A Mesh-Free Finite Element Solution for Unilateral Contact Problems

by

Alexander Grishin

A Dissertation Presented in Partial Fulfillment
of the Requirements for the Degree
Doctor of Philosophy

Approved November 2010 by the
Graduate Supervisory Committee:

Jami Shah, Chair
Joe Davidson
Keith Hjelmstad
Ken Huebner
Gerald Farin
Pedro Peralta

ARIZONA STATE UNIVERSITY

December 2010

ABSTRACT

Current trends in the Computer Aided Engineering (CAE) involve the integration of legacy mesh-based finite element software with newer solid-modeling kernels or full CAD systems in order to simplify laborious or highly specialized tasks in engineering analysis. In particular, mesh generation is becoming increasingly automated. In addition, emphasis is increasingly placed on full assembly (multi-part) models, which in turn necessitates an automated approach to contact analysis. This task is challenging due to increases in algebraic system size, as well as increases in the number of distorted elements – both of which necessitate manual intervention to maintain accuracy and conserve computer resources.

In this investigation, it is demonstrated that the use of a mesh-free B-Spline finite element basis for structural contact problems results in significantly smaller algebraic systems than mesh-based approaches for similar grid spacings. The relative error in calculated contact pressure is evaluated for simple two dimensional smooth domains at discrete points within the contact zone and compared to the analytical Hertz solution, as well as traditional mesh-based finite element solutions for similar grid spacings. For smooth curved domains, the relative error in contact pressure is shown to be less than for bi-quadratic Serendipity elements. The finite element formulation draws on some recent innovations, in which the domain to be analyzed is integrated with the use of transformed Gauss points within the domain, and boundary conditions are applied via distance functions (R-functions). However, the basis is stabilized through a

novel selective normalization procedure. In addition, a novel contact algorithm is presented in which the B-Spline support grid is re-used for contact detection. The algorithm is demonstrated for two simple 2-dimensional assemblies. Finally, a modified Penalty Method is demonstrated for connecting elements with incompatible bases.

DEDICATION

To my mother and father.

ACKNOWLEDGMENTS

The author would first like to thank all the members of his committee for their kind indulgence, as this investigation involves a topic and set of disciplines not currently pursued (at least explicitly) by the faculty of Arizona State University. One hopes this will change in the near future, but in the mean time, the committee has shown tremendous grace and generosity in supporting the author's efforts when it would have been simpler (and justifiable) not to do so.

Thanks must also be given to the author's employer, Phoenix Analysis and Design Technologies (PADT), for giving him a sabbatical to complete his dissertation, and for offering extreme flexibility in his work schedule during the entire four years he pursued his degree. Without this support, it is doubtful he would have been able to find the time and finances to complete it.

Finally, the author will be forever indebted to Dr. Jami Shah, who not only exemplified the generosity shown by the committee, but suggested the research topic to begin with, and pointed the author in the specific direction taken. It should be emphasized that he did this even though this research topic falls outside his area of expertise and funding.

TABLE OF CONTENTS

Page

CHAPTER Page

CHAPTER Page

LIST OF TABLES

LIST OF FIGURES

Figure Page

Figure Page

Figure Page

Figure Page

LIST OF SYMBOLS

Symbol	Meaning
H	The set of points in the half-space defined by $f(x,yz)\leq 0$ (Chapter 1). The set of all basis indices that intersect the domain, Ω (Chapter 4)
$\mathbf{C}$	Space curve over a single parameter, t
b	Function over Ω representing body loads. Minor diameter of contact ellipse. Bi-variate B-Spline basis function
$\mathbf{B}$	Vector of Bézier basis functions (Chapter 2). Vector of selectively normalized FE basis (Chapter 4
$\mathbf{r}$, $\mathbf{r}_b$, $\mathbf{r}_f$, $\mathbf{r}_p^c$, $\mathbf{r}_\varphi$	Residual load vector, residual body load vector over Ω, surface load residual vector over Γ, residual penalty surface load vector over contact surface, residual displacement load vector over Ω
$\mathbf{S}$	parametric tensor-product surface
$\mathbf{N}$	Vector of B-Spline basis functions, or parametric bases other than Bézier
N	B-Spline, or any other parametric basis other than Bézier
$\mathbf{P}$	Bézier or B-Spline curve control points (used to construct $\mathbf{C}$)
$\mathbf{p}$	positioin vector corresponding to points on a contact surface

Symbol	Meaning
P	Reference point
p	Order of del operator in functional (Chapter 2). Pressure
S_i	Screw representing an assembly mating pair, i
f	A general function (Chapter 1). Function over Γ representing external traction.
$\mathbf{f}$	Vector over Γ representing external traction
$\mathbf{F_e}$	Element load vector in global coordinates
$\mathbf{F}$	Global external load vector
$\mathbf{k_e}$	Element stiffness matrix in global coordinates
$\mathbf{K}$	Global stiffness matrix in global coordinates
$\mathbf{x_e}$	Element deflection in global coordinates

Chapter 1

INTRODUCTION

The aim of this research is to assess the feasibility of implementing a mesh-free Finite Element Analysis (FEA) formulation for use as an automated analysis tool within a proposed assembly design system. The research is to be carried out by the Design Automation Laboratory (DAL) at Arizona State University under the guidance of Dr. Jami Shah. Once feasibility of the mesh-free analysis methodology is established, a second phase of research would determine the most proper and robust implementation of this methodology within the larger framework of the automated assembly modeling program. The current phase, however, will focus solely on the selection and testing of the mesh-free system. A mesh-free analysis is chosen for this effort because it is felt that a truly robust assembly design tool should not rely on meshing technology for essentially two reasons: 1.) The authors have found that producing a high quality mesh often results in manual intervention - depending on the quality of the underlying solid model geometry and the desired quality of the mesh. 2.) Many large assemblies, when meshed appropriately, result in systems of equations that are too large for practical solution (even on today's powerful PC's and workstations). One important selection criteria for the automated mesh-free system is that it should provide optimal algebraic system size to achieve a desired accuracy. That is to say, for a given exact solution, the numerical approximation should yield a desired level of accuracy with a minimum number of equations compared to other methods. Within the context of the assembly design system, a major recurring

task of the mesh-free tool would be to resolve contact/interface stresses and reaction forces. The precise algorithm for doing this is expected to be novel for this application and will also be developed in the course of the investigation. What follows is a rough description of the proposed assembly design system, how the mesh-free analysis tool fits in with it, a summary of the current state-of-the art, and an overview of the structure of this investigation.

1.1 An assembly design system

Although mechanical assemblies are very common today (March, 2009), their study as design abstractions began only relatively recently (Nevins and Whitney 1989). Assemblies can be regarded alternately as collections of parts, hierarchical systems, and networks. The proposed assembly design system would maintain a network-centric database utilizing assembly feature models (Bourjault 1984). The nodes of this diagram would link to subassemblies and parts, while the edges would link to connectivity data (such as feature transforms (Whitney 2004)). This information would in-turn be utilized by the mesh-free analysis tool (e.g. automatic surface-surface contact detection would not be necessary. Surface-to-surface contact information would be captured in the part-connectivity algorithm) to calculate interface stresses and forces, as well as part stresses and other system response quantities of interest. A flowchart depicting the top-down assembly design philosophy of the proposed system is depicted Figure 1.1.

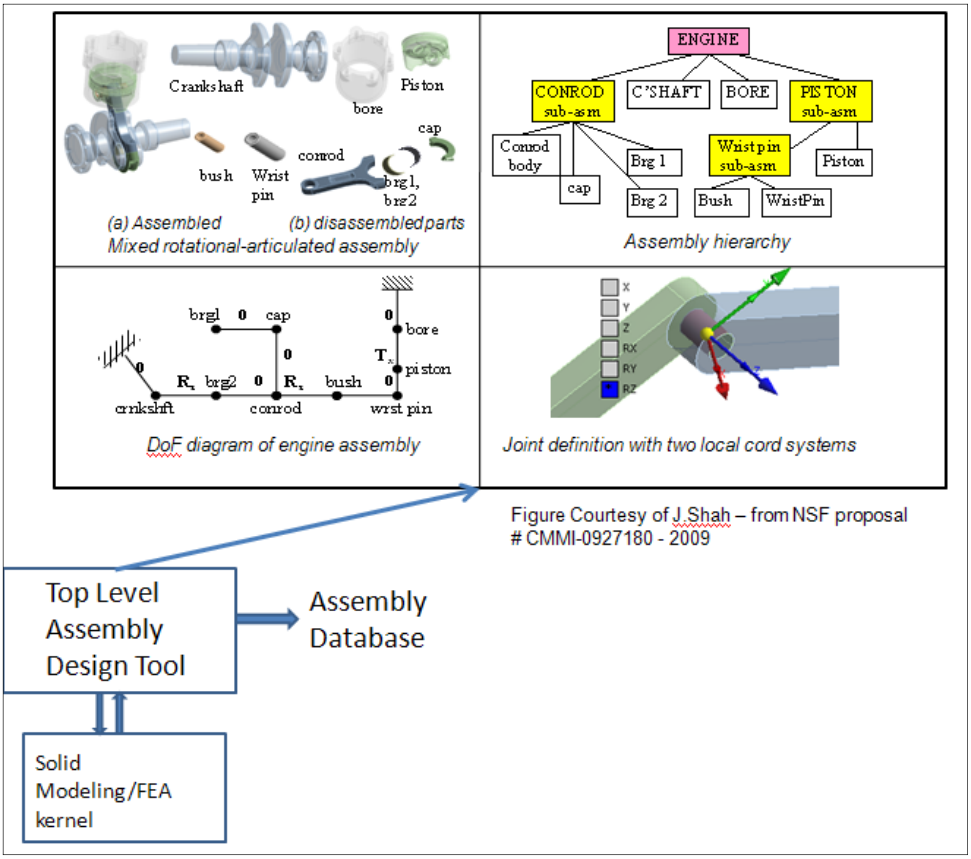


Figure 1.1. Design tool flowchart

1.2 Finite element sub-system architecture

The field of computer simulation is vast and growing. Virtually every industry linked to product development performs various structural, thermal, fluid, or multi-field analyses to their products on a daily basis (or rents this service from dedicated consultants). As this need grows, and computers become powerful, the complexity of the products to analyze also grows. It is not uncommon for the solid-model representation of an assembly to contain over 1000 assembly features. The increasing number of such features, as well as their

increasingly complex geometry, present serious challenges to the structural analyst. To understand why, one must consider the standard practice when performing an analysis of a product. With rare exceptions, the product's geometric and topological characteristics are captured in a 3-dimensional solid model (or CAD model) – due to a process often referred to as "bottom up" design (the individual parts are designed first, and then placed in an assembly which now inherits all the information of its constituents. This is in contrast to "top down" design, in which an assembly's basic heuristics and functional requirements are embodied in a simplified model, whose detailed part descriptions get filled in later).

The following observations extend more broadly to cover 2-dimensonal, as well as non-structural products. However, the current proposal will focus on full, 3-dimensional products represented by solid models analyzed for their structural behavior. The solid model is a mathematical description, the exact form of which is unique to the solid modeling software's geometry "kernel" or "engine". This mathematical description is then converted to "neutral", or some intermediate standard exchange file format, which is then read and interpreted from within a special purpose finite element pre-processor. This pre-processor typically relies (like the CAD program) on a commercial geometry kernel.

The process of geometry/topology transfer from solid model to FE pre-processor is significantly more reliable when both CAD system and FE program utilize the same geometry kernel. In recent years, this is becoming increasingly common, and the geometry transfer step (the "Neutral File" step in Figure 1.2) is

being eliminated. Once the original geometry is reproduced, it must be discretized, or "meshed" (the elements defined), and the physics environment described (material constitutive laws, loads, boundary conditions). The system equations are then assembled and solved. Finally, the analyst queries various calculated model response quantities from a solution database. This last step is referred to as "post-processing". A visual diagram of this procedure is offered in Figure 1.2.

The Finite Element Method (FEM) is by far the dominant mathematical framework for structural analysis, however there are others, as will be discussed. Most other popular mathematical methods rely on the same procedure as described above, differing only in the level of fidelity to the original model's topological features and geometry, and certain details of generating the mesh. When considering computational expense, and even solution accuracy, it is often expedient to "de-feature" a solid model, removing topological or geometric features which are not expected to affect solution, or to replace them with reduced-order elements, such as springs, beams, or shells, which accurately account for structural behavior but do not reflect the solid model's true topology (spring and beam elements are 1-dimensional. Shell elements are two-dimensional). Such considerations are entirely appropriate and will always be necessary for meaningful structural analyses. However, as computational power increases and its corresponding cost decreases, analytical practices tend to favor retaining more structurally redundant topological features and analyzing models of ever increasing size and complexity.

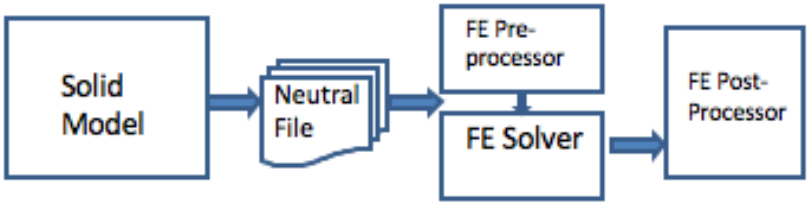


Figure 1.2. Typical structural analysis procedure

This poses serious challenges to the importation and meshing of solid models. These problems are essentially twofold: First, most CAD systems today rely on a system that represents solids as volumes bounded by parametric "trimmed" surfaces (this is referred to as "B-rep", for Boundary representation). This representation, though very flexible, relies on techniques for generating parametric curves that approximate surface-to-surface intersections (the problem of parametric surface intersections is a current topic of research in CAGD (Farin 2002)). These techniques often differ slightly between CAD systems. Because of this, a particular surface in one CAD system will not be trimmed correctly in another CAD system (or FE pre-processor) once it is transferred via a neutral file. Thus, as the number of features and surfaces in a model increases, the potential for import/export corruption rises. Second, and perhaps more importantly, traditional finite element formulations require that elements fall within a certain range of shape quality (usually defined as a ratio between element angles and/or edge lengths) to maintain equation stability. The third and final point is that generating a mesh of acceptable quality is often the most time consuming task

facing an analyst/designer. This requirement generates increasingly more elements (smaller mesh sizes) as the geometry becomes more complex, thus driving up the size of the matrices that are to be generated, stored, and solved.

Resolution of these difficulties requires a radical change to the process shown in Figure 1.2. Even if problems involving trimmed surfaces are eliminated (future CAD systems may employ alternative representations), the meshing problems mentioned above still go unaddressed. Perhaps the best solution to these problems involves the incorporation of a robust mesh-free finite element formulation. Furthermore, such a solution should operate directly on the CAD geometry, without going through the intermediate export step. Thus, the system that is proposed in this investigation would ultimately take the form of a suite of code that would link directly with the CAD geometry kernel as shown in Figure 1.3. In particular, all FE pre-processing and post-processing would take place within the CAD environment, making extensive re-use of commonly available solid-modeling algorithms. Such pre-processing would include assembly feature recognition and part connectivity data.

1.3 Mesh-free finite element feasibility study

Before the analysis system described above can be implemented, a mesh-free methodology must be chosen and tested. Algorithms utilizing the methodology must be developed and optimized. The current state of research in mesh-free finite element formulations has matured (discussed in the next section)

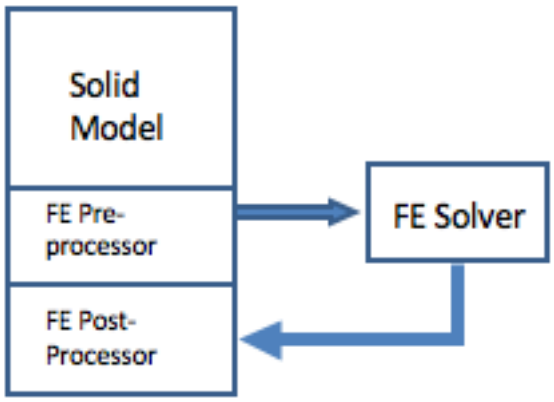


Figure 1.3. Proposed structural analysis architectural framework

to the point where it is believed at least one proposed technology is a good candidate for use in a general-purpose CAD-centric structural analysis system. The selection and testing of one these technologies for use in general unilateral contact problems forms the thesis of this investigation. The selection process begins by supplying the relevant background in assembly design, structural analysis, finite element analysis, and mesh-free technology in Chapter 2. As the primary function of the analysis tool would be the determination of contact stresses and displacements, a review of computational contact mechanics closes this chapter. By the end of this chapter, it will be seen that the current state-of-the art in particle-based mesh-free technology is not robust enough for use in a general assembly design environment. In Chapter 3, it will be seen that there is indeed one mesh-free formulation, investigated by two separate research teams, which seems to hold promise. This is the non-

conforming tensor-product B-Spline grid similar to that of Shapiro/Höllig (Rvachev et al. 2000; K. Hoellig, Reif, U., and Wipper 2001). This prior work will be summarized in the form of a literature review. In Chapter 4, the system is developed and studied for feasibility in a 2-dimensional framework completely within Mathematica®. Problems involving contact between simple solids will be analyzed and compared with exact solutions, as well as traditional finite element solutions. The system will be tested to assess its accuracy in calculated stresses, as well as the robustness of the contact algorithm. The results of this validation study are summarized in Chapter 5. Chapter 6 will summarize the original contributions made by these efforts, and Chapter 7 will outline plans for future work.

Chapter 2

BACKGROUND

Since this investigation seeks ultimately to combine the disciplines of Computer-Aided Geometric Design (CAGD) and structural analysis, the topics of CAGD, assembly design, and structural analysis (in which finite element methods dominate), which are pertinent to the current research are reviewed to provide background. It will be seen that the current state-of-the-art in structural analysis evolved from a history pre-dating that of CAGD, and the emergence of the latter has not had a significant impact on the former. It is believed this is one reason for some of the challenges being faced today in the integration of CAGD and Finite Element Analysis (FEA). Research in each of the two domains tends to focus on issues historically distinct to those domains and one finds very little synthesis of the two in the literature (exceptions are found in the work of Shapiro (Shapiro and Tsukanov 1999a), (Hoellig 2003), and Hughes (Hughes, Cotrell, and Bazilevs 2005)). However, it is believed this is beginning to change as the challenges described in the previous section become more urgent.

2.1 CAGD, solid modeling, and CAD systems

The word CAD is an acronym for Computer Aided Design. Modern CAD systems integrate a number of engineering design and manufacturing tasks. For example, most commercial systems today allow the engineer to create a full 3-dimensional representation of virtually any possible volume (although there are certain topological restrictions). They are also capable of extracting any information about this volume (mass properties, volume, feature sizes, etc.) and

creating engineering drawings and machine tool paths for actually making the part. As the CAE industry continues to mature, doubtless more functionality will be added. This synthesis of various services and functionality tends to obscure the fact that these tasks are usually carried out by entirely separate pieces of software (sometimes acquired from other firms), each with its own unique history. To begin with, it should be mentioned that computerized engineering drawing systems preceded the era of solid modeling. The most sophisticated commercial CAD systems of the 1970's offered "wireframe" modeling capability to enhance their digital drawing software. Wireframe modeling consisted of connecting points in space with lines and curves. Different views could be obtained with straightforward projective geometry algorithms. Although this capability was a significant improvement over strict 2D digital drafting, wireframe models could be ambiguous, as topological characteristics were not addressed. In particular, there was no clear way to automatically determine whether a point lay inside our outside a manifold. During this time, universities and private research organizations were developing the concept of solid modeling.

The term "solid modeling" encompasses a body of theory and techniques focused on the representation of solids by a computer. In principle, these representations should permit any well-defined geometrical property of a solid to be calculated automatically (Shapiro 2001). According to (Requicha and Voelcker 1982), "research in solid modeling became visible in the mid 1960's, and by the mid-1970's a first generation of experimental systems had appeared." The first commercial solid modeling engines were integrated into CAD systems

by the early 1980's and as of this writing, all commercial CAD systems employ one of a handful of commercial or proprietary geometry engines. Table 1.2 provides a list of commercial and experimental solid modeling systems circa 1983 (Requicha and Voelcker 1983) and 2008. Systems which survive today are highlighted in red (even if the names have changed). New arrivals as of 2008 are highlighted in blue. The attrition is due mainly to the emerging dominance of B-rep algorithms and corporate mergers. It is worth noting that Pro/Engineer® entered the field in 1988 with its own proprietary B-rep engine and was extremely successful. It was the first parametric (the word "parametric" here is used to refer to variable dimensions used to drive feature size and placement – not to the parameters used in parametric curves and surfaces), feature-based modeler (discussed later). Its arrival probably accelerated the rate of disappearance of the other solid modelers in Table 2.1 and forced other CAD systems to adopt a similar approach.

2.1 A list of solid modeling engines circa 1983 vs. 2008

1983			2008	
Solid Modeler	Type		Solid Modeler	Type
CATIA	B-rep		CATIA	B-rep
CATSOFT	CSG			
DDM-SOLIDS	B-rep			
EUCLID	B-rep		Open-Cascade	B-rep
GEOMOD-II	B-rep		I-DEAS	B-rep
ICEM SOLID MODELING	CSG			
ICM GMS	B-rep			
MEDUSA	B-rep			
PADL-1,2	CSG			
PATRAN-G	Cell Decomp.			
ROMULUS	B-rep		Parasolid	B-rep
SOLIDESIGN	B-rep			
SOLIDS MODELING-II	CSG			
SYNTHAVISION	CSG			
TIPS-1	CSG			
UNIS-CAD	B-rep			
UNISOLIDS	CSG			
			ACIS	**B-rep**
			Pro/Engineer	**B-rep**

The systems of the early 1980's were very successful in their day, and opened the door for a much larger second generation of development in the late 1980's. Most of the development could be broadly categorized as following under one of three basic representational schemes (There are actually 6 known schemes (Requicha 1980), but only three find wide usage in engineering systems): Constructive Solid Geometry (CSG), Boundary Representation (B-rep), and

Domain (or cellular) Decomposition. This last representational scheme is rarely used today and so it is removed from discussion. The first experimental CAD systems used CSG-based schemes. In the eighties, these were largely abandoned in favor of B-rep algorithms. Even though the CSG approach is no longer used in modern CAD systems as of this writing, research in the field continues (see (Hartmann 1998), for example) and it has direct theoretical and practical applicability to both CAD systems and some mesh-free FE systems (a topic which will be discussed later), and so it will be discussed first.

