Convex Envelope Generation Using a Mix of Gift Wrap and QuickHull Algorithms

Charbel Fares
Holy Spirit University of Kaslik, Lebanon
charbelfares@usek.edu.lb

ABSTRACT

The environment simulation is widely used nowadays. Training in many fields such as medicine and architecture heavily depends on virtual reality techniques. Since objects in real life do not have a deterministic shape it is not possible to have a geometric equation that might model them. Convex Hulls (or Convex Envelopes) are a must in such simulations. The need for convex envelopes rises with the intention of having realistic scenes with exact collision detection between objects in the virtual world. In this paper, four algorithms for generating the convex hull are discussed, implemented and compared. The first three algorithms are the Brute Force, the Gift Wrap and the QuickHull algorithm. The fourth one is a hybrid approach that combines the QuickHull and the Gift Wrap algorithms. Simulations were done in the medical environment, and algorithms are tested with the model of 3D wrist and knee bones.

Keywords: Convex Envelope, Medical Modeling, Computational Geometry, Virtual Reality.

1 INTRODUCTION

Virtual Reality (VR) is an advanced computer-generated technology which allows information to be displayed in a realistic environment that permits users and participants to interact with it. Recent developments in computer technologies have enabled VR to become a powerful product and analysis tool in computational science and engineering. Today, many studies and researches on surgical education depend heavily on VR simulators that become the training method in the medical area [1]. Simulators allow users and participants to examine and study parts and organs of the bodies, offering invaluable education for students and researchers. Unlike cadaver dissection that is not legally nor ethically accepted nowadays, VR models enable the user to perform on human body organs as if in real-world with anatomical accuracy and realism. Therefore, it reduces the risks to surgical patients and avoids the ethical issues associated with animal experimentation. Since computers are a way to simulate a physical environment, those environments are essentially geometric. Many of the computational problems involved in designing and building a VR system are geometric in nature. One principal important problem that must be addressed in order to make VR more realistic is the problem of real-time interactive collision detection.

Convex hull (CH) becomes a choice for modeling physical objects that do not have deterministic shape. The importance of the CH problems not only stems with Collision Detection (CD) but it has many applications such as cluster analysis, image processing and pattern recognition [2]. Other problems can be reduced to CH such as half-space intersections, Delaunay triangulation and Voronoi diagrams. Speeding up algorithms that compute CH is still a challenging issue in many fields and research areas in order to fulfill real-time requirements. In this paper a hybrid method to construct the CH and to decrease its running time is proposed. Simulations were done in the medical environment. The hybrid approach is an output sensitive algorithm that constructs the minimal convex envelope of a set of n points in two as well as three dimensions.

The rest of the paper is structured as follow: section 2 reviews some of the previous convex hull algorithms. Section 3 describes in details three different conventional techniques used to compute the convex envelope. Section 4 discussed the proposed Hybrid technique. In section 5 the results and simulations are shown. Finally in section 6 conclusions are given.

2 PREVIOUS WORK

One of the central problems in computational geometry is the computation of convex hulls. It has been an intensively studied subject up to present days. Early studies dealt primarily with the planer 2D case [3], then comes techniques for calculating CH in 3D space [4]. Brute Force is a simple algorithm that works in both 2D and 3D, it takes $O(n^3)$ running time in 2D and $O(n^4)$ in 3D. A lower bound algorithm presented by Yao [?] for computing the convex hull vertices in the quadratic decision tree model had a complexity of $O(nlogn)$. Another approach known as Grahams scan achieves $O(nlogn)$ running time in 2D. Jarvis March algorithm constructs the convex envelope in $O(nh)$ time, where h denotes the number of vertices of the convex hull. This technique works also in 2D and it is output sensitive because it depends on h in its running time. Furthermore, 2D divide-and-conquer [5] is proposed after the sorting algorithms such as MergeSort and QuickSort and has $O(nlogn)$ running time. Based on this algorithm Preparata and Hong presented their first $O(nlogn)$ time algorithm in 3D. QuickHull [6] is also a fast technique that works in 2D and can be generalized to 3D. It takes as well $O(nlogn)$ running time to compute CH. Moreover, Gift Wrapping is a 3D algorithm that constructs the convex envelope in $O(nh)$ time. This output sensitive method proposed by Chand and Kapur was a generalization of Jarviss march and worked not only in 3D but also in arbitrary dimensions. More involved method in 3D was proposed by Chazelle and Matousek [7], they succeeded to accomplish an $O(nlogh)$ running time algorithm. Edelsbrunner and Shi [8] made-up a deterministic technique having $O(nlog^2h)$ running time. The last two algorithms are not very practical and tend to be complicated thus; the problem of finding optimal and practical algorithms that construct convex envelope in 3D remained. Brute Force, Gift Wrapping and QuickHull algorithms were chosen to be tested, implemented and compared. They were also evaluated with respect to the proposed Hybrid method.

