EUROPEAN ACADEMIC RESEARCH
Vol. II, Issue 6/ September 2014

ISSN 2286-4822
www.euacademic.org

Impact Factor: 3.1 (UIF)
DRJI Value: 5.9 (B+)

Minimum Bounding Containers of 2D Convex Polygon

ASIA MAHDI NASER ALZUBAIDI
Computer Science Department, College of Science
Karbala University, Karbala
Iraq

Abstract:

In Computational Geometry the smallest enclosing shape issue is that of discovering the bounding volume, territory, or edge that outlines a set of objects. The issue of least bounding containers can ubiquitous in differing set of applications for example, collision avoidance, hidden object detection and to accelerate programming execution case in point a straightforward test with an uncomplicated bounding holder can regularly prohibit the likelihood of conjuring a computationally extravagant convergence or regulation calculations, and no further inefficient calculation is required. moreover, the handiness of minimum bounding happens in a mixture of modern applications like pressing and ideal design outline. The calculation could be connected to numerous different fields, running from a clear thought of whether an item will fit into a decided beforehand holder, or whether it might be produced using standard estimated stock. In this paper, we describes a method for determining the minimum bounding container such as bounding box , bounding ball and bounding ellipsoid of a set of 2D solid convex polygon based on divide and conquer technique. The suggested method consists of three steps. Firstly, generate the 2D points set and determine the convex hull polygon that enclosing the given 2Dpoints by dividing them in two equal sets. Then, we compute the convex hull for each set depend on Andrew's Monotone Chain algorithm. Secondly, we use Common Tangent Algorithm to obtain one convex hull by constrict the outlines of convex polygon. the third step of proposed scheme includes finding *the minimum enclosing bounding area of convex polygon bounding. the experimental analysis and results of presented algorithm find that the computational time is significant for any number of vertices of geometric shape.*

Key words: Divide-Conquer algorithm, Convex hull, Monotone Chain algorithm, Common Tangent algorithm.

1. Introduction

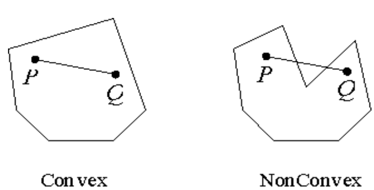
The problem of determining the minimum bounding containers for set of objects in m-dimension has been developed and adopted for use in many areas such as in computational geometry for example in collision detection of objects or in collision avoidance, hidden object detection, in finding area of object, and in an approximation object in spatial indexes like the R-Tree and its alternative. Also, it can be used in industrial applications likes packing and optimum layout design. Moreover, it found in 2D applications of mathematics likes the calculate the placement of an emergency facility for a set of customer modeled as demand points(Roth 2009).In particular calculating the minimum bounding volume containers of shapes has received much attention by researchers because it is the most frequently approximation methods used for spatial indexes of complicated objects (Rye 2014). In this paper we give an insight for computing the minimum bounding box, bounding ball and bounding ellipsoid of 2D convex polygon. So, the proposed algorithm began by generate the 2D points set and then construct the convex hull polygon using divide concept of them in two equal sets and conquer algorithm to determine the convex hull of each set depend on Andrew's Monotone Chain approach. Then we use Common Tangent Line approach to mix the two hulls and construct one convex polygon. the final step of proposed scheme includes finding the minimum bounding box, minimum bounding ball and minimum bounding ellipsoid of convex polygon. This introduction gave a brief outline over the bounding containers algorithms and the rest of this paper is arranged as following: After a discussion of related backgrounds and provides some information about the algorithm's history in Section 2, section 3 discuss the Andrew's Monotone Chain convex hull algorithm. While in section 4, we review the common tangent method that used to merge the two convex hulls in one hull . The actual bounding containers algorithm will be presented in section 5. Section 6 gives the experimental results of suggested algorithms with test examples and section 7 provides the paper's conclusions.