The major motivation behind the CSG approach was to achieve a "correct" and complete definition of a solid based upon a small, finite set of algebraic operators on half-spaces. These half-spaces form an infinite set of the form

$$H = \{(x, y, z) \mid f(x, y, z) \leq 0\} \tag{2.1}$$

where f is a polynomial function. Finite subsets of this set (sometimes called R-sets) are generally accepted as suitable models for solids (Requicha 1980). A further subset of this set (called r-sets) is required for most engineering solids. This subset is characterized by the fact that it is "regular" (i.e. it equals the closure of its interior). The theoretical foundation for this representational scheme was laid at the University of Rochester in the early to mid-1970's and expressed in the experimental programs PADL 1 and 2 (Voelcker 1974; Voelcker 1978). One advantage of this representational scheme is that all topological information is generated automatically as a consequence of Boolean operations on the r-sets (in contrast, if a volume is subtracted from another volume in the B-rep scheme, the

resulting surfaces must be explicitly created). A second advantage is that the algorithm for representing a particular solid is simple and compact: it involves recursive descent down a binary tree – the tree providing the entire database necessary to the store the representation. A third advantage is that "correctness" of a particular part representation is guaranteed by the mathematical soundness of the theory of Boolean operators over sets of type (2.1) (when Boolean operations are modified to handle tangent objects). Figure 2.1 gives a simple example of how a solid is represented using the CSG approach on simple "primitives", such as cylinders and hexahedra.

Opinion seems to differ on why the CSG approach has been abandoned (as of this writing. See (Shapiro 2001) for one such opinion), but it is clear that one serious deficiency of the technique lies in the fact that often solids must be constructed with surfaces which cannot be described by equation (2.1).

The B-rep approach involves representing a solid model by defining its boundaries in terms of parametric surfaces (usually tensor-product surfaces), whose boundaries are in turn defined by parametric trimming curves. Figure 2.2 depicts how the same simple solid of Figure 2.1 may be represented using the B-rep approach.

The trimming curves are themselves parametric space curves. NURBS (Non-Uniform Rational B-Splines) are a particular mathematical form of parametric curves and surfaces. NURBS curves and surfaces are a generalization of a simpler parametric construction called a Bézier curve. A Bézier curve is

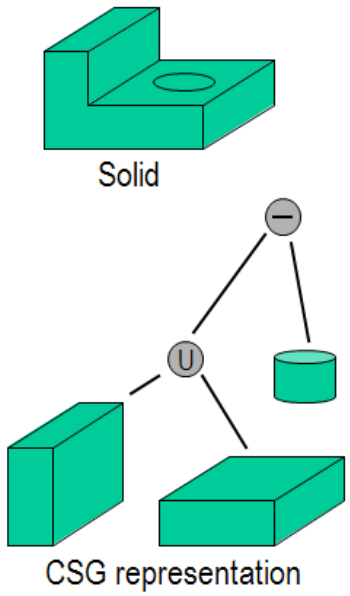


Figure 2.1. CSG representation of a simple solid

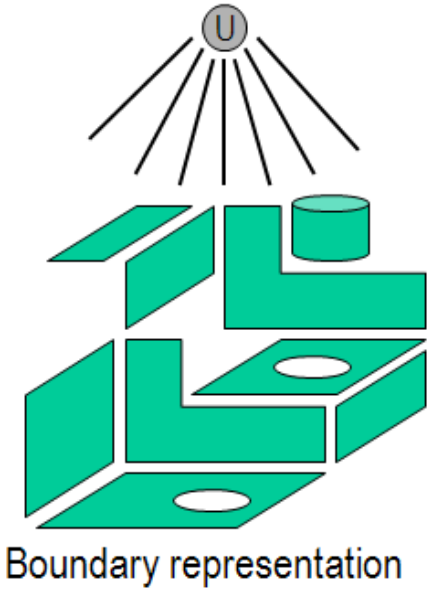


Figure 2.2. B-rep of the same solid

space curve defined by a parameter t and n+1 “control points”, P. The exact form is given in equation (2.2):

$$\mathbf{C}(t) = \sum_{i=0}^{n} \mathbf{B}_{i,n}(t)\mathbf{P}_i, \quad t \in [0,1]$$
$$\mathbf{B}_{i,n}(t) = \binom{n}{i} t^i (1-t)^{n-i}, \quad i = 0,...n \tag{2.2}$$

The curve always interpolates, and is tangent to, the two end control points, C_0 and C_n. If control points are joined via straight lines consecutively, they form a “control polygon”. All bézier curves possess what is called the “Convex Hull Property”, meaning that the curve is guaranteed to lie entirely within the control polygon for parameter values between zero and 1. As an example, a linear Bézier curve would interpolate two control points and yield a straight line segment. As another example, a third-order (n=3) Bézier curve would involve four control points and interpolate the first and fourth ($\mathbf{P}_0$ and $\mathbf{P}_3$), as shown in Figure 2.3.

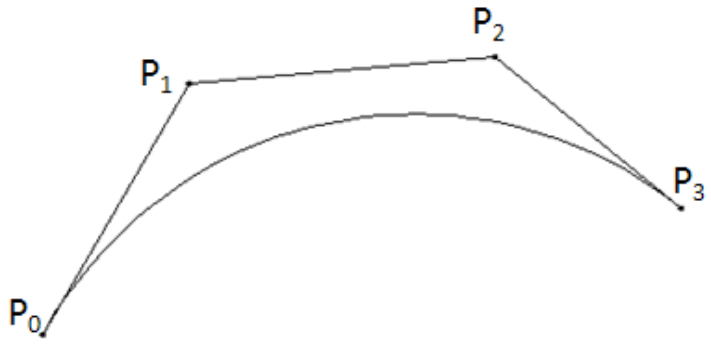


Figure 2.3. Third order Bézier curve with control polygon

The curve in Figure 2.3 can be easily manipulated by simply moving the control points and re-calculating equations (2.2) based on the new positions. This is a very versatile and computationally efficient scheme for representing geometry that may include curves of arbitrary shape. Note also that equations (2.2) represent a linear system of equations. This is convenient for calculating Bézier approximations or interpolations of other types of curves. Furthermore, more complicated curves of a given degree can be constructed by joining Bézier curves together at their endpoints. Curve tangency (often referred to as C_1 continuity) can be ensured by ensuring that line segments P_{n-1},P_n on the first curve are collinear with points P_0,P_1 on the second. Similar constructions are possible to ensure curvature (C_2) continuity. A tensor-product Bézier surface may be easily defined extending the definition (2.2):

$$\mathbf{S}(u,v) = \sum_{i=0}^{n}\sum_{j=0}^{m}\mathbf{B}_{i,n}(u)\mathbf{B}_{j,m}(v)\mathbf{P}_{i,j} \tag{2.3}$$

where the points $\mathbf{P}_{i,j}$ now represent a three-dimensional "control net", with the points **P** forming the vertices of this net. The CATIA B-rep solid modeler was originally designed to incorporate a hierarchical scheme for defining and joining Bézier curves in the manner just mentioned. Surfaces were defined by extruding or revolving these curves along Bézier trajectories. In spite of the versatility of this form of parametric curve in solid modeling, it was soon discovered that adjustments had to made in order to precisely capture certain types of curves such as conic sections and other curves involving non-integer exponents. The

adjustment involved replacing the basis functions, *B* with weighted ratio *C* of the basis:

$$\mathbf{C}(t) = \frac{\sum_{i=0}^{n} \mathbf{B}_{i,n}(t)\mathbf{P}_i w_i}{\sum_{j=0}^{n} \mathbf{B}_{j,n}(t) w_j} \tag{2.4}$$

Curves of type (2.4) may also be referred to as Uniform Rational Bezier curves (replacement of the Bezier basis, *B* with B-Splines, *N* below results in Uniform Rational B-Spline curves). Finally, NURBS were introduced to eliminate the cumbersome and unstable practice of Bézier curve joining (unstable because changes made to particular Bézier segment would have a significant impact on other segments). The exact form of a NURBS curve, *C* is shown in equation (2.5), and equations of this form are the basis of most modern B-rep systems (Farin 2002)

$$\begin{aligned}
\mathbf{C}(t) &= \sum_{i=0}^{k} \frac{\mathbf{N}_{i,n} w_i}{\sum_{j=0}^{k} \mathbf{N}_{j,n} w_j} \mathbf{P}_i, \quad t \in [t_0, t_1, ..., t_m] \\
\mathbf{N}_{i,0}(t) &= \begin{cases} 1 \; if \; t_i \le t \le t_{i+1} \; and \; t_i < t_{i+1} \\ 0 \; otherwise \end{cases} \\
\mathbf{N}_{i,p}(t) &= \begin{cases} \frac{t - t_i}{t_{i+p} - t_i} \mathbf{N}_{i,p-1}(t) + \frac{t_{i+p+1} - t}{t_{i+p+1} - t_{i+1}} \mathbf{N}_{i+1,p-1}(t) \end{cases}
\end{aligned} \tag{2.5}$$

The degree, *d* of this curve and the number, *n* of control points *P* are related via the expression:

$$d \equiv m - n - 1 \tag{2.6}$$

where m is the number of non-decreasing "knot points" making up the parameter space, *t*. This parameter space differs from that of Bézier curves in that it need

not be restricted to the ordered set of rational numbers between 0 and 1. It can incorporate any non-degreasing set of rational numbers (repeated knots are permitted and have a special meaning in that they reduce the continuity at that parameter by a factor of 1 times each repeated instance). It can be shown that curves of the form (2.5) are actually identical to joined Bézier segments of continuity C^{p-1} (Farin 2002). But the latter formulation tends to be more robust and stable than if the Bézier curves are constructed individually and joined later. In addition to being able to model more types of geometry than surfaces of type (2.1), parametric equations of type (2.4) and (2.5) have another advantage in that they possess affine invariance. That is to say, transformations applied to these curves are equivalent to the same transformations applied to their control points. This is an extremely useful property for modeling.

Extension of the NURBS curve definition to tensor-product surface is accomplished exactly as in equation (2.3) for a Bézier surface. An example of a NURBS surface with its control net overlaid is shown in Figure 2.4. The black grid represents the parametric knot space.

The study of parametric curves and surfaces is a large field of research in its own right. CAGD is the field of research that covers the limitations, applications, and properties of these surfaces and curves. The field of CAGD was developed specifically to overcome the limitations imposed by curves and surfaces of the type (1). It relies heavily on the principles of differential geometry and topology, but it typically does not concern itself with solid modeling.

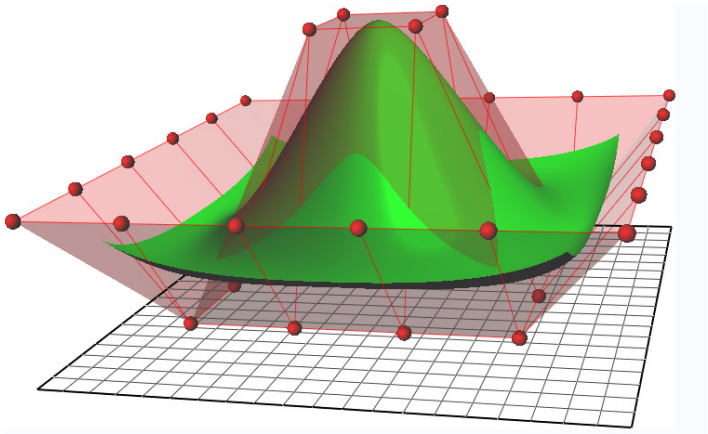


Figure 2.4. A NURBS surface (green) with control net (red)

The tensor-product property of parametric surfaces implies that they possess a rectangular geometry. A technique called surface trimming is used to overcome this limitation. Inconsistency among different trimming algorithms and solid model data structures is responsible for many solid model transfer problems. Surface trimming works by first defining a parametric curve u(t), v(t) in the domain of a parametric surface **s**(u,v). This curve is then mapped onto the surface via **c**(u(t),v(t)). Such a surface-mapped curve is sometimes referred to as a Curve-ON-Surface, or CONS (Woo et al. 1999). If the domain curve has degree *d*, then the CONS has degree (*m*+*n*)*d*, where *m* and *n* are the degree of the tensor product surface, **s.** Trimming curves must always form closed, non-intersecting loops, and a criterion must be used to determine on which side of the trimmed loop the surface should be hidden. The popular, free graphics programming language OpenGL® uses the criterion that surface points *outside* of counterclockwise loops shall be excluded, and surface points *inside* of clockwise loops are included (Woo et al. 1999). The diagram of Figure 2.5 demonstrates this criterion. Figure 2.6

depicts a third order NURBS surface trimmed by an outer loop consisting of four connected piecewise linear segments, and an inner loop consisting of a two piecewise linear segments and a third order curve.

In a typical, modern B-rep solid modeling system, the surface trimming curves are estimated based on surface intersections. For example, the intersection of the two cylinders shown in Figure 2.7 results in the trim curve highlighted in red in Figure 2.8. In feature-based CAD modeling, the resulting geometry shown in Figure 2.8 is the result of "cutting" the bottom cylinder with the top cylinder. With this operation, the trimming curve acts as an inner trim loop on the bottom cylinder, and an outer trim loop for the top cylinder. Operations of this sort

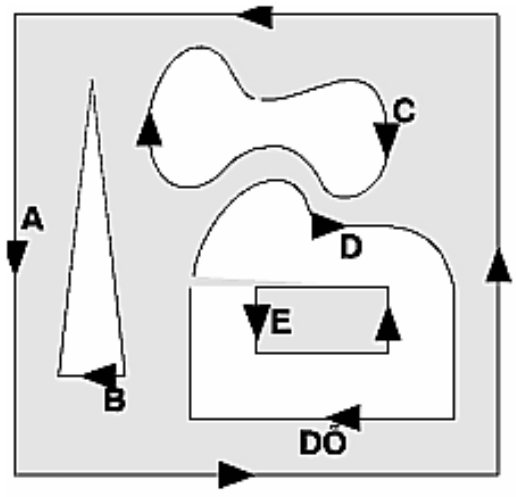


Figure 2.5. Diagram depicting display criteria for trimming curves

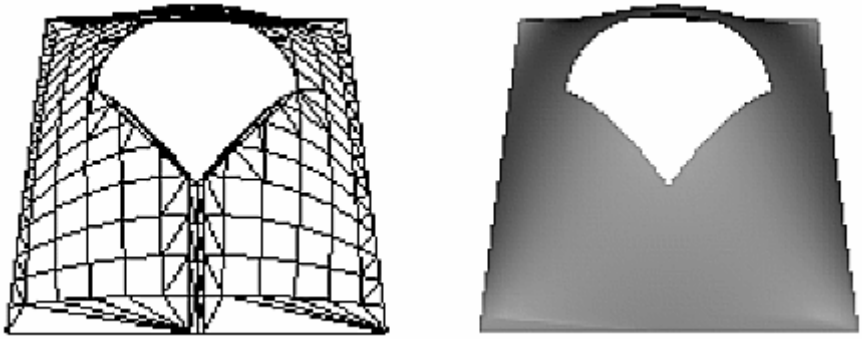


Figure 2.6. Trimmed NURBS surface from the OpenGL Programmer's Guide

necessitate a fairly complex model data structure which must be capable of reflecting a changing model's topology (and related surface trimmings). One such data structure was proposed by Baumgart (Baumgart 1974). It is called the "Winged Edge" data structure and keeps track of face edges and adjacency (this has since

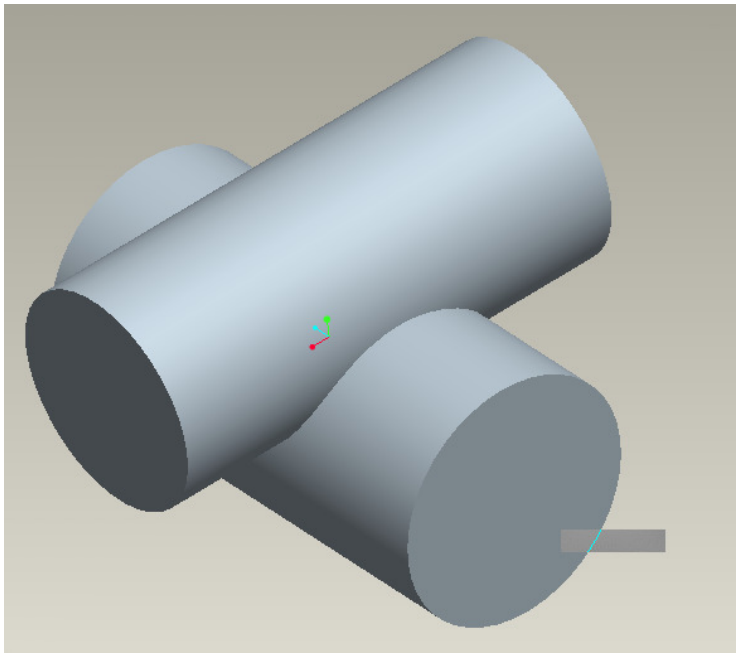


Figure 2.7. Two intersecting cylinders

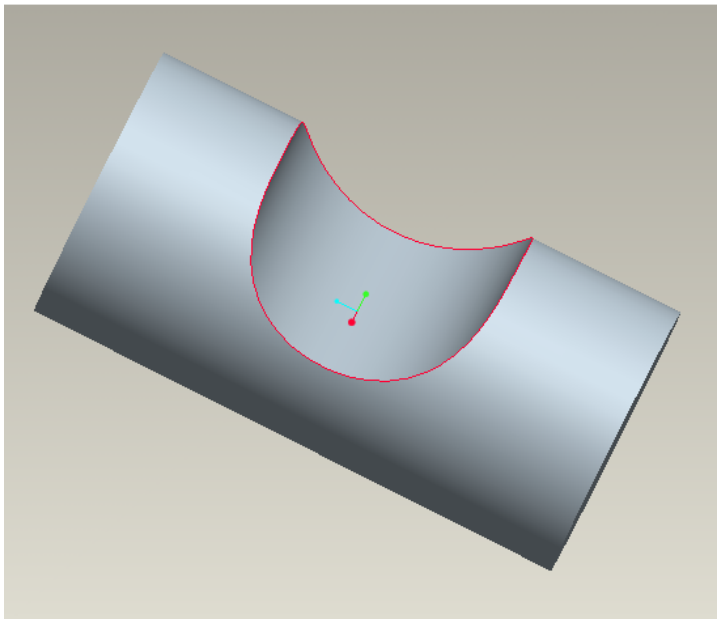


Figure 2.8. Bottom cylinder "cut" by top cylinder

been replaced by the "Half-Edge" dataset. Increasingly, commercial CAD engines, such as ACIS include the full boundary topology dataset).

Usually, additional data, such as vertex number and location must be stored. With such a scheme, it is clear that a model's feature history (The part in Figure 2.8, for example was created with an extrusion followed by a cut) must be maintained, and it must be unambiguously connected to the model's geometric/topological data-structure. An ongoing problem for B-rep solid modeling systems is sometimes referred to as the "Persistent Naming Problem" (Marcheix and Pierra n.d.). As features are added to a solid model, some previously defined topological entities must be identified after the model is re-evaluated. This is a nontrivial problem, which is usually handled via topological

ID mapping, but other methods have been tried (Marcheix and Pierra n.d.). To the CAD user, this problem usually manifests itself by the consequences of deleting or radically modifying a previously defined feature. The extent of the repair work that must be performed after such an operation is often a measure of how robust the CAD system's topological ID system is (the program must evaluate just how much of the model's feature history depend on the change).

2.2 Assembly design

As explained in Chapter 1, the current research is being conducted within a larger context involving the integration of CAD modeling, finite element analysis, and assembly design principles. Background to the latter topic is presented next, and is largely excerpted with permission of J. Shah (Shah 2009).

There are many types of assemblies: static, rotational, articulated and mixed. The specific design process varies with assembly type and function. Major design tasks typically include shape/size design of parts; interfacing of components; layout, packaging; kinematic, dynamic, structural analyses; motion simulation and interference detection. Auxiliary tasks may include manufacturability (DfM) and assemblability (DfA) analysis. In addition to nominal design of an assembly, there is also tolerance design which determines the GD&T scheme and allowable manufacturing variations to ensure proper functioning and assemblability. The project proposed here addresses only nominal design of assemblies.

Contemporary CAD systems support bottom-up assembly design where parts must be created before assemblies. This is the opposite process to human

designers' approach for most types of assemblies. Nevertheless, CAD systems are excellent for detailed part geometry creation, layout design, part positioning, automatically propagating parametric modifications and interference detection. Simulation applications, such as FEA, rigid body dynamics, kinematics, are available either as separate CAE modules or third party packages. CAE accepts geometry data from CAD and then uses it to perform the requisite analysis. This separation prevents the designers from incorporating simulation within their design process and is compounded by incompatibility between applications and specialized expertise needed for FEA. Figure 2.9 shows a mechanical assembly in assembled and disassembled forms. An assembly can be viewed as a hierarchy of sub-assemblies and parts, as shown in Figure 2.10. The grouping of parts into

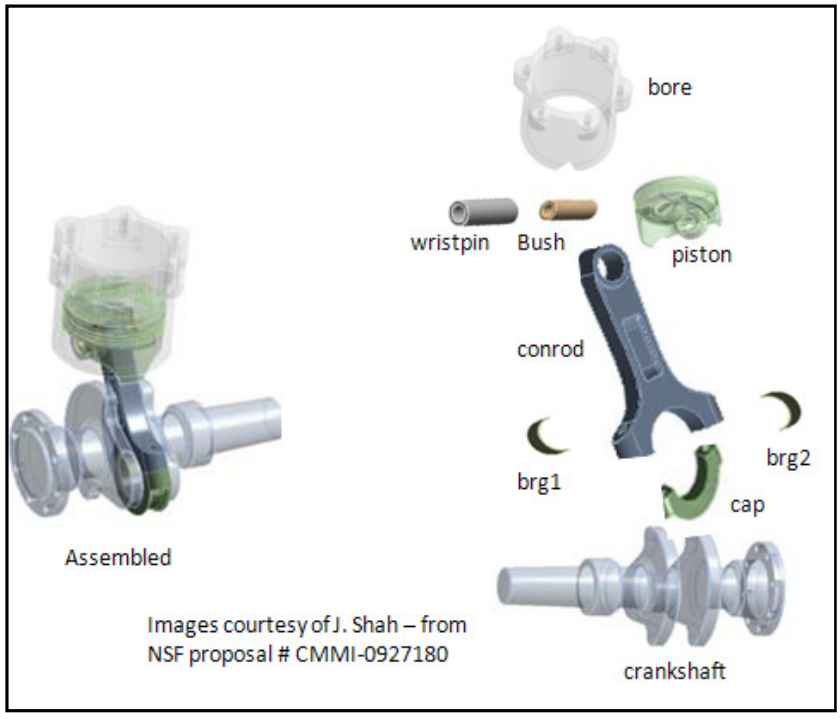


Figure 2.9. Unexploded (left) and exploded view of engine assembly

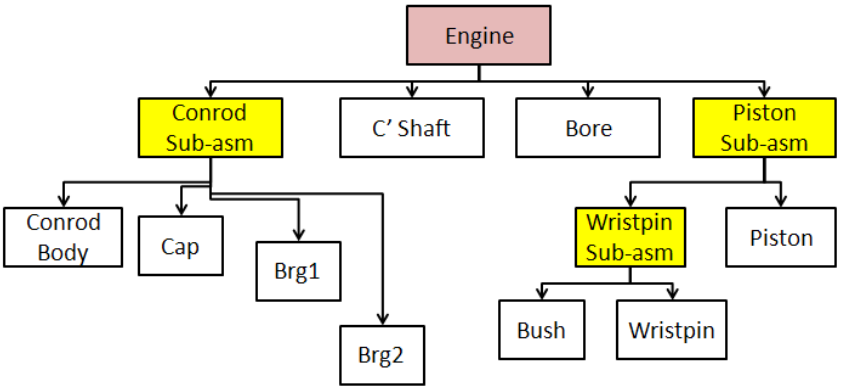


Figure 2.10. Engine assembly hierarchy

sub-assemblies is not unique; it may represent assembly sequence, modularity, common degrees of freedom relative to other parts or subassemblies, or something else.

