COSC262 Algorithms

Assignment: Convex Hulls – Thomas Knight

# Introduction

For this assignment, three different algorithms will be implemented for computing convex hulls of two-dimensional points using the datasets provided. These algorithms comprise of the Gift Wrap, the Graham-Scan and Andrew’s Monotone Chain algorithm. The aim of this report is to discuss and compare the implementation and analysis of these algorithms.

# Algorithm implementation

# Gift Wrap algorithm

The Gift Wrap algorithm constructs the convex hull by iterating through a set of input points to locate the next point to be added to the convex hull. The concept of the Gift Wrap algorithm is to start from the point with the minimum y value that is rightmost and keep wrapping points in counter clockwise direction.

For this assignment a Python list was used to implement the Gift Wrap algorithm which started off as a copy of all the input points. Three key variables were used, to indicate the current list index, minimum angle and the swap index in the list.

To begin the algorithm an O(n) operation was made to locate the starting point which would be the point with the minimum y value that was rightmost. Once the starting point was located it was interchanged with the first point in the list and then appended to the end of the list

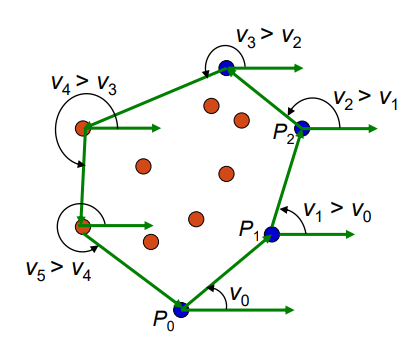
Once the initialisation stage was completed, the Gift Wrap algorithm then iterated through the rest of the points in the list. This method would then locate the next point in the list to ‘wrap’ by determining which point gave the minimum angle with the current point. Another condition to note is that the current minimum angle must be greater than the previous minimum angle as shown in Figure 1, otherwise a point inside the convex hull is selected. Once the next point was found it was then interchanged with the point at the current index in the list. This point then became the current point and was used to find the next point.

Figure 1 - choosing a minimum angle greater than the previous minimum angle

This method keeps iterating until the next point found in the list is the starting point meaning all the points on the convex hull have been established.

The complexity of this algorithm is O(nm) where n is the number of input points and m is the number of points on the convex hull. This makes the Gift Wrap method an output sensitive algorithm where it is favourable when n is small of m is expected to be very small in proportion to n. In general cases this algorithm is outperformed by others.

# Graham-Scan algorithm

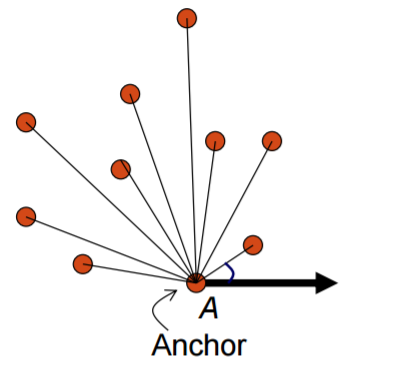
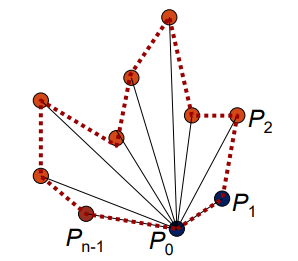
The Graham-Scan algorithm functions by using stacks as well as lists and used a pre-processing step before computing the convex hull. Like that of the Gift Wrap method, the Graham-Scan method also starts from the point with the minimum y value that is rightmost. Once the starting point is selected the algorithm then sorts the remaining points corresponding to the angle they make with the starting point as shown in Figure 2.

Figure 2 - Sorting the remaining input points corresponding to the angle with the starting point

After the input points are sorted, this method would then remove the first three points including the start point from the sorted list and onto a stack. The algorithm them iterates through the points in the sorted list and checks if the current point forms a counter-clockwise turn with the previous two points on the stack. If the current point does from a counter-clockwise turn with the previous two points on the stack, then the current point gets pushed onto the stack otherwise this method would pop the last point off the stack until a counter-clockwise turn is made.

This method keeps iterating until all the points in the sorted list have been processed and the algorithm returns to the starting point in which then the stack contains all of the points of the convex hull.

The complexity of this algorithm is O(nlogn) where n is the number of input points. The complexity of the pre-sorting stage using the Python sort function is O(nlogn) but the complexity of constructing the convex hull is O(n) as this operation just iterates through the list of input points checking if the current point forms a counter-clockwise turn with the previous two points on the stack.

