PARALLEL IMPLEMENTATION OF FIELD VISUALIZATIONS WITH HIGH ORDER TETRAHEDRAL FINITE ELEMENTS

MOHAMMAD HAFIFI HAFIZ BIN ISHAK

UNIVERSITI SAINS MALAYSIA

2013

PARALLEL IMPLEMENTATION OF FIELD VISUALIZATIONS WITH HIGH ORDER TETRAHEDRAL FINITE ELEMENTS

by

MOHAMMAD HAFIFI HAFIZ BIN ISHAK

Thesis submitted in fulfillment of the requirements
for the Degree of
Master of Science

APRIL 2013

ACKNOWLEDGEMENTS

First and foremost, I owe my deepest gratitude to Dr. Razi Abdul Rahman, my thesis advisor and project supervisor, who has introduced me to the subject. His serious attitude to research has always been a reference model for me. He inspired me greatly to work in this project. His willingness to motivate me contributed tremendously to my project. I also would like to thank him for showing me some example that related to the topic of my project. His guidance and patience throughout the tumultuous time of conducting scientific investigations related to this project are much appreciated. Besides, his invaluable support and insightful suggestions, not to mention all the hard work and extra time poured in has resulted in the completion of this project.

I also would like to express my deepest gratitude to Ministry of Higher Education (MOHE) that provides me with MyBrain scheme throughout my candidature period.

Last but not least, my heartfelt gratitude to my family especially my mother,for her continuous support in my pursuit of this master degree and endless love in my life.

TABLE OF CONTENT

PAGE

ACKNOWLEDGEMENTS III

LIST OF TABLES VII

LIST OF FIGURES VIII

LIST OF ABBREVIATIONS XIII

ABSTRAK XIV

ABSTRACT XVI

CHAPTER 1-INTRODUCTION 1

1.1 Introduction 1

1.2 The Need of Parallel Algorithm For High Order Visualization 5

1.3 Problem Statement 6

1.4 Research objectives 7

1.5 Scope of work 7

1.6 Organization of the thesis 8

CHAPTER 2-LITERATURE REVIEW 9

2.1 Introduction 9

2.2 Visualization Method in FEM 11

2.4 Parallel Implementation of Visualization 18

2.4.1 Message Passing Interface 19

2.4.2 Parallel Virtual Machine 21

2.4.2 MPICH2 22

2.4.3 Hyper Threading 23

2.5 Summary 24

CHAPTER 3-IMPLEMENTATION OF POST –PROCESSOR ALGORITHM 25

3.1 Introduction 25

3.2 Software implementation overview 25

3.3 Solver3D module 27

3.3.1 Element Connectivity Data Structure 28

3.3.2 Hierarchical basis function for high order tetrahedron elements. 29

3.3.3 Pre-assembly 32

3.4 The post-processor module “Post3D” 35

3.4.1 Tetrahedron Subdivision 37

3.4.2 Edge-Based Subdivision 39

3.4.3 Volume-based Subdivision 40

3.4.4 The Legacy VTK File Format 48

3.4.5 Construction of Algorithm 50

3.5 Parallelization 53

3.5.1 Message Passing Interface (MPI) 53

CHAPTER 4-RESULTS AND DISCUSSIONS 58

4.0 Overview 58

4.1 Three Dimensional Simulations with Scalar Fields 58

4.2 Weak formulation of partial differential equation using Galerkin’s method 58

4.2.1 Heat conduction in a solid unit cube 59

4.1.2 Accuracy of High Order Visualization 70

4.1.3 Isocontour 73

4.1.4 Heat conduction in L-Shape domain 75

4.1.5 Heat conduction in Cube with cylinder domain 85

4.2 Computational performance of higher order tetrahedron elements 93

4.2.1 Computational performance of MPI parallelism 95

5.3 Effect of Hyper-Threading Technology 106

CHAPTER 5-CONCLUSION AND FUTURE WORKS 109

6.1 Summary 109

6.2 Recommendations for Future Work 110

REFERENCES 111

APPENDICES 116

APPENDIX A: Basis functions 117

APPENDIX B: Third Order algorithm 120

APPENDIX C: Fourth Order algorithm 125

APPENDIX D: Third Order with MPI algorithm 130

APPENDIX D: Third Order with MPI algortihm 136

LIST OF TABLES

		Page
Table 3.1:	Distribution Of Dof In A Tetrahedron	34
Table 3.2:	Distribution Of Dof In A Tetrahedron For `Polyde`	35
Table 3.3:	Tetrahedron And Nodes Number For Subdivide Tetrahedron	43
Table 3.4:	Mapping For All Indices *I* In Children Tetrahedron Into Parent Tetrahedron	44
Table 4.1:	Summaries Of Mesh, Dof For Its Respective P For Tc1	62
Table 4.2:	Summaries Of Mesh, Dof For Its Respective P For Tc2	76
Table 4.3:	Summaries Of Mesh, Dof For Its Respective *P* For Tc3	86
Table 4.4:	Experimental Hardware Platform.	95
Table 4.5:	Summaries Of Result Mpi Implementation On Tc1 For Its Respective P And Dof For Hpc By Using 6 Cores	96
Table 4.6:	Summaries Of Result Mpi Implementation On Tc1 For Its Respective P And Dof For Workstation By Using 4 Cores	97
Table 4.7:	Summaries Of Result Mpi Implementation On Tc2 For Its Respective P And Dof For Hpc By Using 6 Cores	101
Table 4.8:	Summaries Of Result Mpi Implementation On Tc2 For Its Respective P And Dof For Workstation By Using 4 Cores	102