The items enclosed in the boxes represent 'assembly units', the lines represent 'part-of' relations. The hierarchical model, however, does not have any information about mating features. A separate constraint graph is used to code geometric relations between faces (against, parallel, etc), axes (collinear, coincident) and size parameters (*hole_dia* = *pin_dia*). Geometric relations are used for locating parts in assemblies. Lee et al (Lee and Andrews 1985; Lee and Gossard 1985) and Rocheleau (Rocheleau and Lee 1987) developed transformation matrices for part positioning in assemblies which form the basis of assembly modeling in CAD. Size relations are used for propagating dimensional changes from one part to another to maintain compatibility.

Theoretically, it should be possible to extract kinematic DoFs from the part models and mating constraints but CAD systems currently are not able to do that. Instead, the user must interactively define “joints” and pairs of local coordinate systems (LCS)

associated with part geometry (Figure 2.11), resulting in the Degree of Freedom (DoF) representation shown in Figure 2.12. This is typically done in a dynamic analysis module or external application.

Many types of simulation, such as structural, dynamic and thermal, require the geometry to be meshed. Major steps for such analyses are: geometry simplification, meshing, specification of boundary conditions, and solving and reviewing results. By far the most time is consumed in getting good meshes, even though tremendous advances have been made in automatic meshers in the past 25 years (see section 2.4.3). It is not uncommon to see the simulation done part by part, rather than for the assembly as a

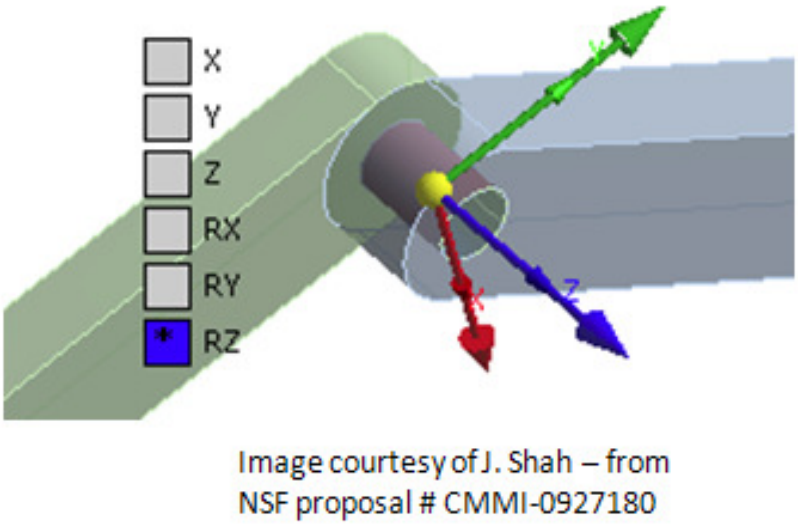


Figure 2.11. Joint definition: reference and moving LCS

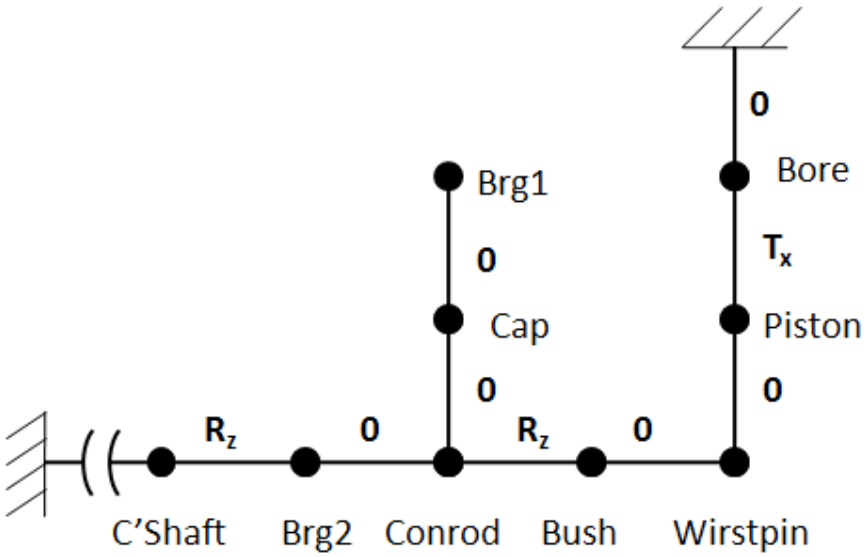


Figure 2.12. DOF diagram of engine assembly

whole. One reason is that multiple parts will contain a larger number of nodes/elements increasing the computational complexity. Another is that including contacting regions in models makes the problem non-linear, requiring iterative solvers. Also, contact modeling is based on ad-hoc methods that vary from system to system. The alternative, modeling parts individually, has its own set of disadvantages, which include the difficulty of accurately estimating boundary conditions at mating regions and inability to optimize at the system level.

Hierarchical models (Figure 2.10) are widely used in commercial CAD systems with the addition of geometric relations between pairs of entities on the part being positioned and the assembly. These relations are not necessarily mating conditions; in fact, relative positioning can be used even with parts not in contact. In the assembly tree, interior nodes are sub-assemblies and leaf nodes are individual parts. The leaf nodes are essentially pointers to part CAD models and a transformation matrix. Multiple instances

of the same part point to the same CAD model but different transformation matrices. Another type of assembly representation is a connectivity or liaison graph in which the nodes represent parts and the arcs represent contact (Bourjault 1984). This simplistic view is incapable of representing multiple contact pairs between the same two parts. Kinematic and dynamic analysis packages do not model the detailed geometry; instead, each part is abstracted as lines connected at joints defined by translational and rotational DoFs based on reference and moving LCS (Figure 2.11). Structural models of assemblies require different types of information; specifically, contact regions, gap, sliding behavior and friction condition. Neither kinematic nor structural models can be derived from hierarchical or connectivity models. Therefore, the user must define them interactively with the help of CAD geometry. The next sections review some new approaches for assembly modeling that not only obviate such manual work but, more importantly, allow assembly design compatible with designer thinking. Proposed advances include top-down design, design with knowledge structures called Generic Functional Interfaces (GFI), automatic completion, recognition and mapping of GFI to support design and simulation tasks.

2.2.1 Generic Functional Interfaces

A part feature has been defined as a stereotypical shape with certain topological and geometric properties (Shah and Mantyla 1995). Similarly, an assembly feature is defined as a stereotypical association between two part features that are on different parts. Whitehead posits that parts get position from location features and keep it with effector features (Whitehead 1954). Although a great deal of literature exists on part feature definition and recognition, the same cannot be said about assembly features. Brunetti (Brunetti and Golob 2000)

developed an assembly model that supports conceptual development according to the Pahl & Beitz procedure (Pahl and Beitz 1997). Gupta et. al. (Gupta, C.J.J. Paredis, and Sinha 2001) created models for assembly sequence generation. Noort et. al. (Noort, Hoek, and Bronsvoort 2002) used assembly features to provide multiple views of a product. Liang (Liang and C. Paredis 2003) used a port model to define interaction between a component and its environment or interfaces between components. DeFazio (De Fazio et al. 1991) proposed a feature-based assembly system to specify the mating relations between components and relative extraction directions for use in assembly planning. Sodhi (Sodhi and J.U. Turner 1991) used assembly features for specification of relations between components at a high abstraction level. Boredegoni (Borgedoni and Cugini 1997) classified assembly features as attachment, pass-through and connect types.

The above review, though not comprehensive, is representative of much assembly modeling literature. Two themes stand out: most of the work is directed at assembly sequence planning when the design is already fixed and not evolving; the various assembly feature definitions used require manual input into the ad-hoc attribute slots. The latter problem creates a disconnect between kinematic/structural simulation and design. The ultimate goal of the current research (of which this investigation is one phase) is to go beyond assembly features by capturing functional definition of generalized part-part interfaces in the form of knowledge structures termed Generic Functional Interfaces (GFI).

Each part may have many GFIs where it interacts with other parts; they combine to produce desired functions. GFIs, like assembly features, encode

mutual constraints on mating features' shape, dimensions, position, and orientation, but more importantly, GFIs are carriers of functional information. Arizona State University's DAL has previously developed a canonical definition of assembly features based on the NIST Open Assembly Model (OAM) (Rachuri 2005; Rachuri et al. 2003) and screw theory (Mahbub 2008). Screw theory first developed by Sir Robert Ball (Ball 1900), and was applied to assemblies by Whitney (Whitney 2004). A system of screws is a way of representing the geometry for motions that a rigid body can undergo or of representing the forces and moments exerted on it.

Screws representing motion are called twists, while screws representing forces are called wrenches. A twist or wrench matrix has 6 columns and 1 to 6 rows, one for each degree of freedom (DoF) being described. Twists and wrenches can be used to describe a wide variety of part-to-part constraints. *N* DoFs are represented by *N* rows, 6 columns each. A twist $T=\{\omega_x, \omega_y, \omega_z, v_x v_y v_z\}$ is a screw that describes the instantaneous motion of a rigid body. The first triplet represents the angular velocity of the body with respect to a global reference frame. The second triplet represents the velocity of a point on the body; at any given instant the linear velocity due to the rotational motion is zero on that point. Similarly, a wrench $W=\{fx, fy, fz, mx\ my\ mz\}$ is a screw that describes the resultant load directions possible.

The first triplet describes the resultant force in a global reference frame; the second triplet represents moments. Screw theory permits one to represent in a precise mathematical way the interactions between two surfaces or bodies in

contact. It can be used to determine the degrees of freedom and the amount of under or over-constraint, and directions in which loads can be transmitted. Screw matrices are key in the definition of GFIs; screws can be pre-defined (Whitney 2004) or extracted by mathematical procedures explained below. Future studies at DAL will investigate how the functional capability of GFIs can be fully exploited in assembly design and simulation. Figure 2.13 shows a library of GFIs.

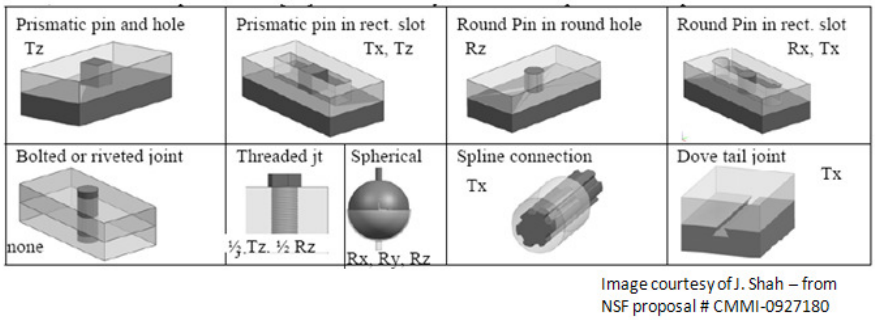


Figure 2.13 Library of common GFI's with their respective DOF's

Although this is not a comprehensive list, it includes the commonly encountered ones. Most GFI's involve mating of multiple pairs of surfaces. For example, the prismatic pin in open rectangular slot has a 3 pairs of plane-plane contact; if the slot was blind, an additional pair would come into play depending on the pin length and slot depth. For the purpose of twist and wrench matrices one does not distinguish between contact with or without clearance. It is also observed that these common GFIs typically involve simple geometric surfaces (planes, cylinders, spheres). Each mating surface pair can be represented by screws. The

net DoFs and load transmission directions can be found from the intersection of all screws S_i representing the mating pairs for a GFI. The same formula can be extended to multiple GFIs between parts:

$$\cap_i (S_i) = Rcp\left\{\cup_{i=1}^{n} \left(Rcp(S_i)\right)\right\} \tag{2.7}$$

where *Rcp* is Reciprocal; twists and wrenches are reciprocals of each other, that is $T \bullet W = 0$. The operations *Rcp,* $\cup$, $\cap$ are mathematically well defined (Davidson and Hunt 2005) and computational procedures exist (Whitney 2004). Figure 2.14 shows the type of information that can be clustered around GFI definition. This goes far beyond what current CAD supports and more comprehensive than assembly features; it is not just stand-alone geometric relations associated with entity pairs when adding part instances to assemblies. It *captures the stereotypical properties of different types of interfaces* that may be encountered between parts in assemblies. It remains to be investigated whether this set of attributes is sufficient for design and simulation applications, how to extend this representation to more complex GFIs and how to support user defined GFIs.

2.2.2 Assembly design scenarios

With GFIs, three different strategies for assembly design will be possible:

- bottom-up design with GFIs
- bottom-up design without GFIs
- top down design with mapping to GFIs

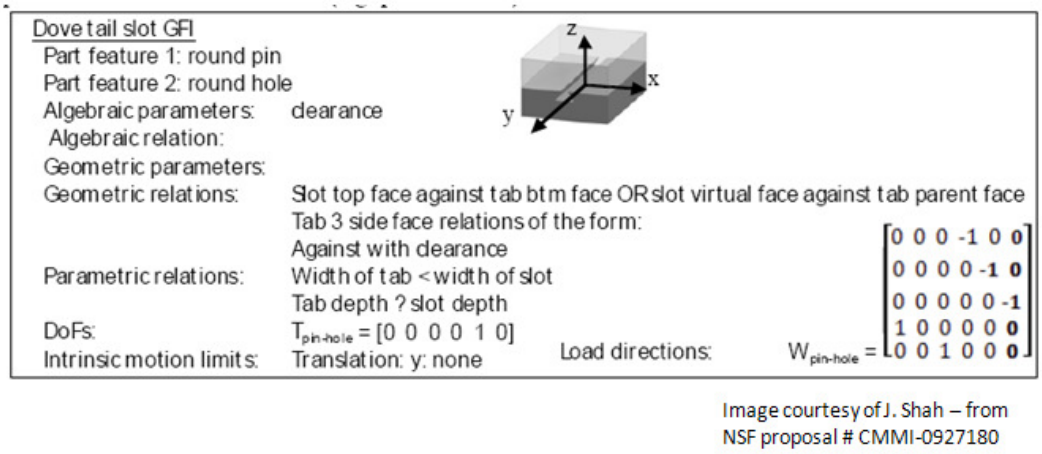


Figure 2.14. GFI definition

In bottom up design part CAD models are created first, followed by assembly design. In contemporary CAD systems generic geometric constraints are used for part positioning within assemblies. *There are no pre-defined or user defined GFIs stored in libraries*, such as those shown in Figure 2.14. One possible scenario is to define assemblies as they are done today without GFIs, and then to recognize them automatically from the assembly CAD model. Another possibility is for the designer to use GFIs interactively when defining the assembly. In either case, a library of GFIs is required and a neutral (implementation independent) language is needed.

The proposed GFI definition is not ad-hoc but physically based (screw theory) which is essential for encoding functions and compatible with kinematic applications. The GFI template should contain slots for: part features that constitute the GFI; assembly parameters (geometric - defined by two geometric

entities directly; algebraic - defined by other parameters, e.g. clearance); and constraints (geometric or algebraic). Kinematic and structural relations can be derived using screw theory; for predefined features they can be pre-coded. Motion limits will need to be derived from collision detection algorithms by simulating part motion in each DoF direction. Note that part feature definitions already contain face properties and feature dimensions (e.g. pin diameter). For bottom up design with GFIs the following procedure is proposed. The designer will instance a GFI and then graphically select one face on each part belonging to part features to be mated. Even if the part features contain multiple faces, and they usually do, only one face needs to be picked. The faces picked do not have to be mating faces. This is because the GFI definition is rich enough to automatically complete the part feature by recognizing adjacent faces, using GFI Completion Algorithm. Also, the GFI definition can be used to determine if the selected faces are consistent with its definition.

For bottom up assembly design without GFIs, as it is done now, it has been proposed to automatically recognize GFIs. In well-established part feature recognition algorithms, attributed face adjacency graphs are used to facilitate topological level recognition. To our knowledge, GFI recognition, or even assembly feature recognition, has never been done before. A new data structure will need to be designed to facilitate GFI recognition and to reduce search space.

One possible approach would be combine AFAGs of individual parts linking with arcs candidate mating faces, which can be found using surface proximity algorithm. For most GFIs depicted in Figure 2.13, the mating surfaces

are of the same type (plane-plane, cylinder-cylinder, spherical-spherical) and these can be searched and recognized with the condition that they be within proximity specification, belonging to different parts and having opposite normals (material direction), nominally coaxial cylinders, concentric spheres. However, there are exceptions, as in the case of the round pin and slot – the mating surfaces are of dissimilar geometry. Therefore, more sophisticated *GFI recognition algorithms* will be investigated in the future to handle any set of user-defined geometric conditions. To support user defined features, both the Recognition and the Completion algorithms must be data-driven. An interesting future possibility is discovery of *new* GFIs from archived assemblies – a process akin to data mining.

2.2.3 Top-down assembly design

There may be different starting points for top-down design depending on assembly type. For articulated assemblies, the design process begins with kinematic synthesis where the linkage configuration, joint types and relative spatial location/orientations of joints is determined. This is done in either a mechanism synthesis package or on paper. The kinematic model shown in Figure 2.15 contains links, identified by numbers, and joints, identified by letters. Only the parts relevant to kinematics are represented and the only part geometry needed is the location of joints. Kinematic diagrams facilitate motion simulation in terms of input/output displacements, velocities and accelerations.

The DAL proposes to investigate how the kinematic model can be used to drive embodiment design of the assembly. The output of kinematic design

consists of generic joint types and LCS; key point positions for joint location; link lengths and geometric relations (e.g. link 2 is parallel to 8; all revolute joint axes are parallel). We want to import this output directly into a CAD assembly and develop a mechanism by means of which the designer can conduct embodiment design. An algorithm will be developed to automatically create an initial skeleton hierarchical assembly model using the links, joint definitions and matching DoFs as the basis for the decomposition. Next, the joints will be mapped to corresponding GFIs, but these will not contain any parameter values, only geometry types and mating constraints. Figure 2.15 depicts this scenario. The designer will then add new features to the skeleton geometry of each part and add secondary parts while geometric and parametric constraints, inherited from GFIs, are maintained as the part geometry evolves. Note that this scenario is the opposite of the process supported in CAD today where the detailed part models are used for assembly design and then used to define the kinematic simulation model. When secondary parts, such as seals, bearings, fasteners, bushing, etc. are added by the user, the process can also be supported by the feature based assembly design described earlier. One research challenge is how to apply/associate constraints defined a priori with geometry that evolves later.

For top-down design of non-articulated assemblies, such as common rotational assemblies (pumps, turbines, gearboxes,..), kinematic synthesis is not needed. Instead, the designer would start with a skeleton assembly hierarchy to

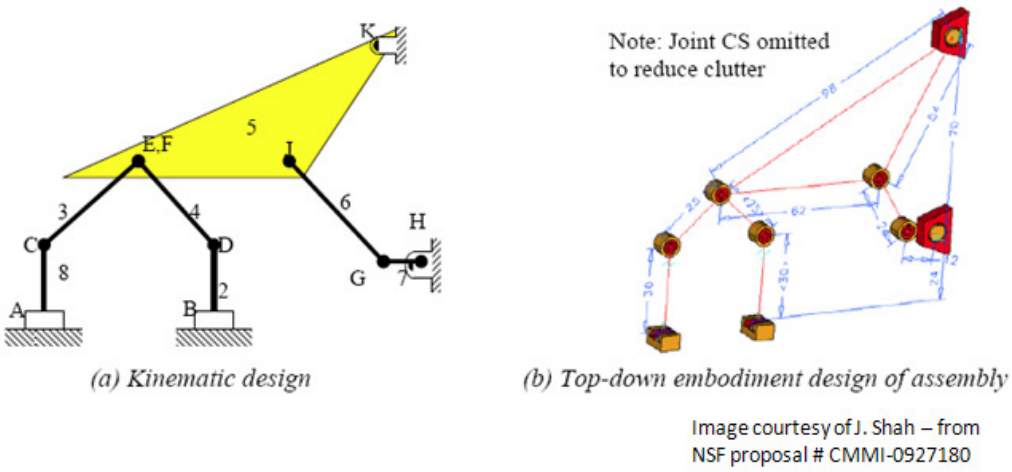


Figure 2.15. Top-down embodiment design of assembly

define the basic product structure. The assembly constraints, features, GFIs and geometric relations would be associated with the parts in the hierarchy before proceeding to detailed geometry definitions. The rest of the process will be similar to that defined above.

2.3 Structural analysis

Structural analysis comprises the set of physical laws and mathematics required to study and predict the behavior of structures. The subjects of structural analysis are engineering artifacts whose integrity is judged largely based upon their ability to withstand loads; they commonly include buildings, bridges, aircraft, and ships. Structural analysis incorporates the fields of mechanics and dynamics as well as the many failure theories. From a theoretical perspective the primary goal of structural analysis is the computation of deformations, internal forces, and stresses. In practice, structural analysis can be viewed more abstractly

as a method to drive the engineering design process or prove the soundness of a design without a dependence on directly testing it (Anonymous n.d.).

