Bonus Assignment: Implementing Convex Hull Algorithms

Sylvia Chen
9752129

Grace Wang
2736411

ABSTRACT

In this paper, we implement the Jarvis March and Graham Scan algorithm to compute the convex hull of a set of n points in R^2 and evaluate them by some experiments. The result shows that for the small size convex hull, especially the number of vertices is less than 20, Jarvis March is faster, and Graham Scan is a better choice for the other size of convex hull.

1 INTRODUCTION

A subset S of the plane is called *convex* if and only if the line segment $\overline{pq}$ is completely contained in S and the **convex hull** $CH(S)$ is the smallest convex set contains S for any pair of points $p, q \in S$ [3]. The problem of constructing convex hull of a set of n points perhaps is the most studied problem in computational geometry [5].

2 IMPLEMENTATION

Let $S = \{p[1], ..., p[n]\}$ be a set of n distinct points in R^2 and $CH(S)$ be the convex hull of S, we say that a point p is "on the convex hull of S" if p is a vertex of $CH(S)$ and L is a list contains the vertices of $CH(S)$. In this section, two of the convex hull algorithms, Jarvis march and Graham scan, were implemented in **Matlab**.

2.1 Jarvis March Algorithm

The Jarvis March algorithm [7], also known as the Gift-wrapping algorithm [4] starts from a point p_1 which is certainly vertice of $CH(S)$. Then, select the point which is closest to the outside in the same direction and add them to L one by one. When returning to the original point p_1, the selected point contained in L is the required convex hull. To be more specific, we start from the leftmost point and we keep wrapping points in clockwise direction. Here we use *Cross product* of vectors to determine the outermost point (see 2.3 for more details).

2.2 Graham Scan Algorithm

The Graham Scan algorithm [6] is an incremental algorithm. We first sort the points. Then, we compute vertices that lie on the *upper hull* performed and the *lower hull* respectively [3]. More details have already been described in the lecture and we also use *Cross product* of vectors to determine left or right turn here.

2.3 Cross Product

In two dimensions, the analog of the cross product for $u = (u_x, u_y)$ and $v = (v_x, v_y)$ is [1]:

$$\mathbf{u} \times \mathbf{v} = u_x v_y - u_y v_x \tag{1}$$

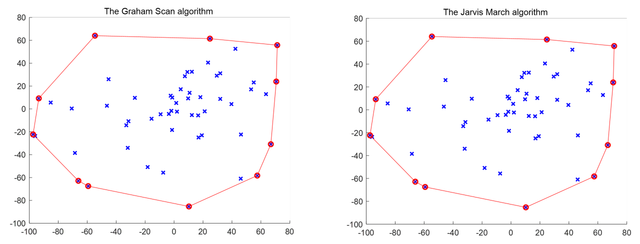
The symbol of the cross product represents the direction in which the vector rotates. Specifically:

* $\vec{u} \times \vec{v} = 0$, u,v are col-linear.
* $\vec{u} \times \vec{v} > 0$, v is to the left of u (or we can say u rotate counterclockwise towards v).
* $\vec{u} \times \vec{v} < 0$, v is to the right of u (or we can say u rotate clockwise towards v).

Until now, we can already use this properties for both algorithms. For the Graham Scan algorithm, assume the last three points be A,B) and C. Then, if $\vec{AB} \times \vec{BC} < 0$, it means the last three points makes a right turn; otherwise they makes left turn. For the Jarvis March algorithm, let point J be the start point and we first compare the relative relationship between point K and point I. Then if $\vec{JK} \times \vec{JI} < 0$, it means I is closer to the outside than K. The comparison of two vectors is transitive. Therefore, for every point p_i, we compare it with the existing point $p_{mostout}$ which is the closest to the outside. If p_i is closer than $p_{mostout}$, we update $p_{mostout}$.

2.4 Correctness

In order to test the correctness of the implemented convex hull algorithm, we set up a drawing function. The input of this function is S and L. We first draw the points in S as a scatter plot. Then we connect the points in L one by one to see whether L constitutes a convex hull. Fig.1 is the convex hull obtained by the two algorithms using the same random points.



(a) The Graham Scan algorithm

(b) The Jarvis March algorithm

Figure 1: Correctness of Two Algorithms

Comparing Fig.1 (a) and (b), for the same point set, the two convex hulls are exactly the same, and conform to the definition of convex hull.

2.5 Special Case Handling

Utill now, we have not dealt with the col-linearity problem, that is, $\vec{AB} \times \vec{BC} = 0$ or $\vec{JK} \times \vec{JI} = 0$. For the Graham Scan algorithm, if the three points A, B, and C are col-linear, delete point B and keep the two points A and C. Specifically, $\vec{AB} \times \vec{BC} < 0$ is rewritten as $\vec{AB} \times \vec{BC} <= 0$. And for the Jarvis March algorithm, when taking the outer point, if the three points J, K, and I are col-linear, we take the point farther from J. Fig.2 gives an example of handling the special cases.

3 TIME COMPLEXITY IN THEORY

According to the theory, the overall time complexity of Jarvis March algorithm is $O(nh)$, where h is the number of vertices in the convex

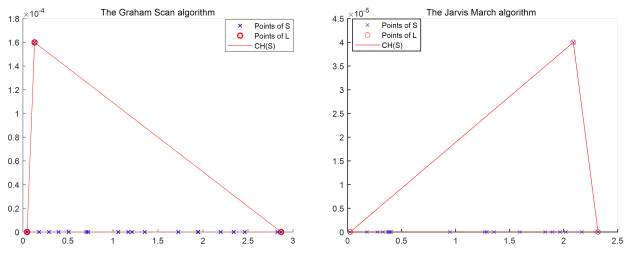


Figure 2: Special Case of Two Algorithms

hull. In the worst case, h equals n and the complexity is $O(n2)$ [2]. Besides, the time complexity of Graham Scan algorithm is $O(n \log n)$ [3].

