SL / SCR Paper

Theodor Mario Henneken

13.08.2021

Expand Convex Polygon inside Non-Convex Polygon

Algorithm to find biggest convex polygon (given a start polygon) inside a non-convex polygon with one hole (e.g., a racetrack)

Introduction

- Goal: expand polygons from track tesselation into biggest convex form, adhering to track limits
- Achieved by Algorithm to the right side
  - Using Algorithm *extend_convex_polygon_into_direction*, which is defined and exemplarily visualized in the following

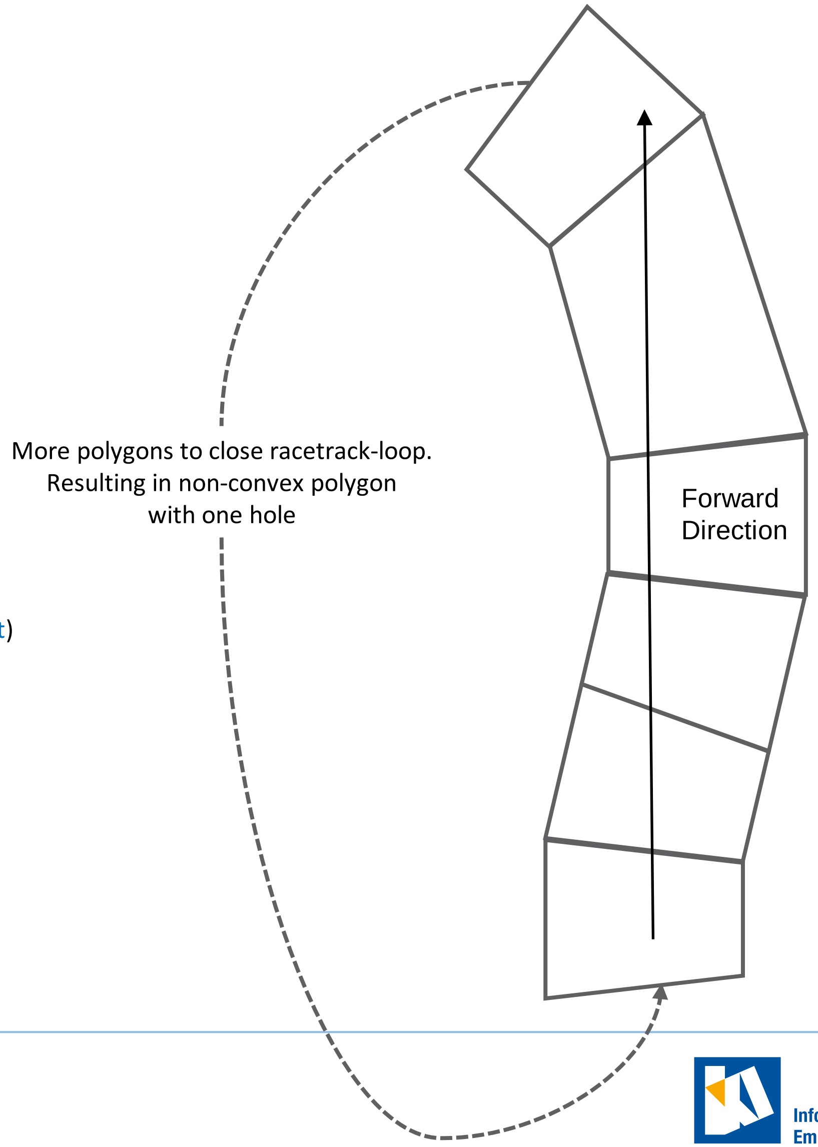
```
function polygons_extended = extend_convex_polygons(polygons)
      polygons (struct): all track polygons
1   for i_p in polygons
1a    track_polygons_extended_forward(i_p) = ...
          extend_convex_polygon_into_direction(polygons , i_p, forward)
1b  track_polygons_extended_backward(i_p) = ...
          extend_convex_polygon_into_direction(polygons, i_p, backward)
2   end
3   for i_p in polygons
3a    polygons_extended(i_p) = ...
          union(track_polygons_extended_forward(i_p), track_polygons_extended_backward(i_p))
4   end
```

Overview

▭ Polygon outline

- Abbreviation: poly = polygon

```
function poly_curr_ext  = extend_convex_polygon_into_direction(polygons, i_curr, direction)
    polygons (struct): all tesselated track polygons (will be used by functions inside this function)
    i_curr (int): index of current polygon
    direction (bool): forward (true) or backward (false)
0 poly_curr = get_current_poly(i_curr)
1 poly_curr_ext = poly_curr.extend_poly_to_next(direction)
2 i_next = get_next_polygon_index(i_curr, direction)
3 while true
3a  poly_next = get_current_poly(i_next)
3b  if poly_curr_ext .overlaps(poly_next)
3b1        poly_next_ext = poly_next.extend_poly_to_next(direction)
3b2        poly_overlap_curr_n_next = poly_curr_ext.intersect(poly_next_ext)
3b3        polys_differences_curr_n_next = poly_curr_ext.subtract(poly_overlap_curr_n_next)
3b4        poly_curr_ext_retain = poly_curr.get_poly_overlapping(polys_differences_curr_n_next)
3b5        poly_curr_ext = poly_curr_ext_retain.union(poly_overlap_curr_n_next)
3c  else
3c1        break
3d  end
3e  i_next = get_next_polygon_index(i_curr, direction)
```

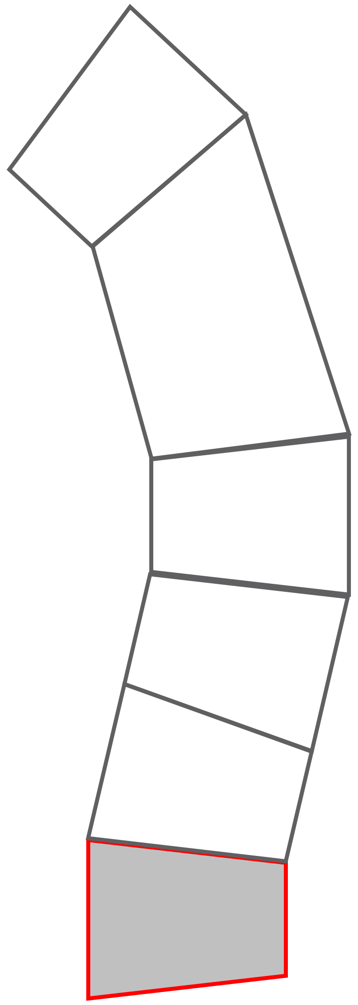

More polygons to close racetrack-loop.
Resulting in non-convex polygon
with one hole
Forward
Direction

Algorithm Run
Preparation

▨ Current step and polygon

- Abbreviation: poly = polygon

```
function poly_curr_ext  = extend_convex_polygon_into_direction(polygons, i_curr, direction)
    polygons (struct): all tesselated track polygons (will be used by functions inside this function)
    i_curr (int): index of current polygon
    direction (bool): forward (true) or backward (false)
0 poly_curr = get_current_poly(i_curr)
1 poly_curr_ext = poly_curr.extend_poly_to_next(direction)
2 i_next = get_next_polygon_index(i_curr, direction)
3 while true
3a  poly_next = get_current_poly(i_next)
3b  if poly_curr_ext .overlaps(poly_next)
3b1         poly_next_ext = poly_next.extend_poly_to_next(direction)
3b2         poly_overlap_curr_n_next = poly_curr_ext.intersect(poly_next_ext)
3b3         polys_differences_curr_n_next = poly_curr_ext.subtract(poly_overlap_curr_n_next)
3b4         poly_curr_ext_retain = poly_curr.get_poly_overlapping(polys_differences_curr_n_next)
3b5         poly_curr_ext = poly_curr_ext_retain.union(poly_overlap_curr_n_next)
3c  else
3c1         break
3d  end
3e  i_next = get_next_polygon_index(i_curr, direction)
```



Informatik 11 Embedded Software RWTH AACHEN UNIVERSITY

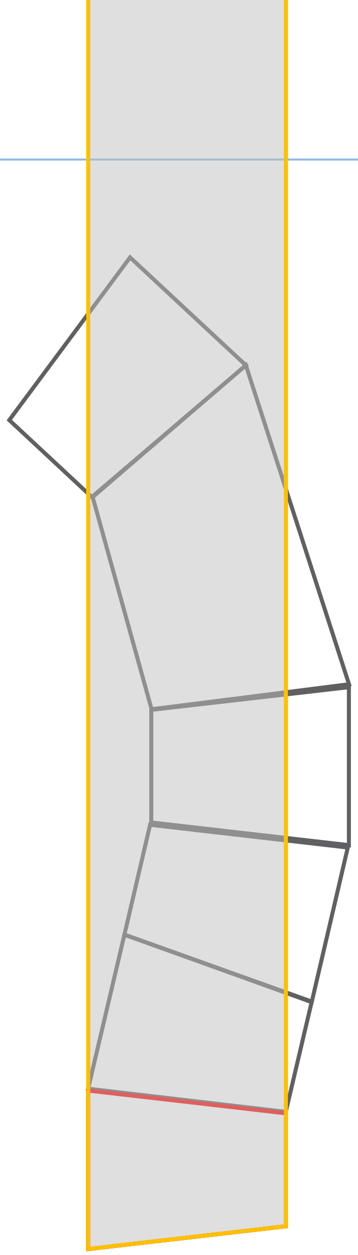
Algorithm Run

Preparation

▨ Current step and polygon

- Abbreviation: poly = polygon

```
function poly_curr_ext  = extend_convex_polygon_into_direction(i_curr, direction)
    i_curr (int): index of current polygon
    direction (bool): forward (true) or backward (false)
0 poly_curr = get_current_poly(i_curr)
1 poly_curr_ext = poly_curr.extend_poly_to_next(direction)
2 i_next = get_next_polygon_index(i_curr, direction)
3 while true
3a  poly_next = get_current_poly(i_next)
3b  if poly_curr_ext .overlaps(poly_next)
3b1         poly_next_ext = poly_next.extend_poly_to_next(direction)
3b2         poly_overlap_curr_n_next = poly_curr_ext.intersect(poly_next_ext)
3b3         polys_differences_curr_n_next = poly_curr_ext.subtract(poly_overlap_curr_n_next)
3b4         poly_curr_ext_retain = poly_curr.get_poly_overlapping(polys_differences_curr_n_next)
3b5         poly_curr_ext = poly_curr_ext_retain.union(poly_overlap_curr_n_next)
3c  else
3c1         break
3d  end
3e  i_next = get_next_polygon_index(i_curr, direction)
```



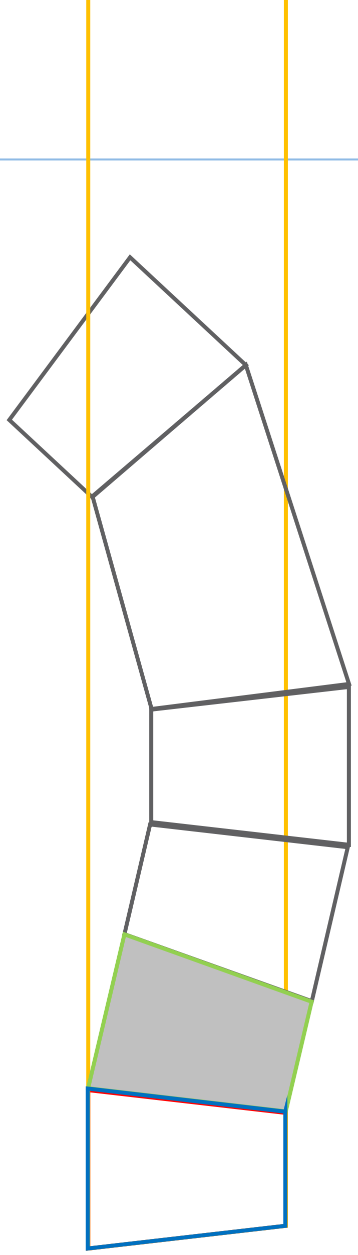
Iteration i_next = 1

Algorithm Run
Iteration i_next = 1

▨ Current step and polygon

- Abbreviation: poly = polygon

```
function poly_curr_ext  = extend_convex_polygon_into_direction(i_curr, direction)
    i_curr (int): index of current polygon
    direction (bool): forward (true) or backward (false)
0 poly_curr = get_current_poly(i_curr)
1 poly_curr_ext = poly_curr.extend_poly_to_next(direction)
2 i_next = get_next_polygon_index(i_curr, direction)
3 while true
3a poly_next = get_current_poly(i_next)
3b if poly_curr_ext .overlaps(poly_next)
3b1     poly_next_ext = poly_next.extend_poly_to_next(direction)
3b2     poly_overlap_curr_n_next = poly_curr_ext.intersect(poly_next_ext)
3b3     polys_differences_curr_n_next = poly_curr_ext.subtract(poly_overlap_curr_n_next)
3b4     poly_curr_ext_retain = poly_curr.get_poly_overlapping(polys_differences_curr_n_next)
3b5     poly_curr_ext = poly_curr_ext_retain.union(poly_overlap_curr_n_next)
3c else
3c1     break
3d end
3e i_next = get_next_polygon_index(i_curr, direction)
```



