Name: Delhi Durai. **Student-ID:** 948639812

Name: Rajeswari. **Student-ID:** 916809371

RUST FINAL PROJECT – CS 510

GEO FENCING AND LOCATION TRACKING

Theory of operation: -

The Geofencing crate is a library that uses coordinate geometry to calculate if a given point is within or outside the fence, the fence is built using several individual coordinates. It uses latitude and longitude as coordinate grids to create the fence and to search.

The crate provides the following features

1. Establishing virtual boundary for Geo fence.
2. Command line interface for user inputs.
3. Search option to search for objects that are within or outside the virtual boundary.

At present it is designed to create fences for three different shapes

a. Polygon
b. Circle
c. Triangle

Geofencing on a Polygon fence: -

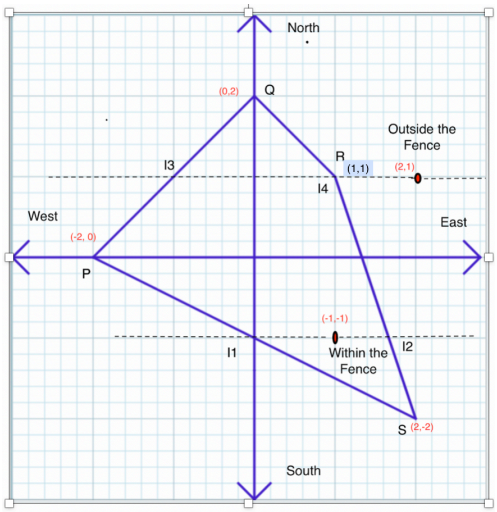
Polygon geofencing uses Ray Casting algorithm along with Cramer's rule, to determine if a point is within or outside the fence. A ray of line, cast in any direction can determine, if it is within or outside the fence, based on the number of times the line intersect with the fence.

If the number of times the line intersects is odd, it is assumed that the point from which the ray is cast is within the boundary/fence. Similarly, if the number of times, the line intersects is even, it is assumed that the point from which the ray is cast is outside the boundary/fence.

For example, the below figure shows a polygon fence drawn in a coordinate plane with North and East being positive and South and West being negative access. P, Q, R and S are the points that forms the polygon fence. A ray of line that is cast on either direction from the point (2,1), one of the ray, will intersect the polygon fence in two places I4 and I3, which is even number and hence the point is determined as outside the fence. On the other hand for the point (-1,1), if a ray of line is cast on either direction horizontally, the line intersects only once I1 for the line that is cast towards west and only once I2 for the line that is cast towards East, this determines that the point is inside the fence.

In the code, this is achieved by first determining the line of equation for each pair of points PQ, QR, RS and SP. Once the line equation is determined, use Cramer rule to find the intersection point from the point coordinate.

A more detailed explanation on Cramer rule is found in Cramer rule and step by step explanation is given Wolfman web Resource



Name: Delhi Durai. **Student-ID:** 948639812

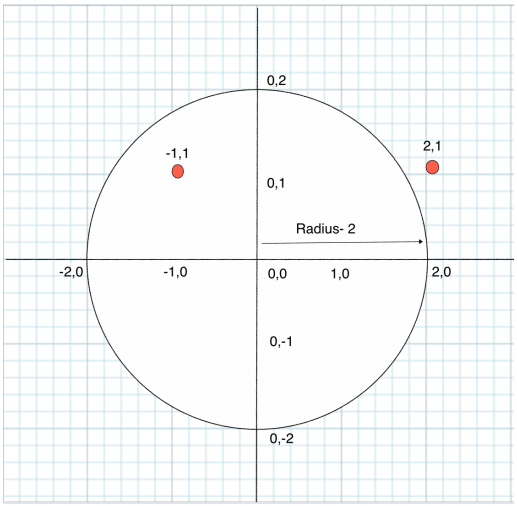
Name: Rajeswari. **Student-ID:** 916809371

RUST FINAL PROJECT – CS 510

Geofencing on a Circular fence: -

Circular geofence uses Pythagorean theorem to determine if a point coordinate is within or outside the circle.
Pythagorean theorem is based on the assumption that a circle can be defined as locus of all points that satisfy the equation $x^2 + y^2 = r^2$, where x and y are the coordinates of origin and the r is radius.
Therefore, if a point ***(p, q)*** is considered to be within the circle, then it should satisfy $(x-p)^2 + (y-q)^2 < r^2$, similarly a point p is considered out of the circle, then it should satisfy $(x-p)^2 + (y-q)^2 > r^2$
For example, the below figure shows a circular fence that originates from the point (0,0) with a radius of 2. The point (2,1) is outside the circle, that can be proved by applying the above equation
$(0-2)^2+(0-1)^2 = 5$, which is greater than $r^2 = 4$, this determines the point is outside the circle.
On the other hand, the point (-1,1) is considered to be within the circle by equating the points $(0-(-1))^2 + (0-1)^2 < 2^2$

A more detailed explanation of Pythagorean theorem on circle is explained in the link mathopenref_pytha