LIST OF FIGURES

		Page
Figure 1.1:	Visualization Of Human Bone(Scholari and Thessaloniki, 2013)	4
Figure 1.2:	Linear Element's And Fourth Order Element's Tetrahedron	5
Figure 2.1:	Different Levels Of Parallelism And Actual Data Layout In The Cpu Implementation.	22
Figure 3.1	Flow Chart Of The 3-D Fem Solution Process	26
Figure 3.2:	Process flow Of The Solver3d Module	27
Figure 3.3:	Netgen Input File	28
Figure 3.4:	Tetrahedron	29
Figure 3.5:	Numbering And Orientation Of A Standard Tetrahedron For `Polyde`3d. The Vertices Are Bold; The Faces Underlined.	32
Figure 3.6:	Mapping Of Non-Isoparametric Tetrahedron In `Polyde` And The Low Order(P=1,2) Isoparametric Tetrahedron Cell Types In Vtk	36
Figure 3.7:	Tetrahedron Cell Types In Vtk: (A) Linear Tetrahedron (B) Quadratic Tetrahedron	36
Figure 3.8:	Process flow Of The `Post3d` Module	37
Figure 3.9:	Tetrahedron Refinement Techniques In 2d	38
Figure 3.10:	Red Refinement In 3d	38
Figure 3.11:	Subdivision Of A Triangle	39
Figure 3.12:	Subdivisions Scheme Configurations For Tetrahedron	40
Figure 3.13:	Mapping Fourth Order Tetrahedron Dof Into Quadratic Tetrahedron, A "Parent" Is Mapped To 8 "Children"	41

Figure 3.14:	New Nodes Number For 8-Tetrahedron	43
Figure 3.15:	Dof Number In Fourth Order Parent Tetrahedron	44
Figure 3.16:	Mapping Of Children Tetrahedron 1 To Quadratic Tetrahedron	45
Figure 3.17:	Porting 20 Node Tetrahedron Dof Into Quadratic Tetrahedron	46
Figure 3.18:	Dof Number In Third Order Parent Tetrahedron	46
Figure 3.19:	Mapping Of Children Tetrahedron 2 To Quadratic Tetrahedron	47
Figure 3.20:	Example Of Vtk Output	49
Figure 3.21:	Cell And Point Data	50
Figure 3.22:	Part Of Second Order Algorithm	51
Figure 3.23:	Part Of Third Order Algorithm	52
Figure 3.24:	Part Of Fourth Order Algorithm	53
Figure 3.24:	Sending And Receiving Process Between Process A And Process B	54
Figure 3.25:	Flow Chart Of Parallel Algorithm	55
Figure 4.1:	Cube (Lx=Ly=Lz) With T=1 On One Side And T=0 At Other Five Sides.	60
Figure 4.2:	Unstructured Mesh Of 24576 Tetrahedron For Test Case 1	61
Figure 4.3:	Temperature Distribution On Tc1 : (A) Boundary (B) Interior	61
Figure 4.4:	Boundary Condition With Mesh Element=48 For P= {1,2,3,4}, The Dof Are In The Parenthesis	63
Figure 4.5:	Contour Line With Mesh Element=48 For P={1,2,3,4}, The Dof Are In The Parenthesis	64

Figure 4.6:	Example Of The Zoom Into An Element Region By The Factor Of 4times Than Normal	65
Figure 4.7:	The Zoom Into An Element Region Contour Line With Mesh Element=48 For P= {1,2,3,4}, The Dof Are In The Parenthesis.	65
Figure 4.8:	Boundary Condition With Mesh Element=49152 For P={1,2,3,4}.The Dof Are In The Parenthesis	67
Figure 4.9 :	Contour Line With Mesh Element=49152 For P={1,2,3,4}, The Dof Are In The Parenthesis.	68
Figure 4.10:	Example Of The Zoom Into An Element Region By The Factor Of 4times	69
Figure 4.11:	The Zoom Into An Element Region Contour Line With Mesh Element=49152 For P= {1,2,3,4}, The Dof Are In The Parenthesis.	69
Figure 4.12:	Temperature Distribution On Tc1with Field Value Plotted On Z-Plane(P=1, 48 Mesh Element)	70
Figure 4.13:	Temperature Distribution On Tl1 For P = {1, 2, 3, 4}For Mesh Element=48.	71
Figure 4.14:	Temperature Distribution On Tl1 For P = {1, 2, 3, 4}For Mesh Element=196608.	72
Figure 4.15:	Isocontours Of 4 For The Isovalues 0.5 (Red) Which Standard Linearization Approach (A), Second Order Approach (B), And Tetrahedron Subdivision Approach (C) , (D)	74
Figure 4.16:	Reentrant Edge At The Corner Of L-Shape Geometry	75
Figure 4.17:	Temperature Distribution On Tc2 : (A) Boundary (B) Interior	76
Figure 4.18:	Boundary Condition With Mesh Element=1088 For P= {1,2,3,4}.The Dof Are In The Parenthesis	77
Figure 4.19:	Contour Line With Mesh Element=1088 And Surface Element=72 For P= {1,2,3,4}.The Dof Are In The Parenthesis.	78

Figure 4.20:	Example Of The Zoom Into An Element Region By The Factor Of 4times	79
Figure 4.21:	The Zoom Into An Element Region Contour Line With Mesh Element=144 For P= {1,2,3,4}, The Dof Are In The Parenthesis.	79
Figure 4.22:	Boundary Condition With Mesh Element=69632 For P= {1,2,3,4}, The Dof Are In The Parenthesis	81
Figure 4.23:	Contour Line With Mesh Element=69632 For P= {1,2,3,4}, The Dof Are In The Parenthesis.	82
Figure 4.24:	Example Of The Zoom Into An Element Region By The Factor Of 4times	83
Figure 4.25:	The Zoom Into An Element Region Contour Line With Mesh Element=69632 For P= {1,2,3,4}, The Dof Are In The Parenthesis	83
Figure 4.26:	Mesh Of 15408 Tetrahedrons For Test Case 3(Tc3)	85
Figure 4.27:	Temperature Distribution On Tc3: (A) Boundary (B) Interior	86
Figure 4.28:	Boundary Condition With Mesh Element=1926 For P= {1,2,3,4},The Dof Are In The Parenthesis	87
Figure 4.29:	Contour Line With E=1926 And Surface Element=123 For P= {1,2,3,4}.The Dof Are In The Parenthesis	88
Figure 4.30:	Example Of The Zoom Into An Element Region By The Factor Of 4times	89
Figure 4.31:	The Zoom Into An Element Region Contour Line With Mesh Elemet=1926 For P= {1,2,3,4},The Dof Are In The Parenthesis.	89
Figure 4.32:	Boundary Condition With Mesh Element=15408 For P= {1,2,3,4}, The Dof Are In The Parenthesis.	90
Figure 4.33:	Contour Line With Mesh Element=15408 For P= {1,2,3,4}, The Dof Are In The Parenthesis.	91