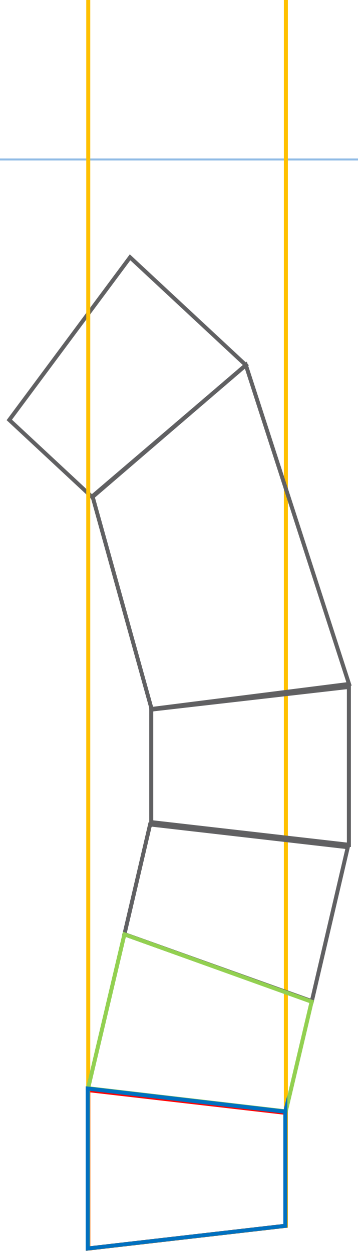
Algorithm Run

Iteration i_next = 1

▨ Current step and polygon

- Abbreviation: poly = polygon

```
function poly_curr_ext = extend_convex_polygon_into_direction(i_curr, direction)
    i_curr (int): index of current polygon
    direction (bool): forward (true) or backward (false)
0 poly_curr = get_current_poly(i_curr)
1 poly_curr_ext = poly_curr.extend_poly_to_next(direction)
2 i_next = get_next_polygon_index(i_curr, direction)
3 while true
3a  poly_next = get_current_poly(i_next)
3b  if poly_curr_ext .overlaps(poly_next)
3b1         poly_next_ext = poly_next.extend_poly_to_next(direction)
3b2         poly_overlap_curr_n_next = poly_curr_ext.intersect(poly_next_ext)
3b3         polys_differences_curr_n_next = poly_curr_ext.subtract(poly_overlap_curr_n_next)
3b4         poly_curr_ext_retain = poly_curr.get_poly_overlapping(polys_differences_curr_n_next)
3b5         poly_curr_ext = poly_curr_ext_retain.union(poly_overlap_curr_n_next)
3c  else
3c1         break
3d  end
3e  i_next = get_next_polygon_index(i_curr, direction)
```



Informatik 11 Embedded Software

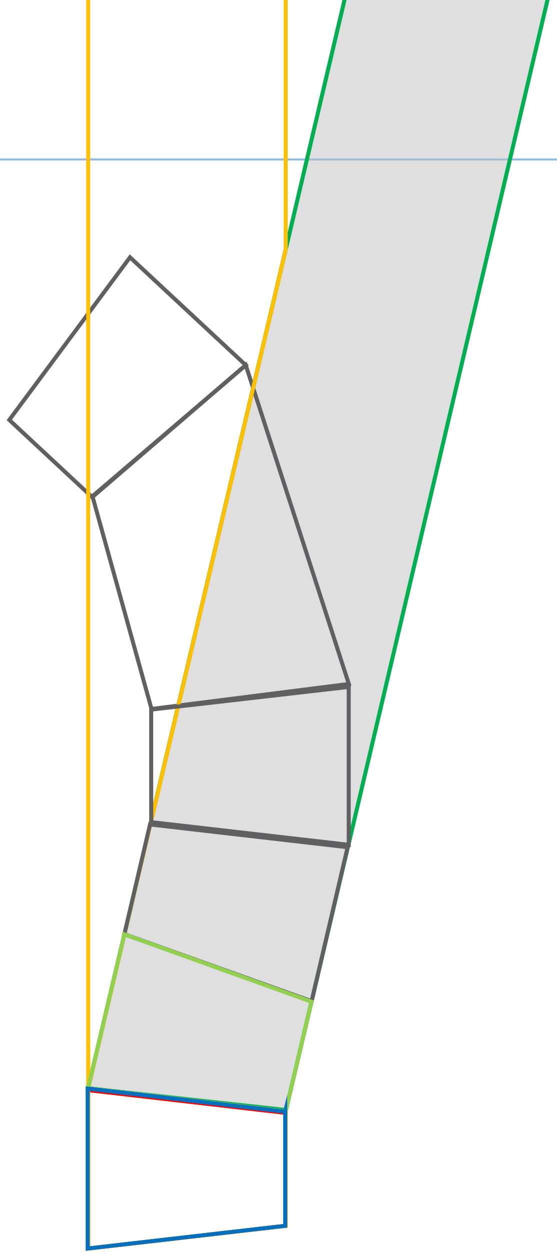
RWTH AACHEN UNIVERSITY

Algorithm Run
Iteration i_next = 1

▨ Current step and polygon

- Abbreviation: poly = polygon

```
function poly_curr_ext  = extend_convex_polygon_into_direction(i_curr, direction)
    i_curr (int): index of current polygon
    direction (bool): forward (true) or backward (false)
0 poly_curr = get_current_poly(i_curr)
1 poly_curr_ext = poly_curr.extend_poly_to_next(direction)
2 i_next = get_next_polygon_index(i_curr, direction)
3 while true
3a  poly_next = get_current_poly(i_next)
3b  if poly_curr_ext .overlaps(poly_next)
3b1         poly_next_ext = poly_next.extend_poly_to_next(direction)
3b2         poly_overlap_curr_n_next = poly_curr_ext.intersect(poly_next_ext)
3b3         polys_differences_curr_n_next = poly_curr_ext.subtract(poly_overlap_curr_n_next)
3b4         poly_curr_ext_retain = poly_curr.get_poly_overlapping(polys_differences_curr_n_next)
3b5         poly_curr_ext = poly_curr_ext_retain.union(poly_overlap_curr_n_next)
3c  else
3c1         break
3d  end
3e  i_next = get_next_polygon_index(i_curr, direction)
```



Informatik 11 Embedded Software RWTH AACHEN UNIVERSITY

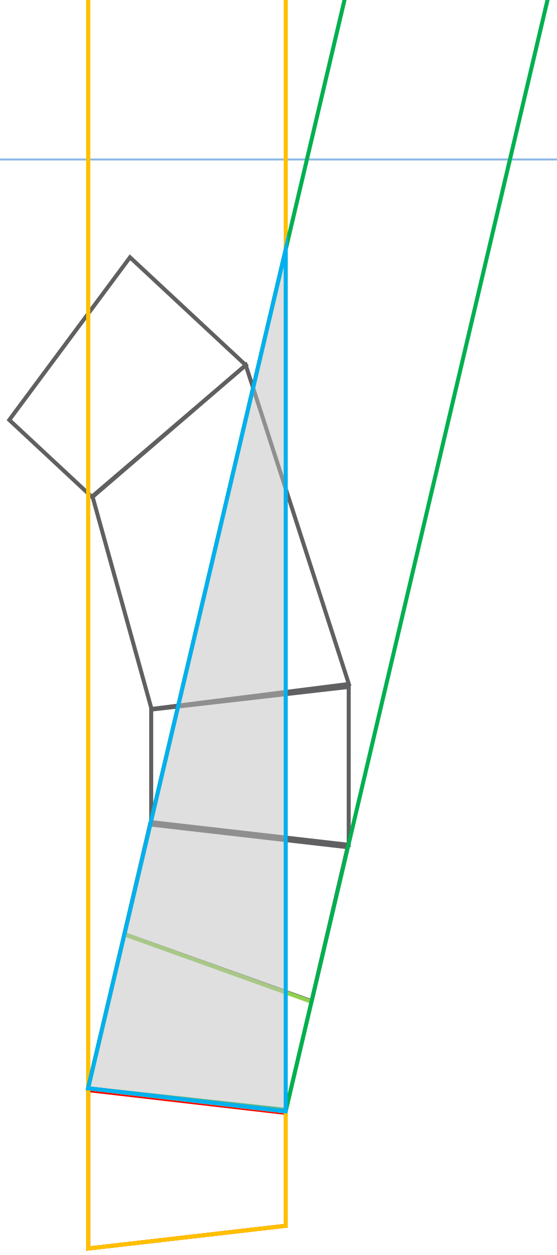
Algorithm Run
Iteration i_next = 1

▨ Current step and polygon

- Abbreviation: poly = polygon

- function poly_curr_ext = extend_convex_polygon_into_direction(i_curr, direction)
  i_curr (int): index of current polygon
  direction (bool): forward (true) or backward (false)

```
0 poly_curr = get_current_poly(i_curr)
1 poly_curr_ext = poly_curr.extend_poly_to_next(direction)
2 i_next = get_next_polygon_index(i_curr, direction)
3 while true
3a  poly_next = get_current_poly(i_next)
3b  if poly_curr_ext .overlaps(poly_next)
3b1       poly_next_ext = poly_next.extend_poly_to_next(direction)
3b2       poly_overlap_curr_n_next = poly_curr_ext.intersect(poly_next_ext)
3b3       polys_differences_curr_n_next = poly_curr_ext.subtract(poly_overlap_curr_n_next)
3b4       poly_curr_ext_retain = poly_curr.get_poly_overlapping(polys_differences_curr_n_next)
3b5       poly_curr_ext = poly_curr_ext_retain.union(poly_overlap_curr_n_next)
3c  else
3c1       break
3d  end
3e  i_next = get_next_polygon_index(i_curr, direction)
```



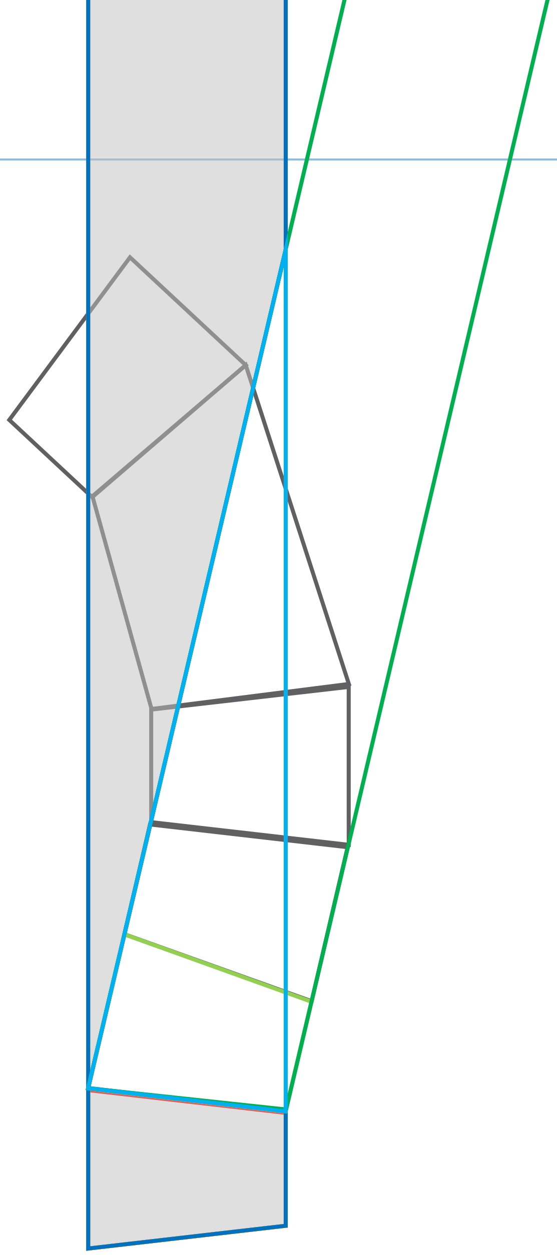
Informatik 11 Embedded Software | RWTH AACHEN UNIVERSITY

Algorithm Run
Iteration i_next = 1

▨ Current step and polygon

- Abbreviation: poly = polygon

```
function poly_curr_ext = extend_convex_polygon_into_direction(i_curr, direction)
    i_curr (int): index of current polygon
    direction (bool): forward (true) or backward (false)
0 poly_curr = get_current_poly(i_curr)
1 poly_curr_ext = poly_curr.extend_poly_to_next(direction)
2 i_next = get_next_polygon_index(i_curr, direction)
3 while true
3a  poly_next = get_current_poly(i_next)
3b  if poly_curr_ext .overlaps(poly_next)
3b1       poly_next_ext = poly_next.extend_poly_to_next(direction)
3b2       poly_overlap_curr_n_next = poly_curr_ext.intersect(poly_next_ext)
3b3       polys_differences_curr_n_next = poly_curr_ext.subtract(poly_overlap_curr_n_next)
3b4       poly_curr_ext_retain = poly_curr.get_poly_overlapping(polys_differences_curr_n_next)
3b5       poly_curr_ext = poly_curr_ext_retain.union(poly_overlap_curr_n_next)
3c  else
3c1       break
3d  end
3e  i_next = get_next_polygon_index(i_curr, direction)
```