3 3D CONVEX HULL GENERATION

The convex hull or convex envelope of a finite set S of n points in the Euclidean space $\Re^d$ of dimension d denoted as $CH(S)$ is defined by the smallest convex set containing all the points or simply the intersection of all half-spaces containing the set S. The convex hull in $\Re^d$ is the set of solutions to a finite system of linear inequalities in d-variables:

$$CH(S) = \{x \in \Re^d : Ax \leq b\} \tag{1}$$

Where $A \in \Re^{n*d}$ and b $\in \Re^n$.

A solution of the above system can be written as the follow:

$$CH(S) = \sum_{i=1}^{d} \lambda_i p_i : \sum_{i=1}^{d} \lambda_i = 1, \lambda_i \geq 0 \tag{2}$$

Two algorithms for constructing the convex envelope in 3D are discussed and described in details. The first algorithm is the Gift Wrapping and the second one is the QuickHull. Then, a hybrid technique based on the last two algorithms is proposed.

3.1 Gift Wrapping Algorithm

The Gift Wrapping algorithm known also as Jarvis March was created to work in arbitrary dimensions. It consists of an initialization phase followed by a series of wrapping steps. The initialization phase begins first by finding a starting edge $(a,\, b)$ by using the 2D algorithm on the projection of the points on the XY plane; it pivots an initial plane P around the edge $(a,\, b)$ of the hull; it finds the smallest angle between the plane P containing the starting edge $(a,\, b)$ and a plane T formed by point p_i and the edge $(a,\, b)$; it replaces the point p_i by c and form a triangular face containing (a, b, c). The plane $(a,\, b,\, c)$ is a facet on the convex hull. All points now lie to the left of this plane. A set F of frontier edges is initially defined and contains the three edges $(a,\, b)$, $(a,\, c)$ and $(b,\, c)$. Each frontier edge in F is associated with a triangle or facet on the convex hull of S. The wrapping steps are repeated recursively for the edges $(a,\, c)$ and $(b,\, c)$ by finding other triangles adjacent to those edges. All steps for every explored edge are repeated until all facets have been explored. The Gift Wrapping algorithm needs $O(nh)$ times operations to construct CH. It is clear that for every hull edge point discovered, the computer needs $O(h)$ time where h is the number of hull points. Hence for n points in the set, the total time complexity is $O(nh)$. The worst case occurs when all of the input set of points occur on CH, the time complexity of the algorithm becomes $O(n^2)$.

3.2 QuickHull Algorithm

The QuickHull algorithm finds the convex hull of n input points by recursively partitioning this given input set. It shares similarities with sorting algorithms such as it is recursive and each recursion step partitions the data sets into several subsets. QuickHull begins by dividing the set of points into two subsets with respect to a plane formed by: the vertices corresponding to the minimum (x_min) and maximum (x_max) coordinate, and the vertex corresponding to the maximum distance (x_d) from the line joining $(x_min,\, x_max)$. From this initial plane, QuickHull creates a polyhedral of new facets, called visible facets, by calculating the point that has the maximum distance (x_dmax) with respect to the plane. Therefore, QuickHull builds new sets of points from the outside set of the located visible facets. If a point is above multiple new facets, one of the new corresponding facets is selected. If it is below all the new facets, the point is inside the convex hull and consequently it can be discarded.

Partitioning also records the furthest points of each outside set. Each point p in the outside set is processed to locate a visible facet. Visible facet means that the point p is above the specified facet. It constructs a polyhedral from the processed point p to the horizon edges of the visible facets. Finally it deletes this facet, therefore adds the newly created polyhedral of facets to the convex hull. The last three steps are repeated recursively for every point in the new outside set.

$O(nlogn)$ time operations are needed to compute the convex envelope using the QuickHull algorithm. The points will be partitioned into two equal sets and hence the depth of the recursion is $(logn)$. At each level of recursion there are $O(n)$ operations. Therefore, the overall average time is $O(nlogn)$.

