Point-in-Polygon Test

Report for the CEGE0096: 1st Assignment

**─**

Optional image here

**Your Name** Jianfan Yang

**Your Portico** **Number** 21110107

**Your Email** ucesjy3@ucl.ac.uk

# 

# Introduction

Use the text highlighted in orange as a guideline and delete it before submitting the report. Please do not just answer the questions without integrating your answers into paragraphs. These questions are there only as a guideline. Please do not change the template of the report. Please do not use colloquial expressions and write in professional English. Remember to be thorough (and honest) in your explanations; this report will be also used to assess what you have done in each task.

Describe your assignment. This section should answer the following questions:

* What does your software do? Which part of the assignment have you completed?
* What are the limitations of your software?
* How did you develop it?
* How long did it take to develop it?

This software automatically classifies the given nodes into three classes, i.e., Inside, Boundary, and Outside, with respect to the plotted polygon area. In this project, I utilized Python to implement the Minimum Bounding Rectangle Algorithm (MBR) and the Ray Casting Algorithm (RCA) for node classification. It takes me approximately one day to finish.

# Project Description

Explain here how to use your software. This section should at least include the answers to the following questions:

* What are the prerequisites of your software? Is Python 3.8 enough or should I install additional libraries?
* How can I run your software? What should I execute? Is your script expecting the presence of any file in the same directory? Does it require parameters?

The prerequisites of my software is Python 3.9 and the libraries matplotlib and csv are needed to plot the graph and read the data from csv file. The software can be easily executed by typing *python3 main\_from\_file.py* or *python3 main\_from\_user.py* in the terminal. These Python files do not need any parameter and should be in the same directory.

# Software

In this section you should write how you developed some of the tasks in the marking scheme. You can copy and paste the function/method/class here and describe how it works. Please do not paste screenshots of your code (penalties will occur if you do that).

## Task 1. The MBR Algorithm

Describe here how you implemented this algorithm. You can copy and paste your code and describe what it does. If you did not do this task state why.

**Codes**

p = (x, y)

category = []

if p[0] < min\_x or p[0] > max\_x or p[1] < min\_y or p[1] > max\_y:

plotter.add\_point(p[0], p[1], 'outside')

category.append('outside')

**Explanation**

p consists of the coordinate of a given node. The node is judged to be outside the polygon when its x-coordinate (y-coordinate) is smaller than the x-coordinate (y-coordinate) of the rectangle.

## Task 2. The RCA Algorithm

Describe here how you implemented this algorithm. You can copy and paste your code and describe what it does. If you did not do this task state why.

**Codes**

t = rca.get\_intersect\_times(p[0], p[1])

if t == -1:

plotter.add\_point(p[0], p[1], 'boundary')

category.append('boundary')

elif t % 2:

plotter.add\_point(p[0], p[1], 'inside')

category.append('inside')

else:

plotter.add\_point(p[0], p[1], 'outside')

category.append('outside')

class RCA:

def \_\_init\_\_(self, boundaries):

self.boundaries = boundaries

def get\_intersect\_times(self, x, y):

cnt = 0

ori\_x, ori\_y = x, y

for b in self.boundaries:

# reset the node to the original position

x = ori\_x

y = ori\_y

if (x == b.sx and y == b.sy) or (x == b.ex and y == b.ey): # the node is a vertex of the boundary

return -1

if (y == b.sy and y == b.ey and min(b.sx, b.ex) <= x <= max(b.sx, b.ex)): # the node is on the harizonal boundary

return -1

if x == b.sx and x == b.ex and b.sy <= y <= b.ey: # the node is on the harizonal boundary

return -1

if (y == b.sy or y == b.ey):

if b.sy != b.ey:

y += 1e-6

else:

continue

if y > b.ey or y < b.sy:

continue

if x >= max(b.sx, b.ex):

continue

if x < min(b.sx, b.ex):

cnt += 1

continue

if b.ex == b.sx: # avoid zero-divide

slope\_b = 1e9

else:

slope\_b = (b.ey - b.sy) / (b.ex - b.sx)

if x == b.sx:

slope\_p = 1e9

else:

slope\_p = (y - b.sy) / (x - b.sx)

if slope\_p > slope\_b:

cnt += 1

elif slope\_p < slope\_b:

continue

else:

return -1 # boundary

return cnt

class Edge:

def \_\_init\_\_(self, start, end):

if start[1] < end[1]:

self.sx = start[0]

self.sy = start[1]

self.ex = end[0]

self.ey = end[1]

else:

self.sx = end[0]

self.sy = end[1]

self.ex = start[0]

self.ey = start[1]

**Explanation**

We first build a class for boundary whose instance consists of the coordinate of the start and end point of a boundary. Next, we compute the interaction times of each node and all boundaries. If the time is odd, it is inside the polygon; otherwise outsides.

## Task 3. The Categorisation of Special Cases

Which cases did you solve and how? If you did not do this task state why.

**Codes**

if (y == b.sy or y == b.ey):

if b.sy != b.ey:

y += 1e-6

**Explanation**

For those nodes which may interact with the vertex of the polygon, we move it up a very small distance (10-6 in the software) to avoid this condition.

## Task 5. Object-Oriented Programming

What did you make object-oriented? If you did not do this task state why.

I made the RCA and the boundary as classes, which is object-oriented programming. (See the codes in task 2 for detail).

## Task 6. On the Use of Git and GitHub

How did you use Git and GitHub? If you did not do this task state why.

I use git to save the edit history.

## Task 9. Plotting

How did you use the Plotter class? Did you implement any additional plotting functionalities? If you did not do this task state why.

I use the Plotter class to store the properties of nodes, for instance, the relative position of node. I did not implement any other plotting functionalities.

## Task 10. Error Handling

Which error handling functionality did you implement? If you did not do this task state why.

There is no error handling module in my software.

## Task 11. Additional Features

Any additional features? If you did not do this task state why.

I create a new class Edge to store each boundary, which contains a start point and an end point of an edge, making the codes can be utilized by other programs with least modification.