Chapter 1

Introduction

Structural analysis involves three important phases: (1) modeling; (2) analysis; and (3) result processing. There exist several computer programs to automate the analysis and result processing phase. However, modeling is still done largely by hand, which is a tedious task. This task is particularly difficult for 3D wood frame structures and becomes more tedious if one intends to do a finite element analysis that requires repeated re-discretization to obtain an accurate solution. The manual creation of these discretizations, also called mesh generation, requires extensive human effort, skill and time. Though the efforts to automatically generate these discretizations can be traced back to the early 1970’s, the most rapid advances have occurred in the last few years. Early efforts in the development of mesh generators were made in procedural programming languages such as C or FORTRAN. The main reason was the easy integration and down streaming of output to finite element analysis codes, which were mostly written in C or FORTRAN. In recent years there has been a paradigm shift from a procedural to object oriented design for finite element analysis programs, further resulting in a huge demand for object oriented mesh generators. Moreover, studies of finite element meshing techniques have shown that the problem lends itself very well to object oriented design philosophies.
This thesis presents the development of an automatic finite element mesh generator, named WoodFrameMesh, for modeling a wide variety of light framed wooden structures with complex geometries. The program is developed in C++ which is one of the most popular languages for object oriented programming. The program data structure makes extensive use of the Standard Template Library (STL) and pointers, which are among the most powerful features of C++. Overall, a finite element mesh generation platform is developed that is based on object oriented design philosophies. The WoodFrameMesh program generates output files in SAP 2000\textsuperscript{1} version 7.xx input format. This format of input file can also be read by another finite element analysis program named WoodFrameSolver\textsuperscript{2}. The WoodFrameSolver program was developed at Virginia Tech by a group of students and professors (WoodFrameSolver team members\textsuperscript{3}) including the author. The analysis of the models developed using the WoodFrameMesh is performed using both the WoodFrameSolver and SAP 2000. Having a SAP 2000 version 7.xx input format for the output file generated by the WoodFrameMesh program greatly facilitated the testing of models using both SAP 2000 and WoodFrameSolver. A chapter discussing the development of the WoodFrameSolver program, its extensibility and its verification is also presented in this thesis.

1.1 Problem Statement

An important task in the analysis of light frame wooden structures is to separate the “structural” and “non-structural” aspects of response because their stiffnesses are similar. Therefore, the modeling of light frame wooden structures poses a big problem as it is difficult and sometimes impossible to separate “structural” and “non structural” components. One of the most accurate ways to model these structures is using finite elements. The last few decades has seen a rise of the finite element method for modeling and analyzing complicated problems. Analyzing structures using finite elements comes with a huge problem of discretization for generation of a valid finite element mesh.

\textsuperscript{1} SAP 2000 is an integrated structural analysis and design program by Computers and Structures Inc., Berkeley, CA.
\textsuperscript{2} WoodFrameSolver is a finite element analysis engine developed at Virginia Tech
\textsuperscript{3} Other WoodFrameSolver team members: Paul W. Spears (former Graduate Student, Virginia Tech), Dr. Finley Charney (Associate Professor, Virginia Tech), Dr. Samuel Kassegne (Visiting Scientist, University of California – San Diego) and Hariharan Shivkumar Iyer (Graduate Student, Virginia Tech)
Another critical issue which arises is the amount of time and effort spent during manual generation of finite element models. The development of finite element models constitutes the generation of a computational grid and setting up boundary conditions. Computational grid generation, also known as mesh generation, is a discretization process, which has become as demanding as the efforts required to perform the computation for which meshing is required. Mesh generation is a tedious task if one wants to do a complete finite element analysis of a structure; in addition it also requires some expertise. We also know that applicability and accuracy of finite element analysis are highly dependent on the validity and quality of meshes generated. Thus it is important to have an acceptable mesh and efficient process of mesh generation. Often it is beneficial to automate such processes when one intends to analyze larger and more geometrically complex structures. Advantages of automatic mesh generation include:

1.) Most of the structures differ very slightly in their geometry and modeling method.
2.) The geometry and shape of realistic engineering structures are intrinsically complex, usually composed of several materials and regions.
3.) Only a tiny fraction of nodes associated with the objects needs to be defined by the user.
4.) The time required to develop a complete finite element model of a structure using the generator would be in seconds as compared to the hours spent developing it manually.
5.) Changes can easily be incorporated in the program and required input information if one decides to alter the modeling method at a later stage and generate new models.
6.) It is cost effective when one has to analyze a large number of structures.
7.) There is an ample amount of literature and applied work available on automatic mesh generation.
Hence, the main focus of this thesis is on the development of an automatic mesh generator for finite element analysis of light frame wooden structures. In the context of the need, the generator should at least allow for:

1.) The generation of a triangular mesh for flat, and sloped diaphragms, and for walls with multiple openings.
2.) The generation of quadrilateral meshes for walls.
3.) Uniform mesh.
4.) Automatic inclusion of boundary conditions at the desired positions, using springs, restraints and non-linear links.
5.) Flexible input file format.
6.) Robust and efficient implementation.
7.) Object oriented architecture with emphasis on the extensibility, computational efficiency and memory requirement.

1.2 Overview of Chapters

The thesis is divided into seven chapters and includes two appendixes. A brief introduction of the next six chapters and appendixes is as follows:

Chapter 2 presents a literature review of available 2D mesh generation algorithms. We see that generating 2D meshes is no longer a challenge as many mesh generation algorithms are available.

Chapter 3 presents an analytical study of the resultant force on the tie down attached to one of the shear walls of a simple structure, and compares this to its maximum ideal value. The analysis is done to understand how lateral forces are transferred to vertical tie downs under certain situations and varying parameters, i.e., treating stud walls as roller supports or springs, varying spring stiffnesses and wall width, etc.

Chapter 4 presents a detailed discussion of mesh generation algorithms implemented in the WoodFrameMesh program. Several crucial computational geometry problems
encountered during code development and their solutions are also discussed in this chapter. Each discussion accompanies a graphical illustration for easy understanding of the solution.

**Chapter 5** presents the architecture of the WoodFrameMesh program and discusses finite element mesh generation using object oriented C++. We see how the general operational units in the program are treated as objects, and not as functions as it is in procedural languages. Some object oriented concepts are also discussed in this chapter.

**Chapter 6** presents the architecture of the WoodFrameSolver finite element analysis engine. The chapter also describes how the developed code can be extended to add new features. Also, a brief discussion is presented of the analysis results obtained using WoodFrameSolver, and the comparison of the results with those obtained using SAP 2000.

**Chapter 7** summarizes all the work done, and presents recommendations for future development of the WoodFrameMesh and WoodFrameSolver programs.

**Appendix A** presents the users manual of the mesh generator program. The manual discusses the program capabilities and the desired input file format. It also presents a tutorial which discusses example models with varying geometries.

**Appendix B** presents the analysis results of some models generated by the WoodFrameMesh program and SAP 2000. The analysis is done using both WoodFrameSolver and SAP 2000 and a comparison of results is presented in the form of tables. Performance of the implemented solver and the equation renumberer is also presented.