2. Literature Survey

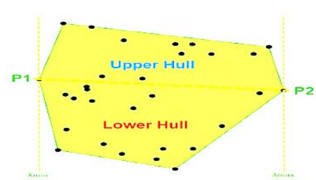
The minimum bounding containers problem was study in widely manner for instance in(Chan et.al.1999) proposed a method for determining the minimum bounding box of an three-dimensional solid shape, this scheme simplifies the complexity of 3D shape by using projection technique of orientations of the contours These orientations are used to approximate the orientation of the solid shape so that its bounding box volume is minimized . While in 2007 both Michael and Yildirim study the problem of computing an approximation to the minimum volume bounding ellipsoid of a convex hull based on symmetric concept with Khachiyan's barycentric coordinate descent technique then the paper described the method used to compute an approximation to the minimum-volume enclosing ellipsoid of the convex hull. They also used dropping technique leads to the same complexity analysis and return smaller core sets as Khachiyan's barycentric method. (Jambawalikar et.al.2008) proposed an algorithm to compute an approximate minimum volume ellipse containing a set of randomly generated ellipses ,they use an augmented real number is the exponent of square matrix multiplication model of computation to perform time complexity as Yildirim and Kumar algorithm.

3. Convex Hull Via Andrew's Monotone Chain Algorithm

Constructing a convex hull is one of the first important geometry algorithms. there are various computing convex algorithms and the most common form of this geometric algorithms involves determining the convex hull of set of m-dimension points in plane using bounding boxes techniques(Sacristan 2012). Mathematically, set S of points P ∈ R^m is named convex and bounded if for any pair of vertices a and b such a,b ∈ P then, ab line segment lies entirely in S, as shown in figure(1). In R^2 to implement the convex hull for Andrew's Monotone Chain scan algorithm we need to use stack structure. But in practice, Andrew's algorithm will execute slightly faster than other algorithms such as Graham scan (Sunday 2012). Andrew's Monotone Chain convex hull algorithm is an algorithm which creates convex hull of a set of 2D points in O(nlogn) time duo to the needing of firstly sort the points lexicographically by their x-coordinate and then constructing upper hull from its rightmost point p_2 to the leftmost point p_1 in counterclockwise order and create the lower hulls of the points from leftmost point to rightmost point in the same orientation with O(n) time(Sacristan 2012). As depicted in figure(2).



Figure(1) Convex and Non Convex hull



Figure(2) Andrews Monotone Chain Algorithm

To build the upper and lower convex hull we need to iteratively test wither the points according to p1p2 line is above, on or under it by fast accurate computation of cross_ Direction routine

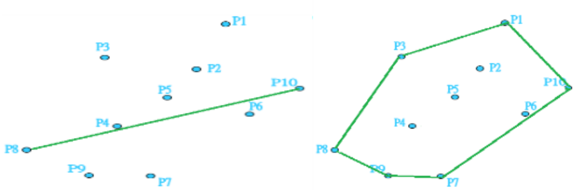
Cross Direction Algorithm (Sunday 2012).
Input: three points P0, P1, and P2 .
Output: value
> 0 for P2 above of the line through P0 and P1
= 0 for P2 on the line through P0 and P1
< 0 for P2 under of the line through P0 and P1

begin

1. $\text{value} = (P1.x - P0.x) \times (P2.y - P0.y) - (P2.x - P0.x) \times (P1.y - P0.y)$
2. return value

finish

Figure (3) show set of 2D points. P8, P10 are the leftmost and rightmost points respectively. P1, P3 for upper hull. P9, P7 for lower hull.