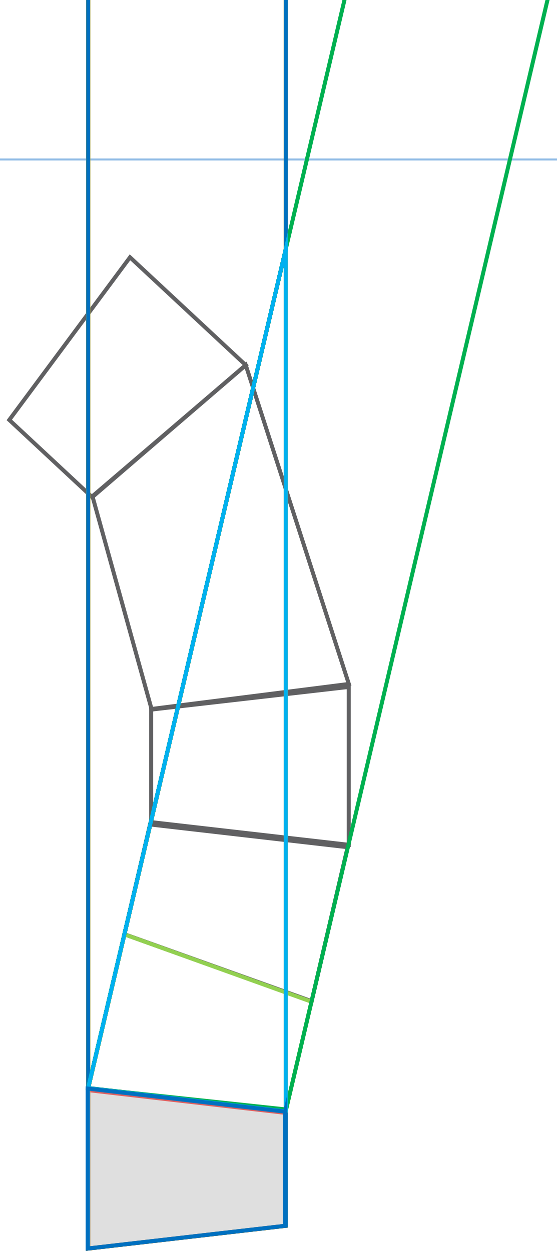


Algorithm Run
Iteration i_next = 1

▨ Current step and polygon

- Abbreviation: poly = polygon

```
function poly_curr_ext  = extend_convex_polygon_into_direction(i_curr, direction)
    i_curr (int): index of current polygon
    direction (bool): forward (true) or backward (false)
0 poly_curr = get_current_poly(i_curr)
1 poly_curr_ext = poly_curr.extend_poly_to_next(direction)
2 i_next = get_next_polygon_index(i_curr, direction)
3 while true
3a  poly_next = get_current_poly(i_next)
3b  if poly_curr_ext .overlaps(poly_next)
3b1       poly_next_ext = poly_next.extend_poly_to_next(direction)
3b2       poly_overlap_curr_n_next = poly_curr_ext.intersect(poly_next_ext)
3b3       polys_differences_curr_n_next = poly_curr_ext.subtract(poly_overlap_curr_n_next)
3b4       poly_curr_ext_retain = poly_curr.get_poly_overlapping(polys_differences_curr_n_next)
3b5       poly_curr_ext = poly_curr_ext_retain.union(poly_overlap_curr_n_next)
3c  else
3c1       break
3d  end
3e  i_next = get_next_polygon_index(i_curr, direction)
```



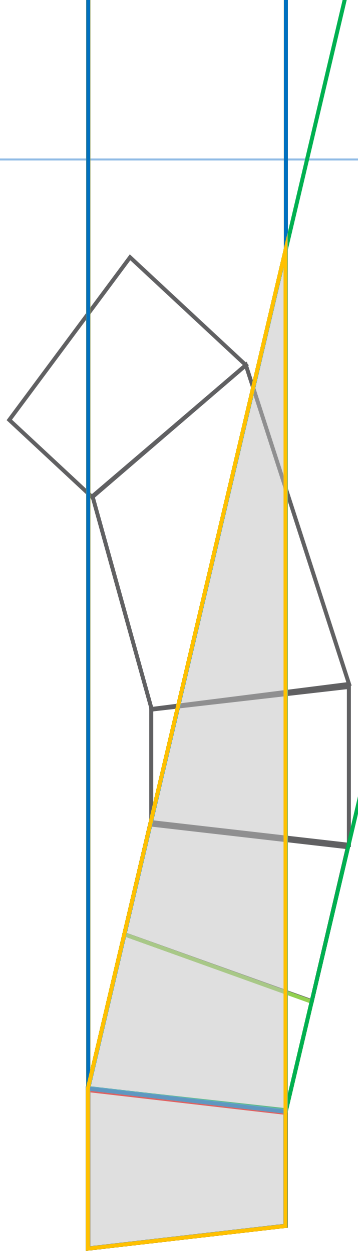
Informatik 11 Embedded Software
RWTH AACHEN UNIVERSITY

Algorithm Run
Iteration i_next = 1

▨ Current step and polygon

- Abbreviation: poly = polygon

```
function poly_curr_ext  = extend_convex_polygon_into_direction(i_curr, direction)
    i_curr (int): index of current polygon
    direction (bool): forward (true) or backward (false)
0 poly_curr = get_current_poly(i_curr)
1 poly_curr_ext = poly_curr.extend_poly_to_next(direction)
2 i_next = get_next_polygon_index(i_curr, direction)
3 while true
3a  poly_next = get_current_poly(i_next)
3b  if poly_curr_ext .overlaps(poly_next)
3b1       poly_next_ext = poly_next.extend_poly_to_next(direction)
3b2       poly_overlap_curr_n_next = poly_curr_ext.intersect(poly_next_ext)
3b3       polys_differences_curr_n_next = poly_curr_ext.subtract(poly_overlap_curr_n_next)
3b4       poly_curr_ext_retain = poly_curr.get_poly_overlapping(polys_differences_curr_n_next)
3b5       poly_curr_ext = poly_curr_ext_retain.union(poly_overlap_curr_n_next)
3c  else
3c1       break
3d  end
3e  i_next = get_next_polygon_index(i_curr, direction)
```



Algorithm Run
Iteration i_next = 1

▨ Current step and polygon

- Abbreviation: poly = polygon

```
function poly_curr_ext  = extend_convex_polygon_into_direction(i_curr, direction)
    i_curr (int): index of current polygon
    direction (bool): forward (true) or backward (false)
0 poly_curr = get_current_poly(i_curr)
1 poly_curr_ext = poly_curr.extend_poly_to_next(direction)
2 i_next = get_next_polygon_index(i_curr, direction)
3 while true
3a  poly_next = get_current_poly(i_next)
3b  if poly_curr_ext .overlaps(poly_next)
3b1       poly_next_ext = poly_next.extend_poly_to_next(direction)
3b2       poly_overlap_curr_n_next = poly_curr_ext.intersect(poly_next_ext)
3b3       polys_differences_curr_n_next = poly_curr_ext.subtract(poly_overlap_curr_n_next)
3b4       poly_curr_ext_retain = poly_curr.get_poly_overlapping(polys_differences_curr_n_next)
3b5       poly_curr_ext = poly_curr_ext_retain.union(poly_overlap_curr_n_next)
3c  else
3c1       break
3d  end
3e  i_next = get_next_polygon_index(i_curr, direction)
```

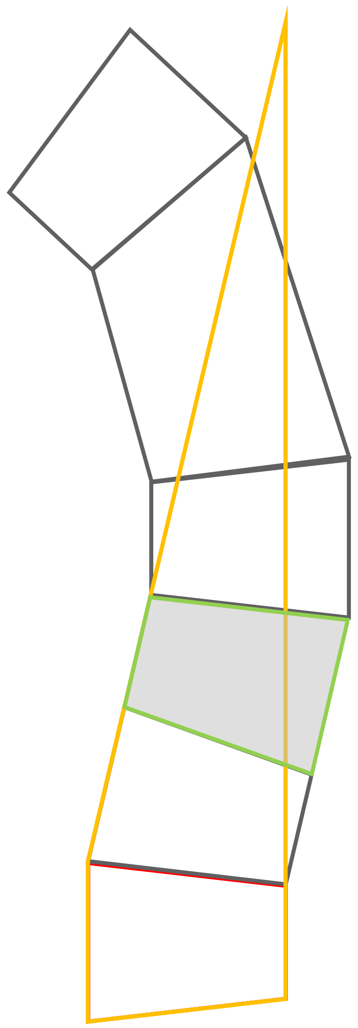
Iteration i_next = 2

Algorithm Run
Iteration i_next = 2

▨ Current step and polygon

- Abbreviation: poly = polygon

```
function poly_curr_ext = extend_convex_polygon_into_direction(i_curr, direction)
    i_curr (int): index of current polygon
    direction (bool): forward (true) or backward (false)
0 poly_curr = get_current_poly(i_curr)
1 poly_curr_ext = poly_curr.extend_poly_to_next(direction)
2 i_next = get_next_polygon_index(i_curr, direction)
3 while true
3a poly_next = get_current_poly(i_next)
3b if poly_curr_ext .overlaps(poly_next)
3b1     poly_next_ext = poly_next.extend_poly_to_next(direction)
3b2     poly_overlap_curr_n_next = poly_curr_ext.intersect(poly_next_ext)
3b3     polys_differences_curr_n_next = poly_curr_ext.subtract(poly_overlap_curr_n_next)
3b4     poly_curr_ext_retain = poly_curr.get_poly_overlapping(polys_differences_curr_n_next)
3b5     poly_curr_ext = poly_curr_ext_retain.union(poly_overlap_curr_n_next)
3c else
3c1     break
3d end
3e i_next = get_next_polygon_index(i_curr, direction)
```

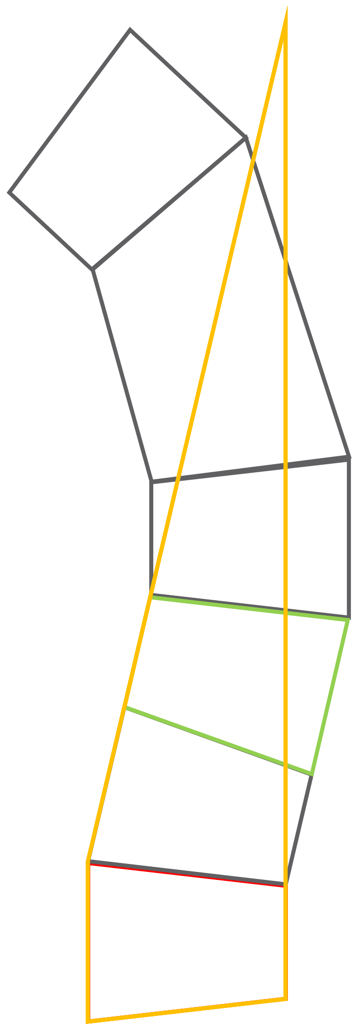


Algorithm Run
Iteration i_next = 2

▨ Current step and polygon

- Abbreviation: poly = polygon

```
function poly_curr_ext  = extend_convex_polygon_into_direction(i_curr, direction)
    i_curr (int): index of current polygon
    direction (bool): forward (true) or backward (false)
0 poly_curr = get_current_poly(i_curr)
1 poly_curr_ext = poly_curr.extend_poly_to_next(direction)
2 i_next = get_next_polygon_index(i_curr, direction)
3 while true
3a  poly_next = get_current_poly(i_next)
3b  if poly_curr_ext .overlaps(poly_next)
3b1       poly_next_ext = poly_next.extend_poly_to_next(direction)
3b2       poly_overlap_curr_n_next = poly_curr_ext.intersect(poly_next_ext)
3b3       polys_differences_curr_n_next = poly_curr_ext.subtract(poly_overlap_curr_n_next)
3b4       poly_curr_ext_retain = poly_curr.get_poly_overlapping(polys_differences_curr_n_next)
3b5       poly_curr_ext = poly_curr_ext_retain.union(poly_overlap_curr_n_next)
3c  else
3c1       break
3d  end
3e  i_next = get_next_polygon_index(i_curr, direction)
```