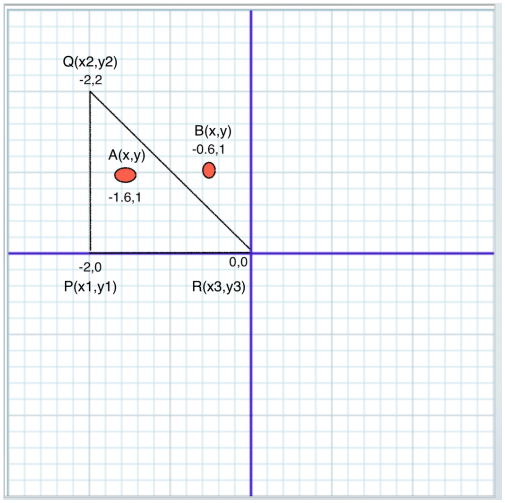
Geofencing on a Triangular fence: -

Triangular geofence determines if a point is within or outside the fence by calculating the barycentric coordinates of the point in question, with the three coordinate points that formed the triangle.
The below algorithm is used to for determining this. From the below figure lets assume P(x1,y1), Q(x2,y2) and R(x3,y3) forms a right angled triangle.

Name: Delhi Durai. **Student-ID:** 948639812

Name: Rajeswari. **Student-ID:** 916809371

RUST FINAL PROJECT – CS 510



A point [A(x,y)] is said to be within or outside the fence when it satisfies the below criteria

1. Calculate the area of PQR, PQR_Area = [x1(y2-y3) + x2(y3 – y1) + x3(y1-y2)]/2
2. Calculate the area of APQ, assign this to APQ_Area
3. Calculate the area of AQR, assign this to AQR_Area
4. Calculate the area of APR, assign this to APR_Area
5. A point is said to be within the fence if PQR_Area = APQ_Area + AQR_Area + APR_Area

Performance of the project: -

Performance testing is done on the crate with the simulated data and the below are our observations. Please note the performance test is done after removing the default 1 second delay from the execute function. The delay is introduced deliberately in the execute function to provide a real time effect for the user.

Run	Fence	Test criteria	Execution Time in milli seconds	Observations
1	Polygon	Polygon fence created using 4 different coordinates, with same number of moving target coordinate.	4.48	Execution time increases as we increase the number of coordinates required to create the fence.
2	Polygon	Polygon fence created using 10 different coordinates, with same number of moving target coordinate.	4.82	
3	Polygon	Polygon fence created using 4 coordinates, the distance between each coordinate is increased by 10% compared to Run #1	4.64	Execution time increases as the circumference of the fence increase.
4	Circle	Circular fence with radius 3	2.46	Execution time increases as we increase the radius of the fence.
5	Circle	Circular fence with radius 6	4.34	
6	Triangle	Triangular fence created using standard three coordinates, with overall fence area as 50	2.11	Execution time increases as the area of the fence increase.
7	Triangle	Triangular fence created using standard three coordinates, with overall fence area as 300	2.75	
8	Triangle	Triangular fence created using standard three coordinates, with overall fence area as 300. The moving target distance is increased by 10 %.	2.02	Don't see any major difference with the increase in distance of the moving targets with respect to the fence

Name: Delhi Durai. **Student-ID:** 948639812

Name: Rajeswari. **Student-ID:** 916809371

RUST FINAL PROJECT – CS 510

Reliability of the project and how this was assessed: -

The project is capable of performing the below operations

1. Create polygon fence using coordinates geometry and accurately validate if a given point (search coordinate) is within or outside the fence. As the project uses line equation to build the fence, the coordinates provided should be sequenced in linear fashion, which should enable to pair points with line equation appropriately.
2. Create circular fence with the given coordinates and accurately validates if a given point (search coordinate) is within or outside the fence.
3. Create triangular fence with the given coordinates and accurately validates if a given point (search coordinate) is within or outside the fence.
4. Accuracy chart of each fence search is listed below

Sno	Fence	Capability	Percentage of accuracy	Reason
1	Polygon	Ability to validate a point (coordinate) if it is with in or outside the fence, on a Polygon fence with smooth edges (with no zig zag, up and down edges)	100	
2	Polygon	Ability to validate a point (coordinate) if it is with in or outside the fence, on a Polygon fence with rough edges (when the fence is not smooth and edges has many irregularity in the shape of the fence)	95	The search some times results in unexpected output especially when the edges of the fence has too many irregularities in the shape.
3	Circle	Ability to validate a point (coordinate) if it is with in or outside the fence	100	
4	Triangle	Ability to validate a point (coordinate) if it is with in or outside the fence	100	

How this was assessed: -

1. Have created multiple unit test case to test most of the possible scenarios that can occur.
2. The test data even though created by us, will cover enough test scenarios to validate the project.

Related work: other similar projects

Inspired from: https://github.com/tidwall/tile38

References:

Polygon: https://github.com/sam-drew/picket

Circle: https://www.geogebra.org/m/hQW8KxXz
https://www.geeksforgeeks.org/find-if-a-point-lies-inside-or-on-circle/

Triangle: https://blackpawn.com/texts/pointinpoly/default.html
https://www.geeksforgeeks.org/check-whether-a-given-point-lies-inside-a-triangle-or-not/