Geometric Algorithms
Assignment 3

Laura Baakman (s1869140)

September 30, 2014

A

B

C

To find the intersected edges and compute the intersection points we have used the method `find_intersected_edges`, see Listing 1. This method calls the method `line_segments_intersect` (**??**) on the edge defined by the points `p0` and `p1` and the edges of the triangulation. If an intersection is found it adds the indices of the intersected edge in the global list `intersected_line_segments` and appends the intersection point to `intersection_points`. The intersected edges of the triangulation and the intersection points are presented in Table 1.

To visualize the results we have added the code in Listing 2, the resulting visualization is shown in Figure 1.

Listing 1: The method `find_intersected_edges()`.

```
def find_intersected_edges():
    """
    Find the edges of the triangulation that are intersected by the linesegment p0-p1.

    The intersected edges are stored as an index in the lists xl and yl
    in the global variable intersected_line_segments.
    The intersection points are stored as list of length two in the
    global parameter intersection_points.
    """
    global intersected_line_segments, intersection_points, edgs, xl, yl, po, p1
    segment_1 = [p0, p1]
    for edge in edgs:
        segment_2 = [
            [xl[edge[0]], yl[edge[0]]],
            [xl[edge[1]], yl[edge[1]]]
        ]
        intersection = line_segments_intersect(segment_1, segment_2)
        if(intersection):
            intersected_line_segments.append(edge)
            intersection_points.append(intersection)
```

Listing 2: Part of the method `display()` that visualizes the intersected edges and the intersections.

```
# draw intersected segments
glColor3f(0.0, 0.0, 1)
glBegin(GL_LINES)
for edge in intersected_line_segments:
    glVertex2f(xl[edge[0]],  yl[edge[0]])
    glVertex2f(xl[edge[1]],  yl[edge[1]])
glEnd()

# draw intersection points on walk line
glColor3f(0.0, 1.0, 0.0)
glPointSize(4)
glBegin(GL_POINTS)
for point in intersection_points:
    glVertex2f(point[0], point[1])
glEnd()
```

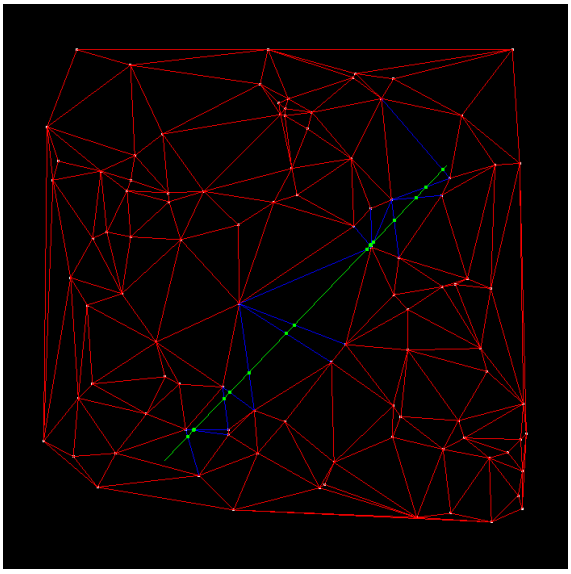

Figure 1: The visualization of the intersection of the line segment s with the edges of the triangulation dt. Each intersected edges is shown in blue, the green dots represent the intersections.

edge	intersection	edge	intersection
(2.26e2, 5.27e2) - (2.43e2, 5.84e2)	(2.29e2, 5.36e2)	(4.57e2, 3.01e2) - (2.93e2, 3.71e2)	(4.51e2, 3.04e2)
(2.26e2, 5.27e2) - (2.80e2, 5.34e2)	(2.36e2, 5.28e2)	(4.35e2, 2.75e2) - (4.57e2, 3.01e2)	(4.55e2, 2.99e2)
(2.26e2, 5.27e2) - (2.79e2, 5.28e2)	(2.37e2, 5.27e2)	(5.43e2, 2.37e2) - (4.81e2, 2.43e2)	(5.12e2, 2.40e2)
(2.72e2, 4.75e2) - (2.79e2, 5.28e2)	(2.74e2, 4.89e2)	(4.81e2, 2.43e2) - (4.91e2, 3.14e2)	(4.85e2, 2.68e2)
(2.72e2, 4.75e2) - (3.11e2, 5.03e2)	(2.81e2, 4.81e2)	(4.81e2, 2.43e2) - (4.57e2, 3.01e2)	(4.59e2, 2.95e2)
(2.93e2, 3.71e2) - (4.07e2, 4.44e2)	(3.51e2, 4.08e2)	(4.56e2, 2.53e2) - (4.57e2, 3.01e2)	(4.56e2, 2.98e2)
(2.93e2, 3.71e2) - (3.11e2, 5.03e2)	(3.05e2, 4.57e2)	(5.54e2, 2.16e2) - (4.81e2, 2.43e2)	(5.24e2, 2.27e2)
(2.93e2, 3.71e2) - (4.25e2, 4.22e2)	(3.61e2, 3.98e2)	(4.69e2, 1.17e2) - (5.54e2, 2.16e2)	(5.45e2, 2.05e2)

Table 1: The edges that were intersected by the line segment between `p0` and `p1` and the point where the line segment intersected the edge. A point $\mathbf{P}$ defined by its x and y coordinate is represented as (x, y). A linesegment between the points $\mathbf{P_1}$ and $\mathbf{P_2}$ is represented as $(\mathbf{P_1}, \mathbf{P_2})$.