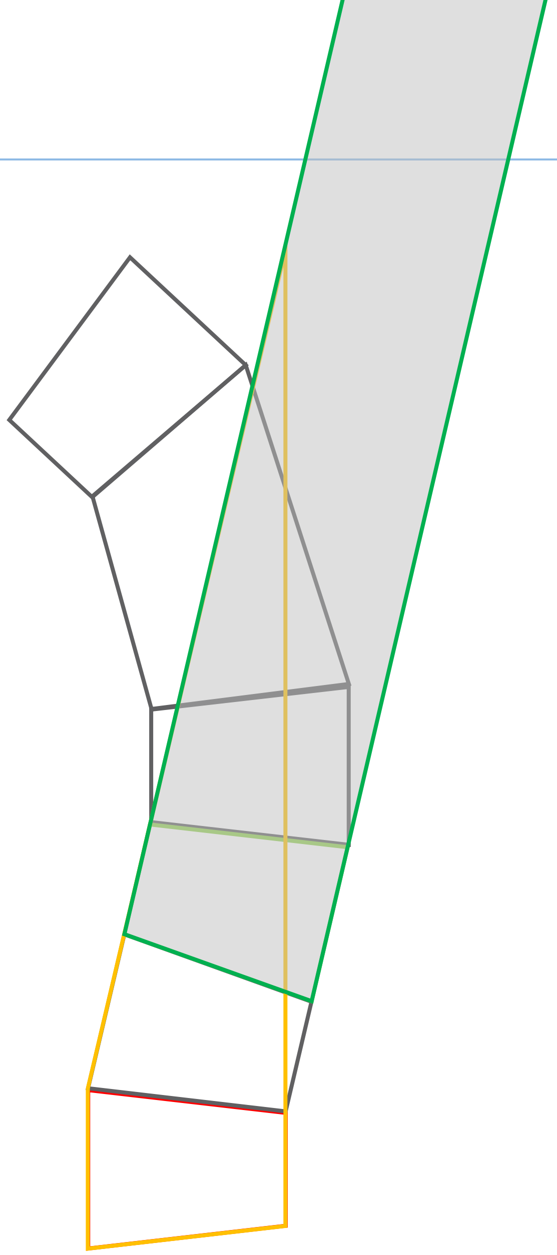


Algorithm Run
Iteration i_next = 2

▨ Current step and polygon

- Abbreviation: poly = polygon

```
function poly_curr_ext  = extend_convex_polygon_into_direction(i_curr, direction)
    i_curr (int): index of current polygon
    direction (bool): forward (true) or backward (false)
0 poly_curr = get_current_poly(i_curr)
1 poly_curr_ext = poly_curr.extend_poly_to_next(direction)
2 i_next = get_next_polygon_index(i_curr, direction)
3 while true
3a  poly_next = get_current_poly(i_next)
3b  if poly_curr_ext .overlaps(poly_next)
3b1       poly_next_ext = poly_next.extend_poly_to_next(direction)
3b2       poly_overlap_curr_n_next = poly_curr_ext.intersect(poly_next_ext)
3b3       polys_differences_curr_n_next = poly_curr_ext.subtract(poly_overlap_curr_n_next)
3b4       poly_curr_ext_retain = poly_curr.get_poly_overlapping(polys_differences_curr_n_next)
3b5       poly_curr_ext = poly_curr_ext_retain.union(poly_overlap_curr_n_next)
3c  else
3c1       break
3d  end
3e  i_next = get_next_polygon_index(i_curr, direction)
```



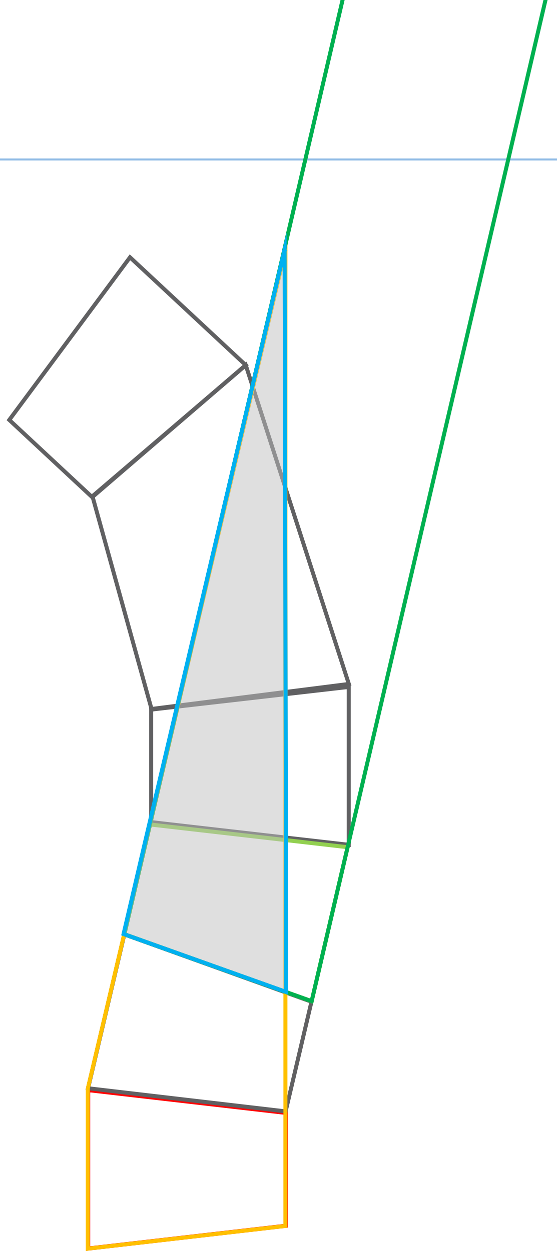
Informatik 11 Embedded Software
RWTH AACHEN UNIVERSITY

Algorithm Run
Iteration i_next = 2

▨ Current step and polygon

- Abbreviation: poly = polygon

```
function poly_curr_ext  = extend_convex_polygon_into_direction(i_curr, direction)
    i_curr (int): index of current polygon
    direction (bool): forward (true) or backward (false)
0 poly_curr = get_current_poly(i_curr)
1 poly_curr_ext = poly_curr.extend_poly_to_next(direction)
2 i_next = get_next_polygon_index(i_curr, direction)
3 while true
3a  poly_next = get_current_poly(i_next)
3b  if poly_curr_ext .overlaps(poly_next)
3b1        poly_next_ext = poly_next.extend_poly_to_next(direction)
3b2        poly_overlap_curr_n_next = poly_curr_ext.intersect(poly_next_ext)
3b3        polys_differences_curr_n_next = poly_curr_ext.subtract(poly_overlap_curr_n_next)
3b4        poly_curr_ext_retain = poly_curr.get_poly_overlapping(polys_differences_curr_n_next)
3b5        poly_curr_ext = poly_curr_ext_retain.union(poly_overlap_curr_n_next)
3c  else
3c1        break
3d  end
3e  i_next = get_next_polygon_index(i_curr, direction)
```



Informatik 11 Embedded Software
RWTH AACHEN UNIVERSITY

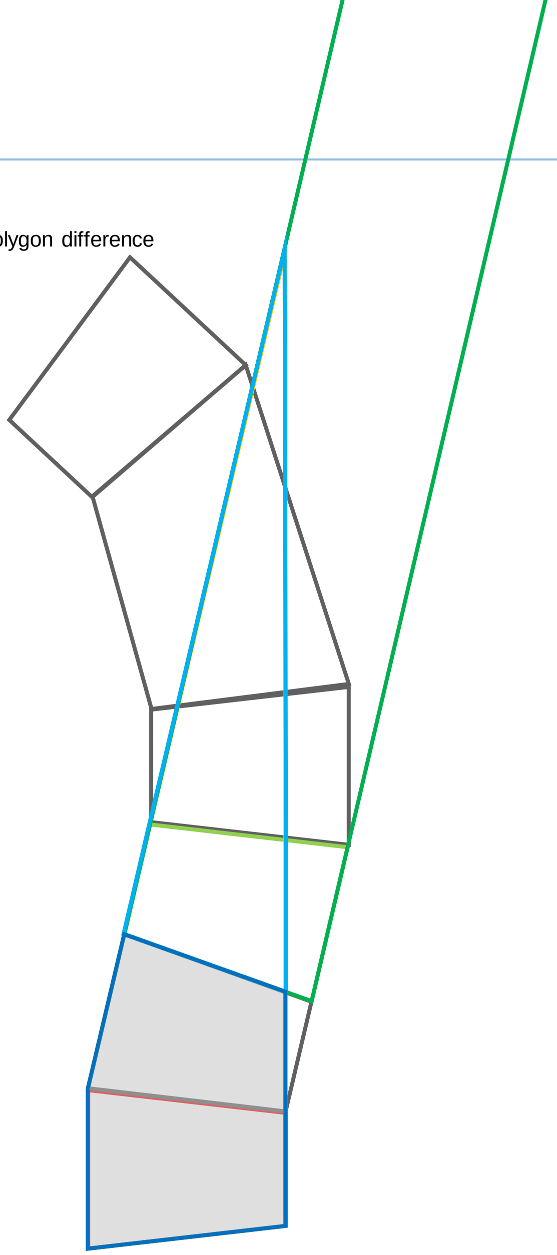
Algorithm Run
Iteration i_next = 2

▨ Current step and polygon

- Abbreviation: poly = polygon

```
function poly_curr_ext  = extend_convex_polygon_into_direction(i_curr, direction)
    i_curr (int): index of current polygon
    direction (bool): forward (true) or backward (false)
0 poly_curr = get_current_poly(i_curr)
1 poly_curr_ext = poly_curr.extend_poly_to_next(direction)
2 i_next = get_next_polygon_index(i_curr, direction)
3 while true
3a  poly_next = get_current_poly(i_next)
3b  if poly_curr_ext .overlaps(poly_next)
3b1       poly_next_ext = poly_next.extend_poly_to_next(direction)
3b2       poly_overlap_curr_n_next = poly_curr_ext.intersect(poly_next_ext)
3b3       polys_differences_curr_n_next = poly_curr_ext.subtract(poly_overlap_curr_n_next)
3b4       poly_curr_ext_retain = poly_curr.get_poly_overlapping(polys_differences_curr_n_next)
3b5       poly_curr_ext = poly_curr_ext_retain.union(poly_overlap_curr_n_next)
3c  else
3c1       break
3d  end
3e  i_next = get_next_polygon_index(i_curr, direction)
```

Note:
Edge case of only one polygon difference

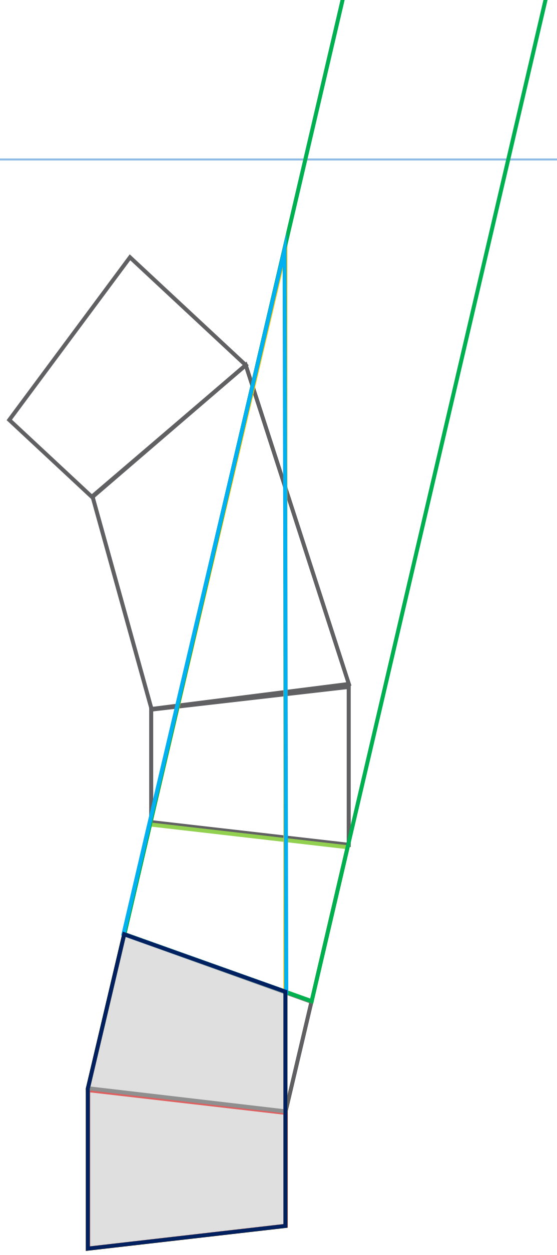


Algorithm Run
Iteration i_next = 2

▨ Current step and polygon

- Abbreviation: poly = polygon

```
function poly_curr_ext  = extend_convex_polygon_into_direction(i_curr, direction)
    i_curr (int): index of current polygon
    direction (bool): forward (true) or backward (false)
0 poly_curr = get_current_poly(i_curr)
1 poly_curr_ext = poly_curr.extend_poly_to_next(direction)
2 i_next = get_next_polygon_index(i_curr, direction)
3 while true
3a  poly_next = get_current_poly(i_next)
3b  if poly_curr_ext .overlaps(poly_next)
3b1         poly_next_ext = poly_next.extend_poly_to_next(direction)
3b2         poly_overlap_curr_n_next = poly_curr_ext.intersect(poly_next_ext)
3b3         polys_differences_curr_n_next = poly_curr_ext.subtract(poly_overlap_curr_n_next)
3b4         poly_curr_ext_retain = poly_curr.get_poly_overlapping(polys_differences_curr_n_next)
3b5         poly_curr_ext = poly_curr_ext_retain.union(poly_overlap_curr_n_next)
3c  else
3c1         break
3d  end
3e  i_next = get_next_polygon_index(i_curr, direction)
```



