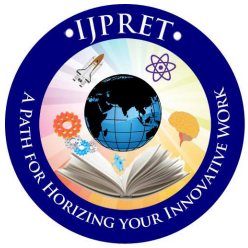
Research Article *ISSN: 2319-507X*
Amitkumar Manekar, IJPRET, 2013; Volume 1(8): 287-298 *IJPRET*



INTERNATIONAL JOURNAL OF PURE AND APPLIED RESEARCH IN ENGINEERING AND TECHNOLOGY

A PATH FOR HORIZING YOUR INNOVATIVE WORK

PERFORMANCE ANALYSIS AND IMPROVEMENT STRATEGY FOR CONVEX ENVELOPE GENERATION ALGORITHM FOR PARALLEL PROGRAMMING

AMITKUMAR S MANEKAR[1], PROF. PANKAJ KAWADKAR[2], PROF. HITESH GUPTA[3], PROF. MALTI NAGLE[4]

1. PG Research Scholar Computer Science and Engineering, PIES.
2. H.O.D. Computer Science and Engineering, PIES.
3. H.O.D. Computer Science and Engineering, PCST.
4. H.O.D. Computer Science and Engineering, PIT, RGPV, Bhopal, M.P, India.

Accepted Date:

27/02/2013

Publish Date:

01/04/2013

Keywords

Open MP,

MPI,

Hybrid (MPI + OpenMP),

Convex Envelope,

Medical Modeling,

Computational Geometry

Virtual Reality,

Parallelism

Abstract

Simulation and virtual reality in medicine, aerospace, weather forecasting, traffic engineering, architecture and many more industries as a reliable and cost worthy alternative for research and development of new technological models. Vector data processing having set of values available for the geometric projection. Objects in real life don't have a deterministic shape and size it is not possible to apply geometric equation that might model them. Many shapes are propagated by Convex Hull or Convex Envelope for a realistic view of real world model. As Brute Force, Gift Wrap and Quick Hull algorithm sequential algorithms are constructed and tested for the convex hull. In this paper we constructed parallel Convex Hull algorithms and check the performance on parallel models (OpenMP, MPI and Hybrid) for the simulation data in medical environment.

Corresponding Author

Mr. Amitkumar S Manekar

I. INTRODUCTION

Today, researches on surgical education depend heavily on VR simulators that become the training method in the medical field [11][12]. simulator play a very important role in learning by allowing user especially medical or biological student and medical practitioner to examine and study organs or human body structure in a way that were not possible before few years earlier. Similarly surgeons (as well as student) can gain a valuable experience by performing a particular surgery with an anatomical accuracy and realism as it is actually achieved in the real world. Thus, the surgeon can practice his operations before he proceeds and operates on the patients. Therefore, it reduces the risks to surgical patients and avoids the ethical issues associated with animal experimentation [1]. Since the real object does not have deterministic shape and size, it is very difficult to determine the geometric equation for the same. To overcome this alternative way is to construct the relative object is constructing the convex envelope with convex hull algorithm and to take off real object with collision detection in real world. On other side every real world object of nondeterministic shape may have several boundary points to calculate these huge data for convex envelope generation several sequential algorithms are available. As Brute Force, Gift Wrap and Quick Hull algorithm sequential algorithms are constructed and tested for the convex hull. In this paper we tried to develop the parallel convex hull algorithm which concludes the huge boundary point in convex envelope by using parallel models (OpenMP, MPI and Hybrid). Now constructed convex envelope is used for the simulation of real world object.

In computer simulation is done with the geometric equation many complex VR (Virtual Reality) problems is based analysis of geometry. An important problem that must be addressed to make VR more realistic is the problem of real-time interactive collision detection [2]. Collision Detection means two virtual objects should not intersect to each other. In VR geometric equations are so precise that it needs high

speed algorithms with good performance. All these algorithm works with convex object. Developing the convex envelope considering collision detection in geometric equation is very hard and this problem is known as NP-Hard. Convex hull (CH) is a best suitable approach to model physical objects that do not have deterministic shape and size. It plays an important role in many applications that are based on cluster analysis, image processing and pattern recognition [13]. As a programmer solving this type of complex memory intensive problem with a great speed is possible in *Parallel Programming*. All these convex hull algorithms are sequential, In this paper we develop the parallel algorithm and check on three basic flavours of parallelism OpemMP,MPI and Hybrid. Every flavour have its own advantages and disadvantages so finally we analyse the result for three flavours in terms of complexity.