Figure 4.34: Example Of The Zoom Into An Element Region By The Factor Of 4times 92

Figure 4.35: The Zoom Into An Element Region Contour Line With Mesh Element=15408 For P= {1,2,3,4},The Dof Are In The Parenthesis. 92

Figure 4.36: Growths Of The Post Times For Tc1 94

Figure 4.37: Screenshot Of Post Time 94

Figure 4.38: Total Post Times For Tc1 For First Order(P=1) With Implementation Of Mpi On Workstation Computer And Hpc 98

Figure 4.39: Total Post Times For Tc1 For Second Order Mpi(P=2) With Implementation Of On Workstation Computer And Hpc 98

Figure 4.40: Total Post Times For Tc1 For Third Order(P=3) With Implementation Of Mpi On Desktop Computer And Hpc 99

Figure 4.41: Total Post Times For Tc1 For Fourth Order(P=4) With Implementation Of Mpi On Desktop Computer And Hpc 99

Figure 4.42: Total Post Times For Tc2 For First Order(P=1) With Implementation Of Mpi On Desktop Computer And Hpc 103

Figure 4.43: Total Post Times For Tc2 For Second Order(P=2) With Implementation Of Mpi On Desktop Computer And Hpc 103

Figure 4.44: Total Post Times For Tc2 For Third(P=3) With Implementation Of Mpi On Desktop Computer And Hpc 104

Figure 4.45: Total Post Times For Tc1 For Fourth Order(P=4) With Implementation Of Mpi On Desktop Computer And Hpc 104

Figure 4.46: Test Environment 106

Figure 4.47: Cpu Post Time Benchmark For P=2 On Tc1 107

Figure 4.48: Cpu Post Time Benchmark For P=3 On Tc1 107

Figure 4.49: Cpu Post Time Benchmark For P=4 On Tc1 108

LIST OF ABBREVIATIONS

FEM	Finite element method
DOF	Degree of freedom
3D	Three dimensional
SORCG	Symmetric Successive Over-relaxation Gradient
BC	Boundary condition
AFEM	Adaptive Finite element method
VTK	Visualization Toolkit
MPI	Message Passing Interface
DVR	Direct Volume Rendering

PERLAKSANAAN SELARI PENGGAMBARAN MEDAN DENGAN UNSUR TERHINGGA TETRAHEDRON DARJAH TINGGI

ABSTRAK

Dalam kaedah penyesuaian unsur terhingga (AFEM), darjah tertinggi unsur terhingga biasanya digunakan dalam pengiraan. Dalam simulasi tiga dimensi, cabaran besar terhadap pasca pemprosesan sejak program perisian data penggambaran sedia ada tidak mempunyai darjah tertinggi—penggambaran biasa hanya boleh menggambarkan sehingga sepuluh nod unsur tetrahedron. Penyelidikan ini mencadangkan dan melaksanakan rangka kerja yang cekap untuk penggambaran data dengan unsur terhingga tetrahedron yang mempunyai fungsi asas hierarki. Satu kaedah umum bagi pasca-pemprosesan medan data dengan tetrahedron darjah tertinggi dibentangkan. Kaedah ini dibina dengan pendekatan perisian sumber terbuka VTK dimana perisian data penggambaran ParaView boleh didapati secara percuma. Dengan menggunakan pembahagian Merah unsur darjah tertinggi, algoritma yang dilaksanakan telah berjaya menunjukkan penggambaran sehingga darjah keempat tetrahedron dengan menggunakan struktur data yang sama untuk tetrahedron darjah kedua yang terdapat di ParaView. Keputusan jelas menunjukkan peningkatan yang sepadan dalam ketepatan penggambaran apabila darjah polinomial telah meningkat, iaitu, garisan medan kontur semakin licin. Keselarian kod dengan pakej penghantaran mesej openMPI juga telah dilaksanakan untuk meningkatkan prestasi pengiraan dalam platform pengkomputeran pelbagai teras. Keputusan menunjukkan bahawa pengiraan masa yang diambil dalam data pasca pemprosesan

berkurangan apabila pemprosesan secara selari diaktifkan. Algoritma yang telah dimajukan telah dinilai pada masalah geometri dengan jumlah nilai yang tidak diketahui di mana ia didapati mempunyai pendekatan yang berskala.

PARALLEL IMPLEMENTATION OF FIELD VISUALIZATIONS WITH HIGH ORDER TETRAHEDRON FINITE ELEMENTS

ABSTRACT

In the adaptive finite element method (AFEM), high order finite elements are usually used in the computations. In three dimensional simulations, post-processing poses considerable challenge since available data visualization software programs do not accommodate such a high order visualization—common data visualizers can only visualize for up to ten-node tetrahedron elements. This work proposes and implements an efficient framework for data visualization with tetrahedron finite elements having hierarchical basis functions. A general method for post-processing of field data with high order tetrahedra is presented. The method builds upon an approach of the open source visualization software VTK where the data visualizer program `ParaView` is freely available. By using Red Partitioning of high order elements, the implemented algorithm successfully enables visualization of up to fourth order tetrahedra while using the same data structure for second order tetrahedra as available in `ParaView`. The results of the implementation clearly show the corresponding increase in accuracy of visualization when the polynomial orders were increased, i.e., the field contour lines are increasingly smoother. Parallelism of the code with the message passing package openMPI was also implemented to increase the computational performance in a multicore computing platform. The results show that computational times taken in the data post-processing significantly decreases when multicore parallel processing is enabled. The developed algorithm was assessed on various problem geometries with considerable high number of unknowns where it is found that the approach is quite scalable.