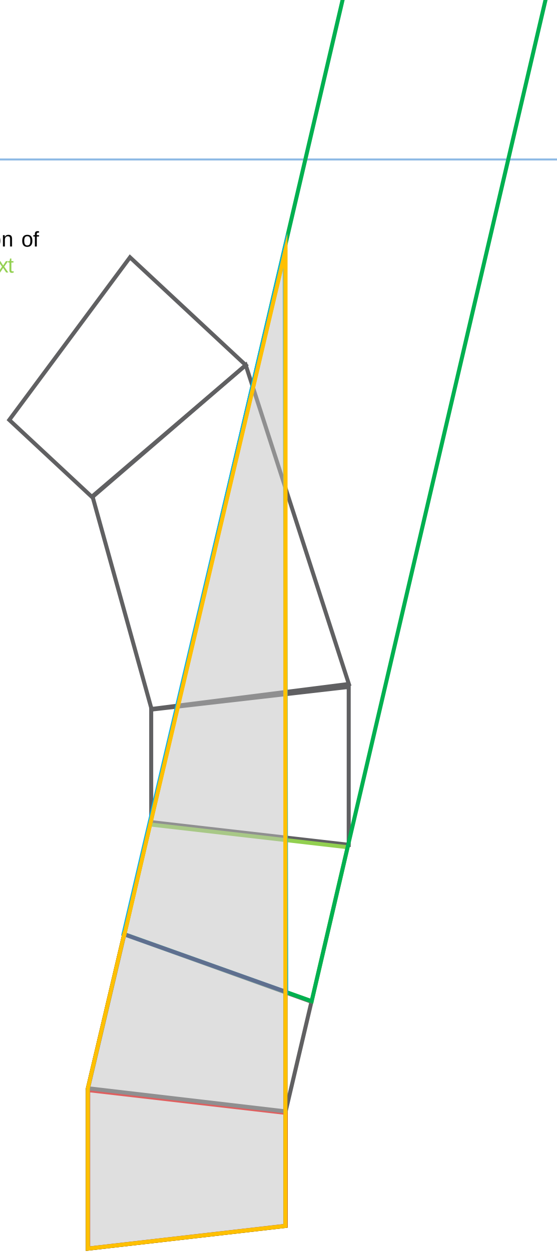
Algorithm Run
Iteration i_next = 2

▨ Current step and polygon

- Abbreviation: poly = polygon

```
function poly_curr_ext  = extend_convex_polygon_into_direction(i_curr, direction)
    i_curr (int): index of current polygon
    direction (bool): forward (true) or backward (false)
0 poly_curr = get_current_poly(i_curr)
1 poly_curr_ext = poly_curr.extend_poly_to_next(direction)
2 i_next = get_next_polygon_index(i_curr, direction)
3 while true
3a  poly_next = get_current_poly(i_next)
3b  if poly_curr_ext .overlaps(poly_next)
3b1         poly_next_ext = poly_next.extend_poly_to_next(direction)
3b2         poly_overlap_curr_n_next = poly_curr_ext.intersect(poly_next_ext)
3b3         polys_differences_curr_n_next = poly_curr_ext.subtract(poly_overlap_curr_n_next)
3b4         poly_curr_ext_retain = poly_curr.get_poly_overlapping(polys_differences_curr_n_next)
3b5         poly_curr_ext = poly_curr_ext_retain.union(poly_overlap_curr_n_next)
3c  else
3c1         break
3d  end
3e  i_next = get_next_polygon_index(i_curr, direction)
```

Note:
Edge case of no restriction of poly_curr_ext by poly_next



Algorithm Run
Iteration i_next = 2

▨ Current step and polygon

- Abbreviation: poly = polygon

```
▶ function poly_curr_ext  = extend_convex_polygon_into_direction(i_curr, direction)
      i_curr (int): index of current polygon
      direction (bool): forward (true) or backward (false)
▶ 0 poly_curr = get_current_poly(i_curr)
▶ 1 poly_curr_ext = poly_curr.extend_poly_to_next(direction)
▶ 2 i_next = get_next_polygon_index(i_curr, direction)
▶ 3 while true
▶ 3a  poly_next = get_current_poly(i_next)
▶ 3b  if poly_curr_ext .overlaps(poly_next)
▶ 3b1        poly_next_ext = poly_next.extend_poly_to_next(direction)
▶ 3b2        poly_overlap_curr_n_next = poly_curr_ext.intersect(poly_next_ext)
▶ 3b3        polys_differences_curr_n_next = poly_curr_ext.subtract(poly_overlap_curr_n_next)
▶ 3b4        poly_curr_ext_retain = poly_curr.get_poly_overlapping(polys_differences_curr_n_next)
▶ 3b5        poly_curr_ext = poly_curr_ext_retain.union(poly_overlap_curr_n_next)
▶ 3c  else
▶ 3c1        break
▶ 3d  end
▶ 3e  i_next = get_next_polygon_index(i_curr, direction)
```

Iteration i_next = 3

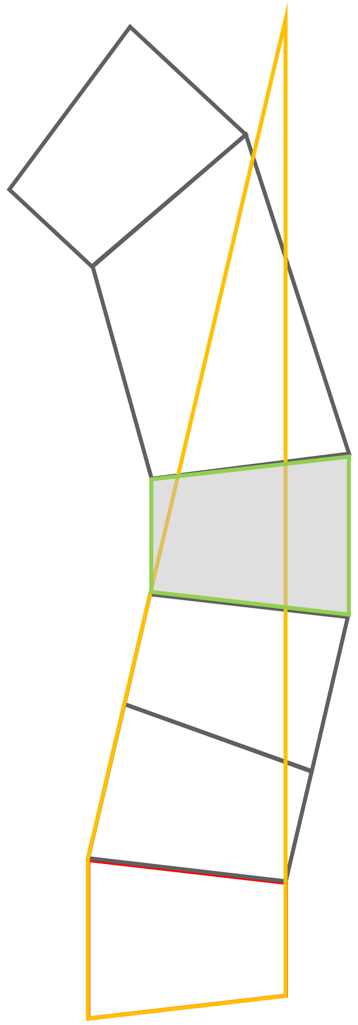
Note: same behaviour as for i_next = 2 → nothing new happening, skip to next section

Algorithm Run
Iteration i_next = 3

▨ Current step and polygon

- Abbreviation: poly = polygon

```
function poly_curr_ext  = extend_convex_polygon_into_direction(i_curr, direction)
     i_curr (int): index of current polygon
     direction (bool): forward (true) or backward (false)
0 poly_curr = get_current_poly(i_curr)
1 poly_curr_ext = poly_curr.extend_poly_to_next(direction)
2 i_next = get_next_polygon_index(i_curr, direction)
3 while true
3a  poly_next = get_current_poly(i_next)
3b  if poly_curr_ext .overlaps(poly_next)
3b1       poly_next_ext = poly_next.extend_poly_to_next(direction)
3b2       poly_overlap_curr_n_next = poly_curr_ext.intersect(poly_next_ext)
3b3       polys_differences_curr_n_next = poly_curr_ext.subtract(poly_overlap_curr_n_next)
3b4       poly_curr_ext_retain = poly_curr.get_poly_overlapping(polys_differences_curr_n_next)
3b5       poly_curr_ext = poly_curr_ext_retain.union(poly_overlap_curr_n_next)
3c  else
3c1       break
3d  end
3e  i_next = get_next_polygon_index(i_curr, direction)
```

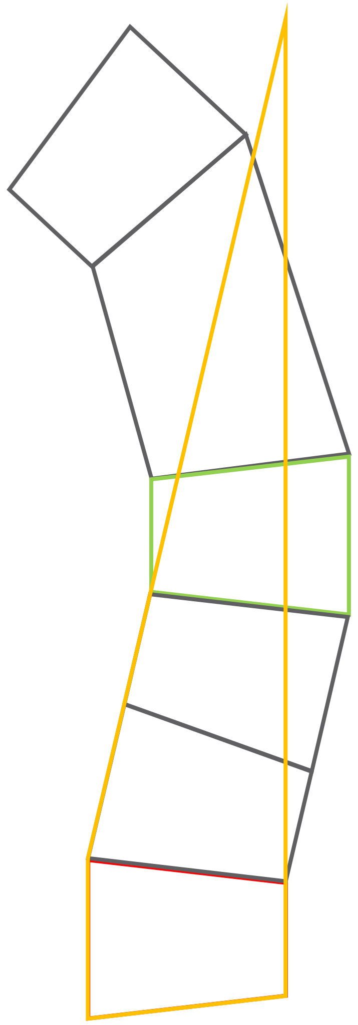


Algorithm Run
Iteration i_next = 3

▨ Current step and polygon

- Abbreviation: poly = polygon

```
function poly_curr_ext  = extend_convex_polygon_into_direction(i_curr, direction)
    i_curr (int): index of current polygon
    direction (bool): forward (true) or backward (false)
0 poly_curr = get_current_poly(i_curr)
1 poly_curr_ext = poly_curr.extend_poly_to_next(direction)
2 i_next = get_next_polygon_index(i_curr, direction)
3 while true
3a  poly_next = get_current_poly(i_next)
3b  if poly_curr_ext .overlaps(poly_next)
3b1       poly_next_ext = poly_next.extend_poly_to_next(direction)
3b2       poly_overlap_curr_n_next = poly_curr_ext.intersect(poly_next_ext)
3b3       polys_differences_curr_n_next = poly_curr_ext.subtract(poly_overlap_curr_n_next)
3b4       poly_curr_ext_retain = poly_curr.get_poly_overlapping(polys_differences_curr_n_next)
3b5       poly_curr_ext = poly_curr_ext_retain.union(poly_overlap_curr_n_next)
3c  else
3c1       break
3d  end
3e  i_next = get_next_polygon_index(i_curr, direction)
```

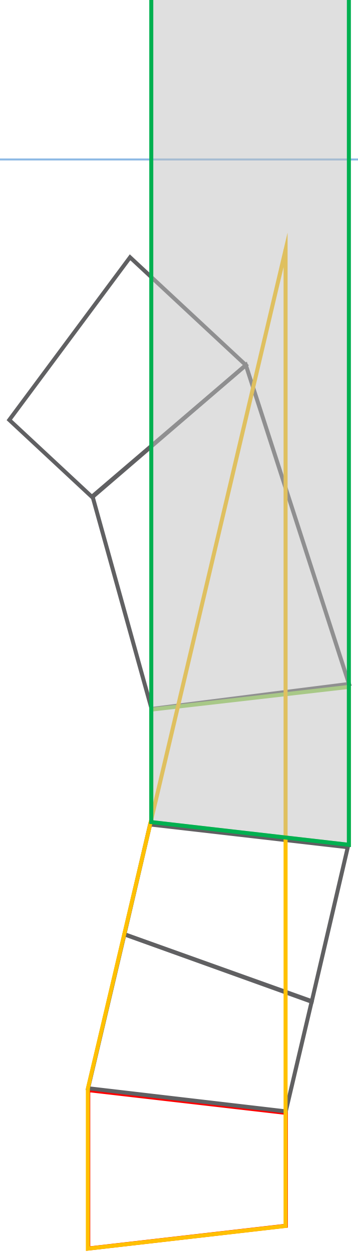


Algorithm Run
Iteration i_next = 3

▨ Current step and polygon

- Abbreviation: poly = polygon

```
function poly_curr_ext = extend_convex_polygon_into_direction(i_curr, direction)
    i_curr (int): index of current polygon
    direction (bool): forward (true) or backward (false)
0 poly_curr = get_current_poly(i_curr)
1 poly_curr_ext = poly_curr.extend_poly_to_next(direction)
2 i_next = get_next_polygon_index(i_curr, direction)
3 while true
3a poly_next = get_current_poly(i_next)
3b if poly_curr_ext .overlaps(poly_next)
3b1     poly_next_ext = poly_next.extend_poly_to_next(direction)
3b2     poly_overlap_curr_n_next = poly_curr_ext.intersect(poly_next_ext)
3b3     polys_differences_curr_n_next = poly_curr_ext.subtract(poly_overlap_curr_n_next)
3b4     poly_curr_ext_retain = poly_curr.get_poly_overlapping(polys_differences_curr_n_next)
3b5     poly_curr_ext = poly_curr_ext_retain.union(poly_overlap_curr_n_next)
3c else
3c1     break
3d end
3e i_next = get_next_polygon_index(i_curr, direction)
```

