Implementation and Analysis of Geometric Algorithms for Line Intersection and Convex Hull Problems

Ali Raza (21K-4703), Syed Saadullah Hussaini (21K-4736), Muhammad Sameed (21K-3100)
BAI-5A

Abstract

In this project, we present an implementation of geometric algorithms for solving two fundamental problems: line segment intersection and convex hull construction. The project provides a user-friendly interface allowing real-time interaction with graphical elements, enabling users to input points and line segments dynamically. The implemented algorithms include slope comparison, cross product test, and counter-clockwise (CCW) test for line segment intersection. On the other hand, Brute force, Jarvis March, Graham scan, Quick Elimination, and Andrew's Monotone for convex hull construction. The report outlines the design, experimental setup, and results of the implemented system, demonstrating the efficacy of various algorithms.

1 Introduction

Geometric algorithms play a crucial role in computer science, offering solutions for spatial problem-solving. This project delves into the implementation of key geometric algorithms in Java, focusing on line intersection and convex hull computations. An interactive GUI allows users to engage directly with these algorithms, providing a hands-on experience for understanding their mechanics and applications. The project aims to bridge theoretical concepts with practical implementation, offering insights into algorithmic efficiency and usability.

2 Programming Design

2.1 Programming Language Used

The system is implemented in Java using the Swing framework on NetBeans. The graphical interface allows users to interactively input points and line segments. A clear diagram illustrates the system architecture, depicting how the algorithms are integrated into the user interface. This section discusses the choice of programming language, GUI framework, and the overall design philosophy.

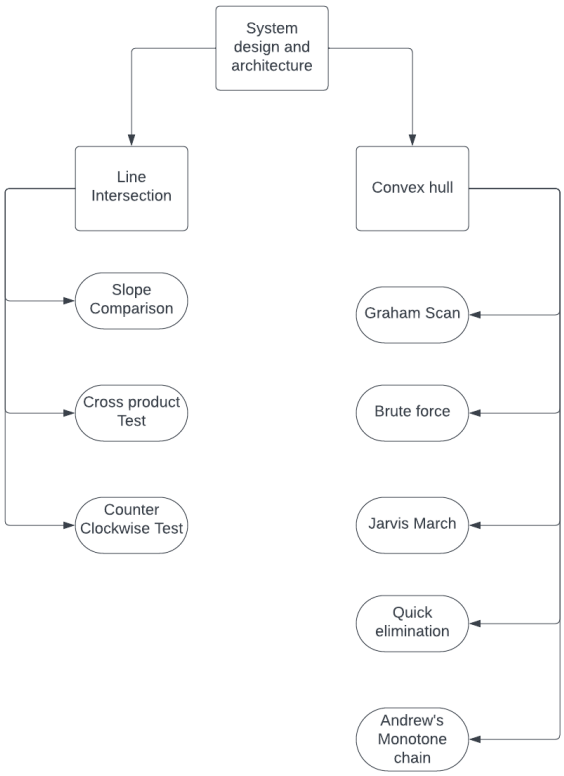

Figure 1: System Architecture Diagram

2.2 System Diagram and Discussion

The system includes a main interface for users to select between line intersection and convex hull problems. For line intersection, methods include Counter Clockwise Test, Slope Comparison, and an additional researched method i.e. Vector Cross Product Method. Convex hull algorithms include Brute Force, Jarvis March, Graham Scan, Quick Elimination, and one algorithm sourced from recent research i.e. Andrew's Monotone Chain. It comprises several key classes: ChoicePage handles user input and algorithm selection, CustomDrawingPanel manages the drawing and visualization of geometric shapes, and LineIntersect implements the logic for determining line intersections. The system's architecture is designed to facilitate easy point selection, drawing, and algorithm execution, providing a seamless user experience.

3 Experimental Setup

The experimental setup involves a user-centric interface, enabling users to interactively draw points and line segments on the screen. The application offers a variety of algorithms for users to choose from, including Brute Force, Graham Scan, Jarvis March, Quick Elimination, and Andrew's Monotone for convex hull solutions, as well as Slope Comparison, Cross Product Test, and Counter-Clockwise Test for line intersection. This setup is designed to cater to both educational and research purposes, allowing for a comprehensive analysis of each algorithm's performance under different scenarios.
System Specs: Core i7 8th gen,8GB RAM 256GB SSD.

4 Results and Discussion

4.1 Line Intersection

4.1.1. Slope Method

Here we calculate the slope intercept of two line segments using the y = mx+b formula, if lines are not parallel, they intersect else if slopes are equal they are either parallel or coincident.

