ECM3428: Algorithms that changed the world - Continuous Assessment I

Quickhull

Student number: 690062247

Abstract

Convex hull problems have many applications including hand movement tracking and sports analysis. Quickhull is a divide-and-conquer algorithm capable of solving these problems, allowing the wide range of applications to be efficiently utilised. This report discusses the principals, complexity and applications of the quickhull algorithm.

Word count: 1104

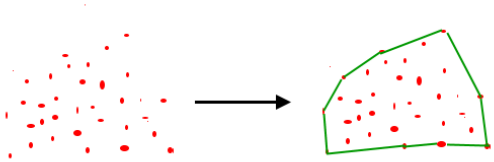
I certify that all material in this report which is not my own work has been identified.

1 The quickhull algorithm

1.1 Main principals

Quickhull is an algorithm used to compute the convex hull of a finite set of points, the smallest convex set so that all of the points are included in the lines of the convex or bounded within it - [1] See Figure 1 for example.

Figure 1: Example of the convex hull of a set of points in two dimensions - [2]



It is a divide-and-conquer algorithm, an algorithm that recursively breaks down a problem into multiple similar, smaller sub-problems that are easier to directly solve than the original, until all the solutions may be recombined to provide the solution to the original problem. This divide-and-conquer approach is similar to that employed by quicksort which is where quickhull derives its name from.

First introduced for two dimensional convex hulls in 1990 by J.S. Greenfield in - [3], it was extended to work in N dimensional space in 1996 by Barber et al in - [1].

For two dimensional problems the algorithm works by finding the two points with maximum and minimum x or y coordinates as it is known that these points must be a part of the convex hull. The line formed by these two points bisects the plane and is then used to divide the set into two subsets of points that will be solved recursively. The point with the largest perpendicular distance from the line is then found on one side, forming a triangle with the bisecting line. The points that are contained within the triangle can now not be a part of the convex hull, allowing them to be ignored going forward. The previous steps of finding the point with the farthest perpendicular distance is then applied to the two new lines created by the triangle. This process is repeated until there are no more points remaining outside and the recursion has stopped, the points that have been selected will make up the convex hull - [3].

For any number of dimensions greater than two, the problem becomes much more complex as a result of the increased number of planes involved.

1.2 Pseudo code

The pseudo code for two dimensional quick hull is shown in algorithm 1.

Algorithm 1 Quickhull for a two dimensional problem - [4]

Input = a set S of n points.
Output = Convex Hull.
Ensure: There are at least 2 points in the input set S of points
procedure QuickHull(S) ▷ Find convex hull from set S of n points.
{
ConvexHull := {}.
Find left and right most points, A and B.
Add A and B to convex hull.
Segment AB divides the remaining (n-2) points into 2 groups S1 and S2:
Where S1 are points in S that are on the right side
of the oriented line from A to B.
Where S2 are points in S that are on the right side
of the oriented line from B to A.
FindHull (S1, A, B).
FindHull (S2, B, A).
}
end procedure
procedure FindHull(Sk, P, Q)
{ ▷ Find points on convex hull from the set Sk of points that are on the right side of the oriented line from P to Q
If Sk has no points
Return.
From the given set of points in Sk, find farthest point, C, from segment PQ.
Add point C to convex hull at the location between P and Q.
P, Q and C partition remaining points of Sk into 3 subsets: S0, S1, and S2.
Where S0 are points inside triangle PCQ.
Where S1 are points on the right side of the oriented line from P to C.
Where S2 are points on the right side of the oriented line from C to Q.
FindHull(S1, P, C).
FindHull(S2, C, Q).
}
end procedure

1.3 Complexity analysis

1.3.1 Time complexity

For problems in which the number of dimensions is less than or equal to three the best case scenario time complexity is $O(n \log n)$ where n is the number of points in the set S. This scenario occurs when the points are always divided evenly, resulting in a smaller ratio of points of the convex hull compared to the number of points contained within the convex hull. This best case time complexity is also the time complexity that is expected to occur in the majority of cases - [3].

The worst case scenario time complexity for dimensions less than or equal to 3 is $O(n^2)$, this will occur in cases where every point is a part of the convex hull and thus there are no points contained within the convex hull - [3].

1.3.2 Space complexity

The space complexity for quickhull with n points, m of which are vertics of the convex hull is $O(n)$ as every time the function is called it will result in the points being partitioned into three subsets, one for each of the two recursive calls of the function on either side of the lines and one for the points contained within. The algorithm will require $O(m)$ calls to complete so this is the maximum number of records that may exist at one time, thus the maximum memory needed at any time will be $O(n)$.

1.4 Applications of quickhull in diverse areas for problem solving

Quickhull is a method that may be used to calculate convex hulls and therefore has many applications to any area in which a convex hull is of use. One such area is in sports analysis such as in football, convex hulls may be used to analyse data about the game such as player positing and movement, one application. of this is to identify player movement and playing patterns, see [5] and [6] for two examples of this in use.

Another application is the area of computer graphics in which convex hulls may be used to simplify object collision meshes in animated scenes. They may also be used in hand and gesture detection, see [7] and [8] for example implementation. This has a range of possible uses, one of which is in casinos, where tracking the dealer and players hand can provide data about the game.

A different application is in animal home-range analysis, data is collected about an animals position at regular intervals and a convex hull is then created using these points in order to estimate the range in which the animal lives. This data allows a better understanding to be gained about the animals being tracked and can be applied in varies areas including conservation. A study in which this is utilised is [9].

2 References

References

[1] C. B. Barber, D. P. Dobkin, and H. Huhdanpaa, "The quickhull algorithm for convex hulls," *ACM Transactions on Mathematical Software (TOMS)*, vol. 22, no. 4, pp. 469–483, 1996.

[2] 2021. [Online]. Available: https://www.geeksforgeeks.org/convex-hull-set-1-jarviss-algorithm-or-wrapping/

[3] J. S. Greenfield, "A proof for a quickhull algorithm," *Electrical Engineering and Computer Science - Technical Reports. 65.*, 1990.

[4] 2021. [Online]. Available: https://iq.opengenus.org/quick-hull-convex-hull/

[5] F. A. Moura, L. E. B. Martins, R. O. Anido, P. R. C. Ruffino, R. M. Barros, and S. A. Cunha, "A spectral analysis of team dynamics and tactics in brazilian football," *Journal of sports sciences*, vol. 31, no. 14, pp. 1568–1577, 2013.

[6] R. Metulini, M. Manisera, and P. Zuccolotto, "Sensor analytics in basketball," *arXiv preprint arXiv:1707.01430*, 2017.

[7] A. Pradhan and B. Deepak, "Obtaining hand gesture parameters using image processing," in *2015 International Conference on Smart Technologies and Management for Computing, Communication, Controls, Energy and Materials (ICSTM)*. IEEE, 2015, pp. 168–170.

[8] A. Dhawan and V. Honrao, "Implementation of hand detection based techniques for human computer interaction," *arXiv preprint arXiv:1312.7560*, 2013.

[9] B. J. Worton, "A convex hull-based estimator of home-range size," *Biometrics*, pp. 1206–1215, 1995.