CHAPTER ONE

INTRODUCTION

1.1 Introduction

Adaptive finite element methods (AFEM) are widely regarded by today's numerical analysts to be effective tools for highly accurate simulations while at the same time keeping the computer resource at a reasonable level. High order finite elements that are usually used in the adaptive and spectral finite element methods are deemed to provide fastest rates of convergence of the finite element solutions (Babuska and Suri, 1994).However, challenges still exist, especially those related to the efficiency of the implementation. One of the challenges is implementing a fast and efficient post-processor for three dimensional problems in large-scale AFEM simulations for example in problems with more than 5 million unknowns.

The proposed research project is motivated by the need to implement efficient computations for data visualization with high order finite elements by combining optimal algorithms with the hardware capabilities of today's desktop computers. Multi-core central processing units (CPUs) and graphics processing units (GPUs) are becoming more affordable as they penetrate further into the consumer market. Multi-core parallel computing paradigm is increasingly widespread, while on the other hand, current software algorithms are still in large part designed for sequential desktop computers (Buttari et al., 2007).

A common technique in FEM is the adaption of meshes which is usually called *h*-adaptive relates to the adjustment of an existing mesh. Another type of adaptive procedure is the called p-adaptive FEM where the polynomial degrees of interpolation functions are increased to improve the solution process. The combination of both *h* and *p*-adaptive procedures is called *hp*-adaptive FEM. *hp*-adaptive FEM is widely regarded to be most effective method in dealing with problem with singularities (Pessolani, 2002).

This work attempts to implement a post-processing code that is practical for data visualization with high order finite elements. To be practical, such a code will be an extension of an existing post-processor software package that is freely available. The focus of the research study is on accurate visualization of finite elements with high order interpolation functions to address one of the visualization issues with high order finite elements which is the lack of tools for visualizing high order basis function in three dimensions. In this research, tetrahedron elements are studied considering they are geometrically versatile and are used in many automatic meshing algorithms in three dimensions.

Over the last three decades, the rapid growth of the computational power has made it possible to simulate more detailed finite element models. The result output data of these programs can be excessive large that require additional post processor to handle the data. Post processing may be defined as investigation of the results of analysis after a finite element model has been prepared and checked, boundary conditions have been applied, and the model has been solved. Post processing of finite

element data generally requires additional software to organize the output such that it is easily understandable whether the construction output is acceptable or not (Lee and Ahn, 2011). It also can be defined as the program that utilizes a set of instructions already in the software program.

High order visualization is visualization methods based on the knowledge that the data was produced by a high-order finite element simulation. With respect to post-processing in high order visualization, current commercial FEM program like ANSYS and Abaqus are limited in their post-processing ability to visualize high order basis functions. Visualization of computed results is often used to understand and evaluate the numerical approximation of the mathematical model. Very few commercial codes provide post-processing with high order visualization for three dimensions. StressCheck is one of the first commercially available FEA software with an integrated pre- and post-processor which is a suite of analysis modules. The codes make use of the p-version of the finite element method (Apostolis, 2012). The idea of p-adaptive FEM is by increasing the order of the approximating polynomials of discrete problems.

The FEM provides a standard process for converting governing energy principles or governing differential equations in to a system of matrix equations to be solved for an approximate solution. For linear problems such solutions can be very accurate and quickly obtained. In post analysis, a post-processing is utilized to manipulate the data for generating deflected shape of the structure, creating stress plots, animation, etc.

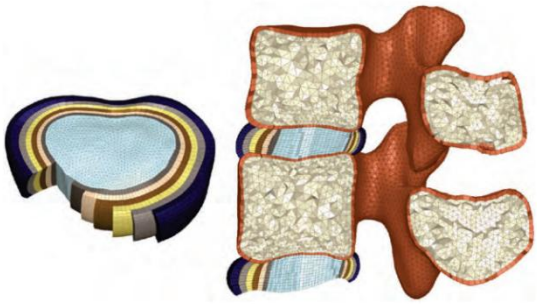
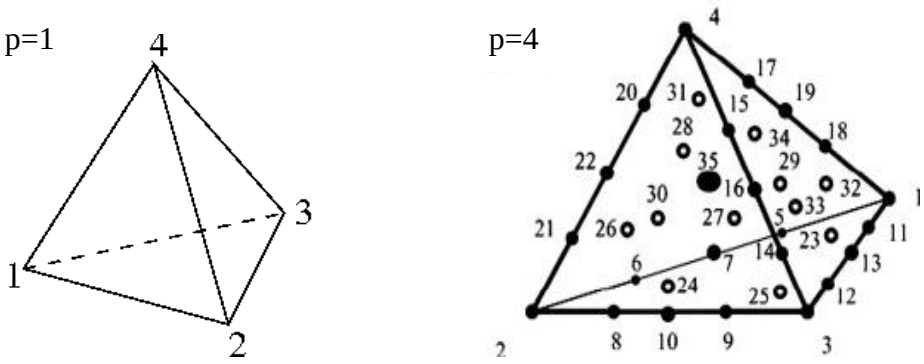


Figure 1.1: visualization of human bone (Scholari and Thessaloniki, 2013).

Figure 1.1 shows the cross section a human bone that is generated from tetrahedron elements. With the post-processing ability such as previewing tool of 3D and 2D graphics, fatigue analysis, calculation of force and moments, shell and volume meshing are very useful for the researcher to understanding behavior of the structure that represent the results with the graphical aid. With more post-processing ability is added, more data can be interpreted and visualized.

In a finite element simulation, each polynomial coefficient must be uniquely solved, and these coefficients are called degrees of freedom (DOF). The number of DOF per finite element governs the accuracy of simulation. Nevertheless, even with high accuracy in the solution, the visualization will not be as accurate if the same number of DOF is not used for visualization. Currently available visualization tools are limited to certain finite element shapes such as tetrahedron, voxel, hexahedron, wedge and pyramid. Until today, most common visualization packages only handle finite elements up to the second order DOF only. When there are not enough degrees of freedom, a valid visualization to the solution is not achieved. This case is depicted

in Figure 1.2 where Figure 1.2 (a) shows the linear tetrahedron element while fourth order element is shown on Figure 1.2 (b). Because visualization tools currently render finite elements of up to degree two only, it is difficult to render the solution for fourth order which has more degrees of freedom. Many current post-processing tools do not have the ability to visualize high order finite element solution of the high order. Correspondingly, visualizing these high order solutions for three dimensional problems is a formidable challenge.