In spite of the generality and insight into structural behavior provided by the Theory of Elasticity, with few exceptions, researchers in the 19th century were unable to derive exact solutions for generic engineering structures which were not "thin". Most of the exact solutions provided by the theory were restricted to idealized point or line loads acting on infinite or semi-infinite domains. An example of this is the 2-body contact solution provided by Hertz (Hertz 1896). Since exact solutions were lacking for most cases, researchers focused instead on approximate solutions.

The 20th century saw work on numerical techniques for solving the governing equations of structures provided by the theories of Mechanics of Materials, and later the Theory of Elasticity and Continuum Mechanics. All of these involved partitioning the domain in some manner into mathematically manageable pieces. The earliest such methods involved some variant of what would today be called a Finite Difference (FD) technique. In its simplest form, this technique requires that the domain in question be subdivided by a uniform lattice of points, or nodes. In two dimensions, the connectivity between these nodes is such that each node at position (x_i,y_j) is connected to four neighbors, (x_{i-1},y_j),(x_{i+1},y_j),(x_i,y_{j+1}), and (x_i,y_{j-1}) where the subscripts i and j denote a well-ordered integer progression of x and y-coordinates as shown in Figure 2.16.

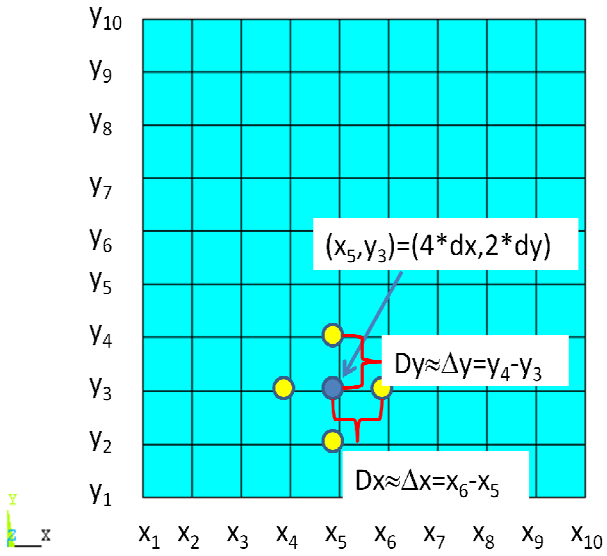


Figure 2.16. Structured grid over a rectangular domain

Such a scheme superimposes a rectangular grid over the domain when the coordinates are Cartesian and is sometimes called a "structured grid" when there are as many rows of points as columns. The use of such a grid provides a straightforward means of replacing the governing differential equation over the domain by a difference equation. For example, the differential expression dy/dx is now replaced with $(y_i-y_{i-1})/(x_i-xi_{-1})$ or $(y_{i+1}-y_i)/(x_{i+1}-x_i)$. Such an approximate solution, though simple to implement, could be very laborious to solve before the advent of computing machines. One limitation is that a minimum of $(O+1)^n$ nodes are required simply to obtain the solution of a differential equation of degree O over a finite domain of dimension n. Domain topologies which are not simply connected, as well as discontinuous loads and boundary conditions, often require many more nodes than this because the FD approximation provides

solutions only at nodal points. The resulting domain solution is piecewise linear and thus requires many nodes to resolve strong gradients. Early implementations of this technique were solved by hand iteratively without the use of matrix methods (Felippa 2001). Another deficiency of the technique is that it produces erroneous results near domain discontinuities (it only produces good results within smooth, homogeneous continua).

The dawn of the 20th century saw researchers searching for more efficient approximate solutions which could overcome these difficulties. Instead of focusing on discretizing the governing equations over a continuum (as had been the common practice until then), these researchers focused instead on joining the solutions of simpler structural elements together via equations of compatibility. It appears that this approach may be followed along two very different governing principles: The first involves assembly of matrix equations F=Kx by adding the stiffness contributions of each element. Here F is the external load vector (known), K is the assemble stiffness matrix (known), and x is solved for by standard matrix procedures. This is known as the Direct Stiffness Method (DSM. In early works, often referred to as the Matrix Displacement Method, or DM), and is the one used in modern FEA systems. The other principle, however, is the one which first caught traction. It involves several steps. The first being the assembly of system Q=fq, where Q is the global system deformation vector. f is the total system flexibility matrix 1/k, and q are total system internal force vector. For systems which are statically indeterminate (i.e. most systems), this results in an

over constrained system. To alleviate this, degrees of freedom are chosen to augment the equations resulting in a positive definite solvable system.

The final step involves applying compatibility equations via use of the unit dummy force method in order to retrieve displacements (from these, strains may be calculated). This more complicated approach is referred to as the Flexibility Method (FM. In early works, often referred to as the Matrix Force Method – in which the same acronym was used. For an explanation of this method, the reader is referred to one of the few sources on this topic still in publication – Przemieniecki (Przemieniecki 1968)).

For other developments in this early period, a nice summary of the historical development of structural analysis is given by C.A. Felippa (Felippa 2001). For a very readable history of the development of the theory of Mechanics of Materials, see Timoshenko (Timoshenko 1983). For a historical account of developments specific to finite element analysis, see (Zienkiewicz 1995). Apparently the first published papers dealing with the mechanical analysis of a structure by dividing it into several topologically simple pieces, and assigning degrees of freedom to each piece, and solving these as a matrix system were due to R.A. Frazer and W. J. Duncan in 1934 and 1935 (Duncan and Collar 1934; Duncan and Collar 1935). These papers marked in important milestone in numerical analysis in that they are the first true matrix methods, in which matrix stiffness terms (or "influence coefficients") are derived from known analytical solutions. Such methods provide high accuracy and less labor in solving the resulting algebraic systems (solutions equivalent to FD schemes could be found

with much smaller equation systems). This approach, however crucial to the subsequent development of finite element analysis seems to be the source of the bias in that field toward the construction of a "mesh", which must interpolate the geometric domain. This bias can be seen in early statements by Courant (Courant 1943): "…The vagueness as to the accuracy of the approximation obtained is only one of the objections to the Rayleigh-Ritz method that may be raised. More annoying is that a suitable selection of the coordinate functions is often very difficult and that laborious computations are sometimes necessary. For these reasons, alternative methods must be studied." It doesn't seem to have occurred to researchers in Western Europe, the United Kingdom, and United States that the Rayleigh-Ritz method may be saved instead by a suitable modification of the basis functions to accommodate boundary conditions. This is in contrast to the former Soviet Union, in which Kantorovich et al championed a method by which basis functions are multiplied by weighting functions which equal zero at domain boundaries (Kantorovich and Krylov 1958). Further, Kantorovich showed how to construct such functions for arbitrary domains defined by implicit functions. Although such an arrangement is only capable of solving equations with homogeneous boundary conditions, it was clearly an important advance, which would ultimately lead to the method of Solution Structures (Rvachev 1975), in which any boundary constraint may be exactly represented, regardless of whether the basis interpolates the geometric domain or not.

The finite element method was formally introduced (by general concensus) in a paper by Turner et al (M.J. Turner et al. 1956). A important

discovery, called the Direct Stiffness Method, was added by Turner in (M.J. Turner 1959). This is a method of assembling element coefficient entries into a global system, regardless of element type or dimension, as long as the degrees of freedom are compatible. The method may be stated simply as:

$$\mathbf{F}_e = \sum_e (\mathbf{k}_e \mathbf{x}_e + \mathbf{b}_e) \tag{2.8}$$

where $\mathbf{k}_e$ are the element stiffness matrices (representing element stiffness with regard to all its degrees of freedom) in the global reference system, $\mathbf{x}_e$ are the element displacements in the global reference, and $\mathbf{b}_e$ are element external forces (body forces) expressed in the global reference. The equations are assembled once and then solved. By the 1970's, the finite element method seemed to reach maturity, producing advances such as general procedures for constructing basis functions and constructing algebraic systems associated with the weak form of the governing differential equations. Significant contributions were made by Argyris, Turner, Clough, Argyris, Babuska, and others. Without citing references, some of these are discussed in the next section. Readers who are interested in attributions to some of these methods are once again referred to (Zienkiewicz 1995) .

2.4 The Finite Element Method

Felippa (Felippa 2001) points out that Matrix Structural Analysis and the Finite Element Method share some common points of reference in their historical development. These points may be summarized as Matrix Formulations, Element Formulations, and Solution Procedures. Both MSA and FEM are characterized by the fact that they discretize the solution of a partial differential equation, as

opposed to earlier techniques which discretized the differential equation itself. As such, methods must be chosen to assemble these discrete element solutions. The MSA saw various approaches to doing this in its history, with the main competing techniques being the Direct Stiffness Method, and the Flexibility Method.

The modern FEM relies almost exclusively on the Direct Stiffness Method of matrix assembly. The techniques for solving the assembled system of equations in MSA had direct application to FEM (indeed FEM may be viewed as simply another method of MSA). The emergence of the digital computer prompted research into better matrix system solution algorithms, such as iterative methods, as well as the efficient storage and manipulation of sparse systems. Today, these efforts continue, focusing largely on efficient parallel and distributed computing. Since the key distinguishing features of the FEM are its continuum-based element formulations, and its near-exclusive use of DSM in matrix assembly, we discuss these a bit further.

2.4.1 The Galerkin formulation

The element formulations in modern FEM are usually derived by the use of a variational formulation. This may be stated as follows: The solution of a governing differential equation (D)

$$\mathbf{D}^{(o)}(u) = b(\Omega) + f(\Gamma) \tag{2.9}$$

may be approximated by solving an equivalent (weak) form:

$$\int_{\Omega} \left(\mathbf{D}^{(o)}(u) v(\Omega) - b(\Omega) v(\Omega) \right) d\Omega - \int_{\Gamma} f(\Gamma) v(\Omega) d\Gamma = 0 \tag{2.10}$$

where **D** is some differential operator of degree o, u is the primary variable of interest (displacement, in the case of structural analysis), *b* is a continuously distributed load through the domain, Ω, *f* is a load on the free boundary, Γ (as opposed to the portion of the boundary Γ_d, where essential boundary conditions are applied), and v is a "trial" function which, in general, belongs to a space of such functions with following properties:

$$V = \left\{ \begin{matrix} v : v \text{ is continuous on } \Omega,\ v' \text{ is piecewise continuous} \\ \text{and bounded on } \Omega, \text{ and } v(\Gamma_d) = \phi_0 \end{matrix} \right\} \qquad (2.11)$$

The last part of (2.10) is critical to the solution. It states that *the trial function must take on values of φ_0 only on the portion of the boundary, Γ_d where boundary conditions are applied*. This is an important restriction which is usually enforced by setting the trial function coefficients (i.e. the nodal degrees of freedom) to the value φ_0 after matrix assembly. This is possible because of property (2.15). However, in mesh-free methods where the shape functions do not interpolate the nodal degrees of freedom directly (and in some cases, there are no "nodal degrees of freedom"), care must be taken to modify them in some way to satisfy (2.11). In particular, this is the case with the method of Shapiro/Höllig. Integrating equation (2.10) by parts, and making use of the restriction (2.11) on v, results in an equation of the form:

$$\begin{aligned} &\int_\Omega \nabla^{(p)}(u)\nabla^{(p)}(v)d\Omega = \int_\Omega \lambda(v(\Omega))d\Omega + \int_\Omega b(\Omega)v(\Omega)d\Omega \\ &+ \int_\Gamma f(\Gamma)v(\Gamma)d\Gamma \end{aligned} \qquad (2.12)$$

The order, p of the del operator ∇ is given by $p=o/2$. λ is a linear residual term, which may be "left over" from the construction of the bilinear form on the left-hand-side, depending on the exact form of the governing equation. If the governing equation D admits a variational form, then the linear form involving λ is zero, which will be the assumption in what follows. This derivation is a generalization of one found in (C. Johnson 1992), and is usually referred to in the literature as the Galerkin formulation. The method is characterized by the invocation of a trial or weighting function, as in equation (2.10) without appeal to variational principles (indeed, the method is more general than the variational approach). The solution represents a minimization of the difference between the left-hand side and right-hand side (called the "residual"). Further characteristics of equation (2.12) may be found in a discussion of the variational calculus, such as (Heinbockel 2006). The solution, u is assumed to have the same form as v ($u \in V$) and differ from it by an unknown factor c, such that equation (2.12) becomes

$$\int_{\Omega} c\nabla^{(p)}(v)\nabla^{(p)}(v)d\Omega = \int_{\Omega} b(\Omega)v(\Omega)d\Omega + \int_{\Gamma} f(\Gamma)v(\Gamma)d\Gamma \qquad (2.13)$$

The unknown c may now be solved directly, as it is the only unkown. The next step in the FEM is to discretize equation (2.13). To make this equation useful for domains of arbitrary (not necessarily continuous) topologies, the trial function space V is replaced with a finite-dimensional subspace, $V_h \subset V$ made up of orthogonal basis functions, N that interpolate solution values (displacement, for

example) at discrete points $\mathbf{x_i}$. If we call the discrete solution values, $\varphi_i = v(\mathbf{x_i})$, the discretized version of trial function v may be written:

$$v(x) = \sum_{i=1}^{M} N_i(\mathbf{x})\varphi_i$$
$$or \quad \mathbf{v}(x) = \mathbf{N}^T \Phi \tag{2.14}$$

where M is the total number of discrete points (times the number of degrees of freedom at those points) chosen to subdivide the domain. The functions, *N* belong to an orthogonal function space (Lagrange polynomials are a popular choice) which interpolate the discrete solution values ϕ_i. An obvious requirement is that v must be C^0 continuous on Ω. Another useful property of the shape functions *N* is their so-called "Kronecker-Delta" property (which, as will be seen, mesh-free methods do not possess):

$$N_j(\mathbf{x}_i) = \delta_{ji} = \begin{cases} 1 \text{ if } j = i \\ 0 \text{ if } j \neq i \end{cases} \tag{2.15}$$

The discrete points, **x** in Ω, called "nodes", are each connected to non-overlapping finite elements which subdivide, or partition the domain and form a subspace $T_h = \{T_1, \ldots T_m\}$ of Ω such that

$$\Omega = \bigcup_{i=1}^{m} T_i \tag{2.16}$$

This partitioning allows one to re-write equation (2.13) in a discretized form by substituting equation (2.14) into equation (2.13) for each element:

$$\int_{T_i} \nabla^{(p)}(\mathbf{N}^T\Phi)^T \nabla^{(\mathbf{p})}(\mathbf{N}^T\bar{\Phi}) dT_i = \int_{T_i} b(\Omega)\mathbf{N}^T\bar{\Phi} dT_i$$
$$+ \int_{\Gamma_i} f(\Gamma)\mathbf{N}^T\bar{\Phi} d\Gamma_i \tag{2.17 a}$$

Note that if the trial function (2.11) is modified to include boundary conditions, an additional term is introduced to equation (2.16 a). For example, if a trial function:

$$v' = \omega v + \phi$$

is used, where the ω is a weighting function which forces v to zero on the boundary, and φ captures non-homogeneous effects within the domain, then substituting this equation into equation (2.13) and following the same steps as before results in:

$$\begin{aligned}\int_{T_i} \nabla^{(\mathbf{p})}(\omega \mathbf{N}^T \Phi)^T \nabla^{(\mathbf{p})}(\omega \mathbf{N}^T \bar{\Phi}) dT_i &= -\int_{T_i} \nabla^{(\mathbf{p})}(\omega \mathbf{N}^T \bar{\Phi})^T \nabla^{(\mathbf{p})}(\phi) dT_i \\ &+ \int_{T_i} b(\Omega) \omega \mathbf{N}^T \bar{\Phi} dT_i + \int_{\Gamma_i} f(\Gamma) \omega \mathbf{N}^T \bar{\Phi} d\Gamma_i\end{aligned} \tag{2.17 b}$$

Using the definition:

$$\nabla^{(p)}(\mathbf{N}) = \mathbf{A} \tag{2.18}$$

and noting that $\bar{\Phi}$ cancels from both sides of equation (2.17 a), this equation may be written:

$$\int_{T_i} \mathbf{A}\mathbf{A}^T \Phi dT_i = \int_{T_i} b\mathbf{N} dT_i + \int_{\Gamma_i} f\mathbf{N} d\Gamma_i \tag{2.19}$$

Equation (2.19) now represents a linear set of equations for each element T_i. Furthermore, the matrix on the left-hand side is symmetric, positive definite for static structures (ignoring any gyroscopic effects or fluid interfaces), thus guaranteeing a unique solution. The full system of equations over Ω:

$$\mathbf{K}\mathbf{x} = \mathbf{r}_b + \mathbf{r}_f \tag{2.20}$$

is assembled using the Direct Stiffness Method ($\mathbf{r}_b$ and $\mathbf{r}_f$ are the body and surface load residuals, respectively):

$$K_{ij} = \sum_{i=1}^{m} \int_{T_i} \mathbf{A}\mathbf{A}^T \Phi dT_i \tag{2.21}$$

Similarly,

$$\mathbf{r}_b + \mathbf{r}_f = \sum_{i=1}^{m} \left(\int_{T_i} b\mathbf{N} dT_i + \int_{\Gamma_i} f\mathbf{N} d\Gamma_i \right) \tag{2.22}$$

2.4.2 Shape (basis) functions

Many function-interpolating solution function spaces have been used in FEM throughout its history – from polynomials, to splines, to trigonometric functions. However, the most popular choice involves interpolating polynomials, most of which can be derived using Lagrange Interpolation (or its variant, Hermite Interpolation). Lagrange interpolation works in the following way: Given a set of k+1 data points in 1-dimension $\{(x_0,p_0),\ldots(x_k,p_k)\}$, a polynomial of order k, called a Lagrange form, L, may be constructed to interpolate those points:

$$L(x) := \sum_{j=0}^{k} p_j l_j(x) \tag{2.23}$$

where:

$$l_j(x) := \prod_{i=0, i\neq j}^{k} \frac{x - x_i}{x_j - x_i} \tag{2.24}$$

The extension to two dimensions is straightforward (and similarly for higher dimensions):

$$L(x, y) = \sum_{i=1}^{n} \sum_{j=1}^{m} u_{ij} l_i(x) l_j(y) \tag{2.25}$$

As a simple example, if one has two points, a and b at x_0 and x_1, respectively, and a linear curve is sought which passes through the two points, equation (2.24) yields:

$$l_0(x) = \frac{x - x_1}{x_0 - x_1}$$
$$l_1(x) = \frac{x - x_0}{x_1 - x_0} \tag{2.26}$$

By equation (2.25), the linear segment between x_0 and x_1 which passes through points a and b is given by substituting equation (2.26) into equation (2.23):

$$L(x) = \left(\frac{x - x_1}{x_0 - x_1} \right) a + \left(\frac{x - x_0}{x_1 - x_0} \right) b \tag{2.27}$$

where l_0 and l_1 could form the basis for a 1-dimensional line element. If such an element has length L and element origin at x=0, the shape functions of equation (2.26) would become:

$$l_0(x) = \frac{L - x}{L}$$
$$l_1(x) = \frac{x}{L} \tag{2.28}$$

For higher-dimensional elements, it becomes convenient to find the "natural" or "barycentric" coordinates, ξ_i of the element (see (Cook, Malkus, and Plesha 1989), Chapter 5). These coordinates represent the distance from an arbitrary point, P on the element to its nodes. For example, the linear line element with origin at x=0 and length L could be interpolated with coordinates ξ_i, where $\xi_1 = L_1/L$, and $\xi_2 = L_2/L$, as shown in Figure 2.18. Since $L_1 + L_2 = L$, the coordinates ξ_1 and ξ_2 are not independent. They satisfy the constraint relation $\xi_1 + \xi_2 = 1$. ξ_1

and ξ_2 can be used to state the position of an arbitrary point *P* on the line element in terms of x_1 and x_2:

$$x = \xi_1 x_1 + \xi_2 x_2 \tag{2.29}$$

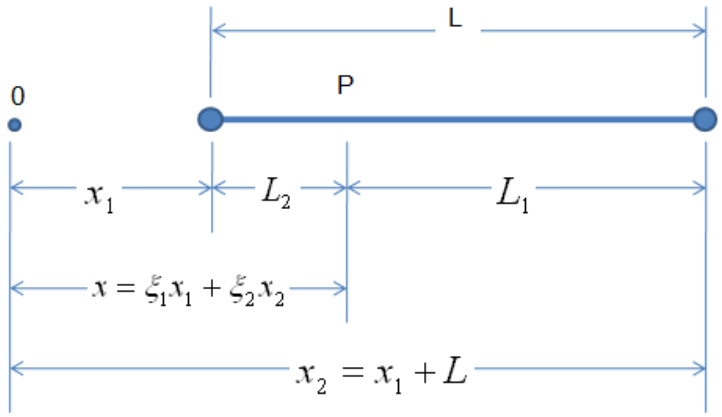


Figure 2.17. Barycentric coordinates for a linear line element

As an example, consider *P* at the centroid of the line element. In this case, $\xi_1=\xi_2=1/2$, where $x=(x_1+x_2)/2$. Equation (2.29) may be combined with the constraint relation $\xi_1 + \xi_2 = 1$, to obtain the system:

$$\begin{Bmatrix} 1 \\ x \end{Bmatrix} = \begin{pmatrix} 1 & 1 \\ x_1 & x_2 \end{pmatrix} \begin{Bmatrix} \xi_1 \\ \xi_2 \end{Bmatrix} \tag{2.30}$$

and its inverse:

$$\begin{Bmatrix} \xi_1 \\ \xi_2 \end{Bmatrix} = \frac{1}{L} \begin{pmatrix} x_2 & -1 \\ -x_1 & 1 \end{pmatrix} \begin{Bmatrix} 1 \\ x \end{Bmatrix} \tag{2.31}$$