II. Previous Work

Many researchers have tried to find probable methods to apply arrangement for large data sets. The sequential minimal optimization (SMO) [14] breaks the large QP problem into a series of smallest size QPP. Computing the convex hull of a static set of n point set can be done in optimal O (n log n) time, e.g., with Graham's scan [4] or Andrew's vertical sweep line variant of it. Optimal output sensitive algorithms are due to Kirkpatrick and Seidel also to Chan, they achieve O(n log n) running time, where h denotes the number of vertices on the convex hull [5]. A large number of shape matching methods can be found in the literature. Some of them employ elastic deformation of templates [6] [10], comparison of directional histograms [8], or shocks and skeletal representations of object shape [9]. Lee proposed an approach that combines polygon curve representation with Fourier descriptors for shape-based retrieval in images of vertebrae[7].on other side Parallel computers are going main stream because clusters of SMP (Symmetric Multiprocessors) nodes provide support for an ample collection of parallel programming paradigms. MPI and OpenMP are the trendy flavours in a parallel programming. The conventional parallel programming practice involves a pure Shared Memory Model [13].

Usually using the OpenMP API [18], in shared memory architecture, or a pure message passing model [13] using MPI API [15], on distributed memory system .old approach of doing parallelism involves pure shared memory models. Usually in shared memory architecture uses of OpenMP API. For distributed MPI API once [17]. In this paper we revive the parallel programming models in high performance computing (HPC) with classification of parallel programming models used today.

III. Parallel Processing

Parallel processing means use of two or more processor for solving single problem. Traditionally software has been written for serial computation, to be run on a single computer having a single Central Processing Unit (CPU), a problem is broken into a discrete series of instructions. Instructions are executed one after another, only one instruction may execute at any moment in time. In ***parallel computing*** is the simultaneous use of multiple compute resources to solve a computational problem, to be run using multiple CPUs ,a problem is broken into discrete parts that can be solved concurrently. Each part is further broken down to a series of instructions, Instructions from each part execute simultaneously on different CPUs [16]. There are several parallel programming models; ***Parallel programming models exist as an abstraction above hardware and memory architectures*.** Classification of parallel programming models using a pure shared or distributed memory approach, shared memory OpenMP, and distributed memory Message Passing models(MPI) is a specification for message passing [19][18][20][21]. The real art of parallel programming is *Load Balancing*. Table-1 explains the basic ***difference between these hardware*** strategies.

Depending on the memory used for communication parallelism is grouped in basic two levels distributed and shared explain with diagram in Fig-1 and Fig-2 respectively.

Distributed Memory

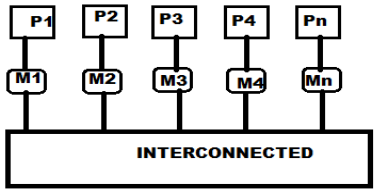


Fig 1. Distributed Memory Model

Each processor has its own memory and parallel programming by message passing (MPI).

Shared Memory **-** Each processor shared memory and two parallel programming approaches

- *Message passing (MPI)*
- *Directives-based interface – OpenMP*

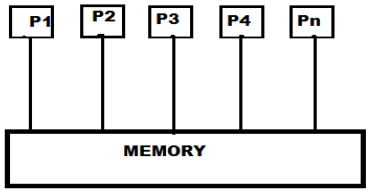


Fig 2. Shared Memory Model

OpenMP **-** OpenMP (Open-Multi-Processing) an API (Application Programming Interface) used Multi threaded parallel processing. OpenMP used shared memory Multi Processor (Core) Computer. This type requires less code modification than using MPI, OpenMP directives can be treated as comments if OpenMP is not available and Directives can be added incrementally.

Limitation

- OpenMP codes not suitable for distributed memory, (exception is Intel's OpenMP)
- Special type support compiler
- Gives lower parallel efficiency [21].

To achieve parallelism using OpenMP two main approaches loop level and parallel region parallelism also called *fine grained parallelism(Larger Components),* and *coarse-grained parallelism respectively(Fewer Components).*

MPI (Message Passing Interface) **-** A standard Message Passing Interface (MPI) is originally for distributed memory environments. MPI runs on both distributed and shared memory model with portability. Coarse grain parallelism is present with explicit messaging.

Limitation

• Large overhead due to communication needs to be minimized as well as global operations can be very expensive that's why transfer between the serial and parallel code difficult.

• Dynamic load balancing is often difficult in MPI.