# Monotone Chain algorithm

The Monotone Chain algorithm can be seen as a variant of the Graham Scan algorithm in which the points are sorted lexicographically by their coordinates and then upper and lower hulls of these points are constructed. Alike the Graham scan method the points are sorted using the Python sort function however they are done so by x-coordinate (in the case of a tie, sort by y-coordinate). Two Python lists were used the contain the points of the upper and lower hulls.

An upper hull is the part of the convex hull that is visible from above and runs from the rightmost point to the leftmost point in counter-clockwise order. The lower hull is the remaining part of the convex hull. This can be see as shown by Figure 3

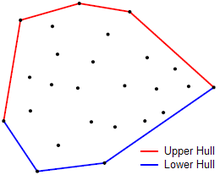


Figure 3 - Upper and lower hulls of a set of points

The upper hull was constructed by iterating through the sorted list of points in reversed order and checking if the current point forms a counter-clockwise turn with the previous 2 points. If the current point does not form a counter-clockwise turn, then the last point is popped off the upper hull list. After this check the current point is then added to the upper hull list and the method repeats until n points have been processed. The lower hull is constructed in the same fashion however the sorted list of points is not in reversed order. Once both the upper and lower hulls have been made, the last point of each list is removed (it’s the same as the first point of the other list) and then the lists are concatenated together to obtain the convex hull of the input points.

The complexity of this algorithm is O(nlogn). Similar to that of the Graham Scan algorithm it takes O(nlogn) time to sort the points using the Python sort function, and then constructing the upper and lower hulls take O(n) time.

# Algorithm analysis

The two data sets set A and set B vary in the proportion of input points and points on the convex hull. In set A the number of points on the convex hull remain roughly the same as the total number of input points increase however in set B the number of points on the convex hull increase linearly as the total number of input points does. This can be seen from Figure 4.

Figure 4 - Number of points in the set vs number of vertices on hull

As discussed previously the complexity of the Gift Wrap method is O(nm) whilst the complexity of the Graham-Scan method is O(nlogn). This suggests that in general the Graham-Scan method is faster than the Gift Wrap method, which can be seen when testing the time complexity of these two algorithms on Set B shown by Figure 5.

Figure 5 - Complexity between Gift Wrap and Graham-Scan on Set B

On the other hand, when using Set A where the amount of points on the convex hull are small, both the Gift Wrap method and the Graham-Scan method result in similar times as shown by Figure 6. This is due to the fact that the Gift Wrap algorithm is output sensitive, meaning the complexity depends partially on the number of points on the convex hull. When the number of points on the convex hull is small in comparison to the amount of input points like in Set A the algorithm is faster compared to when the number of points on the convex hull is large in comparison to the amount of input points like in Set B.

Figure 6 - Complexity between Gift Wrap and Graham-Scan on Set A

As discussed previously the complexity of both the Graham-Scan methods and the Monotone Chain methods are O(nlogn). The general time taken for these methods can be seen from Figure 7 where they both result in almost identical times, both being much faster than the Gift Wrap method.

Figure 7 - Complexity between all three algorithms on Set B

# 

Despite the fact that when using Set B, the Graham-Scan and Monotone Chain methods perform similarly, when using Set A it is apparent that the Graham-Scan method has a slightly better performance than the Monotone Chain method as shown by Figure 8. This is because the Monotone Chain algorithm iterates through the sorted input list twice to construct both the upper hull and lower hull. However both methods still perform better than the Gift Wrap method.

Figure 8 - Complexity between all three algorithms on Set A

# Conclusion

In summary, the objective of this report was to implement and analyse separate methods for constructing convex hulls. Three algorithms were used, namely the Gift Wrap method, the Graham-Scan method and the Monotone Chain method. These methods were applied on two different sets of input points, Set A and Set B which had a varying proportion of the number of total points to the number of points on the convex hull.

Using these different sets of input points, the complexity of these algorithms differs. Using Set A where the number of points on the convex hull remain roughly the same as the total number of input points all three methods perform very similarly. The reason for this is that the Gift Wrap method is an output sensitive algorithm and results in a faster time when the number of points on the convex hull is small.

However when using Set B where the number of points on the convex hull increase linearly as the total number of input points does, the Gift Wrap method which has a complexity of O(nm) is processed notably slower than the Graham-Scan and Monotone Chain methods which have a complexity of O(nlogn). Furthermore although the Monotone Chain algorithm is slightly slower than the Graham-Scan algorithm, as it must iterate through the list of sorted input points twice to construct both the upper and lower hulls, in general the algorithms both result in almost identical times.

# References

<http://learn.canterbury.ac.nz/mod/resource/view.php?id=130112> - S1\_1\_ConvexHull.pdf Lecture slides

<https://en.wikibooks.org/wiki/Algorithm_Implementation/Geometry/Convex_hull/Monotone_chain>