(a) (b)

Figure 1.2: linear and quartic (fourth order) tetrahedra depicting respective degrees of freedom.

1.2 The Need of Parallel Algorithm For High Order Visualization

Parallel computation has been regarded as effective to enhance the computational performance of FEM solution.MPI implementations on computational

performance are theoretically proven by (Thakur et al., 2005) where they were able to optimize parallel processing by 50% faster compared to single processing. With an MPI implementation, these procedures can be done automatically and effectively. Due to the rapid growth of multi-core processor in computing hardware, it is deemed highly desirable to develop post processing algorithm that can maximize the parallel aspect of the multi-core hardware.

1.3 Problem Statement

In the *hp*-adaptive FEM, one of major issues is the lack of tools for visualizing high order basis function in three dimensions. Traditionally, once the finite element solution is obtained, it is visualized using common visualization practice which is based on cutsurfaces, isosurfaces, or volume rendering. Recent development in finite element methods increases both the mesh size (h) and polynomial (p) level of detail or degrees of freedom during a simulation. Finite element solvers that incorporate *hp*-adaptivity are quickly becoming popular since they often converge to a solution with fewer total degrees of freedom than either h or p-adaptivity alone. Nevertheless, one of the problems for the analysts who work with simulations with cubic or higher order elements is the scarcity of post processor tools to enable visualization of higher order solutions in three dimensions. As the basis functions become more complex, it becomes increasingly difficult if not impossible to re-implement them within the visualization system. This problem is crucial issues because it is unlikely that higher order finite elements simulations will be widely used and become popular if the

problem remain unsolved. For instance, tools such as `ParaView` and *Ensight* can currently render finite elements of degree two only. It is the goal of the current work to develop a practical code to contribute in the development of such visualization capability.

1.4 Research objectives

The objectives of this research are:

1. To develop a procedure to extract high order finite element solution data from tetrahedron elements based on a publically available post-processor package and implement the algorithm for high order visualization of scalar fields in 3D.
2. To acquire a level of parallelism in the code and as well as the accuracy of visualization and its efficiency.

1.5 Scope of work

The code development is based on an open source Finite Element solver research code called `PolyDE` (Kasper., 2006). This work represents an extension of the code to enable 3D visualization of solution a scalar field like in the problem of steady state heat conduction in solid. The solution method includes high order tetrahedron elements and visualization with the software program `ParaView`. The visualization is concerned with scalar-valued field variable. Hence only temperature field distribution can be evaluated. All the materials are assumed to be temperature

independent and isotropic. The scope of the code development only covers tetrahedron elements and a mesh with a uniform order of the elements. Parallelism of code is implemented with the Message Passing Interface (MPI) instruction sets and is concerned with the post processing code only, not in finite element solver code itself.

1.6 Organization of the thesis

This thesis provides an in-depth knowledge of the visualization process based on the implementation of high order with FEM methods. This thesis consists of six chapters. The outlines of the remaining five chapters are as follow. Chapter 2 is explained in details the literature review related to this study. The reviews on this chapter are based on high order in FEM, visualization tools in FEM and parallelization. Chapter 3 elaborated in details the hierarchical basis function, the computer code implementation and the theory of parallelization of a system. In this chapter also, the ways on how to implement the post-processor algorithm is shown. In Chapter 4, the case study that have been raised while searching for a model based on algorithms discussed in Chapter 3 is demonstrated. The results based on the implementation on post-processor also demonstrated . Finally, chapter 5 concluded this thesis. The concluding remark together with summarized discussion, addressing issues and limitations justified. The proposal of extension for future work also be stated in this chapter.

CHAPTER 2

LITERATURE REVIEW

2.1 Introduction

In this chapter, a brief introduction about high order finite element method is first presented. By high order, it is meant for third order elements and beyond. The second part reviews the current visualization image processing software programs and their limitation. Then a review on parallelism with message passing interface is presented followed by summary of the literature reviews is given at the end of this chapter to support the simulation work this research study.

Due to the growing popularity of the hp-adaptive FEM to solve a multi-physics problems, higher-order finite element have become essential in these types of simulation. The rapid improvement of computing and processor hardware technology rapidly growth also provide scientist and researcher with powerful tools to solve more complex equations and elements. This is where visualization is required because it is impractical to analyze and understand the solutions without being able to see them.

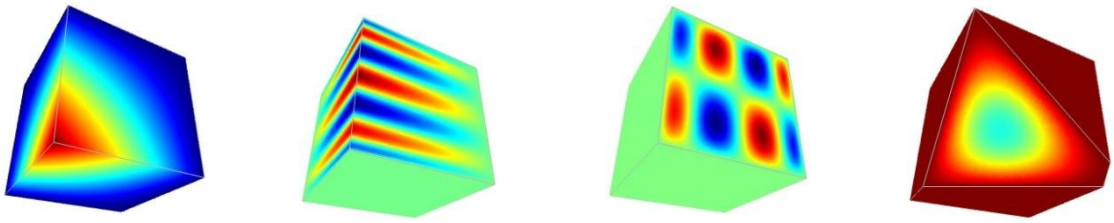
In 1995, Ivo Babuska et al. published a paper in which they described that the finite element method converges exponentially fast when the mesh is refined using a suitable combination of h-refinements (dividing elements into smaller ones) and p-

refinements (increasing their polynomial degree)(Babuska et al., 1995).This pioneering work then makes the method a very attractive since it has a much higher accuracy with a coarser mesh. Numerous researcher observe the hp-FEM in areas like solid and fluid mechanic, geomechanic, electromagnetic and even quantum mechanics (Melenk, 2002, Schwab, 1998).