These equations provide a linear mapping between the x and ξ coordinate systems. Interpolation of a function ϕ along line *1-2* in terms of nodal values ϕ_1 and ϕ_2 may now be performed in natural coordinates:

$$\varphi = \mathbf{N}^T \begin{Bmatrix} \varphi_1 \\ \varphi_2 \end{Bmatrix} \tag{2.32}$$

where:

$$\mathbf{N} = \begin{Bmatrix} \xi_1 \\ \xi_2 \end{Bmatrix} \tag{2.33}$$

This procedure extends generally to 2 and 3-dimensional elements. For a linear triangular element as shown in Figure 2.19, natural coordinates are expressed in terms of area ratios: $\xi_1=A_1/A$, $\xi_2=A_2/A$, $\xi_3=A_3/A$. From this, it is clear that $\xi_1 + \xi_2 + \xi_3 = 1$, and equations (2.32) and (2.33) have the analogs:

$$\begin{Bmatrix} 1 \\ x \\ y \end{Bmatrix} = \begin{pmatrix} 1 & 1 & 1 \\ x_1 & x_2 & x_3 \\ y_1 & y_2 & y_3 \end{pmatrix} \begin{Bmatrix} \xi_1 \\ \xi_2 \\ \xi_3 \end{Bmatrix} \tag{2.34}$$

or:

$$\{x\} = \mathbf{A}\{\xi\} \tag{2.35}$$

and the inverse:

$$\begin{Bmatrix} \xi_1 \\ \xi_2 \\ \xi_3 \end{Bmatrix} = \frac{1}{\det \mathbf{A}} \begin{pmatrix} x_2 y_3 - x_3 y_2 & y_2 - y_3 & x_3 - x_2 \\ x_3 y_1 - x_1 y_3 & y_3 - y_1 & x_1 - x_3 \\ x_1 y_2 - x_2 y_1 & y_1 - y_2 & x_2 - x_1 \end{pmatrix} \tag{2.36}$$

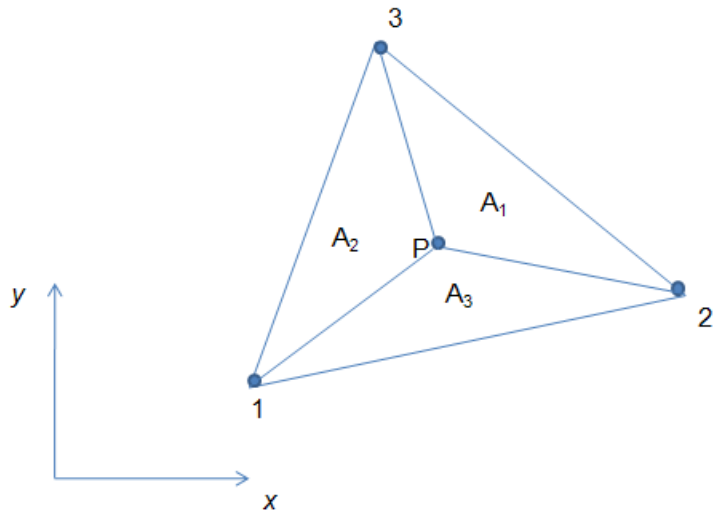


Figure 2.18. Barycentric coordinates for a linear triangle

The functions N are always unity at the nodes (via property (2.14)). Equations (2.20) and (2.21) show that element solution variable ϕ must be differentiated and integrated over each element T_i. The first procedure is easily accomplished via the chain rule, and integration is usually carried out via Gaussian Quadrature. For a more detailed explanation of this procedure, see (Cook, Malkus, and Plesha 1989). The determination of higher-order shape functions for a triangle (which generalize to simplices of any dimension) is straightforward. One may use the Lagrange Form (equation (2.24)), or simply by inspection and solution of constraints on the natural coordinates. In general, one seeks shape functions $N_i=N_i(\xi_1,\xi_2,\xi_3)$ in the relation:

$$\varphi = \mathbf{N}^T \{\varphi\} \tag{2.37}$$

where $\{\phi\}=\phi_i$ are the nodal degrees of freedom. In 2 dimensions, ϕ will be a polynomial in x and y such that $\phi=\phi(\xi_1,\xi_2,\xi_3)=\phi(x,y)$ and expressed by the expansion:

$$\varphi = \sum_{i=1}^{n} a_i \xi_1^q \xi_2^r \xi_3^s \tag{2.38}$$

in which *q*, *r*, *s* are nonnegative integers that range over the n possible combinations for which $q+r+s=p$. Thus ϕ is a complete polynomial of degree *p* in Cartesian coordinates (Cook, Malkus, and Plesha 1989). For example, for the quadratic triangle as shown in Figure 2.20, $n=6$ and $p=2$, and:

$$\begin{aligned}\varphi &= a_1 f(\xi_1)^2 + a_2 f(\xi_2)^2 + a_3 f(\xi_3)^2 + a_4 \xi_1 \xi_2 \\ &+ a_5 \xi_2 \xi_3 + a_6 \xi_3 \xi_1\end{aligned} \tag{2.39}$$

The constants a_1 thru a_6 are easily found by the requirement that at each node, $\phi=1$ and zero elsewhere. The coefficients a are set equal to ϕ at the nodes. Thus for the corner nodes, we have:

$$\begin{aligned}&\text{Node 1:}\quad \varphi = 0 \;@\; \xi_1 = 0, \frac{1}{2};\quad f(\xi_1)^2 = a_1 \xi_1 \left(\xi_1 - \frac{1}{2}\right) = 0 \\ &\varphi = 1 \;@\; \xi_1 = 1;\quad a_1(1)(1/2) = 1;\quad a_1 = 2\end{aligned} \tag{2.40}$$

$$\begin{aligned}&\text{Node 2:}\quad \varphi = 0 \;@\; \xi_2 = 0, \frac{1}{2};\quad f(\xi_2)^2 = a_2 \xi_2 \left(\xi_2 - \frac{1}{2}\right) = 0 \\ &\varphi = 1 \;@\; \xi_2 = 1;\quad a_2(1)(1/2) = 1;\quad a_2 = 2\end{aligned} \tag{2.41}$$

$$\begin{aligned}&\text{Node 3:}\quad \varphi = 0 \;@\; \xi_3 = 0, \frac{1}{2};\quad f(\xi_3)^2 = a_3 \xi_3 \left(\xi_3 - \frac{1}{2}\right) = 0 \\ &\varphi = 1 \;@\; \xi_3 = 1;\quad a_3(1)(1/2) = 1;\quad a_3 = 2\end{aligned} \tag{2.42}$$

and for the mid-side nodes,

$$\text{Node } 4: \quad \varphi = 1 \;@\; \xi_1, \xi_2 = \frac{1}{2}, \frac{1}{2}; \quad a_4 \left(\frac{1}{2}\right)\left(\frac{1}{2}\right) = 1$$
$$\varphi = 0 \text{ elsewhere : automatically satisified;} \quad a_4 = 4 \tag{2.43}$$

$$\text{Node } 5: \quad \varphi = 1 \;@\; \xi_2, \xi_3 = \frac{1}{2}, \frac{1}{2}; \quad a_5 \left(\frac{1}{2}\right)\left(\frac{1}{2}\right) = 1$$
$$\varphi = 0 \text{ elsewhere : automatically satisified;} \quad a_5 = 4 \tag{2.44}$$

$$\text{Node } 6: \quad \varphi = 1 \;@\; \xi_3, \xi_1 = \frac{1}{2}, \frac{1}{2}; \quad a_6 \left(\frac{1}{2}\right)\left(\frac{1}{2}\right) = 1$$
$$\varphi = 0 \text{ elsewhere : automatically satisified;} \quad a_6 = 4 \tag{2.45}$$

Thus, the shape functions for the quadratic triangle are:

$$\begin{aligned}
\mathbf{N}_1 &= \xi_1(2\xi_1 - 1) \\
\mathbf{N}_2 &= \xi_2(2\xi_2 - 1) \\
\mathbf{N}_3 &= \xi_3(2\xi_3 - 1) \\
\mathbf{N}_4 &= 4\xi_1\xi_2 \\
\mathbf{N}_5 &= 4\xi_2\xi_3 \\
\mathbf{N}_6 &= 4\xi_3\xi_1
\end{aligned} \tag{2.46}$$

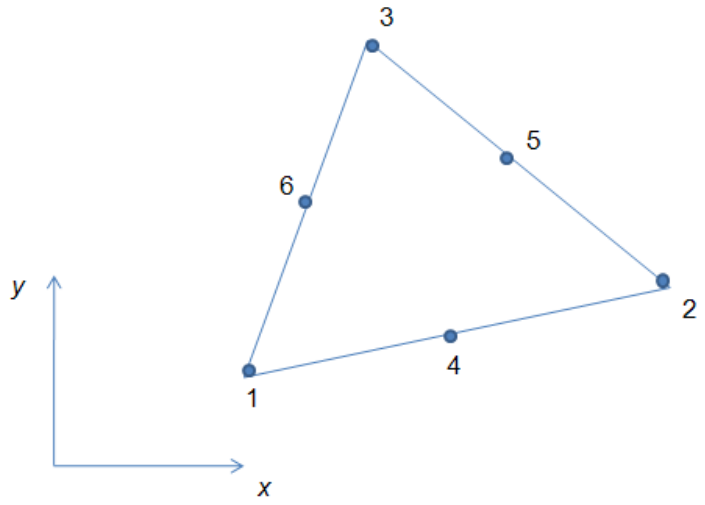


Figure 2.19. Quadratic triangle

2.4.3 Meshing

Meshes for domains of numerical problems are of two basic types – structured and unstructured. A structured mesh (often referred to as a grid) has the property that each node (or vertex) can be addressed by n indices (i,j,k,…,l) in n-dimensions. In addition, each node is mapped to a global coordinate reference via $(x_1,x_2,x_3,\ldots,x_n) = ((i-1)*dx_1,(j-1)*dx_2,(k-1)*dx_3,\ldots,(l-1)*dx_n)$, where the dx_i are the mesh spacing (here considered constant for each dimension – but this need not be the case in general). Such meshes (a simple example given in Figure 2.16) are very convenient for finite difference solutions as the finite differences are easily calculated from the grid points. However, their use in FE applications – especially structural problems with complex domains has been problematic. This is because if the grid is to conform to arbitrary domain topologies, no robust, consistent, automatic algorithm has yet been proposed to accomplish this. An alternative is to use a non-conforming, structured mesh, but the algorithms for such schemes are complicated and suffer from a lack of high accuracy. Thus, in the discussion that follows, we focus on the state-of-the art in unstructured mesh creation.

The previous sections have described how the finite element method is capable of modeling field problems in structural mechanics by discretizing the weak form of the problem and solving the problem at "nodes". These nodes usually represent the vertices of simple solids or simplices, which interpolate piecewise polynomial (shape) functions over the entire domain. However, the

problem of discretizing, or meshing, the domain is not trivial. As of this writing, the only types of subdomains (elements) which can be automatically generated to discretize a 3-dimensional domain for structural problems are tetrahedral. To generate hexahedra, as is required for accuracy in some types of FE problems, the analyst must still intervene to manually subdivide the domain into six-sided volumes (it should be mentioned that algorithms DO exist for generating hexahedra from tetrahedra by simply "gluing" two neighboring tetrahedra together. Algorithms which do this usually generate elements of poor quality, or leave a region of tetrahedra somewhere within the interior of the domain). Today's state-of-the art FE programs either utilize their own proprietary software for automatic creation of triangles and tetrahedra, or lease it from third parties. Most commercial CAD geometry kernels come equipped with tools for automatically meshing surfaces and volumes. What follows is survey of the state-of-the art in meshing technology based on an online review paper by Steve Owen of Carnegie-Mellon University (Owen n.d.).

2.4.3.1 Delaunay triangulation

The oldest and most widespread automatic tetrahedral meshing algorithms are those utilizing the Delaunay criterion (Delaunay 1934). This approach deals mainly with triangulating a given set of points, without consideration of the point locations themselves. Figure 2.21 shows a random set of points in 2D triangulated using this approach.

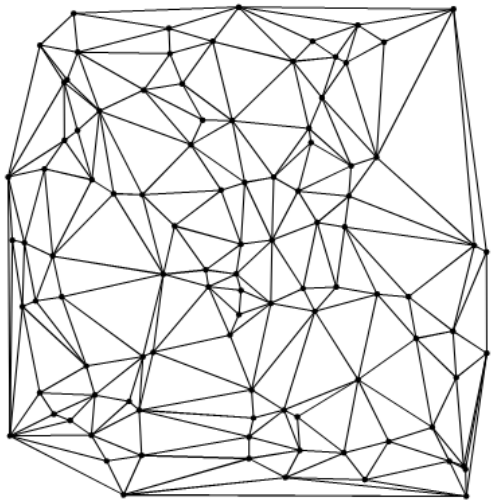


Figure 2.20. Random set of 100 points triangulated with Delaunay Criterion

The Delaunay criterion states that any node under consideration as a triangle or tetrahedron vertex must not be contained within the circumsphere of any other node which is already being used as a vertex. A circumsphere is defined as the sphere passing through the four vertices of a tetrahedron in 3D, or the circle passing through the vertices of a triangle in 2D. In two dimensions the algorithm begins by picking three points circumscribed by an "empty circle" – i.e. one not containing any other points. A neighboring point is then chosen and another triangle created containing the new point and adjacent edge. The Delaunay condition is now checked by checking if the sum of opposite angles α and γ is less than 180° (see Figure 2.22). If this condition does not hold, the adjacent edge is "flipped". That is to say the other two vertices of the four points in question are used as the adjacent edge – thus producing two different triangles containing the same four points which do meet the criterion. This process is

demonstrated in Figure 2.22 thru Figure 2.24. Once two adjacent triangles meet the Delaunay criterion, the next adjacent point is triangulated and the procedure continues in this fashion until all points in the set are triangulated.

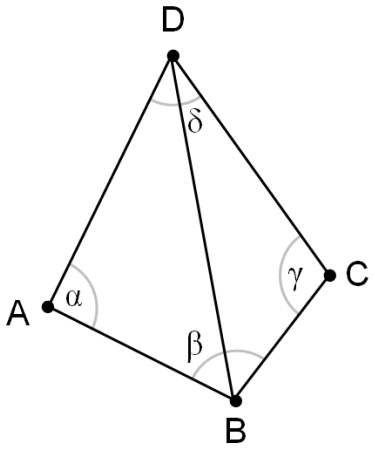


Figure 2.21. Opposite angles α and γ of candidate triangles

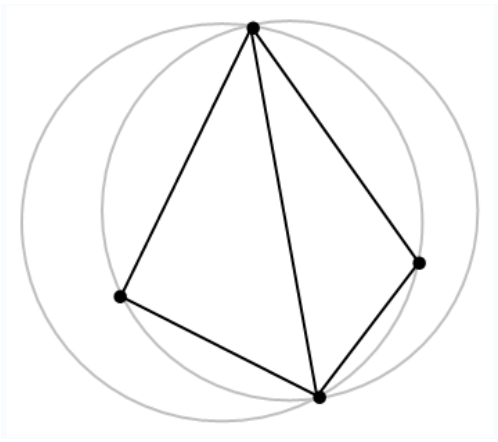


Figure 2.22. The two triangles do not meet the Delaunay Criterion

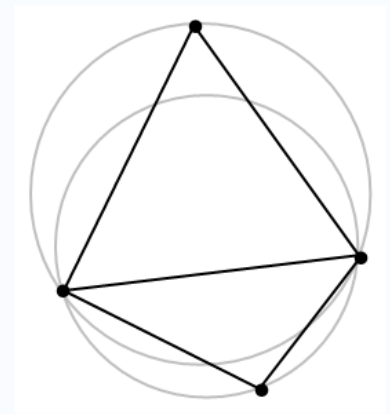


Figure 2.23. After flipping, the new triangles do meet the criterion

An important feature of the Delaunay triangulation is that in the plane, this algorithm maximizes the minimum angle. Compared to any other triangulation of the points, the smallest angle in the Delaunay triangulation is at least as large as the smallest angle in any other. However, the Delaunay triangulation does not necessarily minimize the maximum angle. This algorithm has broad application in computer graphics (in generating surface facets for rendering as an example) as well as finite element analysis, but in spite of its simplicity and speed, it is generally not favored for generating high-quality elements for FE applications. One reason for this is that, even though the algorithm produces the best meshes for predefined sets of points, far better meshes can be produced by controlling point location as well as connectivity. Here "mesh quality" is somewhat loosely defined in FE applications. It can be defined by aspect ratio (ratio of longest edge or angle to smallest edge or angle), or element Jacobian. In the former case, aspect ratios close to 1 are favorable. In the latter case, values close to zero are considered pathological. Better meshes may be achieved with the addition of a

point insertion algorithm on top of the Delaunay triangulation. See (Owen n.d.) for a list of such algorithms.

2.4.3.2 Octree mesh generation

The octree technique was primarily developed in the 1980's by Mark Shepard et al (Yerry and Shepard 1984; Shepard and Marcel 1991). With this method, cubes containing the geometric model are recursively subdivided until the desired resolution is reached. Figure 2.25

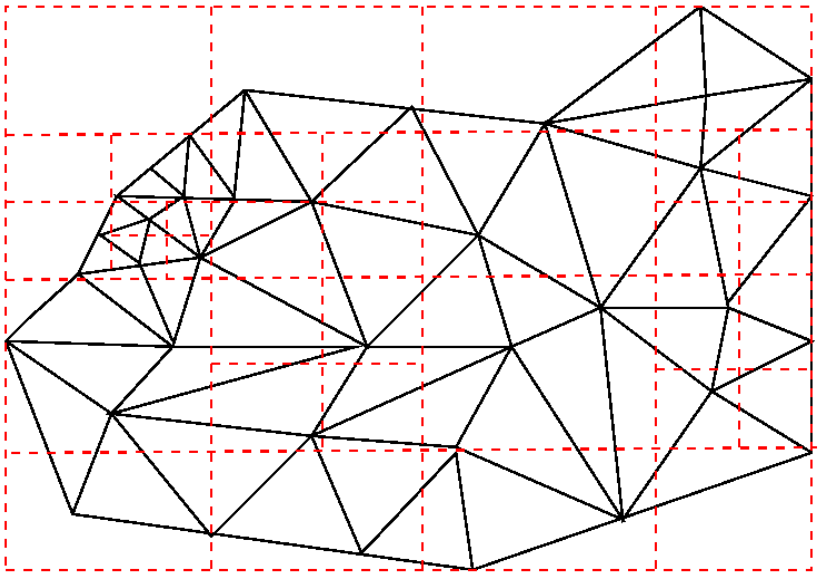


Figure 2.24. Quadtree and associated mesh

shows the equivalent two-dimensional quadtree decomposition of a model. Irregular cells are created at surface boundaries. Tetrahedra are then created from the both the irregular cells on the boundary and the internal regular cells. The octree technique does not match a predefined surface mesh, as an advancing front or Delaunay mesh might. Instead, surface facets are formed wherever the internal octree structure intersects the boundary. The resulting mesh also will change as

the orientation of the cubes in the octree structure is changed. To ensure element sizes do not change too dramatically, a maximum difference in octree subdivision level between adjacent cubes can be limited to one. Smoothing and cleanup operations can also be employed to improve element shapes. ICEM CFD's TETRA employs this technique in the mesh generator of their baseline product. Because it does not depend on surface meshing, this technique can even handle poor or broken CAD geometries. It also tends to produce fewer elements than the advancing front technique for a given element size. However, this comes at the expense of element quality.

2.4.3.3 Advancing front

The two main contributors to this method are Rainald Lohner (Lohner, Parikh, and Gumbert 1988; Lohner 1996) at George Mason University and S.H. Lo (Lo 1991a; Lo 1991b) at the University of Hong Kong. In this method, the tetrahedra are built progressively inward from the triangulated surface. An active front is maintained where new tetrahedral are formed. Figure 2.26 is a simple two dimensional example of an advancing front, where triangles have been formed at the boundary. As the algorithm progresses, the front will advance to fill the remainder of the area with triangles. In three dimensions, for each triangular facet on the front, an ideal location for a new fourth node is computed. Also determined are any existing nodes on the front that may form a well-shaped tetrahedron with the facet. The algorithm selects either the new fourth node or an existing node to form the new tetrahedron based on which will

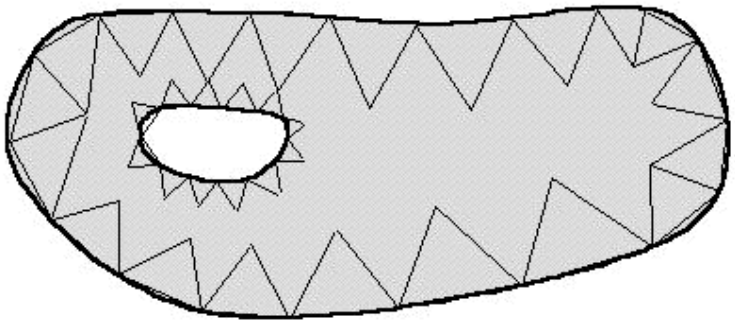


Figure 2.25. Two dimensional depiction of advancing front algorithm (first front shown)

form the best tetrahedron. Also required are intersection checks to ensure that tetrahedra do not overlap as opposing fronts advance towards each other. A sizing function can also be defined in this method to control element sizes. Lohner (Lohner 1996) proposed using a coarse Delaunay mesh of selected boundary nodes over which the sizing function could be quickly interpolated. The method is widely considered to produce the best quality meshes of all three discussed, but is less robust and relies crucially on B-rep model surface integrity. A version of S.H. Lo's advancing front mesh generator is available with the ANSYS® suite of mesh generation tools.

2.5 Mesh-free methods (particle-based)

A search of the literature on mesh-free structural analysis reveals research overwhelmingly focused on particle-based methods. The emphasis on eliminating the mesh involved in traditional finite element analysis is motivated usually by the desire to analyze physical phenomena that involve large-scale deformation of a magnitude that may completely transform geometries and topologies of the

systems under consideration, involve topological features that are difficult to characterize with existing meshing technology (cracks and dislocations, for example), or that involve new topologies that evolve from the phenomena. In such situations, it is of benefit to eliminate the errors incurred by excessively distorted elements. One finds, for example, that these particle-based methods are being studied within the context of high-impact, explicit dynamics analyses, coupled-field phase change studies, or crack propagation. Only a small percentage of the mesh-free research is focused on design-automation and CAD integration (i.e. the focus of this investigation). On may suspect there is more work being done in this area than the literature reveals due to the direct commercial implications of such work, and that some such work may be carried out in the private sector under a proprietary heading.

In any case, the field of particle-based mesh-free finite element methods is fairly large and growing. Table 2.2 gives a list of the some of the more common particle-based mesh-free methods encountered in the literature. This section will describe the two highlighted in red (and underlined), as these are earliest and most fundamental .

A common feature of all mesh-free particle methods is a weight function. A weight function is defined to have compact support, i.e. the subdomain over which it is nonzero is small relative to the rest of the domain. Each subdomain, $\Delta\Omega_i$ is associated with a node *i*. The support is often called the domain of influence of a node. The most commonly used subdomains are discs or balls.