4 THE HYBRID ALGORITHM

Since the running time of CH algorithms depends on the number of points n that constitutes the object, many methods are used to speed up them by preprocessing the input points. Some techniques start by dividing the input points into two arbitrary sets, right and left, then computing the final convex hull. Divide-and-Conquer is one of those algorithms that starts recursively by computing the convex envelope of the right then the left set and finally merging the two hulls into a final convex output. Other techniques used to divide the input points into many subsets, Timothy Chan proposes the Chan algorithm [?] that starts by dividing the input points into (n/N) arbitrary disjoint subsets each of size N. Then, it computes the convex envelope of each group, to have an N partial hulls and then integrating the overall into a final output. The idea behind those techniques was always dividing the sets into many subsets trying to speed up the running time of the algorithm and to reduce its complexity. This paper proposes a hybrid technique that starts first by reducing the number of input set of points, then computing the corresponding convex envelope. It is based on QuickHull and Gift Wrapping algorithms. The wrapping step in the Gift Wrapping algorithm can be done faster if the set of input points is preprocessed. Therefore, preprocessing the input points by reducing their number will be a step forward to speed up the algorithm. The hybrid methods initiates by applying the QuickHull so that the input points are divided into two subsets (upper and lower) with an initial plane, then creates a polyhedron of new facets by calculating the point having the maximum distance with respect to this plane. Points that are inside the polyhedron are consequently inside the convex envelope and it is discarded. The same step is repeated for the lower set. This leads to the reduction of the number of input points constituting a new data set.

The new set is used as a new input for the Gift Wrapping algorithm. Consequently, wrapping steps were done by scanning the new data yielding into a final convex output. Furthermore, after preprocessing the input set of points and reducing its number, the hybrid method computes the facets of the hull one at a time, in counter clockwise order by the sequence of the wrapping steps. This algorithm is shown in figure 1.

5 RESULTS AND SIMULATIONS

Table 1 contains the number of vertices and facets constituting the 3D original model of many wrist bones in addition to vertices and facets constituting their corresponding convex hulls. It also shows the execution time of the conventional algorithms and the proposed Hybrid one. The Time shown in second is for computing the convex hull and drawing it with its corresponding model in the scene. Algorithms are tested on many bones that represent the 3D wrist model. One can remark that the new proposed hybrid algorithm is quicker then QuickHull and Gift Wrapping. Figure 2 shows the results of the Hamate and Ulna, two wrist bones.

Brute Forces takes long time to construct the convex envelope compared to other techniques. Therefore, this algorithm is not practically used specially in real time processing. On the other hand, Gift Wrapping and QuickHull are very fast in computing the CH for all the wrist bones. Moreover, the hybrid method outperforms the three conventional algorithms. Noticed that for the 3rdMetacarpal, the hybrid method decreases first the number of vertices from 675 to 559 then the wrapping process is used to construct the convex envelope. This yields the reduction of the running time to 0.12s compared to 0.26s for the Gift Wrapping and 0.21s for the QuickHull. The number of vertices in Scaphoid for example decreases form 2890 to 2539 yielding to decrease the running time from 1.31s for the Gift Wrapping and 1.22s for the QuickHull to 0.88s for the hybrid method.
For the 3rdMetacarpal, the proposed technique decreases the running time up to 53.87% compared to Gift Wrapping and 42.85% compared to QuickHull. Similarly, for the Scaphoid, the running time of the proposed algorithm decreases up to 32.82% compared to Gift Wrapping and 27.86% compared to QuickHull. The proposed technique performs very fast on small objects compared to big ones.