It is clear from earlier text book focusing on numerical solutions that investigation with high order element in the field of solid mechanics is a great potential (Szabó and Babuška, 1991). Several works focus on specific utilizations of high order elements such as powder compaction analysis (Heisserer et al., 2008), crack propagations and analysis of plate and shell problems (Arciniega and Reddy, 2007, Ainsworth and Pinchedez, 2002, Hakula et al., 1996).In most of these research, the focus is on the capacity of high order element in certain conditions and their contributions to exponential convergence of error.

The higher order element can be described by the degree of basis functions for the element for example linear, quadratic and cubic. The basis functions are used for interpolations of the finite element solution. As an example, a linear tetrahedron has four DOFs attached at each vertex. As the degree of the basis function increased, the order of the element increases equally. The key to the higher order approximation is by adding more DOF into the elements, the higher order elements will give better interpolation of the solution. In lower order FEM, the researchers only work with basis functions associated with grid vertices (vertex functions). In contrast to that, in the higher order FEM researchers moreover consider edge functions, face

functions (corresponding to element faces - 3D only), and bubble functions (higher-order polynomials which vanish on element boundaries). Figure 2.1 depict the higher order basis functions for hexahedral element that are defined in the entire element interior.



vertex function edge function face function bubble function

Figure 2.1:higher order basis functions for hexahedral element in the element interior.

There are a few types of basis functions such as Lagrange basis functions (Berrut and Trefethen, 2004), Hermitian basis functions and hierarchical basis functions (Ingelstroem et al., 2006).Hierarchical basis functions are preferred because of its ease in the implementation of adaptive FEM despite the Lagrange based function may result more orthogonal basis functions. Section 3.5 discusses hierarchical basis functions in detail.

2.2 Visualization Method in FEM

In computational analysis with FEM, the final part is commonly referred to as post-processing. This step mainly involves visualization of the solution of field distribution and other parameters of interest. To the best of the knowledge of the author, visualizing algorithms for high order basis functions is still an open research problem – only very limited discussions on precedures and algorithms are found in

the literature. Much of the existing work is devoted to the development of high order finite element discretization techniques such as high order Discontinuous Galerkin Methods (Ainsworth, 2004, Remacle et al., 2003, Shu, 1989) or high order finite elements (Farin, 1993, Bürger and Gillies, 1989).The present study for visualizing high order solutions includes the issues of contouring, iso-surfacing and cutting. Visualizing high order solution using existing data visualizer may not include the ability to contour and cut. Since high order finite elements significantly affect the approximation and computational performance, the following discussion revolves first around high order basis functions.

A classical post-processor method is as described by Akin and Gray for generating contour lines on isoparametric finite elements by marching along a path of non-varying isovalue, also computed in parametric coordinates (Akin and Gray, 1977). A general non-linear interpolation procedure is presented for contouring any iso-parametric surface. Such an approach adopted the techniques for streamline generation. By the late eighties, the approach is further refined by Gallagher and Nagtegaal by incorporating higher precision isocontour generation in finite element meshes (Gallagher and Nagtegaal, 1989). This is achieved by developing a technique that extends existing 3-D result visualization methods which is isosurfaces on lines for use with discretized volumes. It represents the results as smooth isosurfaces within the volume, using bi-cubic polynomials. Higher accuracy volume renderer for unstructured data which is common in the finite element method was proposed by Williams et al. (Peter, 1998). They incorporated a system that handles meshes whose cells are either linear, quadratic tetrahedron, bricks, prisms, wedges, and pyramids.

However, as higher order elements were being used in three dimensional problems, these approaches which is volume rendering system for unstructured data were not sufficient to completely incorporate the data.

Despite the wide use of high order elements in numerical computation, simulation and computer modeling, far too little attention has been paid for tools for visualizing these solutions in three dimensions (Schroeder et al., 2006). So far, the methods are quite common for visualizing linear basis function on simple elements like triangle, tetrahedron, quadrilaterals and hexahedra. Several popular commercial and open source FEA software, for example PHLEX kernel by Alttair Engineering, Elmer by IT Center for Science (CSC) (Peter Raback, 2001), ANSYS and MSC Nastran have included higher order element in their program (ANSYS®, 2013). Other than that, COMSOL has the capability to solve multi physic problem uses high order element up to third order (Li et al., 2009). Tessellations of basis function with fixed a level of subdivision are one of the most widely used solutions today. The tessellation approaches are promising because current visualization systems are optimized for linear functions but unfortunately very few to describe visualization algorithm so far. This is because tessellation approach creates problem which is hard to correctly tessellating data to produce output compatible with the visualization system that accurately captures the results. Furthermore, the tessellation approach lack of integration between simulation packages and visualization systems that becomes acutely problematic as the complexity of the basis increase when dealing with higher order bases.

Earliest approach for the adaptive finite element method was developed by Eriksson and Johnson which is tailored for parabolic problem in a linear model problem (Mortenson, 1985). By choosing space and time discretization an adaptive algorithm is presented in a finite element method for a linear parabolic problem. For comparison, there are various of technique generated optimal linear grid that can be rendered with standard visualization tools from adaptively subdivision higher-order mesher. In 1995, Ma published a paper which describes a parallel architecture by using ray casting volume rendering technique for tetrahedron grids (Ma, 1995). The algorithm that he proposed are evenly distributed among two to 128 processors resulting to show 60% of efficiency. Previous studies have reported that quadratic tetrahedron with flat faces can be visualize resulting from FEM simulations with DVR accurately (P. L. Williams, 1998). Different cell types can be handle by the system such as bricks, prisms, wedges, and pyramids, but not with high quality visualizations.