Figure(3) set of points and their convex hull

Andrews Monotone Chain Algorithm (Sacristan 2012), (PAVZAV 2010).
Input: set $P = \{p = (p.x, p.y)\}$ of n points.
Output: set H = the output convex hull of set P, k number of output convex points
in set H.
Begin

1. Sort points set P based on their X coordinate in counter-clockwise order,
2. Let P1 is leftmost point, P2 is the rightmost point, k=3.
3. Add both points p1,p2 to output array of convex points H as H[1]=p1,H[2]=p2.
4. Divides the set of points into upper and lower hull based on the line p1p2.
5. Build upper hull as follow

- for I = 1 to n-1
- begin
- while (k >= 3) and (cross_Direction (H[k-2], H[k-1], p[I])<= 0) do
- k = k - 1
- H[k] = P[I]
- k= k + 1
- End

6. Build lower hull as follow:

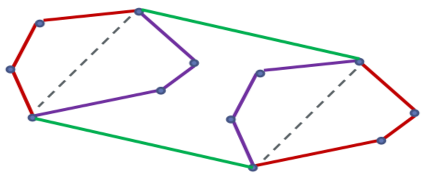
- t= k + 1
- for I:=n-2 down to 1 do
- begin
- while (k >= t)and(cross_Direction(H[k-2], H[k-1], P[I]) <= 0) do
- k =k - 1
- H[k] = P [I]
- k = k + 1
- End

7. return H and k.

Finish

4. Common Tangent Lines Algorithm

Finding the upper and lower Tangent lines between two disjoint convex polygons is a fundamental geometric operation and it resembles to find the maximum and minimum extreme vertices of the polygon (Sunday 2012). The algorithm uses to find independently the upper and lower tangent lines between the two convex hulls that resulted from Andrew's Monotone Chain algorithm in linear time. The right hull vertex is sorted in counter clockwise ordered while the points of left hull are sorted in clockwise oriented. Then discard any points in the rectangular formed by the two lines. The remaining points can be easily added to merge hull in the form of a clockwise sequence (Vertlieb 2012), as depicted in Figure 4.



Figure(4): Common Tangent Lines Algorithm

Common Tangent Algorithm (LH, RH)
Input: RH :right hull points , LH :left hull points.
Output: a,b left and right most points.
begin

1. Let a the right most point in left hull and b the left most point in the right hull.
2. Let done is a Boolean number and with false as initial value
3. while done = 0
4. done = 1.
5. while cross_Direction (RH[b], LH[a], LH[a-1]) <= 0
6. a=a-1.
7. while cross_Direction (LH[a], RH[b], RH[b+1]) >= 0
8. b=b+1.
9. done = 0 .
10. End while .
11. return a, b

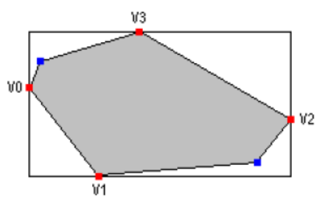
Finish

5. Minimum Bounding Containers

There are finite linear objects, such as points, segments, triangles, polygons, polyhedra, circles and ellipses. Also, the collections of them that a programmer can be selected to solve the problem of bounding containers. These objects are specified by linear and quadratic combinations of their vertices, and their complexity can be measured by the total number of vertices they have. In this paper, we consider two type of bounding container: linear containers for instance rectangular box of convex polygon and quadratic containers such as circles and ellipses(Sunday 2012).

5-1 Minimum Bounding Box

In 2D space, bounding box of a finite linear geometric shape(points, segments, polygons), referred to as a box shape with minimal area that contains a given linear objects or collections of them. Mathematically, box can be defined as rectangular region with edges paralleled to the coordinate axis of x y plane. So, a 2D box is given by the extreme points(xmin ,ymin),(xmax ,ymax) which are its (bottom, left) and (top, right) corners and these values are easily computed in O(n) time with a single scan of all the vertices of convex polygon since it do not involve any arithmetic operations and only needs to compare coordinates values of convex hull vertices with the previous computed min and max constants of x and y axis(Rye 2014). figure(5) show an example of minimum bounding box of point sets.