These are shown in Figure 2.27. The following descriptions SPH and EFG are loosely excerpted from (T. Belytschko et al. 1996).

Table 2.2. List of the more common mesh-free techniques

Common Abbreviation	Full Name
CSPH	Corrected Smooth Particle Hydrodynamics
DEM	Diffuse Element Method
EFG	Element Free Galerkin
FPM	Finite Point Method
GFEM	Generalized Finite Element Method
GMLS	Generalized Moving Least Squares
MFEM	Meshless Finite Element Method
MLPG	Meshless Local Petrov-Galerkin
MLS	Moving Least Squares
NEM	Natural Element Method
PUFEM	Partition of Unity Finite Element Method
RKPM	Reproducing Kernel Particle Method
SPH	Smooth Particle Hydrodynamics

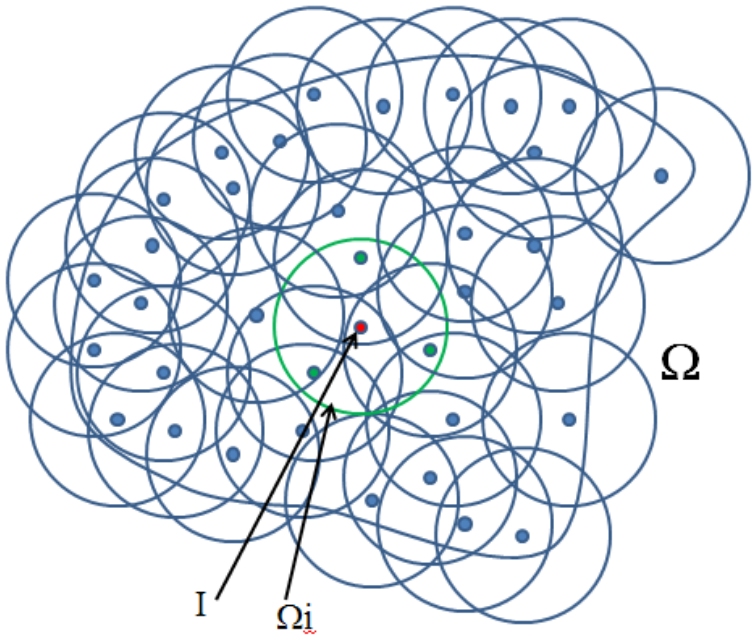


Figure 2.26. SPH domains with circular support

2.5.1 Smooth Particle Hydrodynamics (SPH)

The original particle method which seems to have the longest continuous history is the smooth particle (SPH) method (Lucy 1977). It was originally used for modeling astrophysical phenomena without boundaries such as exploding stars and dust clouds. Compared to other methods, the rate of publications was very modest for many years and is mainly reflected in the papers of Monaghan and coworkers (Monaghan 1982; Monaghan 1988). In these papers, the method was explained as a kernel estimate to provide an approximation for a function u(x) over a small compact domain Ω_i. A rationale for this method (Monaghan 1982) was provided by invoking an approximation for some function u(**x**):

$$u^h(\mathbf{x}) = \int_{\Omega} w(\mathbf{x}-\mathbf{x}', h)u(\mathbf{x}')d\Omega_{\mathbf{x}'} \tag{2.47}$$

where **x** is the coordinate of some point within the local domain Ω, and **x**' is the coordinate of the particle, or node to be interpolated in that domain. Here $u^h(\mathbf{x})$ is the approximation, w(**x**-**x**',h) is a kernel or weight function, and h is a measure of the size of the support (the subdomain). According to Monaghan (Monaghan 1982), the kernel is required to satisfy the following conditions:

- w(**x**-**x**',h)>0 on a subdomain of Ω_i
- w(**x**-**x**',h)=0 outside the subdomain Ω_i
- A normality property: $\int_{\Omega} w(\mathbf{x}-\mathbf{x}', h)d\Omega = 1$
- w(s,h) is a monotonically decreasing function,

where s=||**x**-**x**'||.

Two commonly used weight functions are the exponential and the cubic spline. Consider the isotropic, or polar weight functions, where the supports are circular as shown in Figure 2.27. The argument of $w(x)$ is s = ||x – x'||; let $\bar{s} = s/s_{max}$, where s_{max} is the radius of the support. Examples of exponential and cubic weight functions for this support are:

$$\text{Exponential}: \quad w(\bar{s}) = \begin{cases} e^{-(\bar{s}/\alpha)^2} & \text{for } \bar{s} \le 1 \\ 0 & \text{for } \bar{s} > 1 \end{cases} \tag{2.48}$$

$$\text{Cubic}: \quad w(\bar{s}) = \begin{cases} \frac{2}{3} - 4\bar{s}^2 + 4\bar{s}^2 & \text{for } \bar{s} \le 1/2 \\ \frac{4}{3} - 4\bar{s} + 4\bar{s}^2 - \frac{4}{3}\bar{s}^3 & \text{for } 1/2 < \bar{s} \le 1 \\ 0 & \text{for } \bar{s} > 1 \end{cases} \tag{2.49}$$

The exponential weight is actually C^{-1} continuous since it is not equal to zero at $\bar{s}$ =1, but for numerical purposes it resembles a weight with C^1 continuity or higher. Early researchers seemed to prefer the exponential weight function (Monaghan 1992), but Belytschko et al have found that this function is computationally more demanding than cubic and quartic functions (T. Belytschko et al. 1996), which are constructed to have C^2 continuity.

For developing approximations, discrete analogs of equation 2-45 are needed. The discrete form of this equation is obtained by numerical quadrature of the right-hand side. Since the idea is to obtain a simple formula for $u^h(x)$ in terms of nodal values $u_I=u(x_I)$, $I=1$ to n_N, the most straightforward quadrature

approaches are usually used. Thus, in one dimension the quadrature can be performed by the trapezoidal rule, which gives:

$$u^h(x) = \sum_I w(x - x_I) u_I \Delta x_I \tag{2.50}$$

The sum need be taken only over the points x_I where $w(x-x_I)>0$. In multiple dimensions, the quadrature is more difficult to come to grips with. Generally, formulas of the type:

$$u^h(\mathbf{x}) = \sum_I w(\mathbf{x} - \mathbf{x}_I) u_I \Delta V_I \tag{2.51}$$

are used, where ΔV_I is some measure of the domain surrounding node I. One difficulty in applying equation (2.50) is the development of robust techniques for assigning ΔV_I to each of the nodes. Typically, a grid is superimposed on the set of nodes and basis functions purely for the purpose of performing quadradure. Once quadrature is performed, the approximation can then be written in a form readily recognized:

$$u^h(\mathbf{x}) = \sum_I \varphi_I(\mathbf{x}) u_I \tag{2.52}$$

where:

$$\phi_I(\mathbf{x}) = w(\mathbf{x} - \mathbf{x}_I) \Delta V_I \tag{2.53}$$

The $\phi_I(x)$ are the SPH shape functions of the approximation. In most cases, $u_I \neq u^h(x_I)$, so the parameters, u_I cannot be treated exactly like nodal values, and the shape functions are not true interpolants since they do not pass through the data (T. Belytschko et al. 1996). As mentioned previously, there are numerous different particle-based formulations, but they can all be classified as “kernel”

methods (like SPH), or related to MLS (that is: their shape function formulation follows a Moving Least Squares procedure (T. Belytschko et al. 1996)). Table 2.3 lists some common formulations:

Table 2.3. Common mesh-free shape function formulations

Method	Discrete Form $u^h(\mathbf{x}) = \sum_I N_I(\mathbf{x}) u_I$	Continuous Form
SPH	$N_I(\mathbf{x}) = w(\mathbf{x}-\mathbf{x}_I)\Delta V_I$	$u^h(\mathbf{x}) = \int_\Omega w(\mathbf{x}-\mathbf{x}')u(\mathbf{x}')d\Omega_{x'}$
MLS	$N_I(\mathbf{x}) = \mathbf{p}^T(\mathbf{x})\mathbf{D}(\mathbf{x})$	$u^h(\mathbf{x}) = \int_\Omega \mathbf{C}(\mathbf{x},\bar{\mathbf{x}})w(\mathbf{x}-\bar{\mathbf{x}})u(\bar{\mathbf{x}})d\Omega_{\bar{x}}$
RKPM	$N_I = \mathbf{a}^T\mathbf{p}(\mathbf{x}-\mathbf{x}_I)w(\mathbf{x}-\mathbf{x}_I) = \mathbf{p}^T\mathbf{D}$	$u^h(\mathbf{x}) = \int_\Omega \mathbf{C}(\mathbf{x},\mathbf{x}')w(\mathbf{x}-\mathbf{x}')u(\mathbf{x}')d\Omega_x$
Hp Clouds	$N_I = \phi_I^k(u_I + \mathbf{b}_I^T\mathbf{p}(\mathbf{x}))$	
PUFEM	$N_I = \phi_I^0(u_I + \mathbf{b}_I^T\mathbf{p}(\mathbf{x}))$	

So far, the discussion of mesh-free methods has focused on formulations. Other distinguishing features of the various methods involve generating the discrete system equations. There are essentially two basic types of algorithms: Point Collocation and the Galerkin Method. Point collocation is the most straightforward and works as follows: At any node J, one determines the nodes which contain node J within their domain of influence. In SPH, the construction of the discrete form of the Poisson equations then simply involves calculating the second derivatives of the shape functions of all relevant nodes at node *J*, i.e. at $x = X_J$. This procedure is very fast for most formulations, but Belytschko et al (T.

Belytschko et al. 1996) claim that it suffers from instability. The Galerkin Method is described next.

2.5.2 Element-Free Galerkin (EFG)

The application of a particle-based mesh-free method to problems in elastostatics requires some scheme to generate the discrete system equations. In addition to the point collocation method just described, variational or Galerkin methods are also very popular. Discretization by the Galerkin method follows essentially the same procedure as described in section 2.4. Namely, it requires a weak form or variational principle. The procedure simply involves substituting one of the shape functions of Table 2.3 into equations 2.12 thru 2.20.

One major problem in mesh-free methods involves how to evaluate the integrals in equation 2.17. Belytschko (T. Belytschko et al. 1996) identifies three approaches:

1.) Nodal integration, where the integral is evaluated by

$$\int_{\Omega} f(\mathbf{x}) d\Omega_{\mathbf{x}} = \sum_{I=1}^{n_N} f(\mathbf{x}_I) \Delta V_I$$

2.) Cell or octree quadrature, where a regular array of domains in the background is used for quadrature

3.) A background finite element mesh is used for quadrature

In the methods 2) and 3), the quadrature points do not need to coincide with the nodal degrees of freedom. The first method is the fastest, but "like nodal collocation appears to suffer from instability" (T. Belytschko et al. 1996). The second method is stable but less precise than the third method. The third method

is the most precise of the three but suffers from all the drawbacks of traditional meshing.

2.5.3 Boundary conditions in particle methods

One the biggest problems remaining in the mesh-free particle methods is the accurate application of essential boundary conditions. The problem arises because of the non-interpolatory character of the mesh-free approximation. That is, the approximation does not pass through the nodal parameters - the shape functions do not obey the Kronecker-Delta property introduced in section 2.4 (equation (2.15)). As a consequence, the imposition of boundary conditions on the dependent variable, i.e. Dirichlet or essential boundary conditions, is quite awkward. The following methods have been used by various researchers to address the problem:

1.) Lagrange Multiplier approaches (T. Belytschko, Y.Y. Lu, and Gu 1994)
2.) Modified variational principles (T Belytschko, Y.Y Lu, and Gu 1994)
3.) Penalty Methods
4.) Perturbed Lagrangian (Chu and Moran 1995)
5.) Finite Element Coupling (Krongauz and T. Belytschko 1996)

The disadvantage of the first approach is that the discrete equations for a linear self-adjoint PDE are no longer positive definite nor banded. However, of the five methods listed, the Lagrange multiplier method is the most accurate method for imposing Dirichlet boundary conditions and is therefore quite useful

for smaller problems, such as smaller two-dimensional problems where the cost of solving the equations is immaterial (T. Belytschko, Y.Y. Lu, and Gu 1994). The approach based on the modified variation principle results in banded equations, but the boundary conditions are not imposed with as high a degree of accuracy. For practical purposes, the coupled EFG/finite element approach appears to be most satisfactory. In this approach, elements are placed around the boundary of the domain and the mesh-free approximation is coupled to the finite element approximation, and essential boundary conditions are applied to finite element nodes by standard methods. This approach is particularly useful when finite elements are used as a background mesh for quadrature, since then a finite element mesh is already available. Note however, that only the elements adjacent to the boundary are used for purposes of approximation. Another approach to imposing boundary conditions has been developed by Gosz (T. Belytschko et al. 1996) who imposes the condition by forcing the weight function to go to zero on Dirichlet boundaries. In this technique, nodes cannot be placed near the boundary. It should be mentioned that Gosz seemed unaware of the similar method articulated in (Kantorovich and Krylov 1958), not to mention (Rvachev 1975).

2.5.4 Particle methods overview

The differences between most of the methods listed in Table 2.2 have to do with formulation of the interpolating functions. Of these, only SPH was described as the details of particle-based interpolating functions do not directly concern this research. These details are however relevant to the history of mesh-

free methods in general, and so the following offers a condensed overview – the details of which we will not be pursued.

A parallel path to SPH in constructing meshless approximations, which commenced in the early nineteen-nineties, is the use of moving least-square approximations. Nayroles et al. (Nayroles, Touzot, and Villon 1992) were evidently the first to use moving least square approximations in a Galerkin method called the diffuse element method (DEM). Belytschko et al. (T. Belytschko, Gu, and Y.Y. Lu 1994) refined and modified the method and called their method EFG, element-free Galerkin. This class of methods is consistent and in the forms proposed quite stable, although substantially more expensive than SPH. Later, Oden (Duarte and Oden 1995) and Babuska and Melenk (Babuska and Melenk 1996) recognized that the methods based on moving least squares are specific instances of partitions of unity. This unified the field somewhat and gave birth to another strand of research into particle-based methods known as PU or PUFEM (Partition-of-Unity Finite Element Method).

We conclude by noting that all the particle-based methods seem to suffer from a lack of linear consistency (T. Belytschko et al. 1996). That is to say, for a 2^{nd} order partial differential equation (such as is encountered in elastostatics), linear and constant pressures are not reproduced exactly (the patch test is failed). Furthermore, in the EFG approach, the assembled system matrices suffer from high condition numbers, as well as inefficient assembly and inaccuracy when non-uniform node spacing is used.

2.6 Contact mechanics

Any comprehensive assembly design/analysis system must incorporate a method of resolving the stresses and strains induced at regions where component surfaces touch one-another. Such problems are frequently encountered in structural analysis, and are generally referred to as "contact problems". Finite element methods have been used for years to solve contact problems (Zienkiewicz and Taylor 2000) and the traditional finite element approach to such problems is well-established and mature. According to Zienkiewicz,

> "…contact problems are inherently nonlinear since, prior to contact, boundary conditions are given by traction conditions (often the traction being simply zero) whereas during contact kinematic constraints must be imposed which prevent penetration of one boundary through the other, called the 'impenetrability condition'. The solution of a contact problem involves first identifying which points on a boundary interact and second the insertion of appropriate conditions to prevent the penetration."

As will be seen, the second part of the problem gives rise to two popular methods – the "Penalty Method", and the "Lagrange formulation". However, before discussion of these methods can begin, some important results from the Theory of Elasticity, due to Hertz (Hertz 1896) should be discussed.

2.6.1 The Hertz contact problem

When the region of contact between two bodies is simply connected, smooth, easily identified, and resulting displacements are small, Hertz discovered an analytical solution under the following additional assumptions:

- Loading is perpendicular to the surface. That is, the effect of surface shear is neglected.
- The contact area dimensions are small compared to the radii of curvature of the bodies under load.
- The radii of curvature of the contact areas are very large compared to the dimensions of these areas.
- The shape of the resulting contact region is an ellipse

Hertz's work was the first to give a reasonable analytical description of the elastic fields resulting from two smooth bodies in contact under a compressive load. In particular, his solution predicted the extent of the resulting contact surface, as well as the displacements, stresses and strains within this ellipsoidal "contact zone". A full discussion of this solution is beyond the scope of the present work, but some of the important results are summarized below, as these are essential in assessing the accuracy and validity of any numerical contact analysis procedure. The following results, based on Hertz's solution, can be found in accessible form in (Harris 1991) (and the equations presented below are taken directly from this source). Major and minor diameters, a and b of the contact ellipse, are given by:

$$a = a^* \left[\frac{3Q}{2\Sigma\rho} \left(\frac{(1-\nu_1^2)}{E_1} + \frac{(1-\nu_2^2}{E_2} \right) \right]^{1/3} \tag{2.54}$$

$$b = b^* \left[\frac{3Q}{2\Sigma\rho} \left(\frac{(1-\nu_1^2)}{E_1} + \frac{(1-\nu_2^2}{E_2} \right) \right]^{1/3} \tag{2.55}$$

The maximum normal displacement within the contact zone is given by:

$$\delta = \delta^* \left[\frac{3Q}{2\Sigma\rho} \left(\frac{(1-\nu_1^2)}{E_1} + \frac{(1-\nu_2^2}{E_2} \right) \right]^{2/3} \frac{\Sigma\rho}{2} \tag{2.56}$$

where Q is the applied force over the contact surface, ν_1, ν_2, E_1, and E_2 are Poisson's ratio and the elastic modulus associated with contact surfaces 1 and 2, respectively. $\Sigma\rho$ is the "curvature sum" of the two surfaces defined by:

$$\Sigma\rho = \frac{1}{r_{I1}} + \frac{1}{r_{I2}} + \frac{1}{r_{II1}} + \frac{1}{r_{II2}} \tag{2.57}$$

where r is the radius of curvature at the center of contact, and the indices I and II indicate principle directions of curvature. The dimensionless quantities a* and b* are defined as:

$$a^* = \left(\frac{2\kappa^2 \wp}{\pi} \right)^{1/3} \tag{2.58}$$

$$b^* = \left(\frac{2\wp}{\pi\kappa} \right)^{1/3} \tag{2.59}$$

$$\delta^* = \frac{2\Im}{\pi} \left(\frac{\pi}{2\kappa^2 \wp} \right)^{1/3} \tag{2.60}$$

Here, κ=a/b, and $\Im$ and $\wp$ are the complete elliptic integrals of the first and second kind, respectively:

$$\Im = \int_0^{\pi/2} \left[1 - \left(1 - \frac{1}{\kappa^2} \right) \sin^2 \varphi \right]^{-1/2} d\varphi \tag{2.61}$$

$$\wp = \int_0^{\pi/2} \left[1 - \left(1 - \frac{1}{\kappa^2} \right) \sin^2 \varphi \right]^{1/2} d\varphi \tag{2.62}$$

Equations (2.61) and (2.62) can be performed numerically, and used to produce tables for the dimensionless parameters a*, b*, and δ*, which may then be interpolated to calculate the contact ellipse properties. Once this is done, the maximum scalar contact stress (or pressure) is given by:

$$p_{\max} = \frac{3Q}{2\pi ab} \tag{2.63 a}$$

for two spheres in contact, or:

$$p_{\max} = \frac{2Q}{\pi a} \tag{2.63 b}$$

for two cylinders with coincident common axis. Furthermore, the normal stress within the contact zone is given by:

$$p = \frac{3Q}{2\pi ab} \left[1 - \left(\frac{x}{a} \right)^2 - \left(\frac{y}{b} \right)^2 \right]^{1/2} \tag{2.64 a}$$

for two spheres, or:

$$p = \frac{2Q}{\pi a} \left[1 - \left(\frac{x}{a} \right)^2 \right]^{1/2} \tag{2.64 b}$$

for two cylinders, from which it is easily seen that the maximum contact stress occurs at the center of the ellipse (x=y=0).