Gerstner and Pajarola found that coarse-to-fine refinement scheme can be used for preserving the topology of an extracted isosurface (Gerstner and Pajarola, 2000). This is achieved by assuming linear interpolation within a tetrahedron. Despite being used to extract topology-preserving isosurfaces, their algorithm can also be used to perform controlled topology simplification. There are several reason tetrahedron meshes chosen in many visualization applications compare to other cell type. This is because tetrahedron are simple, elegant, and posses crack preventing adaptive refinement properties. Gregorski has successfully managed to extract adaptive

isosurface from volume adaptive contouring by using a technique called fine-to-coarse and coarse-to-fine mesh refinement (Gregorski et al., 2002).

Corrêa et al. have developed a methodology for render the large models on low end computer system(Corrêa et al., 2002).They are using Out Of Core approach which find the nodes of the model by using appearance algorithm before sends it to the geometry cache, which reads nodes from disk into memory. Appropriate and efficient multiresolution visualization method for piecewise higher order polynomial data on locally refined computational grids was proposed by Haasdonk and colleagues(Haasdonk et al., 2003). This approach uses some error indicator from which it efficiently extracts a continuous h-p-adaptive projection with respect to the visual error. By providing polynomial error indicators and extending the saturation procedure they obtain adaptive projections on locally varying grid levels and polynomial degrees. This projection can then be processed by various local rendering methods, e.g., color coding of data or isosurface extraction. An level of detail(LOD) technique has been found by Callahan et al that eliminated used of tetrahedron topological information and hierarchical data structures by using subset of mesh face (Callahan et al., 2005) Then, the LOD technique is expanded by Vo et al (Solove, 2006). They has present a system for rendering large of volume unstructured grids datasets with multiple capability to view in multiple display called *iRun*.

A more recent work by Jean-François Remacle1 and Nicolas Chevaugeon focuses on classical automatic mesh refinement(AMR) technique to capture the complexity of high order (Remacle et al., 2007).With the AMR technique, the element are divided into sub-element and linear approximations are implemented on each of element. In 2010, Üffinger et al. developed a distributed visualization system which allows for interactive exploration of non-conforming unstructured grids resulting from space-time discontinuous Galerkin simulations, in which each cell has its own higher-order polynomial solution. Their system employs GPU-based ray-casting for direct volume rendering of complex grids which include non-convex, curvilinear cells with varying polynomial degree (Üffinger et al., 2010). Another approach was developed by Sakamoto et al. where they proposed a technique for visualizing a large-scale irregular volume dataset that is generated from a Large Eddy Simulation (LES) based CFD simulation (Sakamoto et al., 2010) . They have extended the particle-based volume rendering (PBVR) technique to fit the distributed computing environment.

Amongst various visualization libraries, the open-source Visualization Toolkit (VTK) is a more established one. VTK supports a wide variety of visualization algorithms including scalar, vector, and volumetric methods. It also provide with advanced modeling techniques such as implicit modeling, polygon reduction, mesh smoothing, cutting, contouring, and Delaunay triangulation. VTK has been shown to be very promising to for FEM with high-order elements (Labs et al., 2010). By using current unstructured mesh class (*UnstructuredGrid),* space is reserved for point coordinates, cell connectivity, cell type and offset into the connectivity array. For a given field defined on the mesh, there are DOF in addition to those defined linearly.

Based on VTK, `ParaView` is a visualization application that can be used to read and display data written in VTK format. Visualization of high order on VTK have been reported by Thompson et al. in 1995(David Thompson, 2005). They are using critical points to partition a finite element into regions by make use of higher order interpolation. The framework that supported an edge based and topology tessellation method was first studied by Shroeder et al. in 2006(Schroeder et al., 2006).The framework use the tessellation of general basis functions that adapt to software architecture in combination with edge-based subdivision algorithm. However, a major problem with this kind of tessellation has resulting increase of data size. Their implementation code however is closed source, while their publications do not completely elaborate the implementation.

As to date, a high order plug-in for `ParaView` to visualize high order data was created by S. Blaise (Blaise, 2012).The plug-in take the advantage from GMSH tools that the element repeatedly split into smaller elements when the relative error is too high until the maximum error is achieved. The new element nodes and field data were calculated by using the high order shape function. However this plug-in currently in development stage and few elements were supported. Presently the high order plug-in use the Lagrange basis functions to rebuild the geometry and the elements that works with it were triangles, quads and hexahedron only. The plug-in have the refinement level feature which is used to calculate error in computation and tolerance.

2.4 Parallel Implementation of Visualization

The relatively rapid growth in microprocessor technology over the last decade has lead to the development of massively parallel hardware and software capable of solving large computational problem. For example, in 1999 Cemal Köse points out that parallel implementation of volume visualization is possible using the volume partitioning method on a large distributed memory multiprocessor system (Köse, 1999). As the simulations move from terascale into the petascale, simulation on pipeline problem description and interpretation of the output have taken a less important to the solver when it comes to performance. Tiankai et al presented their implementation executed in parallel on all simulation components like meshing, partitioning, solver, and visualization with shared data structures and no intermediate I/O (Tiankai, 2006). For the interface between the user, they have work out a computational database system that can be used to generate unstructured hexahedral octree-based meshes with billions of elements. Meanwhile on the server-side, they have developed special I/O strategies that effectively hide I/O costs when transferring individual time step data to memory for rendering calculations. All this approaches are run in parallel and are highly scalable. Despite that, in oil and gas industry, the development and parallelization of multi-phase 3D oil-water reservoir visualization tool has enabled the user to simulate and fetches the large data sets of grid coordinate by using nVIDIA GPU (Siba et al., 2012).Traditional data visualization methods usually deliver poor results when applied to large datasets. Boogaerts et al. introduced a visualization tool for interactive and efficient exploration of high dimensional data using parallel coordinates

(Boogaerts et al., 2012). An algorithm is developed to find an optimal permutation of dimensions, which allows the data miner to immediately see the most important features or irregularities in the dataset.