Hybrid (OpenMP+MPI) **-** Hybrid rational model takes both advantages from the MPI/OpenMP with explicate decomposition of task placement it achieve simple and fine-grain parallelism. Match Current hardware trend wit industry standard MPI and OpenMP. By adding OpenMP limitation of MPI (scalability) is overcome. Assign different no threads by OpenMP for load balancing to achieve synchronization.

Limitation:-

• Programming overhead as mixed mode implementation for Communication.

• Not a solution to all parallel programs but quite suitable for certain algorithms [21].

IV. Convex Hull Algorithm

Geometric algorithms take geometric objects of various types as their inputs. The basic geometric objects in the plane are points, lines, segments, and polygons.

Definition: ***T e convex hull of a set P of points is the smallest convex set containing P. (****Fig-3 for* A set of points and its convex hull *and* Fig 4. Convex Hull*)*

We are given a set P of n points in the plane. We want to calculate something called the convex hull of P. convex polygon containing the points: ***polygon:*** A region of the plane bounded by a cycle of line segments, called edges, joined end-to-end in a cycle. Points where two successive edges meet are called vertices.

Convex: For any two points p; q inside the polygon, the line segment pq is completely inside the polygon. ***Smallest:*** Any convex proper subset of the convex hull excludes at least one point in P.

This implies that every vertex of the convex hull is a point in P (See Fig 4.).

A polygon is defined to be convex if for any two points P1 and P2 inside the polygon, the directed line segment from P1 to P2 is fully contained in the polygon.

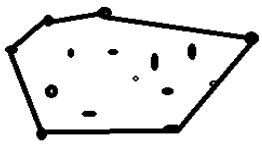


Fig 3. A set of points and its convex hull

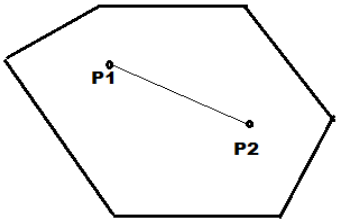


Fig 4. Convex Hull

Constructing Convex Hull

Given P: Set of n points.

Convex hull of P: CH (P), smallest polyhedron structure contains all elements of P on or in the interior of CH (P).

Point P1, P2, P3 is sorted by Quick Hull algorithm on 3-Cordinate system as shown in Fig-5. Now calculate Maximum volume for each set by eliminating trapezium as shown in Fig-6.

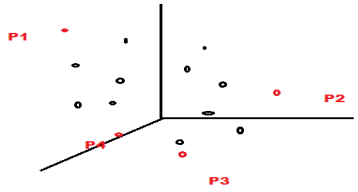


Fig 5- Finding the extreme point a triangle for each set of point p1, p2, p3.

All non- trapezium point consider for next evaluation. With maximum co-ordinatesP1-X, P2-Y ,P3-Z and P4 for maximum volume as shown in Fig-7

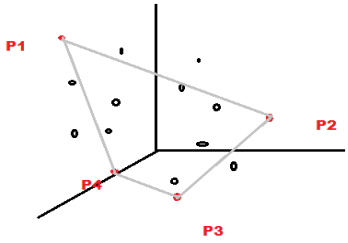


Fig 6- Take max volume and eliminate the point inside the trapezoid and go for the further computation as considering new trapezoid P1- P2-P3- P4

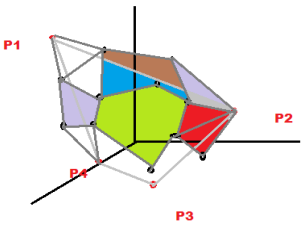


Fig 7. After considering all the boundary points the final 3D convex hull

IV. RESULT ANALYSIS

All result taken on Second Generation Intel Core i5 with 4 GB RAM system. Fig-8 shows that Sequential processing takes more time as compared to OpenMP. It has been observed that hybrid implementation for a larger point gives better result than smaller point data set. In both cases if the point data is large we choose parallel processing with variant of data analysing.

power, Fig-8 is performance analysis of all these aspects.

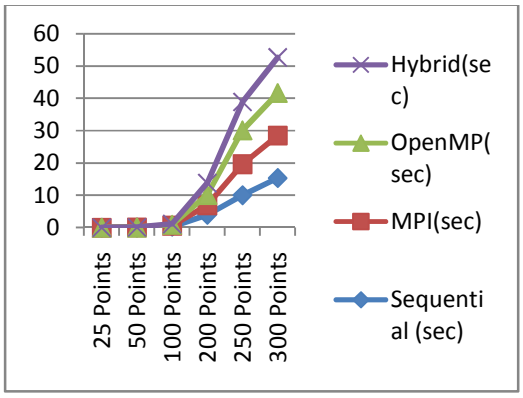


Fig 8 – Performance Analysis of Sequential, Open MP, MPI and Hybrid.