6 CONCLUSIONS

Many 3D objects do not have a shape that could be modeled using precise mathematical equations, convex hull algorithms are a solution for these kinds of issues. The needs of convex hull algorithms rise with the intention of having realistic scenes with real-time interactive collision detection between objects in the virtual world. Since fast collision detection systems work almost exclusively with convex objects, quick convex hull algorithms are implemented in order to fulfill real-time requirements. In this paper, a Hybrid approach to construct the convex hull and speeding up it execution time is proposed and compared with three published methods: Brute Force, Gift Wrapping and QuickHull. The four techniques are implemented, discussed, tested and compared. The results are comparable in terms of execution time for each technique. 3D wrist and knee bones were shown with their corresponding convex envelopes. The proposed hybrid technique decreases the running time of the convex envelope computation for

```
1  find an initial plane from the min and max abssice and the max distance with respect to ($x_min$, $x_max$)
2  construct a polyhedron from the initial plane and the max distance to this plane
3  for each facet F of the polyhedra do
4      for each unassigned point p do
5          if p is above F then
6              assign p to Fś outside set
7          end if
8      end for
9  end for
10 Discard all points inside the polyhedron forming a new imput set ($n_{new}$)
11 find a starting edge $(a, b)$ using the 2D Gift Wrapping algorithm on the XY projection
12 for i=1 to $n_{new}$ do
13     find point $p_i$ corresponding to min angle bewteen plane $P$ in XY containing $(a, b)$ and plane T = $(a, b, p_i)$
14     replace $c \leftarrow p_i$
15     save $(a, b, c)$ into Q
16     wrap the edge $(a, c)$
17     if facet has been explored then
18         wrap the edge $(b, c)$
19         if facet has been explored then
20             return
21         end if
22     end if
23 end for
```

Figure 1: The Hybrid Approach

3D Model	Original Model		Convex Hull		Brute Force	Gift Wrap	QuickHull	Hybrid
	# vertices	# facets	# vertices	# facets	time (s)	time (s)	time (s)	time (s)
1^{st} Meta.	2179	4320	379	790	19014.22.1	0.66	0.61	0.43
2^{nd} Meta.	1168	2258	300	596	12015.41	0.47	0.38	0.25
3^{rd} Meta.	675	1272	150	296	2330.77	0.26	0.22	0.12
4^{rd} Meta.	532	1002	147	290	2302.72	0.21	0.19	0.09
Hamate	2812	5620	394	784	19231.20	0.89	0.82	0.71
Ulna	977	1864	312	620	12153.03	0.41	0.37	0.22
Scaphoid	2890	5784	530	1056	22125.1	1.31	1.22	0.88
Capitate	3026	6048	635	1266	25300.35	1.46	1.38	1.25
Radius	2454	4754	288	572	11260.12	0.78	0.72	0.62

Table 1: Comparison of Execution Time For Computing The 3D Convex Hull

all objects in the scenes. The Hybrid method is an output sensitive algorithm that works in 2D as well as in 3D. It was shown that this method was very efficient, practical and usefulness in modeling 3D medical data and simulating their models in virtual environments.

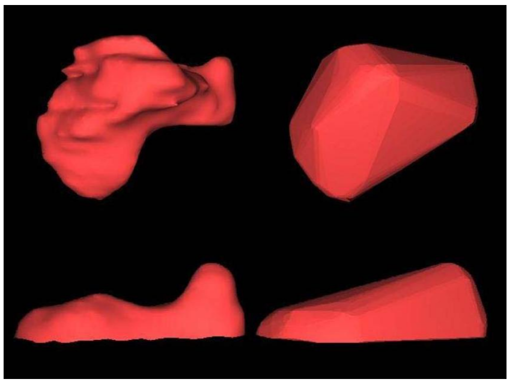
Figure 2: Hamate and Ulna with their convex hulls

REFERENCES

[1] S. Haque, S. Srinivasan, A Meta-Analysis of the Training Effectiveness of Virtual Reality Surgical Simulators, IEEE Transaction on IT in Biomedicine, (2006).

[2] S. Akl, Efficient Convex Hull Algo. for Pattern Recognition, $4^t h$ Int. Conf. on Pattern Recognition, (1978).

[3] V. Bayer, Survey of Algorithms for the Convex Hull Problem, Depart. of CS Oregon State Univ., (1999).

[4] A. Day, Implementation of an Algo. to Find the Convex Hull of a Set of 3D Points, ACM (1990).

[5] F. Preparata, S. Hong, Convex Hulls of Finite Sets of Points in Two and Three Dimensions, ACM Transactions on Mathematical Softwares, (1977).

[6] C. Barber, D. Dobkin, H. Huhdanpaa, The QuickHull Algorithm for Convex Hulls, ACM Transactions on Mathematical Softwares, (1996).

[7] B. Chazelle, J. Matousek, Derandomizing an Output-Sensitive Convex Hull Algorithm in 3D, Computational Geometry Theory and Applications, (1995).

[8] H. Edelsbrunner, W. Shi, An $O(nlog^2h)$ Time Algorithm for the Three-Dimensional Convex Hull Problem, SIAM Journal on Computing, (1991).