2.4.1 Message Passing Interface

MPI is a language-independent communications protocol used to program parallel computers, and message-passing application programmer interface, together with protocol and semantic specifications for how its features must behave in any implementation. In this project, implementation of message passing interface (MPI) is introduced to the source code to study the performance, scalability, and portability of the high performance computer. Currently, the standard has several popular versions: version 1.3 (commonly abbreviated MPI-1), which emphasizes message passing and has a static runtime environment, and MPI-2.2 (MPI-2), which includes new features such as parallel I/O, dynamic process management and remote memory operations (Gropp et al., 1999).The Message Passing Interface (MPI) resulted from the efforts for performing distributed-memory parallel computing. Since the MPI become standardize in 1994, it is used widely as standard library in various field such as academic, research and industry.MPI has become the prime model for high-performance computing because it offer the programmer with the ability to point-to-point communication, collective communication, one-sided communication and parallel I/O. Sending and receiving are the two foundational concepts of MPI.

These routines overview usage are found in almost every distributed library or language (Forum, 2009).

When writing a parallel code, positioning of the communication routines within the code play major effect despite the semantic correctness of message passing. So, to minimize the problem, Fahringer and Mehofer has investigated in the past and make as common practice by schedule non-blocking communication process as soon as possible in order to maximize communication/computation overlap between processor (Fahringer and Mehofer, 1997).There are many technical aspects which may have effects in determining a decent program point to place send or receive routines. For example, the request synchronous between send and receive routines will affect the performance for small message sizes. Other than that Pellegrini et al. demonstrate the CPU caches can have a serious impact on communication between processor. In such circumstance, the sent and received call information should be acquired instantly after the communication routines before data get deleted from the caches (Pellegrini et al., 2012) .

With the development of parallel technology and application, the performance analysis and visualization of parallel computing is one of the most important parts. In 2010, Zhang published a paper that visualize and analyze the results of magnetotelluric parallel forward modeling program based on MPI which used finite element method (Zhang, 2010). The parallelization succeeded in collecting the data of program execution and communication time, visualizing and analyzing the speedup and efficiency. A visual analysis method which is animation-based interactive

visualization technique was introduced by Sigovan et al (Sigovan et al., 2013). They streamed the data into an animation that renders individual communication and group process. By doing this, they make it easier to detect potential communication slowdowns when the streamed data that should be aligned fall out of synch. However their MPI implementation ran into over-plot issues which is multiple point end up sharing the same space, positioned on top of one another even though there were only as many particles as processes on screen.

2.4.2 Parallel Virtual Machine

Parallel Virtual Machine (PVM) is a software tool for parallel networking of computers (Geist, 1994). It is designed to allow a network of Unix and/or Windows machines to be used as a single distributed parallel processor. Hence large computational problems can be solved more cost effectively by using the accumulate processing power and memory of many computers.

A performance study to show the parallel analysis program performance running in different computational platforms was carried out in 2000 by Moreti and his collogue (Moretti et al., 2000). They are using a finite element program called FEMOOP (Finite Element Method - Object Oriented Programming) to analyze the structure of data. The software have capabilities to work in a distributed memory environment which are PVM (Parallel Virtual Machine) and MPI (Message Passing Interface). The results shown from their study indicated that the parallel analysis system of PVM and MPI performances of both are very close.

Parallel Virtual Machine (PVM) has been used as a message passing software with Finite Element Method (FEM) to solve the initial stages of deformation due to the stress and strain of a material (Islam and Alias, 2010). They proposed finite element application used to solve one dimension crack propagation problem along with the C programming source code, parallel computation and performance measurement. For performance of PVM, the results have proven that the parallel computation using multi-processor is more efficient than the sequential computation.

2.4.2 MPICH2

MPICH2 is an open source of MPI-1 and MPI-2 (including dynamic process management, one-sided operations, parallel I/O, and other extensions) implementation that supports multiple concurrent user thread and multiple options for handling process (Balaji et al., 2012). It also serve as a standard for message-passing for distributed memory applications used in parallel computing.MPI implementation for the processor can shown in Figure 2.1.

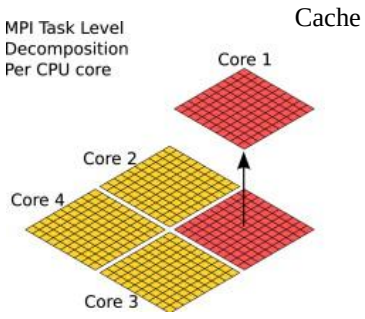


Figure 2.2: Different levels of parallelism and actual data layout in the CPU implementation.

MPICH2 is one of the most popular implementations of MPI and become the pioneer of other major MPI implementation such as Intel MPI, MVAPICH2 and many more.

2.4.3 Hyper Threading

A Hyper-Threading Technology (HT Technology) allowed a computer system shows up to have more processors than it really has. With this technology, one physical processor with a single-core is regarded as two logical processors in the computer system. To make this happen, Hyper-Threading imitate the architectural state on each processor while combining the single set of resources (Marr et al., 2002). As a result, Operating system (OS) will declare to have two CPUs instead of one. Therefore, user programs can schedule processes or threads to logical processors as they would on conventional physical processors in a multiprocessor system.

However, the effects of HT Technology on computer system performance vary according to applications characteristics and the system configuration.(Onur Celebioglu, 2003) has shown that parallelized applications have the performance increase in some cases despite small degradation performance on other cases. They have identified that Hyper-Threading may cause overhead on cache, memory and communications traffic that affecting computer performance. Several attempts have been made to measure the impact of HT on processor utilization (Saini et al., 2011).They have revealed that unstructured-grid software benefit from HT whereas the structured-grid application have performance degradation. In the 1990s, Vuk Marojevic reported the HT impacts on CPU performances (Marojevic, 2004). He discovered that CPU performance increases when the application(VMware) supports

multithreading (MT).Despite that a performance decrease is also documented, when running on application or operating system that has unsupported HT feature.

2.5 Summary

The parallel implementations in computer system have proven their capability in representing the real world applications. Many improvements have been made in order to obtain a concrete approach to increase efficiency of a system. The variety of visualization technique also has shown promising achievement in solving many system applications. Inspiring by the benefits served by the visualization limitation and capability of high order, thus it will be the main objective that will be researched in this thesis. Here the focus is to extend the visualization capability from second order limitation into high order with implementation of parallelization.