2)$ its length *l* is a rational number [Wein2]
- Construction of the offset edge possible in exact manner
- Still left: $(y_1 \neq y_2)$ and $(x_1 \neq x_2)$

Approximation scheme

- Approximating the offset edge by two line segments by finding two points v'_1 and v'_2 with rational coefficients
- v'_j shall lie on the circle $(x-x_j)^2+(y-y_j)^2=r^2$ for j = 1, 2
- To accomplish this we are „pushing the roof"

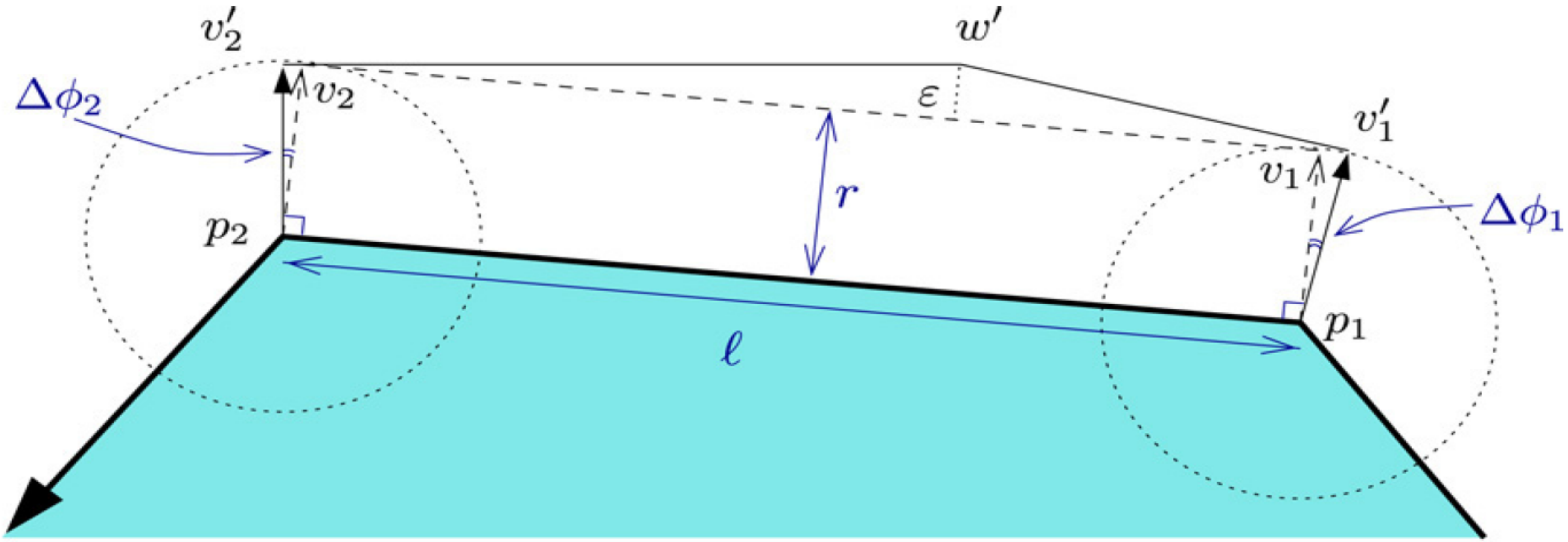


Fig. 2. Approximating the offset edge induced by the polygon edge $p_1 p_2$.

Adapted by Wein2

Summary

- Minkowski Sums
- Decomposing and Convolution
- Convolution also usable in offset polygons
- Exact representation
- Approximation scheme

Content based on

- [Agarwal] Polygonal Decomposition for Efficient Construction of Minkowski Sums
- [Flato] Robust and Efficient Construction of Planar Minkowski Sums
- [Wein] Exact and Efficient Construction of Planar Minkowski Sums using the Convolution Method
- [Wein2] Exact and approximate construction of offset polygons
- [LaValle] Planning Algorithms
- [Pallaschke] Bruchrechnung mit konvexen Mengen
- [Korcz] Visualisierung der Rechnungen auf konvexen Mengen
- Few hints from my advisor and wikipedia