V. Conclusion

From this analysis we conclude that convex hull algorithm using sequential, OpenMP, MPI and Hybrid is tasted for the variant of data set and finally OpenMP gives good result as compared to Sequential Processing. Hybrid is surprisingly better for large data set with a better load balancing techniques. Also we can improve the performance by proper load balancing at hybrid paradigm.

REFERENCES

1. Fadi Yaacoub, Yskandar Hamam, Antoine Abche and Charbel Fares “Convex Hull in Medical Simulations: A NewHybrid Approach”IEEE- 1-4244-0136-4/06 pp3308-3313,2006.

2. Charbel Fares “Convex Envelope Generation Using a Mix of Gift Wrap and QuickHull Algorithm”I nternational Conferences in Central Europe on Computer Graphics, Visualization and Computer Vision”, ISSN 1213-6972, 2012.

3. R. L. Graham. An efficient algorithm for determining the convex hull of a finite planar set. Information Processing Letters, 1(4):132–133, 1972.

4. Riko Jacob" Dynamic Planar Convex Hull", IEEE, Proceedings of the 43 rd Annual IEEE Symposium on Foundations of Computer Science (FOCS'02) 0272-5428/02, 2002.

5. A. Del Bimbo, et al., "Image Retrieval by Elastic Matching of Shapes and Image Patterns", Proceedings of Multimedia '96, 1996, pp. 215-218.

6. Nikolay M. Sirakov et al," Search Space Partitioning Using Convex Hull And Concavity Features For Fast Medical Image Retrieval", Ieee- 0-7803-8388-5,pp 796-799,2004.

7. D. Androutsas, et al., "Image Retrieval Using Directional Detail Histograms". Proc. SPIE, Storage and Retrieval for Image and Video Databases VI, v. 3312, 1999, pp.129-137.

8. S. Tirthapura, et al., "Indexing Based on Edit- Distance Matching of Shape Graphs", Proc. SPIE Multimedia Storage and Archiving Systems III (C. C. J. Kuo, et al., eds), v. 3527, 1998, pp. 25-36

9. A. Pentland, et al., "Photobook: Tools for Content-Based Manipulation of Image Databases", International Journal of Computer Vision, v. 18, n. 3, 1996, pp.233-254.

10. R. S. Haluck and T. M. Krummel, "Computers and virtual reality for surgical education in the 21st century", Arch. Surgery,vol. 135, pp. 786- 792, 2000.

11. S. Haque and S. Srinivasan, "A Meta-Analysis of the training effectiveness of virtual reality surgical simulators", IEEE Transaction on information technology in biomedicine, January 2006.

12. OpenMP. "API Specification for Parallel Programming", Oct. 2011.

13. J.Platt, Fast Training of support vector machine using sequential minimal optimization, Advances in Kernel Methods: Support Vector Machine, MIT Press, Cambridge, MA, 1998.

14. Sandip V.Kendre, Dr.D.B.Kulkarni: Optimized Convex Hull With Mixed (MPI and OpenMP) Programming On HPC, IJCA (0975 –8887), Volume 1 – No. 5,2010

15. J. Diaz, C. Munoz-Caro and A. Nino: A Survey of Parallel Programming Models and

Tools in the Multi and Many-core Era, IEEE, Parallel and distributed system, Vol- 23 no 8 pp 1369-1386. Aug, 2012

16. W. Gropp, S. Huss-Lederman, A. Lumsdaine, E.Lusk, B. Nitzberg, W. Saphir and M. Snir, MPI: The Complete Reference, 2nd Edition,Volume 2 - The MPI-2 Extensions. The MIT Press, Sep. 1998.

17. M. J. Sottile, T. G. Mattson and C. E. Rasmussen,Introduction to Concurrency in Programming Languages.CRC Press, 2010

18. W. Gropp, E. Lusk, and A. Skjellum, Using MPI:Portable ParallelProgramming with the Message-Passing Interface, 2nd ed. MIT Press,Cambridge, MA, 1999.

19. A Grama,A Gupta: An Introduction to Parallel Computing: Design and Analysis of Algorithms 2nd Edition. Pearson Publication 2007.

20. T. G. Mattson, B. A. Sanders and B. Massingill Patterns for Parallel Programming. Addison-Wesley Professional, 2005

21. P. S. Pacheco, Parallel Programming with MPI, Morgan Kaufmann, San Francisco, 1996.