Figure(5): Minimum Bounding Box

5-2 Minimum Bounding Ball

Actually, in 2D space a minimum bounding ball of a finite linear set of points resembles to the algorithm of computing the simplex shape of linear programming. The thought behind the calculation could be imagined as a balloon strictly holding all the produced arbitrarily objects and afterward collapse it until it can't shrivel any longer without losing of any point(Fischer 2005). The bounding ball B of a linear geometric objects set S and specified by a center point C and a radius R also called the "minimal spanning sphere" of the objects can be computed efficiently with running time O(n). figure(6) show some steps of minimum bounding ball algorithm.

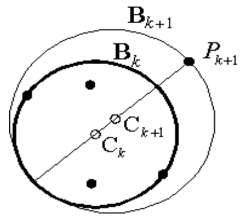
Minimum Bounding Ball Algorithm

Input: linear geometric objects set S.

Output: B represent the Minimum Bounding Ball of S.

Begin

1. a good initialization for bounding ball B is made by estimation two purposes of S that are remote from one another by selecting ones on inverse extremes of the bounding ball for S, and utilizing the line between them as an introductory estimation of measurement then, the middle of distance across is the starting ball center C, and a large portion of the length of the breadth is the beginning ball radius R.
2. each one point P of S is tried for vicinity in the current ball this is carried out by basically checking its distance from the center C is short of what or equivalent to the radius R. In the event that the following point P_{k+1} is in the current B_k, then $B_{k+1}=B_K$ and one simply proceeds to the following point. while if P_{k+1} is outside B_k, then B_k is expanded just enough to include both itself as well as the point P_{k+1}. This is carried out by drawing a line from P_{k+1} to the current center Ck of Bk and stretching out it further to meet the furthest side of B_k. This line is then used as the new diameter for an expanded ball B_{k+1}. As shown in figure(6), it clearly contains the prior ball B_k and all points of S already considered, and no additional recursion is needed(Ritter1990),(Gartner 2008).



Figure(6): Minimum Bounding Ball

3. After we having R and C from previous step using any algorithm to draw circle shape, in this paper we use polar method as following:

Polar Algorithm

Input: C center point , R radius.

Output : draw circle shape.

Begin

1. Two_pi=2*pi.
2. Theta=0.
3. Dtheta=1.0/R.
4. While Theta<= Two_pi
   - X= C_X + R* cos (Theta).
   - Y=C_Y + R * cos (Theta).
   - Plot(round(X),round(y)).
   - Theta= Theta + Dtheta
5. End while.

Finish

5-3 Minimum Bounding Ellipsoid

In 2D space, bounding ellipse of a finite geometric shape(points, lines, polygons, circles and ellipses), referred to as an ellipse shape with minimal area that contains a given linear or quadratic objects or collections of them. mathematically, bounding ellipse is an outstanding container since it is not difficult to test inclusion of a point in an ellipse, particularly in 2D where the entirety of the separations of a point from the two principle focuses is a consistent on the limit of the oval.

Minimum bounding ellipse Algorithm (Sunday 2012)

Input: P represents the 2D convex polygon points .

Output : find and draw an ellipse that outline convex polygon points.

Begin

1. [d N] = size of matrix P.
2. Q = Add a row of 1s to the 2xN matrix P to initialize matrix Q.
3. Initialize count = 1 , err = 1 and tolerance=0.005;
4. u = an Nx1 vector where each element is 1/N
5. while (err > tolerance)
   - X = matrix multiplication of Q*diag(u)*Q' where diag(u)* means places the elements of u in the diagonal of an NxN matrix of zeros
   - M = diag(Q' * inv(X) * Q) where inverse function returns the matrix inverse of X
   - [maximum j]= Find the value and location of the maximum element in the vector M
   - step_size = Calculate the step size for the ascent.
   - new_u = Calculate the new_u by take the vector u, and multiply all the elements in it by (1-step_size).
   - new_u(j) = new_u(j) + step_size
   - err = norm(new_u - u)
   - count = count + 1
   - u = new_u
6. end of while
7. U = diag(u).
8. A = (1/d) * inv(P * U * P' - (P * u)*(P*u)').
9. c = center point of ellipse
10. theta= 0.
11. [U Q V] = find stander deviation of matrix A
12. r1 = x-axis radius.
13. r2 = y-axis radius.
14. dth=change in angles between outline points of ellipse
15. while theta<=2*pi