2.6.2 Weak form – Lagrange Multiplier Approach

The general static structural contact problem is most commonly approached by stating the weak form of the problem with the use of Lagrange multipliers. The following form of the problem may be found in (Bhatti 2006) (Chapter 10). Consider two elastic bodies in contact, as shown in Figure 2.28. The bodies meet in a surface making up the contact zone, denoted by Γ, whose center is marked *P*. One body is denoted as the master (body 1 in Figure 2.28 This is the body whose surface-normal is used for integration), while the other is called the slave (body 2). The variational form for the deformation of both bodies may be expressed as:

$$\delta H(\mathbf{u}) + \delta \Pi_c(\mathbf{g}, \mathbf{F}) = 0 \tag{2.65}$$

where δH contains the combined elastic deformation of the target and contactor (slave and master, respectively) bodies:

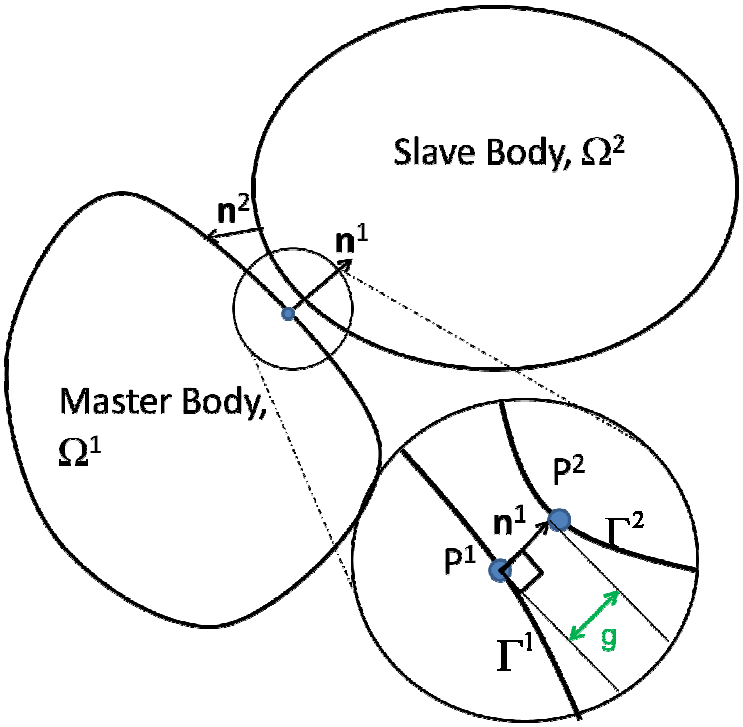


Figure 2.27. Two elastic bodies in contact. Inset depicts a point P^1 on body 1 (the master), and its nearest neighbor, P^2 on body 2.

$$\delta H(\mathbf{u}) = \delta H(\mathbf{u}^2) + \delta H(\mathbf{u}^1) \tag{2.66}$$

and the variation $\delta\Pi$ contains the deformational energy owing only to the contacting surfaces of both bodies. For simplicity, we assume that the contact is frictionless.

$$\delta\Pi_c(\mathbf{g},\mathbf{F}) = \delta\Pi_c(\mathbf{u}_n^2,\mathbf{F}_n^2) + \delta\Pi_c(\mathbf{u}_n^1,\mathbf{F}_n^1) \tag{2.67}$$

The Lagrange multiplier approach imposes the contact constraint (the constraint imposed by the contacting surface of one body on the other) by applying the reaction force necessary to impose the constraint (and solving for

that instead of the displacement-based constraint). Thus the functional associated with the contacting surfaces takes the form:

$$\Pi_c = \int_{A_c} F_n g dA \tag{2.68}$$

where g is the contact gap (about which more is said in the next section):

$$g = u_n^2 - u_n^1 \tag{2.69}$$

The variation δH is familiar from the Theory of Elasticity and given by:

$$\delta H(\mathbf{u}) = \int_V (\delta\boldsymbol{\varepsilon})^T \boldsymbol{\sigma} dV - \int_V (\delta\mathbf{u})^T \mathbf{b} dV - \int_{A_q} (\delta\mathbf{u})^T \mathbf{f} dA \tag{2.70}$$

Thus, for the target (slave) body, the internal and contact terms may be combined to yield

$$\begin{aligned} &\delta H(\mathbf{u}^2) + \delta\Pi_c(\mathbf{u}^2, \mathbf{F}^2) = \int_{V^2} \left(\delta\boldsymbol{\varepsilon}^2\right)^T \boldsymbol{\sigma}^2 dV - \int_{V^2} \left(\delta\mathbf{u}^2\right)^T \mathbf{b}^2 dV \\ &- \int_{A_q} \left(\delta\mathbf{u}^2\right)^T \mathbf{f}^2 dA + \int_{\Gamma^1} (\delta u_n^2 F_n^2 + u_n^2 \delta F_n^2) dA \end{aligned} \tag{2.71}$$

and for the contactor (master), we do the same, but note that the contact forces are equal and opposite:

$$F_n^2 + F_n^1 = 0 \tag{2.72}$$

Which, in turn yields:

$$\begin{aligned} &\delta H(\mathbf{u}^1) + \delta\Pi_c(\mathbf{u}^1, \mathbf{F}^1) = \int_{V^1} \left(\delta\boldsymbol{\varepsilon}^1\right)^T \boldsymbol{\sigma}^1 dV - \int_{V^1} \left(\delta\mathbf{u}^1\right)^T \mathbf{b}^1 dV \\ &- \int_{A_q} \left(\delta\mathbf{u}^1\right)^T \mathbf{f}^1 dA - \int_{\Gamma^1} (\delta u_n^1 F_n^1 + u_n^1 \delta F_n^1) dA \end{aligned} \tag{2.73}$$

Combining the two:

$$\delta H(\mathbf{u}) + \delta\Pi_c(\mathbf{g},\mathbf{F}) = \int_V (\delta\boldsymbol{\varepsilon})^T \boldsymbol{\sigma} dV - \int_V (\delta\mathbf{u})^T \mathbf{b} dV$$
$$- \int_{A_q} (\delta\mathbf{u})^T \mathbf{f} dA + \int_{A_c} (\delta u_n^2 - \delta u_n^1) F_n dA + \int_{A_c} (u_n^2 - u_n^1)\delta F_n dA \geq 0 \tag{2.74}$$

The inequality arises due to the fact that the last term must be greater than or equal to zero in order to impose an impenetrability constraint. That is to say, if the gap becomes negative (within the course of iterative calculation), the gap constraint force must also become negative ($\delta F \leq 0$) in order to keep the residual positive:

$$\int_{\Gamma^1} (u_n^2 - u_n^1)\delta F_n dA \geq 0 \tag{2.75}$$

It is thus convenient to separate the displacement-based variational terms from the force-based terms into two independent equations:

$$\delta H(\mathbf{u}) + \delta\Pi_c(\mathbf{u},\mathbf{F}) = \int_V (\delta\boldsymbol{\varepsilon})^T \boldsymbol{\sigma} dV - \int_V (\delta\mathbf{u})^T \mathbf{b} dV$$
$$- \int_{A_q} (\delta\mathbf{u})^T \mathbf{f} dA + \int_{\Gamma^1} (\delta u_n^2 - \delta u_n^1) F_n dA = 0 \tag{2.76}$$

$$\int_{\Gamma^1} (u_n^2 - u_n^1)\delta F_n dA \geq 0 \tag{2.77}$$

2.6.3 The finite element formulation

A typical finite element implementation of equations (2.76) and (2.77) may proceed as follows. The nodal displacements and displacement variations are given by:

$$\mathbf{u} = \mathbf{N}^T \mathbf{d} \Rightarrow \delta\mathbf{u} = \mathbf{N}^T \tag{2.78}$$

However, the contact forces must also be explicitly interpolated (the choice of shape function N_F depends on the particular formulation):

$$\mathbf{F}_n = N_{F1}F_1 + N_{F2}F_2 + ... = \mathbf{N}_F^T\mathbf{F} \tag{2.79}$$

Where $\mathbf{F}=\mathbf{F}_1+\mathbf{F}_2+\mathbf{F}_3+\ldots$ are the nodal forces at the contact interfaces. The normal displacements, $\mathbf{u}_n$ can be computed by taking the dot product of displacements with the normal vector. So, for the slave surface (superscript 2):

$$\mathbf{u}_n^2 = (\mathbf{n}^1)^T\mathbf{u}^2 = (\mathbf{n}^1)^T(\mathbf{N}^2)^T\mathbf{d}^2; \quad \delta u_n^2 \Rightarrow (\mathbf{n}^1)^T(\mathbf{N}^2)^T$$

$$\mathbf{F}_n^2 = (\mathbf{N}^2)_F^T\mathbf{F}^2 \Rightarrow \delta\mathbf{F}_n^2 \Rightarrow (\mathbf{N}^2)_F^T$$

$$\int_{\Gamma^1}\mathbf{u}_n^2\delta\mathbf{F}_n^2 dA = \int_{\Gamma^1}(\delta\mathbf{F}_n^2)^T\mathbf{u}_n^2 dA = \int_{\Gamma^1}\mathbf{N}_F^2(\mathbf{n}^1)^T(\mathbf{N}^2)^T\mathbf{d}^2 dA = \mathbf{k}_c^2\mathbf{d}^2$$

$$\int_{\Gamma^1}\delta\mathbf{u}_n^2\mathbf{F}_n^2 dA = \int_{\Gamma^1}(\delta\mathbf{u}_n^2)^T(\mathbf{N}^2)_F^T\mathbf{F}^2 dA$$
$$= \int_{\Gamma^1}\mathbf{N}^2\mathbf{n}^1(\mathbf{N}^2)_F^T\mathbf{F}^2 dA = (\mathbf{k}_c^2)^T\mathbf{F}^2$$

And similarly, for the master surface:

$$\mathbf{u}_n^1 = (\mathbf{n}^1)^T\mathbf{u}^1 = -(\mathbf{n}^1)^T\mathbf{N}^T\mathbf{d}^1; \quad \delta u_n^1 \Rightarrow -(\mathbf{n}^1)^T(\mathbf{N}^1)^T$$

$$\mathbf{F}_n^1 = (\mathbf{N}^1)_F^T\mathbf{F}^1 \Rightarrow \delta\mathbf{F}_n^1 \Rightarrow (\mathbf{N}^1)_F^T$$

$$-\int_{\Gamma^1}\mathbf{u}_n^1\delta\mathbf{F}_n^1 dA = -\int_{\Gamma^1}(\delta\mathbf{F}_n^1)^T\mathbf{u}_n^1 dA = -\int_{\Gamma^1}\mathbf{N}_F^1(\mathbf{n}^1)^T(\mathbf{N}^1)^T\mathbf{d}^1 dA = -\mathbf{k}_c^1\mathbf{d}^1$$

$$-\int_{\Gamma}\delta\mathbf{u}_n^1\mathbf{F}_n^1 dA = -\int_{\Gamma^1}(\delta\mathbf{u}_n^1)^T(\mathbf{N}_F^1)^T\mathbf{F}^1 dA$$
$$= -\int_{\Gamma^1}\mathbf{N}^1\mathbf{n}^1(\mathbf{N}_F^1)^T\mathbf{F}^1 dA = -(\mathbf{k}_c^1)^T\mathbf{F}^1$$

The combined contact stiffness contribution (from both bodies) is:

$$\mathbf{k}_c = \left\{ \int_{\Gamma^1} \mathbf{N}_F^2 (\mathbf{n}^1)^T (\mathbf{N}^2)^T \, dA, -\int_{\Gamma^1} \mathbf{N}_F^1 (\mathbf{n}^1)^T (\mathbf{N}^1)^T \, dA \right\} \tag{2.80}$$

Thus, equation (2.74) becomes:

$$\mathbf{Kd} - \mathbf{r}_b - \mathbf{r}_q + \mathbf{k}_\mathbf{c}^T \mathbf{F} = \mathbf{0} \tag{2.81}$$

where the usual finite element formulation gives:

$$\mathbf{K} = \int_V \mathbf{ACA}^T \, dV$$

where the matrix C corresponds to a constitutive relation, and:

$$\mathbf{r}_b = \int_V \mathbf{Nb} \, dV$$

$$\mathbf{r}_f = \int_{A_c} \mathbf{Nf} \, dA$$

Equation (2.77) becomes:

$$\int_{\Gamma^1} (u_n^2 - u_n^1) \delta F_n \, dA \geq 0 \Rightarrow \mathbf{k}_c \mathbf{d} \leq \mathbf{0} \tag{2.82}$$

Equations (2.81) and (2.82) are combined to give:

$$\begin{pmatrix} \mathbf{K} & \mathbf{k}_c^T \\ \mathbf{k}_c & \mathbf{0} \end{pmatrix} \begin{Bmatrix} \mathbf{d} \\ \mathbf{F} \end{Bmatrix} = \begin{Bmatrix} \mathbf{r} \\ \mathbf{0} \end{Bmatrix} \tag{2.83}$$

These equations can be assembled using the usual finite element assembly process. This method is completely general and does not require the slave and master surfaces to have conforming meshes. One drawback is that the method involves inequalities, which necessitate some optimization technique.

2.6.4 Gap monitoring

The contact surface reaction force, **F** in equation (2.81) is an unknown which must be solved for iteratively. The particular choice of starting value for **F** will in general depend on an assessment of the gap between contacting surfaces. Note, however, that the form of the gap relation (equation (2.67)) depends on master and slave normal deformations only. This form may only be usefully employed once *two surfaces have already made contact* and local deformation results.

Thus, any iterative solution to equation (2.81) must also constantly monitor which surfaces or elements have come into contact. Indeed, this is a prerequisite to solving (2.81). Such a calculation involves determining points on element surfaces which may be in potential contact. One requirement for such surfaces is that the element normals on the target and contactor surfaces are equal and opposite:

$$\mathbf{n}^2 = -\mathbf{n}^1 \tag{2.84}$$

This requirement would be met, for example, at point P^1 in Figure 2.28 once the gap becomes zero. The requirement (2.83) *allows a single normal surface (the master suface) to be used for the gap normal direction*. The total distance between such potential contacting points is then calculated. This can be written as:

$$\begin{aligned} g_i &= (\overline{\mathbf{u}}_i^2 - \mathbf{u}_i^i) \cdot \mathbf{n}_i^1 + g_{0i} \\ g_{0i} &= (\overline{\mathbf{x}}_i^2 - \mathbf{x}_i^1) \cdot \mathbf{n}_i^1 \end{aligned} \tag{2.85}$$

where the subscript *i* ranges over the number of element pairs in potential contact.

Various criteria have been used (see (Wriggers 2002)) to determine which elements on each body should be monitored for potential contact (stored as pairs, applying equation 2.85 to assess the gap value). This monitoring process leads to constraint application according to:

$$\mathbf{K} = \mathbf{K} + \mathbf{k}_i^c, \quad g_i \le 0 \tag{2.86}$$

2.6.5 The Penalty Method

As seen in equation (2.81), the Lagrangian contact formulation leads to a larger system of equations than would otherwise be seen for two or more elastic bodies. In particular, an additional term is required for every degree of freedom on the contacting surfaces. One way around this is, instead of treating the contact force as an unknown, to associate this force with a penalty parameter, β such that:

$$F_n = \begin{cases} 0 & if \ \ gap > 0 \\ \beta(u_n^2 - u_n^1) & if \ \ gap \le 0 \end{cases} \tag{2.87}$$

Where the gap is computed as described above. Thus the last term of equation (2.76) becomes:

$$\int_{\Gamma^1} (\delta u_n^2 - \delta u_n^1) F_n dA = \int_{\Gamma^1} \delta u_n^2 \beta u_n^2 dA - \int_{\Gamma^1} \delta u_n^1 \beta u_n^1 dA$$

where, for the slave surface:

$$\int_{\Gamma^1} \delta u_n^2 \beta u_n^2 dA = \int_{\Gamma^1} \beta \mathbf{N}^2 \mathbf{n}^1 (\mathbf{n}^1)^T (\mathbf{N}^2)^T \mathbf{d}^2 dA = \mathbf{k}_p^2 \mathbf{d}^2$$

and for the master surface:

$$-\int_{\Gamma^1} \delta u_n^1 \beta u_n^1 dA = -\int_{\Gamma^1} \beta \mathbf{N}^1 \mathbf{n}^1 (\mathbf{n}^1)^T (\mathbf{N}^1)^T \mathbf{d}^1 dA = -\mathbf{k}_p^1 \mathbf{d}^1$$

(Recall that the minus sign comes from the gap condition (2.84) and (2.85)).

Thus, the matrix, $\mathbf{k}_p^c$ is calculated as:

$$\mathbf{k}_p^c = \beta \begin{pmatrix} \int_{\Gamma^1} \mathbf{N}^1 \mathbf{n}^1 (\mathbf{n}^1)^T (\mathbf{N}^1)^T dA & -\int_{\Gamma^1} \mathbf{N}^1 \mathbf{n}^1 (\mathbf{n}^1)^T (\mathbf{N}^2)^T dA \\ -\int_{\Gamma^1} \mathbf{N}^1 \mathbf{n}^1 (\mathbf{n}^1)^T (\mathbf{N}^2)^T dA & \int_{\Gamma^1} \mathbf{N}^2 \mathbf{n}^1 (\mathbf{n}^1)^T (\mathbf{N}^2)^T dA \end{pmatrix} \tag{2.88}$$

The corresponding residual vector is:

$$\mathbf{r}_p^c = \beta \begin{pmatrix} \int_{\Gamma^1} g \mathbf{N}^1 \mathbf{n}^1 dA \\ -\int_{\Gamma^1} g \mathbf{N}^2 \mathbf{n}^2 dA \end{pmatrix} \tag{2.89}$$

The equations of equilibrium may now be written in simplified form:

$$(\mathbf{K} + \mathbf{k}_p)\mathbf{d} = \mathbf{r} \tag{2.90}$$

No additional equations are needed (the constraint force is now captured in the penalty term). This equation is of the same size as would be obtained for two or more elastic bodies with no contact. However, proper choice of penalty factor, β is now crucial. If β is too large, the equations may have difficulty converging in an iterative solution. If it is too small, excessive penetration may occur.

2.6.6 Implementation

Several issues arise when implementing equation (2.81) or (2.90) in a finite element model. These may be categorized as issues relating to contact region discretization, and issues relating to solution algorithm. In the second case, as mentioned in section section 2.6.3, the Lagrangian contact problem is one of constraint optimization. If the Penalty Method is chosen, a traditional Newton-Raphson-type solution algorithm may be employed (Wriggers 2002), but

additional logic (and potentially additional computer resources) must be added to enforce the constraint condition (2.87).

In the case of problems relating to discretization, most of the issues have to do with calculation of a non-ambiguous normal direction for gap calculations, and with the connectivity of dissimilar meshes (other issues relate to contact constitutive laws, which are beyond the scope of this research). Of these the latter problem is a topic of ongoing research, with the latest entry being that of the so-called Mortar Methods (Wriggers 2002; Wriggers and Panagiotopoulos, P.D. 1999). The most popular method currently employed to connect two dissimilar contact regions is the *node-to-surface* contact method, described in the next section.

2.6.6.1 Contact discretization and connectivity and mesh-free contact

The first problem faced in discretized contact problems has to do with calculating surface normal vectors as in equation (2.82) (used for gap monitoring). This can be problematic due to the discrete nature of a finite element mesh. If the discretization is not isoparametric with degree higher than 1, a unique normal direction does not exist for nodes on curved boundaries. Several methods have been proposed to treat this problem. One method averages the normal directions obtained from the underlying element parameterizations on either side a singularity. Another method interpolates a smooth parametric curve across the contact nodes and calculates a normal direction from this curve rather than the underlying mesh. This solution fails at true singularities, however (corners).

A second problem involves the connection of dissimilar meshes (a problem which seems to have an analogy in some particle-based mesh-free approaches). Most often, a concept of nodal or segment projection is used when nodes on one contact surface do not have clear counterparts on the other contact surface. The procedure involves selecting the node on a target contact segment with the smallest contacting segment-normal projection as in Figure 2.29. Such a procedure is often referred to as a *node-to-surface* algorithm. Gap and reaction force calculations are done on this basis. There may be several nodes nearer one segment than any other. A connectivity matrix is then constructed between the node(s) and the segment. The details of how this is done vary. Figure 2.30 shows a degenerate situation which suffers from both of the problems mentioned above (i.e: a) it is not clear to which segment point P belongs, and b) there is no unique normal distance from target segments to point P). A solution to the second problem may involve joining segments together according to some criterion, such as the Babuska-Brezzi condition (Babuska 1973). This solution involves generating coupling matrices between an underlying mesh's nodal degrees of freedom and joined surface segments. There are apparently other issues surrounding the node-to-surface implementation. Some authors have pointed out that the resulting nodal contact forces are much higher than they should be, and associated surface tractions are not correct (Cescotto and Charilier 1992). This has led to schemes which involve defining contact elements by associating nodes on one surface to their nearest point normal to the opposing surface. As the nodes and surface normal segments alternate, there are no uniqe master and slave

bodies. Such schemes are often referred to as *surface-surface*, or *surface-segment* methods (Wriggers 2002), as shown in Figure 2.31.

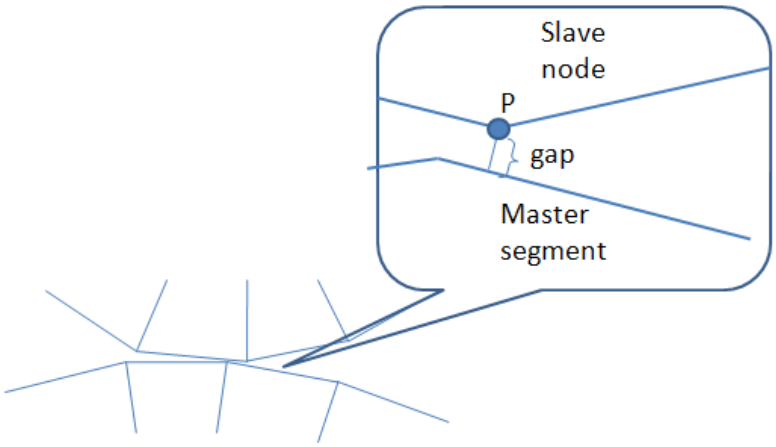


Figure 2.28. Nodal projection in Node-to-Surface Algorithm

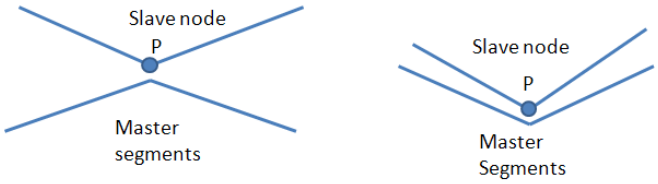


Figure 2.29. Two degenerate cases for nodal projection

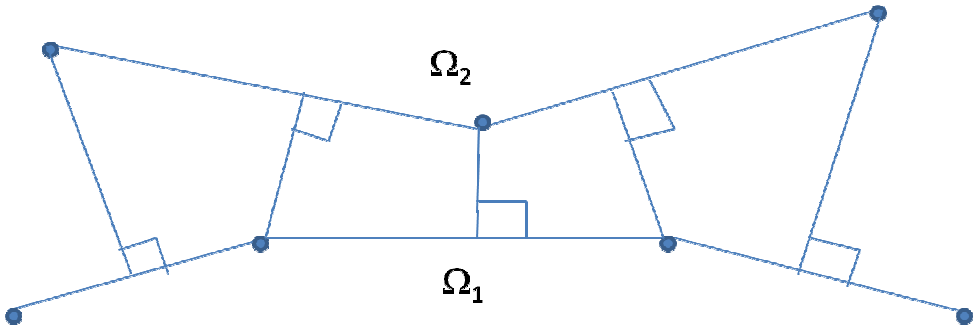


Figure 2.30. Alternating nodal projections used to construct contact segments (Surface-to-Surface Algorithm)

If a one-to-one correspondence between contact surface nodes can be established, both problems mentioned above are eliminated. If the contact problem does not involve large displacements, or contact surface constitutive laws, the existence of a matching contact-target mesh results in straightforward implementation of equations (2.81) and (2.90). Because nodal degrees of freedom on the contacting body have explicit counterparts on the target body, the matrix coupling (for the Penalty Method) between contact and target surface may be expressed simply by:

$$\begin{Bmatrix} \mathbf{f}^1 \\ \hline \mathbf{f}^2 \end{Bmatrix} = \left(\begin{array}{c|c} \mathbf{k}_p^1 & \mathbf{k}_p^{1,2} \\ \hline \mathbf{k}_p^{2,1} & \mathbf{k}_p^2 \end{array} \right) \begin{Bmatrix} \mathbf{u}^1 \\ \hline \mathbf{u}^2 \end{Bmatrix} \tag{2.91}$$

where the superscripts 1 and 2 refer to master and slave bodies, respectively, and each element in the matrix in (2.91) represents a partitioned assembly of all the degrees of freedom on the contact surfaces. Furthermore, this partitioning is identical on both bodies. The tangent stiffness and residual terms for all contacting surfaces are obtained from (2.88) and (2.89). In higher

dimensions, this type of solution (one in which there is a direct one-to-one correspondence between degrees of freedom on the master and slave surfaces expressed in the contact element pairs) is referred to as an *Isoparametric Contact Discretization* (Wriggers 2002), since there is both a one-to-one correspondence between master and slave degrees of freedom, as well as an identical parameterization of both surfaces. Elimination of the integral in equation (2.88) and (2.89) results in a *Node-to-Node*, or gap element formulation, as shown in Figure 2.32.