4.1.2. CCW Test Method

The CCW test is a specific geometric algorithm used to determine the orientation of three points in a two-dimensional plane. Given three points, the CCW test involves calculating the cross product of two vectors formed by these points. The decision is based on the sign of the cross product: if cross product ¿ 0, the points are in a counterclockwise order; if cross product ¡ 0, they are in a clockwise order; and if cross product=0, the points are collinear.

4.1.3. Cross Product Method

Here we represent the line segments as vectors and compute its cross products.If the cross product is 0 the lines are collinear or parallel, else they intersect.

4.2 Convex Hull

4.2.1. Graham Scan Method

Starts from selecting a lower y axis point, sorting the remaining points on their polar angels and then joining the points by checking CCW orientation, if counter clockwise, add the point to the hull.

4.2.2. Jarvis March

We use the orientation approach to find the convex hull in Jarvis march algorithm i.e. we take a point a, another point b, and a point c from b. we check the orientation if its counter clockwise, point c becomes the point b and it iterates like this until it finds the first point again.

4.2.3. Quick Elimination Method

Finds extreme points, divide these points into regions. Then checks its orientation from each extreme point to the points outside the region and apply a modified graham scan approach to form the convex.

4.2.4. Andrew's Monotone Chain Method

The algorithm operates by first sorting the input points lexicographically, a key step that simplifies subsequent hull construction. The convex hull is then constructed in two stages: building the upper hull and the lower hull.

4.2.5. Brute Force Method

The brute-force convex hull algorithm iterates through all pairs of points and checks if any other points lie on the line segment between them. If no other points lie on the segment, it is added to the convex hull. Optionally, intersections between the newly added line and previously formed lines are checked and highlighted during the animation.

4.3 Execution Time and Comparison

Figure	Execution Time (s)
5	Graham Scan: 0.0017
6	Jarvis March: 4.0E-4
7	Quick Elimination: 0.0012
8	Andrew's Monotone Chain: 1.0E-4
9	Brute Force: 0.0147

Table 1: Execution Times for Convex Hull Algorithms on 505 Points

4.4 Time and Space Complexities

Algorithm	Best Case	Worst Case	Average Case	Space Complexity
Brute Force	$O(n^3)$	$O(n^4)$	$O(n^4)$	$O(1)$
Graham Scan	$O(n \log n)$	$O(n^2)$	$O(n \log n)$	$O(n)$
Jarvis March	$O(nh)$	$O(n^2)$	$O(nh)$	$O(1)$
Quick Elimination	$O(n \log n)$	$O(n^2)$	$O(n \log n)$	$O(n)$
Andrew's Monotone	$O(n \log n)$	$O(n \log n)$	$O(n \log n)$	$O(n)$

Table 2: Time and Space Complexity of Convex Hull Algorithms (h is the number of vertices in the convex hull)

Algorithm	Best Case	Worst Case	Average Case	Space Complexity
Slope Comparison	$O(1)$	$O(1)$	$O(1)$	$O(1)$
Cross Product Test	$O(1)$	$O(1)$	$O(1)$	$O(1)$
CCW Test	$O(1)$	$O(1)$	$O(1)$	$O(1)$

Table 3: Time and Space Complexity of Line Segment Intersection Algorithms

5 Conclusion

The project underscores the blend of theoretical algorithms with practical software solutions, highlighting the significance of geometric computations in real-world applications. The successful execution of these algorithms in a user-friendly Java application paves the way for future exploration, particularly in optimizing these algorithms for high-performance computing and expanding their applicability in fields like artificial intelligence and machine learning.

6 References

1. "Andrew's Monotone Chain Convex Hull Algorithm," Wikibooks. [Online]. Available: `https://en.wikibooks.org/wiki/Algorithm_Implementation/Geometry/Convex_hull/Monotone_chain`