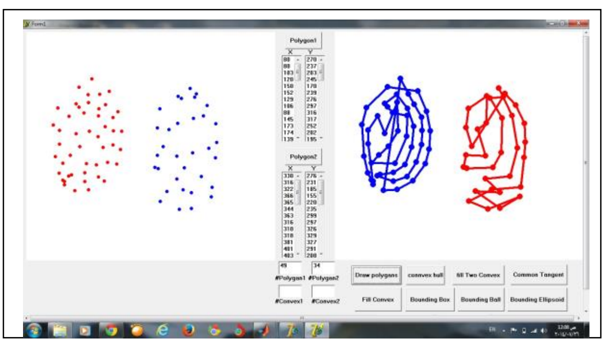
- x= c(1) + r1 * cos(theta);
- y= c(2) + r2 * sin(theta);
- Plot(round(x),round(y)).
- theta=theta+dth;

16. end of while.

finish

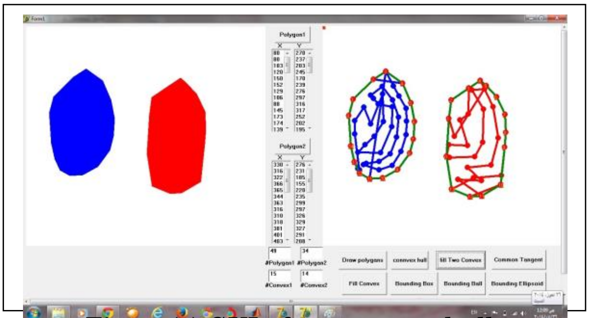
6. Experimental Analysis And Results

In this section, more details can be found of system software implementation. We explain the results of software implementation with an example. The first step of system begins by generating two sets of randomly points on 2D space, each set elected to construct one of convex hull polygons. Actually any user of system could utilize the mouse move and mouse down functions to generate of 2D point sets. Figure (6)depicted the generation operation of two 2D point sets with 49, 34 respectively points number and their drawing polygons.



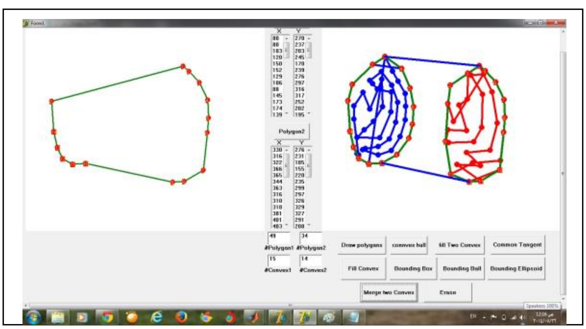
Figure(6) GUI of two point sets with their polygons

After having the 2D points for each polygon we sort these points relatively to x-coordinate value and then, applying Andrew's Monotone Chain algorithm to construct their convex hull. Basically, from our implementation of system we could see that the Monotone Chain algorithm could deal with any number of polygon points in the same effectiveness since it depend on divide and conquer algorithm so it splits points set into upper and lower hulls and then construct the convex hull in O(n log n) running time. Figure(7) show the resulted convex hulls based on Monotone Chain algorithm with 15 and 14 points respectively.



