

PORTLAND STATE UNIVERSITY

CS350

ALGORITHMS AND COMPLEXITY

---

# ConvexHull Analysis: BruteForce vs. QuickHull

---

*Author:*

Wes RISENMAY  
Josh WILLHITE

*Instructor:*

Dr. Andrew BLACK

March 11, 2014

## Contents

<b>1</b>	<b>Objective</b>	<b>1</b>
<b>2</b>	<b>ConvexHull Overview</b>	<b>1</b>
<b>3</b>	<b>Algorithm Explanation</b>	<b>1</b>
3.1	Brute Force . . . . .	1
3.2	QuickHull . . . . .	1
<b>4</b>	<b>Algorithm Implementation</b>	<b>1</b>
4.1	Brute Force . . . . .	1
4.2	QuickHull . . . . .	3
<b>5</b>	<b>Analysis of Complexity</b>	<b>3</b>
5.1	Brute Force . . . . .	3
5.2	QuickHull . . . . .	3
5.3	Expected Outcome . . . . .	3
<b>6</b>	<b>Automated Testing</b>	<b>3</b>
6.1	Algorithm Correctness . . . . .	3
6.2	Data Generation . . . . .	3
6.3	Timing Execution . . . . .	3
<b>7</b>	<b>Results</b>	<b>3</b>
<b>8</b>	<b>Conclusion</b>	<b>3</b>

## 1 Objective

## 2 ConvexHull Overview

## 3 Algorithm Explanation

### 3.1 Brute Force

### 3.2 QuickHull

## 4 Algorithm Implementation

### 4.1 Brute Force

As the name implies the brute force solution to the convexhule problem is very straight forward, as shown below in Figure 1.

```

public static LinkedList<Point2D> bruteForceConvexHull(Point2D points[]) {
    LinkedList<Point2D> convexHull = new LinkedList<Point2D>();

    double curr_x_prod, prev_x_prod;
    boolean on_hull;
    for(Point2D a: points) {
        for(Point2D b: points) {
            if(a != b) {
                prev_x_prod = Math.PI;
                on_hull = true;
                for(Point2D c: points) {
                    if(c != a && c != b) {
                        curr_x_prod = ((a.getX() - b.getX())*(b.getY() - c.getY()) - (b.getX() - c.getX())*(a.getY() - b.getY()));
                        if(curr_x_prod > 0) {
                            curr_x_prod = 1;
                        }
                        else if(curr_x_prod < 0) {
                            curr_x_prod = -1;
                        }
                        else {
                            curr_x_prod = 0;
                        }
                        if (curr_x_prod == prev_x_prod || prev_x_prod == Math.PI || curr_x_prod == 0) {
                            prev_x_prod = curr_x_prod;
                        }
                        else {
                            on_hull = false;
                            break;
                        }
                    }
                }
                if(on_hull) {
                    if(!convexHull.contains(a)) {
                        convexHull.add(a);
                    }
                    if(!convexHull.contains(b)) {
                        convexHull.add(b);
                    }
                }
            }
        }
    }

    return convexHull;
}

```

Figure 1: Brute Force Convex Hull

4.2	QuickHull
5	Analysis of Complexity
5.1	Brute Force
5.2	QuickHull
5.3	Expected Outcome
6	Automated Testing
6.1	Algorithm Correctness
6.2	Data Generation
6.3	Timing Execution
7	Results
8	Conclusion