4 EXPERIMENT RESULT AND DISCUSSION

In this section, we build different data sets and test the running time of the two algorithms.

4.1 The General Data Set

In the general data set, we take (x_0, y_0) as the center of the circle and R as the radius, and generate N random points **inside** the circle. We set N from 50 to 8000 and each algorithm run five times to get the mean of running time. Fig.3 shows the result.

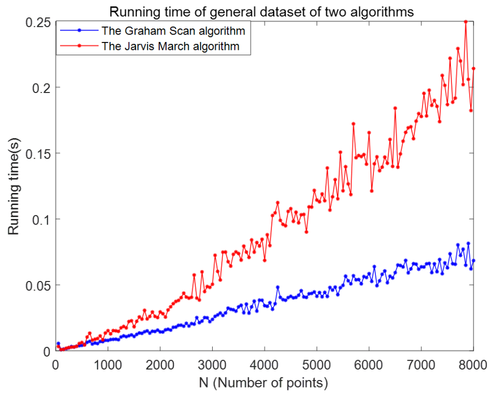


Figure 3: Running Time of General Data Set

From Fig.3, as the number of random points N increase , the running time of the two algorithms increase. However, with the same N the running time of the Graham Scan algorithm is lower than Jarvis March algorithm and the difference between the two algorithms with same N increase as N increase.

4.2 Comparing Data Sets

According to section 3, the time complexity of Jarvis March algorithm is related to the number of convex hull vertices. To verify this theory, we set up two comparative data sets.

In the first data set, we generate N random points **on** the circle. In other words, the number of random points input is equal to the number of convex hull vertices. So, at this time, the Jarvis March algorithm is in its worst situation, that is, $h = n$. The result is shown in Fig.4. As we can see, as the number of input points, all of which

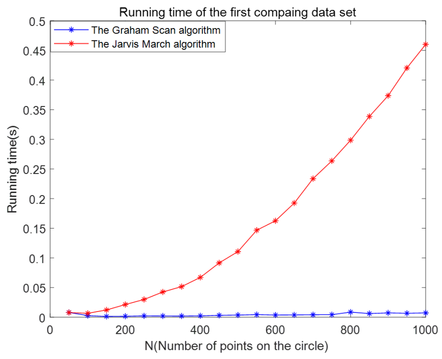


Figure 4: Running Time of The First Data Set

are in the convex hull, goes up, the running time of the Graham Scan algorithm has not been greatly affected, while the running time of the Jarvis March has increased rapidly.

In the second data set, we generate N' random points located inside a circle $C1$ with a radius of $R1$ and n random points located on a circle $C2$ with a radius of $R2$. Here, $C1$ and $C2$ has the same center (x_0, y_0) and $R1 < R2$ and our data set N includes $N' + n$ random points. We set n to a fixed value, increase N, and test the two algorithms separately. Besides, we do the linear fit in order to better observe the trend of change.

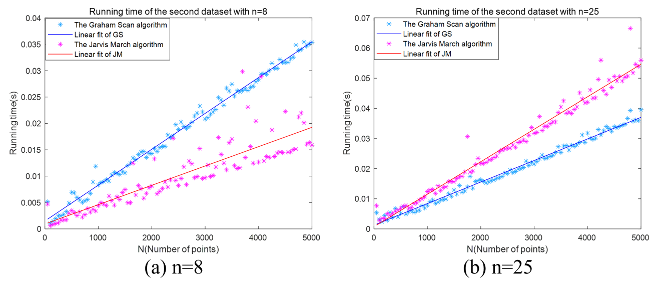


Figure 5: Running Time of The Second Data Set

As we can see from Fig.5, the running time of Jarvis march algorithm is lower than the Graham Scan when the value of n (size of convex hull) is small (n<=20).

5 CONCLUSION

From the experiments, it can be concluded that if the size of the convex hull is known to be small , Jarvis march is faster than Graham. However, if the size of the convex hull and the number of input points are close, that is, the input points are mostly distributed on the convex hull, Jarvis March will be much slower.

In summary, we recommend using Jarvis march algorithm for datasets with small convex hull sizes, and here ’small’ means no more than 20. For other cases, or when the size of the convex hull is unknown, we recommend to use Graham Scan algorithm.

REFERENCES

[1] George B Arfken and Hans J Weber. *Mathematical methods for physicists*. 1999.

[2] Maher Atwah, Johnnie W Baker, and Selim Akl. "An associative implementation of classical convex hull algorithms". In: *Proc. of the Eighth IASTED International Conference on Parallel and Distributed Computing Systems*. 1996, pp. 435–438.

[3] Mark de Berg et al. *Computational Geometry Algorithms and Applications*. Springer, 2008.

[4] Thomas H Cormen et al. *Introduction to algorithms*. MIT press, 2009.

[5] Mujtaba R Ghouse and Michael T Goodrich. "In-place techniques for parallel convex hull algorithms (preliminary version)". In: *Proceedings of the third annual ACM symposium on Parallel algorithms and architectures*. 1991, pp. 192–203.

[6] Ronald L. Graham. "An efficient algorithm for determining the convex hull of a finite planar set". In: *Info. Pro. Lett.* 1 (1972), pp. 132–133.

[7] Ray A Jarvis. "On the identification of the convex hull of a finite set of points in the plane". In: *Information processing letters* 2.1 (1973), pp. 18–21.