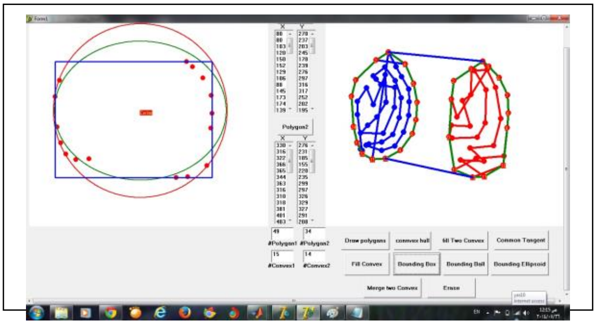
Figure(7) GUI of two convex hulls

Then, we obtain the final convex hull with 15 point by merging the two previous hulls using upper and lower common tangent lines algorithm and discard any point between these two lines as shown in figure(8)



Figure(8) GUI of common tangent method of two

Finally, Figure (9) shows the GUI of implementation of bounding containers including bounding box(blue color), bounding ball(red color) and bounding ellipsoid(green color) of previous example of points set.



Figure(9) GUI of Bounding Containers

7. Conclusion

This paper describes an alternative method for determining the minimum bounding containers including bounding box, bounding ball and bounding ellipsoid of 2D convex polygon. The algorithm suggested above can be used for calculating the minimum bounding containers of any 2D points in fast, robust, and high efficiently manner. Therefore, it may be suitable for real-time graphics and visualization applications, collision avoidance and hidden object detection. Also, The usefulness of such containers is sometimes only given passing mention as an obvious preprocessing test to make before attempting other algorithms. However, there are a number of different choices for bounding containers that a programmer should be familiar with.

BIBLIOGRAPHY:

Roth, L. B., Des. 2009. "Optimal Containment", Doktor Dissertation, Technische Universitat Munchen Zentrum Mathematic.

Rye.E, 2014"MINIMUM BOUNDING BOX (MBB)", University of Minnesota. Undergraduate Computer Science E2 Esten Rye 2866704 G01.

Michael .J. T, E. A. Yildirim,April-2007,"On Khachiyan's algorithm for the computation of minimum-volume enclosing ellipsoids ",Discrete Applied Mathematics 155 , P1731 – P1744,available online at www.elsevier.com/locate/dam

Jambawalikar. s, Kumar. P,2008,"A note on Approximate Minimum Volume Enclosing Ellipsoid of Ellipsoids",IEEE xplore,Perugia,P478 - p487,Print ISBN:978-0-7695-3243-1.

Chan, C. K,S.T.Tan, 1999,"A minimum bounding box algorithm and its application to rapid prototyping", Proceedings of the 10th Solid Freeform Fabrication Symposium,p163-p170.

Sacristan V. 2012. "CONVEX HULL IN 2D", available at www.ma2.upc.es/vera/wp-content/uploads/2012/10/CH-2D.pdf .

Sunday.D, 2012. "The Convex Hull of a 2D Point Set or Polygon", accessed July 13, www.softsurfer.com/Archive/algorithm_0109/algorithm_0109.htm

PAVZAV, 2010. "Andrew's Monotone Chain Convex Hull Algorithm", accessed July 20, http://pavzav.blogspot.com/2010/12/andrews-monotone-chain-convex-hull.html.

Sunday.D,2012,"bounding containers for point sets", accessed on July 14 , http://geomalgorithms.com/a08-_containers.html .

Vertlieb, A. 2012. "Parallel Convex Hull using MPI." Available athttp://www.cse.buffalo.edu/faculty/miller/Courses/CSE633/Vertlieb-Fall-2012 CSE633.pdf

Fischer .K ,2005,"Smallest enclosing balls of balls",Swiss Federal Institute of Technology, Doctor of Technical Sciences dissertation.

Ritter.j, 1990,"An Efficient Bounding Sphere" In Andrew Glassner ,editor, Graphics Gems,chapter 5,p301-p303,AP Professional,Bosten.

Gartner.B,2008, “Approximate Smallest Enclosing Balls”,Available at http://www.ti.inf.ethz.ch/ew/lehre/ApproxGeom08/notes/05-Smallest_Balls.pdf

Lecture Notes for Approximate Methods in Geometry (2008); esp Chapter 5.