7 Screenshots

Figure 2: Main Page

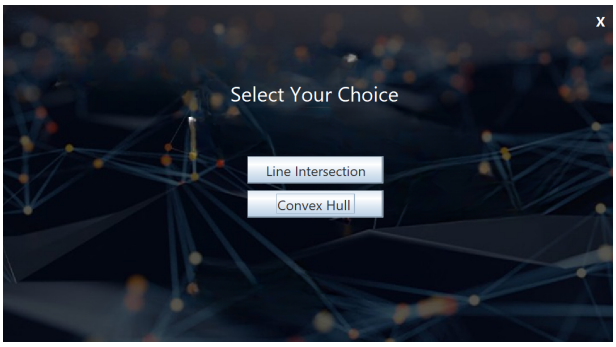

Figure 3: Choice Page

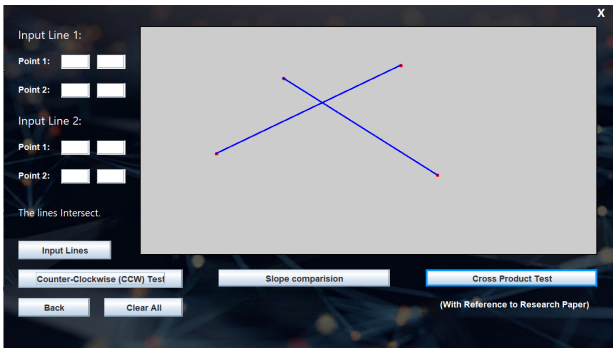

Figure 4: Line Intersection

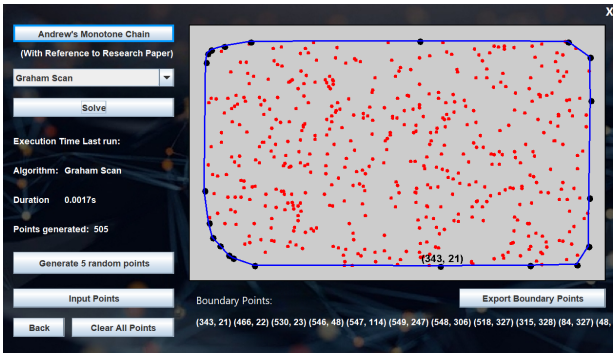

Figure 5: Graham Scan

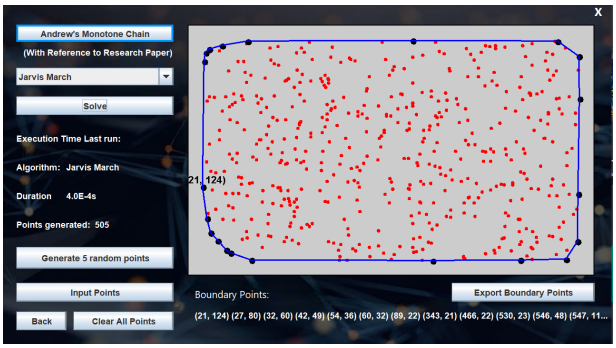

Figure 6: Jarvis March

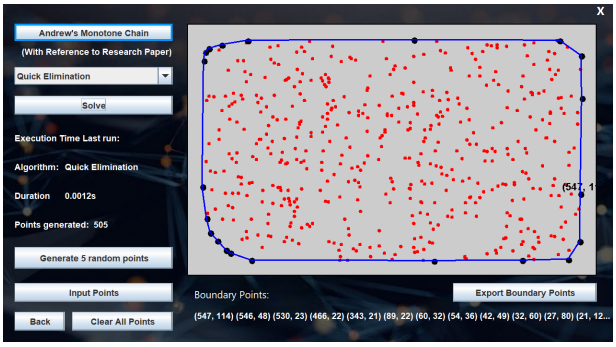

Figure 7: Quick Elimination

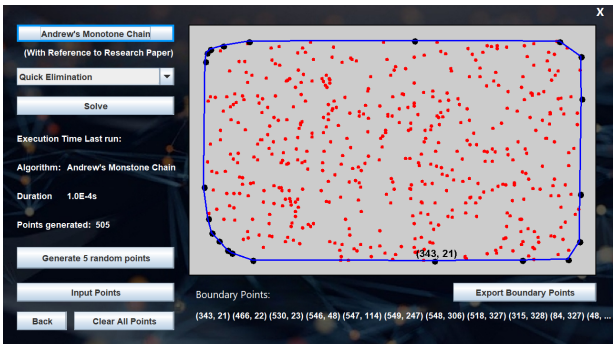

Figure 8: Andrew's Monotone Chain

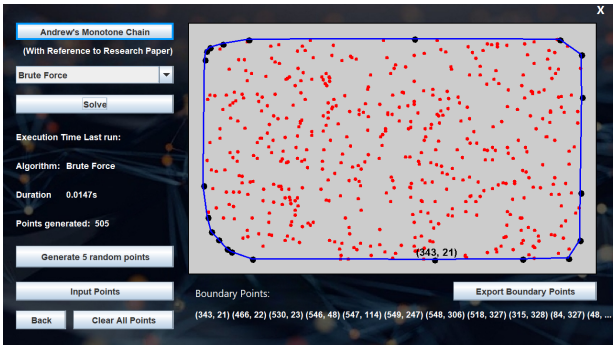

Figure 9: Brute Force