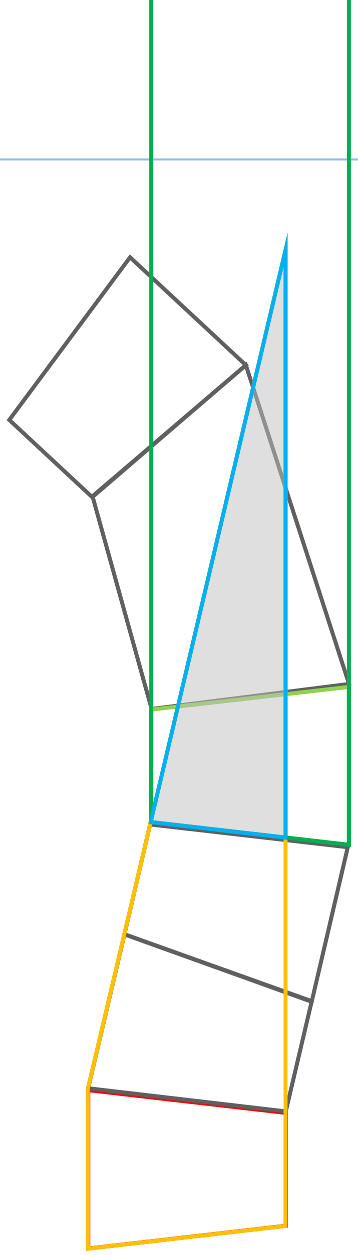


Algorithm Run
Iteration i_next = 3

▨ Current step and polygon

- Abbreviation: poly = polygon

```
function poly_curr_ext  = extend_convex_polygon_into_direction(i_curr, direction)
    i_curr (int): index of current polygon
    direction (bool): forward (true) or backward (false)
0 poly_curr = get_current_poly(i_curr)
1 poly_curr_ext = poly_curr.extend_poly_to_next(direction)
2 i_next = get_next_polygon_index(i_curr, direction)
3 while true
3a  poly_next = get_current_poly(i_next)
3b  if poly_curr_ext .overlaps(poly_next)
3b1         poly_next_ext = poly_next.extend_poly_to_next(direction)
3b2         poly_overlap_curr_n_next = poly_curr_ext.intersect(poly_next_ext)
3b3         polys_differences_curr_n_next = poly_curr_ext.subtract(poly_overlap_curr_n_next)
3b4         poly_curr_ext_retain = poly_curr.get_poly_overlapping(polys_differences_curr_n_next)
3b5         poly_curr_ext = poly_curr_ext_retain.union(poly_overlap_curr_n_next)
3c  else
3c1         break
3d  end
3e  i_next = get_next_polygon_index(i_curr, direction)
```



Informatik 11 Embedded Software
RWTH AACHEN UNIVERSITY

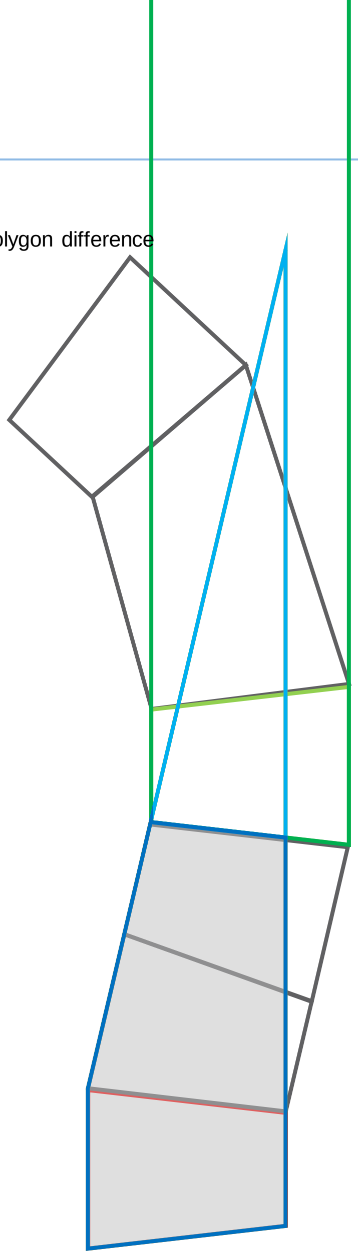
Algorithm Run
Iteration i_next = 3

▨ Current step and polygon

- Abbreviation: poly = polygon

```
▶ function poly_curr_ext  = extend_convex_polygon_into_direction(i_curr, direction)
      i_curr (int): index of current polygon
      direction (bool): forward (true) or backward (false)
▶ 0 poly_curr = get_current_poly(i_curr)
▶ 1 poly_curr_ext = poly_curr.extend_poly_to_next(direction)
▶ 2 i_next = get_next_polygon_index(i_curr, direction)
▶ 3 while true
▶ 3a  poly_next = get_current_poly(i_next)
▶ 3b  if poly_curr_ext .overlaps(poly_next)
▶ 3b1       poly_next_ext = poly_next.extend_poly_to_next(direction)
▶ 3b2       poly_overlap_curr_n_next = poly_curr_ext.intersect(poly_next_ext)
▶ 3b3       polys_differences_curr_n_next = poly_curr_ext.subtract(poly_overlap_curr_n_next)
▶ 3b4       poly_curr_ext_retain = poly_curr.get_poly_overlapping(polys_differences_curr_n_next)
▶ 3b5       poly_curr_ext = poly_curr_ext_retain.union(poly_overlap_curr_n_next)
▶ 3c  else
▶ 3c1       break
▶ 3d  end
▶ 3e  i_next = get_next_polygon_index(i_curr, direction)
```

Note:
Edge case of only one polygon difference



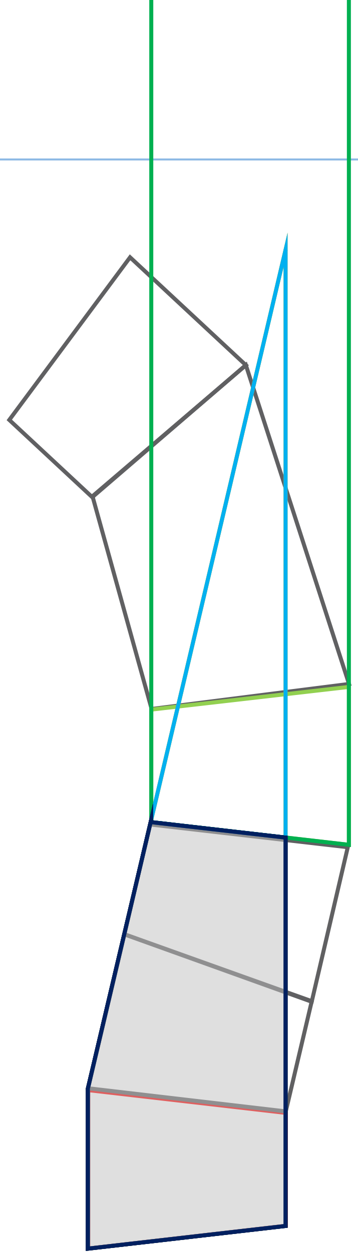
Informatik 11 Embedded Software | RWTH AACHEN UNIVERSITY

Algorithm Run
Iteration i_next = 3

▨ Current step and polygon

- Abbreviation: poly = polygon

```
function poly_curr_ext  = extend_convex_polygon_into_direction(i_curr, direction)
    i_curr (int): index of current polygon
    direction (bool): forward (true) or backward (false)
0 poly_curr = get_current_poly(i_curr)
1 poly_curr_ext = poly_curr.extend_poly_to_next(direction)
2 i_next = get_next_polygon_index(i_curr, direction)
3 while true
3a  poly_next = get_current_poly(i_next)
3b  if poly_curr_ext .overlaps(poly_next)
3b1       poly_next_ext = poly_next.extend_poly_to_next(direction)
3b2       poly_overlap_curr_n_next = poly_curr_ext.intersect(poly_next_ext)
3b3       polys_differences_curr_n_next = poly_curr_ext.subtract(poly_overlap_curr_n_next)
3b4       poly_curr_ext_retain = poly_curr.get_poly_overlapping(polys_differences_curr_n_next)
3b5       poly_curr_ext = poly_curr_ext_retain.union(poly_overlap_curr_n_next)
3c  else
3c1       break
3d  end
3e  i_next = get_next_polygon_index(i_curr, direction)
```



Algorithm Run

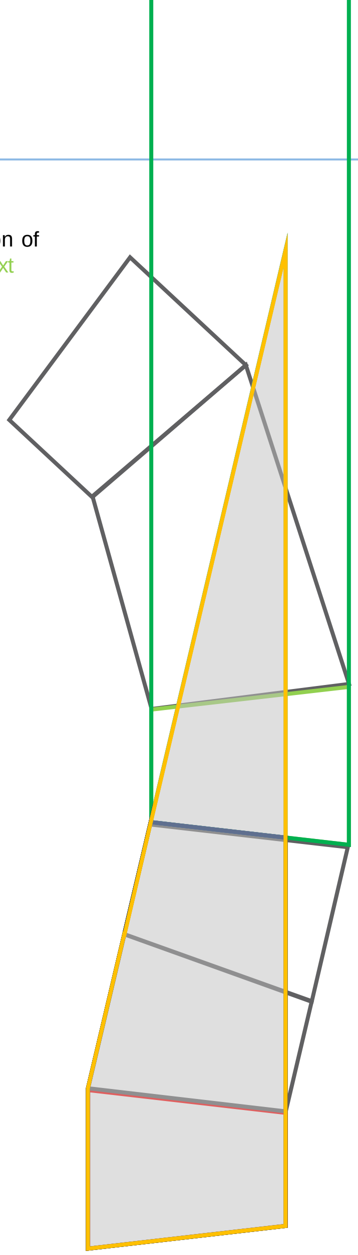
Iteration i_next = 3

▨ Current step and polygon

- Abbreviation: poly = polygon

```
function poly_curr_ext = extend_convex_polygon_into_direction(i_curr, direction)
    i_curr (int): index of current polygon
    direction (bool): forward (true) or backward (false)
0 poly_curr = get_current_poly(i_curr)
1 poly_curr_ext = poly_curr.extend_poly_to_next(direction)
2 i_next = get_next_polygon_index(i_curr, direction)
3 while true
3a poly_next = get_current_poly(i_next)
3b if poly_curr_ext .overlaps(poly_next)
3b1     poly_next_ext = poly_next.extend_poly_to_next(direction)
3b2     poly_overlap_curr_n_next = poly_curr_ext.intersect(poly_next_ext)
3b3     polys_differences_curr_n_next = poly_curr_ext.subtract(poly_overlap_curr_n_next)
3b4     poly_curr_ext_retain = poly_curr.get_poly_overlapping(polys_differences_curr_n_next)
3b5     poly_curr_ext = poly_curr_ext_retain.union(poly_overlap_curr_n_next)
3c else
3c1     break
3d end
3e i_next = get_next_polygon_index(i_curr, direction)
```

Note:
Edge case of no restriction of poly_curr_ext by poly_next



Informatik 11 Embedded Software | RWTH AACHEN UNIVERSITY

Algorithm Run
Iteration i_next = 3

▨ Current step and polygon

- Abbreviation: poly = polygon

```
function poly_curr_ext  = extend_convex_polygon_into_direction(i_curr, direction)
    i_curr (int): index of current polygon
    direction (bool): forward (true) or backward (false)
0 poly_curr = get_current_poly(i_curr)
1 poly_curr_ext = poly_curr.extend_poly_to_next(direction)
2 i_next = get_next_polygon_index(i_curr, direction)
3 while true
3a  poly_next = get_current_poly(i_next)
3b  if poly_curr_ext .overlaps(poly_next)
3b1     poly_next_ext = poly_next.extend_poly_to_next(direction)
3b2     poly_overlap_curr_n_next = poly_curr_ext.intersect(poly_next_ext)
3b3     polys_differences_curr_n_next = poly_curr_ext.subtract(poly_overlap_curr_n_next)
3b4     poly_curr_ext_retain = poly_curr.get_poly_overlapping(polys_differences_curr_n_next)
3b5     poly_curr_ext = poly_curr_ext_retain.union(poly_overlap_curr_n_next)
3c  else
3c1     break
3d  end
3e  i_next = get_next_polygon_index(i_curr, direction)
```

Iteration i_next = 4

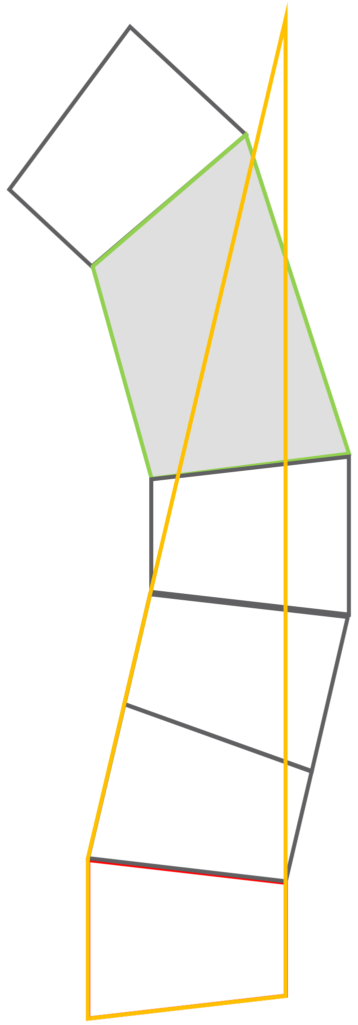
Algorithm Run

Iteration i_next = 4

▨ Current step and polygon

- Abbreviation: poly = polygon

```
function poly_curr_ext  = extend_convex_polygon_into_direction(i_curr, direction)
    i_curr (int): index of current polygon
    direction (bool): forward (true) or backward (false)
0 poly_curr = get_current_poly(i_curr)
1 poly_curr_ext = poly_curr.extend_poly_to_next(direction)
2 i_next = get_next_polygon_index(i_curr, direction)
3 while true
3a poly_next = get_current_poly(i_next)
3b if poly_curr_ext .overlaps(poly_next)
3b1        poly_next_ext = poly_next.extend_poly_to_next(direction)
3b2        poly_overlap_curr_n_next = poly_curr_ext.intersect(poly_next_ext)
3b3        polys_differences_curr_n_next = poly_curr_ext.subtract(poly_overlap_curr_n_next)
3b4        poly_curr_ext_retain = poly_curr.get_poly_overlapping(polys_differences_curr_n_next)
3b5        poly_curr_ext = poly_curr_ext_retain.union(poly_overlap_curr_n_next)
3c else
3c1        break
3d end
3e i_next = get_next_polygon_index(i_curr, direction)
```



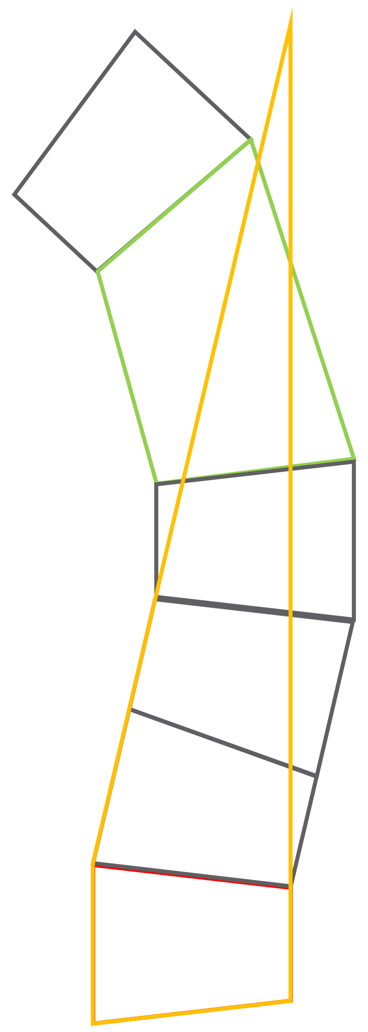
Informatik 11 Embedded Software | RWTH AACHEN UNIVERSITY

Algorithm Run
Iteration i_next = 4

▨ Current step and polygon

- Abbreviation: poly = polygon

```
function poly_curr_ext  = extend_convex_polygon_into_direction(i_curr, direction)
    i_curr (int): index of current polygon
    direction (bool): forward (true) or backward (false)
0 poly_curr = get_current_poly(i_curr)
1 poly_curr_ext = poly_curr.extend_poly_to_next(direction)
2 i_next = get_next_polygon_index(i_curr, direction)
3 while true
3a  poly_next = get_current_poly(i_next)
3b  if poly_curr_ext .overlaps(poly_next)
3b1         poly_next_ext = poly_next.extend_poly_to_next(direction)
3b2         poly_overlap_curr_n_next = poly_curr_ext.intersect(poly_next_ext)
3b3         polys_differences_curr_n_next = poly_curr_ext.subtract(poly_overlap_curr_n_next)
3b4         poly_curr_ext_retain = poly_curr.get_poly_overlapping(polys_differences_curr_n_next)
3b5         poly_curr_ext = poly_curr_ext_retain.union(poly_overlap_curr_n_next)
3c  else
3c1         break
3d  end
3e  i_next = get_next_polygon_index(i_curr, direction)
```



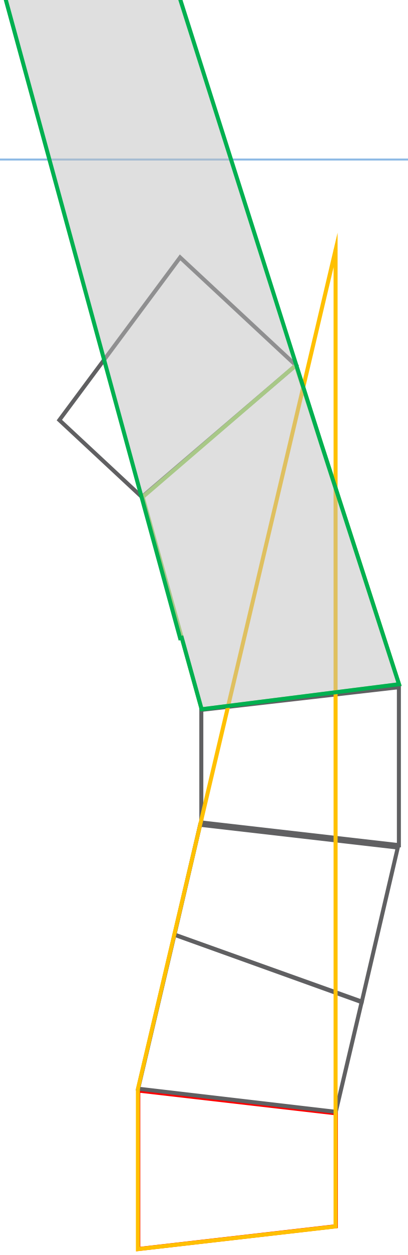
Informatik 11 Embedded Software | RWTH AACHEN UNIVERSITY

Algorithm Run
Iteration i_next = 4

▨ Current step and polygon

- Abbreviation: poly = polygon

```
function poly_curr_ext  = extend_convex_polygon_into_direction(i_curr, direction)
    i_curr (int): index of current polygon
    direction (bool): forward (true) or backward (false)
0 poly_curr = get_current_poly(i_curr)
1 poly_curr_ext = poly_curr.extend_poly_to_next(direction)
2 i_next = get_next_polygon_index(i_curr, direction)
3 while true
3a  poly_next = get_current_poly(i_next)
3b  if poly_curr_ext .overlaps(poly_next)
3b1       poly_next_ext = poly_next.extend_poly_to_next(direction)
3b2       poly_overlap_curr_n_next = poly_curr_ext.intersect(poly_next_ext)
3b3       polys_differences_curr_n_next = poly_curr_ext.subtract(poly_overlap_curr_n_next)
3b4       poly_curr_ext_retain = poly_curr.get_poly_overlapping(polys_differences_curr_n_next)
3b5       poly_curr_ext = poly_curr_ext_retain.union(poly_overlap_curr_n_next)
3c  else
3c1       break
3d  end
3e  i_next = get_next_polygon_index(i_curr, direction)
```

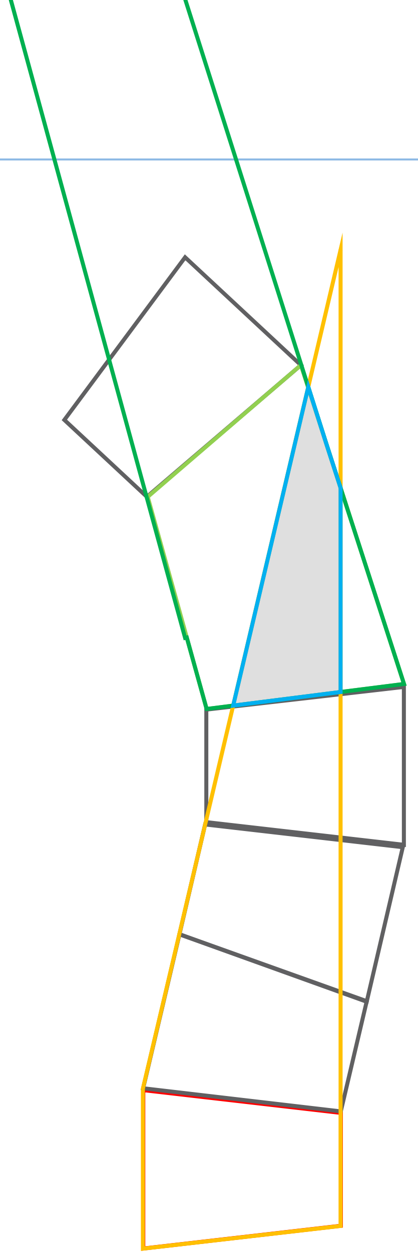


Algorithm Run
Iteration i_next = 4

▨ Current step and polygon

- Abbreviation: poly = polygon

```
function poly_curr_ext = extend_convex_polygon_into_direction(i_curr, direction)
    i_curr (int): index of current polygon
    direction (bool): forward (true) or backward (false)
0 poly_curr = get_current_poly(i_curr)
1 poly_curr_ext = poly_curr.extend_poly_to_next(direction)
2 i_next = get_next_polygon_index(i_curr, direction)
3 while true
3a  poly_next = get_current_poly(i_next)
3b  if poly_curr_ext .overlaps(poly_next)
3b1     poly_next_ext = poly_next.extend_poly_to_next(direction)
3b2     poly_overlap_curr_n_next = poly_curr_ext.intersect(poly_next_ext)
3b3     polys_differences_curr_n_next = poly_curr_ext.subtract(poly_overlap_curr_n_next)
3b4     poly_curr_ext_retain = poly_curr.get_poly_overlapping(polys_differences_curr_n_next)
3b5     poly_curr_ext = poly_curr_ext_retain.union(poly_overlap_curr_n_next)
3c  else
3c1     break
3d  end
3e  i_next = get_next_polygon_index(i_curr, direction)
```



Informatik 11 Embedded Software
RWTH AACHEN UNIVERSITY

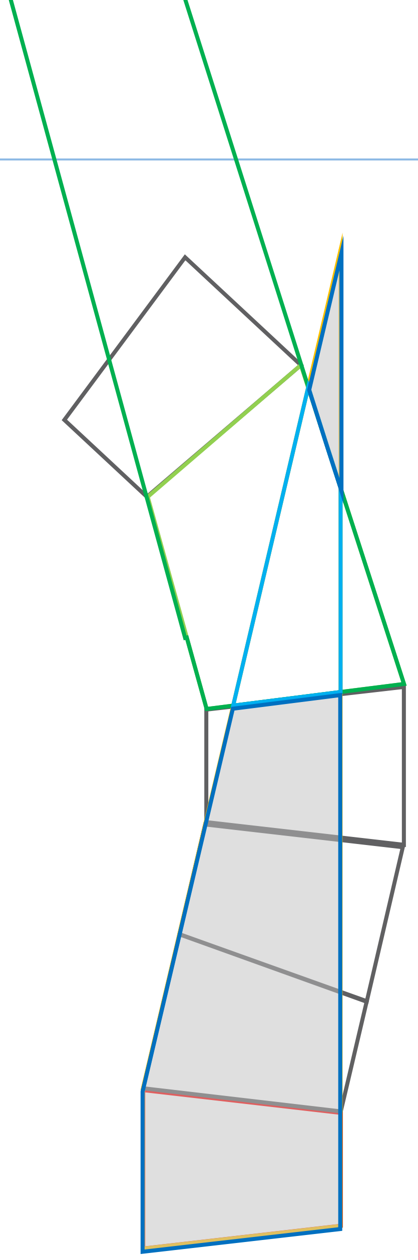
Algorithm Run
Iteration i_next = 4

▨ Current step and polygon

- Abbreviation: poly = polygon

- function poly_curr_ext = extend_convex_polygon_into_direction(i_curr, direction)
  i_curr (int): index of current polygon
  direction (bool): forward (true) or backward (false)

```
0 poly_curr = get_current_poly(i_curr)
1 poly_curr_ext = poly_curr.extend_poly_to_next(direction)
2 i_next = get_next_polygon_index(i_curr, direction)
3 while true
3a  poly_next = get_current_poly(i_next)
3b  if poly_curr_ext .overlaps(poly_next)
3b1       poly_next_ext = poly_next.extend_poly_to_next(direction)
3b2       poly_overlap_curr_n_next = poly_curr_ext.intersect(poly_next_ext)
3b3       polys_differences_curr_n_next = poly_curr_ext.subtract(poly_overlap_curr_n_next)
3b4       poly_curr_ext_retain = poly_curr.get_poly_overlapping(polys_differences_curr_n_next)
3b5       poly_curr_ext = poly_curr_ext_retain.union(poly_overlap_curr_n_next)
3c  else
3c1       break
3d  end
3e  i_next = get_next_polygon_index(i_curr, direction)
```



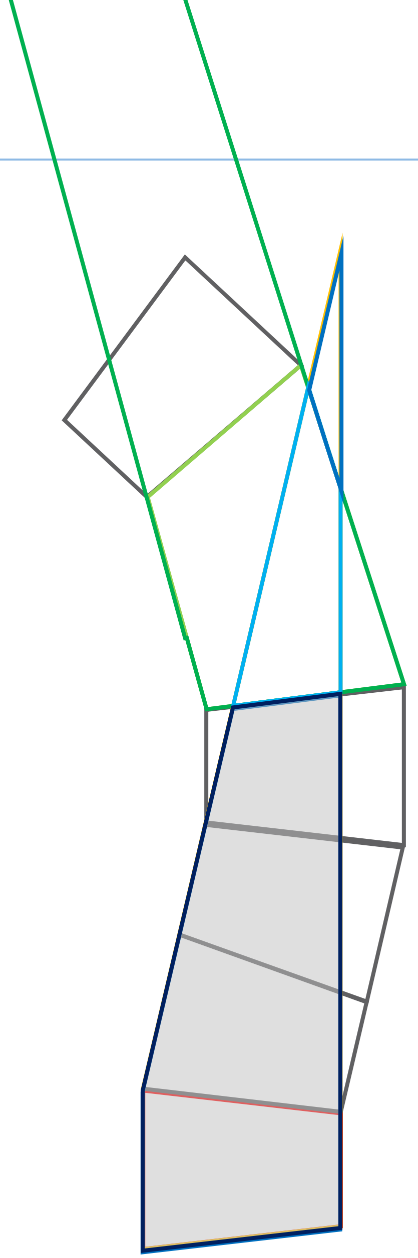
Algorithm Run

Iteration i_next = 4

▨ Current step and polygon

- Abbreviation: poly = polygon

```
function poly_curr_ext  = extend_convex_polygon_into_direction(i_curr, direction)
    i_curr (int): index of current polygon
    direction (bool): forward (true) or backward (false)
0 poly_curr = get_current_poly(i_curr)
1 poly_curr_ext = poly_curr.extend_poly_to_next(direction)
2 i_next = get_next_polygon_index(i_curr, direction)
3 while true
3a  poly_next = get_current_poly(i_next)
3b  if poly_curr_ext .overlaps(poly_next)
3b1       poly_next_ext = poly_next.extend_poly_to_next(direction)
3b2       poly_overlap_curr_n_next = poly_curr_ext.intersect(poly_next_ext)
3b3       polys_differences_curr_n_next = poly_curr_ext.subtract(poly_overlap_curr_n_next)
3b4       poly_curr_ext_retain = poly_curr.get_poly_overlapping(polys_differences_curr_n_next)
3b5       poly_curr_ext = poly_curr_ext_retain.union(poly_overlap_curr_n_next)
3c  else
3c1       break
3d  end
3e  i_next = get_next_polygon_index(i_curr, direction)
```



Informatik 11 Embedded Software

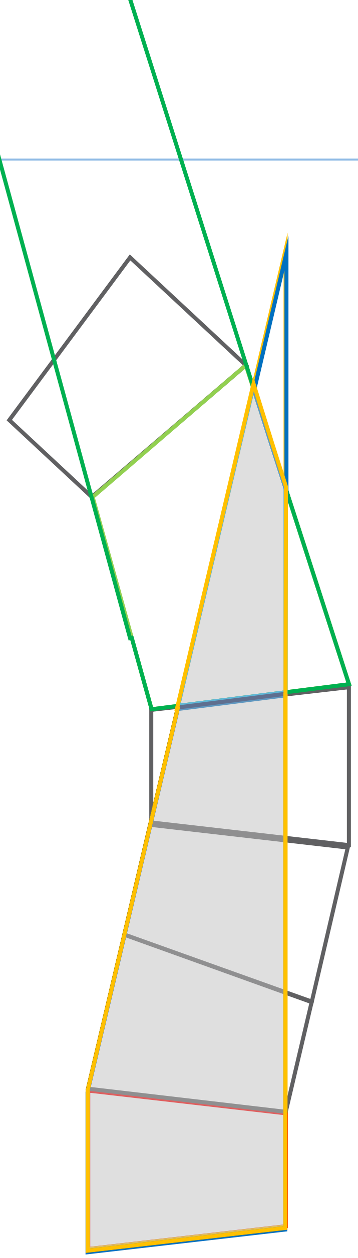
RWTH AACHEN UNIVERSITY

Algorithm Run
Iteration i_next = 4

▨ Current step and polygon

- Abbreviation: poly = polygon

```
▸ function poly_curr_ext  = extend_convex_polygon_into_direction(i_curr, direction)
      i_curr (int): index of current polygon
      direction (bool): forward (true) or backward (false)
▸ 0 poly_curr = get_current_poly(i_curr)
▸ 1 poly_curr_ext = poly_curr.extend_poly_to_next(direction)
▸ 2 i_next = get_next_polygon_index(i_curr, direction)
▸ 3 while true
▸ 3a  poly_next = get_current_poly(i_next)
▸ 3b  if poly_curr_ext .overlaps(poly_next)
▸ 3b1       poly_next_ext = poly_next.extend_poly_to_next(direction)
▸ 3b2       poly_overlap_curr_n_next = poly_curr_ext.intersect(poly_next_ext)
▸ 3b3       polys_differences_curr_n_next = poly_curr_ext.subtract(poly_overlap_curr_n_next)
▸ 3b4       poly_curr_ext_retain = poly_curr.get_poly_overlapping(polys_differences_curr_n_next)
▸ 3b5       poly_curr_ext = poly_curr_ext_retain.union(poly_overlap_curr_n_next)
▸ 3c  else
▸ 3c1       break
▸ 3d  end
▸ 3e  i_next = get_next_polygon_index(i_curr, direction)
```



Iteration i_next = 5

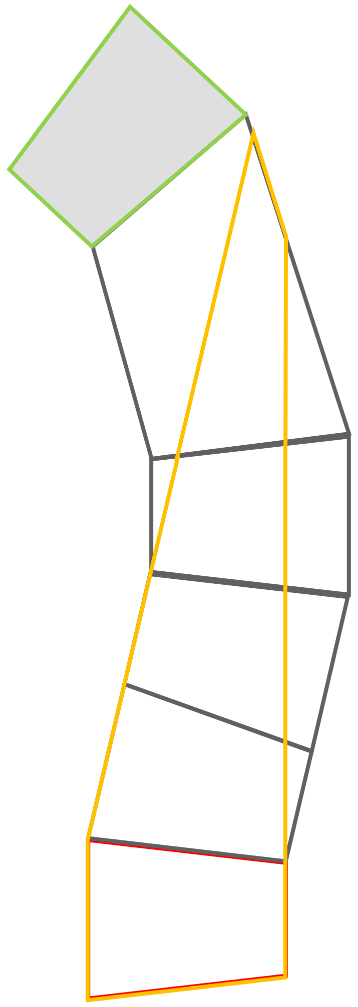
Algorithm Run

Iteration i_next = 5

▨ Current step and polygon

- Abbreviation: poly = polygon

```
function poly_curr_ext  = extend_convex_polygon_into_direction(i_curr, direction)
    i_curr (int): index of current polygon
    direction (bool): forward (true) or backward (false)
0 poly_curr = get_current_poly(i_curr)
1 poly_curr_ext = poly_curr.extend_poly_to_next(direction)
2 i_next = get_next_polygon_index(i_curr, direction)
3 while true
3a poly_next = get_current_poly(i_next)
3b if poly_curr_ext .overlaps(poly_next)
3b1     poly_next_ext = poly_next.extend_poly_to_next(direction)
3b2     poly_overlap_curr_n_next = poly_curr_ext.intersect(poly_next_ext)
3b3     polys_differences_curr_n_next = poly_curr_ext.subtract(poly_overlap_curr_n_next)
3b4     poly_curr_ext_retain = poly_curr.get_poly_overlapping(polys_differences_curr_n_next)
3b5     poly_curr_ext = poly_curr_ext_retain.union(poly_overlap_curr_n_next)
3c else
3c1     break
3d end
3e i_next = get_next_polygon_index(i_curr, direction)
```

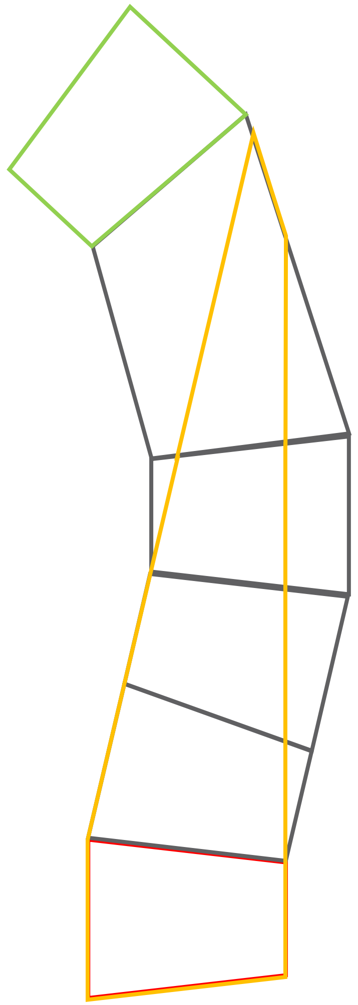


Algorithm Run
Iteration i_next = 5

▨ Current step and polygon

- Abbreviation: poly = polygon

```
function poly_curr_ext  = extend_convex_polygon_into_direction(i_curr, direction)
    i_curr (int): index of current polygon
    direction (bool): forward (true) or backward (false)
0 poly_curr = get_current_poly(i_curr)
1 poly_curr_ext = poly_curr.extend_poly_to_next(direction)
2 i_next = get_next_polygon_index(i_curr, direction)
3 while true
3a  poly_next = get_current_poly(i_next)
3b  if poly_curr_ext .overlaps(poly_next)
3b1        poly_next_ext = poly_next.extend_poly_to_next(direction)
3b2        poly_overlap_curr_n_next = poly_curr_ext.intersect(poly_next_ext)
3b3        polys_differences_curr_n_next = poly_curr_ext.subtract(poly_overlap_curr_n_next)
3b4        poly_curr_ext_retain = poly_curr.get_poly_overlapping(polys_differences_curr_n_next)
3b5        poly_curr_ext = poly_curr_ext_retain.union(poly_overlap_curr_n_next)
3c  else
3c1        break
3d  end
3e  i_next = get_next_polygon_index(i_curr, direction)
```



Informatik 11 Embedded Software
RWTH AACHEN UNIVERSITY

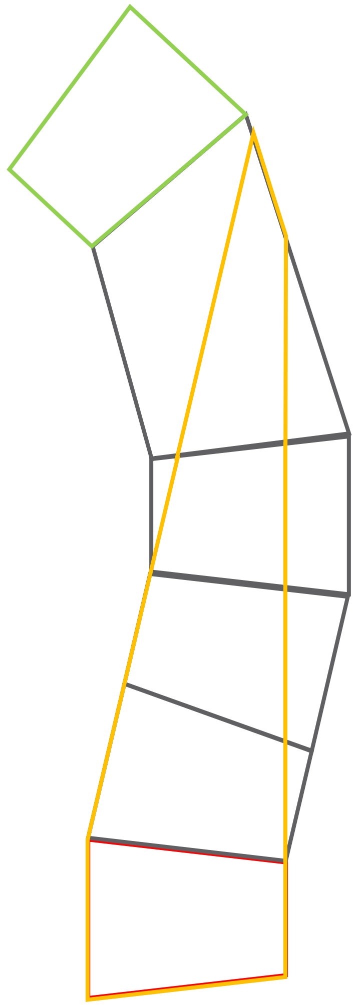
Algorithm Run

Iteration i_next = 5

▨ Current step and polygon

- Abbreviation: poly = polygon

```
function poly_curr_ext = extend_convex_polygon_into_direction(i_curr, direction)
    i_curr (int): index of current polygon
    direction (bool): forward (true) or backward (false)
0 poly_curr = get_current_poly(i_curr)
1 poly_curr_ext = poly_curr.extend_poly_to_next(direction)
2 i_next = get_next_polygon_index(i_curr, direction)
3 while true
3a  poly_next = get_current_poly(i_next)
3b  if poly_curr_ext .overlaps(poly_next)
3b1       poly_next_ext = poly_next.extend_poly_to_next(direction)
3b2       poly_overlap_curr_n_next = poly_curr_ext.intersect(poly_next_ext)
3b3       polys_differences_curr_n_next = poly_curr_ext.subtract(poly_overlap_curr_n_next)
3b4       poly_curr_ext_retain = poly_curr.get_poly_overlapping(polys_differences_curr_n_next)
3b5       poly_curr_ext = poly_curr_ext_retain.union(poly_overlap_curr_n_next)
3c  else
3c1       break
3d  end
3e  i_next = get_next_polygon_index(i_curr, direction)
```



Informatik 11 Embedded Software

RWTH AACHEN UNIVERSITY

Algorithm finished

Biggest convex polygon inside track starting from red polygon in forward direction found