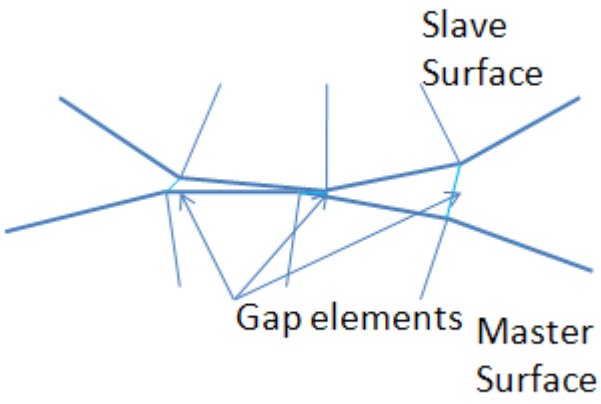


Figure 2.31. Node-to-Node gap elements

Penalty-based contact solutions have been demonstrated for some particle-based methods, such as EFG (Guangyao and T. Belytschko 2001; T. Belytschko and Fleming 1999). More complicated matrix coupling have been demonstrated for other methods (see (Chen and Wang 2000) , for the case of the RKPM for

example). None of these solutions focus on elastic surface-to-surface contact problems, however, and it is expected that results for such applications would vary, depending on such factors as whether consistency conditions are met. A study of the EFG's ability to accommodate essential boundary conditions (a closely related issue) may be found in (Fernandez-Mendez, S. and Huerta 2004). In the case of trimmed B-Spline domains (next chapter) by contrast, consistency is guaranteed by the completeness of the B-Spline formulation (proofs of this consistency may be found in many sources. One is (Hoellig 2003)) .

A mesh-free solution based on non-conforming tensor-product grids enjoys the advantage that element connectivity is trivial (shown in the next chapter). Partly because of this property, this formulation offers the possibility that *any* surfaces in contact may be treated analogously to the Isoparametric Contact Discretization, with the exception that, in general two different finite element bases are used (basis stabilization schemes such as weB-Splines, as well as the one proposed in the current work are both geometry dependent, and therefore a basis whose support intersects one domain should not be expected to be compatible with the same basis which overlaps a separate domain, as the two will in general have different normalizations). It should thus be emphasized that equations (2.88) and (2.89) have been derived with this proviso in mind. If the two bases have equal normalizations, equations (2.88) and (2.89), the contact problem will be truly analogous to the Isoparametric Contact Discretization, as described in (Wriggers 2002).

2.6.6.2 Contact solution algorithm

As has been mentioned, if the Lagrange formulation is used, solution of the contact problem is equivalent to a constraint optimization problem (Bhatti 2006; Wriggers 2002). The required solution algorithm could be quite complicated, or a specialized constraint function could be employed (Bathe and Bouzinov n.d.). If the Penalty formulation is used, a standard Newton-Raphson-type algorithm may be used. The stiffness contribution ($\mathbf{k}_p^c$ in equation (2.88)) of the contact surfaces will depend on the result of a gap calculation (this reflects the nonlinear nature of the solution) which must be updated within each Newton-Raphson iteration. The Newton-Raphson algorithm proceeds in steps by calculating the total system internal reaction force due to the externally applied loads, and then using the difference between the external load and internal reaction to calculate an incremental displacement (or strain). This incremental displacement is then added to the previously calculated displacement, and then the stiffness matrix is updated to obtain a new reaction force. This process is repeated until the difference between the external and internal forces is less than some numerical tolerance. This process is shown in Figure 2.33 for a single iteration, i where $\mathbf{r}_a$ is the external load, $\mathbf{r}_i^{nr}$ is the calculated reaction force from the previous iteration, and $\mathbf{K}_i$ is the tangent stiffness matrix at iteration, i. The stiffness matrix gets updated with incremental displacement, Δu (u_{i+1}-u_i), because, as the gap closes, more contact elements get "switched on", due to criterion (2.87).

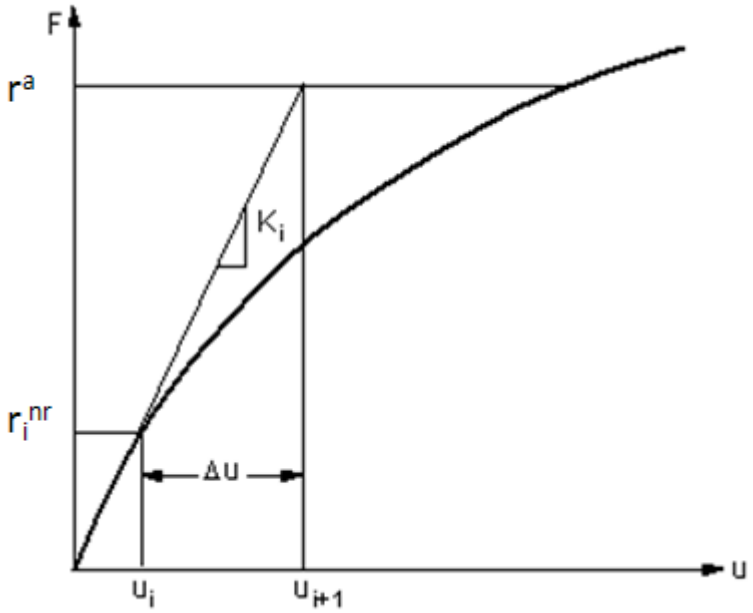


Figure 2.32. A single Newton-Raphson iteration, *i*

Chapter 3

LITERATURE REVIEW

A background in CAGD, Assembly Design, Structural Analysis, the Finite Element Method, Mesh-Free Methods, and Contact Mechanics has been given. It is the ultimate goal of this research to marry all of these disciplines within the framework of a comprehensive engineering design system in which each is enhanced by the other in a seamless, efficient, and robust manner. Commercial systems are available today which may seem to meet these requirements (e.g. UGS NX+Nastran®, SolidWorks+Cosmos®, ProE+Mechanica®), but they all suffer from the deficiencies pointed out in the introduction.

Perhaps one of the biggest current obstacles to meeting the objectives stated above lies in the requirement by modern commercial finite element systems to generate a high-quality mesh over a B-rep solid model. As a response to the substantial challenges posed by this requirement, there is a tendency for many commercial systems to sacrifice element quality (and hence solution quality) for automation and ease-of-use. A successful design/analysis system must not make this compromise. It is for this reason that a suitable mesh-free finite element formulation be identified and assessed. The mesh-free particle methods reviewed in the last section, although successful in the analysis of specialized problems, still suffer from the problems mentioned in that section, and so are removed as candidates for the mesh-free system sought in this investigation. In an automated structural assembly-design environment, the numerical analysis system must be stable, consistent, and reliable. We have found only one mesh-

free methodology which seems to fulfill these requirements within the context of structural assembly design. It involves finite elements defined over a non-conforming tensor-product B-Spline grid (in this context, "non-conforming" simply means that the basis does not interpolate the geometric domain. All that is required is that the basis support contains the domain). Such a framework has emerged relatively recently by two separate research groups, and will be introduced next in the way of a literature review.

3.1 B-Splines as finite elements

Interest in B-Spline functions as a finite element basis is probably as old as the finite element method itself (the former being older than the latter). This investigation will not recount all the accomplishments, but a good reference is (Sabin 1997). To list three highlights from early work: DeBoor used cubic splines to solve certain one-dimensional boundary value problems in his doctoral dissertation in 1966 (De Boor, C.E. 1966). The first application of B-Splines to rectangular plate-bending problems was found in (Antes 1974). A finite element application of B-Splines to hyperbolic problems was achieved by Davies in 1978 (Davies 1978). After that, various groups applied the method to two dimensional rectangular domains. For these types of problems, matrix condition numbers and boundary conditions were not considered problematic (these were not mesh-free finite elements, and it was understood that Dirichlet boundary conditions could be applied directly to repeated knots).

More recently, a few researchers (Hughes, Cotrell, and Bazilevs 2005) have utilized NURBS basis functions in a meshed finite element formulation

(using repeated knots to interpolate geometry). The method simply involves replacing the polynomial interpolants in the trial shape function of equation (3.6) with the NURB bases (see equation (2.5)). The usual finite element discretization procedure (equations 3.9 thru 3.11) then proceeds unmodified.

Some motivations for this formulation seem to be:

1.) H and p refinement are handled easily within the NURBS bases themselves through the mechanism of degree-elevation and knot insertion
2.) The minimal support property of B-Splines guarantees the smallest possible algebraic subspace for approximation (the minimum number of bases to approximate data within a given interval and discretization)
3.) Solution fields may be reproduced on exact geometrical representations.

Some interesting features of this methodology are that the finite element "mesh" is now associated not with nodal degrees-of-freedom, but to the span of knot points. The primary independent field variable (the coefficients of the basis functions) is associated with the control points. Thus when the discrete system equations of this formulation are solved, the solution is given in the form of control point values.

The motivations behind this formulation are very close to those of our own current effort; however, serious problems remain. One primary objection is that a mesh must still be created (this is not a mesh-free technique, but we include it in

this review for completeness and because we may be able to utilize some results in our own work). The use of a NURBS basis reinforces a tantalizing suggestion that perhaps the same functions describing the CAD solid model may somehow be re-used for analysis.

3.2 The method of Shapiro/Höllig

Shapiro et al (Shapiro and Tsukanov 1999b), and later Höllig et al (K. Hoellig, Reif, U., and Wipper 2001) have demonstrated a mesh-free approach that involves B-Splines to interpolate the solution space. In contrast to the traditional finite element mesh which discretizes the physical domain, this approach utilizes a function space whose domain overlaps, but does not necessarily conform to a particular geometric domain as in Figure 3.1.

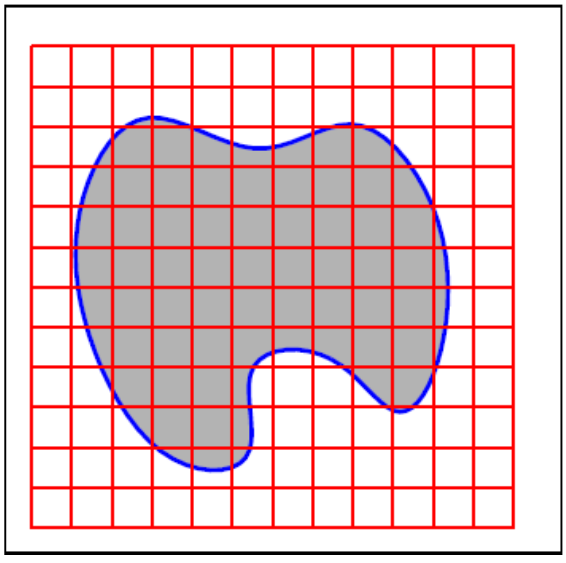


Figure 3.1. Non-conforming tensor-product B-Spline grid overlapping a geometric domain

The solution space is composed with weighting functions to address boundary conditions. Although the basic approach seems to be applicable to all classes of boundary value problems currently addressed by traditional finite element procedures (Hoellig 2003), this dissertation will focus solely on elliptical problems with non-homogeneous Dirichlet boundary conditions (more specifically, it will focus on the majority of problems encountered in the application of elastostatics to engineering structures). The work by Rvachev, Shapiro et al. follows the earlier insight by Kantorovich and Krylov (Kantorovich and Krylov 1958) mentioned in section 2.3, that the solution to differential equations with homogeneous boundary condition

$$u_{|\partial\Omega} = 0 \tag{3.1}$$

can be represented in the form

$$u = \omega\Phi \tag{3.2}$$

where ω is a known function that takes on zero values on the boundary of the domain $\partial\Omega$ and is positive in the interior Ω. Φ is some (unknown) function which is piecewise continuous and differentiable (at least up to the order of the differential equation) over the domain. A simple construction of such functions ω involves calculating the distance from a boundary to all points in the interior of the domain (such functions are guaranteed to vanish at the boundary).

A challenge posed by this approach is finding a way to blend such distance functions from all boundaries in a smooth manner which preserves the distance property. Doing so in a way which preserves the exact original distances to each boundary at all points within the domain is impossible, but this

requirement turns out to be too strict. It suffices to produce a smooth blended function which preserves distance values close to boundaries. An additional challenged is posed in how to apply non-homogeneous boundary conditions.

3.2.1 R-Functions and Solution Structures

The R-Function Method (RFM) of Rvachev (Rvachev 1975) offers a relatively simple algorithm for composing a smooth function for the entire domain which satisfies the requirements stated above, while *approximating* boundary distances close to a boundary. The method works by applying R-functions – the continuous analog of Boolean functions- to implicit functions representing distances to each boundary segment, *i* of the form $\omega_i = \text{f(x,y,z)} \geq 0$ (in three dimensions. f(x,y)≥0 suffices for two). Although there are apparently many families, or branches, of R-functions, the more common ones explored by Rvachev et al are of the form

$$\begin{aligned} \omega_1 \wedge_\alpha \omega_2 &\equiv \frac{1}{1+\alpha}\left(\omega_1 + \omega_2 - \sqrt{\omega_1^2 + \omega_2^2 - 2\alpha\omega_1\omega_2}\right) \\ \omega_1 \vee_\alpha \omega_2 &\equiv \frac{1}{1+\alpha}\left(\omega_1 + \omega_2 + \sqrt{\omega_1^2 + \omega_2^2 - 2\alpha\omega_1\omega_2}\right) \end{aligned} \tag{3.3}$$

where $\alpha(\omega_1, \omega_2)$ is an arbitrary function such that $-1 < \alpha(\omega_1, \omega_2) \leq 1$. According to Rvachev, "…the precise value of α may or may not matter, and often it can be set to a constant. For example, setting $\alpha = 1$ yields the functions min and max respectively, but setting $\alpha = 0$ results in much nicer functions that are analytic everywhere except $\omega_1 = \omega_2 = 0$…" (Rvachev 1975) (i.e.: the boundary). Figure 3.2 shows example of the method used to apply a homogeneous Dirichlet

boundary condition to a simple domain. Here a function ω is created via the Boolean operation:

$$\omega = \omega_1 \Lambda_0 (-(\omega_2 \Lambda_0 \omega_3)) \tag{3.4}$$

where:

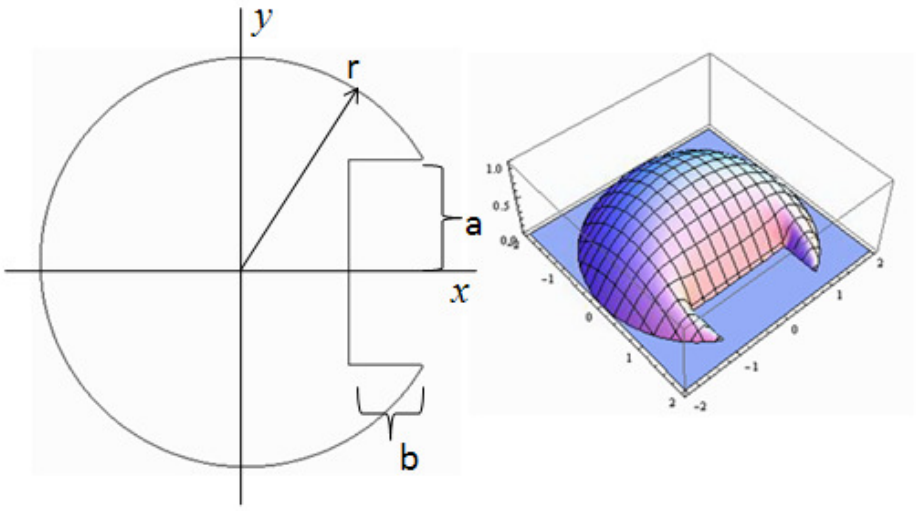


Figure 3.2. R-functions over a simple domain

$$\begin{aligned} \omega_1 &= \frac{1}{2r}(r^2 - x^2 - y^2) \\ \omega_2 &= x - r + b \\ \omega_3 &= \frac{a^2 - y^2}{2a} \end{aligned} \tag{3.5}$$

The Boolean operation is carried out via repeated use of equation (3.4), as it is only a binary operation (the result of the first operation is used as an argument to the equation, along with a new second argument. This new result is used again as

an argument, and so on…). Non-homogeneous Dirichlet boundary conditions could be represented in the form

$$u_{|\partial\Omega} = \phi_0 \tag{3.6}$$

where φ_0 is a piecewise representation of different values on the boundary. Rvachev showed (Rvachev 1975) that a differential equation with such a boundary value description has a solution of the form

$$u = \omega\Phi + \phi \tag{3.7}$$

where φ represents the extension of φ_0 inside the domain. This simple solution structure is capable of solving most problems found in elastostatics. One Rvachev's innovations was in developing an automatic construction of such a function φ given its piecewise boundary description φ_0:

$$\phi = \frac{\sum_{i=1}^{m} \phi_i \prod_{j=1 \neq i}^{m} \omega_j^{\mu_j}}{\sum_{i=1}^{m} \prod_{j=1 \neq i}^{m} \omega_j^{\mu_j}} \tag{3.8}$$

The exponents μ_j control the behavior of the interpolating function at the boundary intersections. When $0 \leq \mu_j \leq 1$, the interpolant is not differentiable at the jth intersection. "Values $\mu_j > 1$ assure that the interpolant is differentiable $\mu_j - 1$ times at the jth intersection, but it has a flat spot there" (Rvachev 1975).

The system equations are generated on a non-conforming, structured grid. Thus the trial solution (2.11) is independent of the geometric domain. Shapiro uses *n* x *n* grids of B-Splines (often bi-cubic) which cover the entire domain. The

equations of such a grid may be obtained by equation (2.16 b). Using the substitution:

$$\mathbf{A} = \nabla^{(p)}(\omega \mathbf{N}) \tag{3.9}$$

This results in:

$$\int_{T_i} \mathbf{A}\mathbf{A}^T \Phi dT_i = -\int_{T_i} \mathbf{A}\nabla^{(p)}(\phi) dT_i + \int_{T_i} \omega \mathbf{N} b dT_i + \omega \mathbf{N} f \tag{3.10}$$

which can be written in algebraic form:

$$\mathbf{K}\mathbf{x} = -\mathbf{r}_\varphi + \mathbf{r}_b + \mathbf{r}_f \tag{3.11}$$

As can be seen in equation (3.10), generating the matrix terms K, $\mathbf{r}_\varphi$, $\mathbf{r}_b$, and $\mathbf{r}_f$ involves differentiation, followed by integration of each term. It is worth pausing a moment here to note that the term $\mathbf{r}_\varphi$ is unique to the R-function method. It represents a body load produced by the imposition of the displacement field φ, which is required in order to comply with the boundary condition, φ_0.

The differentiation step involved in this mesh-free technique is more challenging than in the case of traditional finite element approaches because the weighting function ω depends on the implicit form of the boundary, as well as the actual boundary condition (although this is possible with symbolic algebra software, the resulting polynomial expression for ω and φ could become prohibitively large). The differentiation could be done numerically, but Shapiro states that such approaches "…are not sufficiently accurate for most applications especially when high order derivatives have to be computed." He suggests using *automatic differentiation* (Shapiro and Tsukanov 1999b). Several open source automatic differentiation codes are available (see, for example, FABDAB++ at

http://www.fadbad.com/fadbad.html), and they are also fairly simple to write and to implement (an introductory explanation can be found at http://en.wikipedia.org/wiki/Automatic_differentiation).

The integration step is analogous to traditional finite elements in that the solution must be integrated numerically over each B-Spline grid cell. Once again, however, an additional challenge is posed by the fact that the grid does not necessarily conform to the geometric domain. This means that code must be written to determine how and where the domain intersects grid cells. If Gaussian Quadrature is used for numerical integration, then care must be taken to place quadrature points in the correct locations for such cells (Shapiro and Tsukanov 1999b). Once differentiation and integration has been performed over the entire domain, an algebraic equation like that of equation (3.63) must be solved. If B-Splines are used over a uniform grid (possibly sub-divided around small features), such systems will be sparse and can be solved with familiar public-domain code specially designed for that purpose (UMFPACK and SuperLU, for example).

Although Shapiro and Rvachev use B-Spline basis functions to solve for the unknown field Φ (equation (3.62)), they do not discuss any issues or limitations of this choice on the solution. In particular, they do not address the stability or order of convergence with this choice of basis. They stress that B-Spline bases are just one valid choice, and their approach could easily accommodate others (Shapiro and Tsukanov 1999b),.

In contrast, Höllig points out that the choice of B-Spline bases for arbitrary domains (although otherwise advantageous) is in general potentially

unstable (Hoellig 2003). This is due to the fact that outer splines – those splines whose support overlaps the geometric domain in less than one complete grid cell – yield poor approximations in those regions and lead to high matrix condition numbers (in the resulting algebraic system of equations Ax=b, A has a condition number proportional to h^{-n}, where n is the order of the differential equation). Before revealing Höllig's solution to this problem, we introduce his notation, as well as a more detailed description of the B-Spline basis functions used by both Höllig and Rvachev/Shapiro. Both teams of researchers chose bi-variate (for the case of 2D domains – tri-variate for 3D) tensor product B-Splines as their bases.

3.2.2 B-Spline Conventions and Notations

It is worth briefly describing some notational conventions used by Shapiro and Hoellig et al, because some of the same forms and conventions are used in the current investigation. The notational convention chosen by both teams is shown below

$$u = \omega\Phi = \omega\sum_{i=1}^{n} C_i \chi_i \quad \text{(Rvachev/Shapiro)} \tag{3.12}$$

$$u_h = \sum_i c_i B_i \quad \text{(Hoellig)} \tag{3.13}$$

The subscript, h in (3.13) refers to the dependency of the solution on uniform grid width, h. However, Höllig allows more freedom in the construction of such functions (Hoellig 2003) (he constructs surfaces algorithmically without resorting to polynomial inequalities). Note, however in both cases, only one index is used as in the univariate case. With this notational shorthand, the coefficients C_i and c_i in equations (3.12) and (3.13), respectively represent a

coefficient at grid location *i*. This follows degree-of-freedom numbering conventions in traditional finite element applications, but this notational shorthand warrants further explanation. The spline χ_i or N_i actually refers to the scaled tensor product B-Spline translate (and must therefore be defined on a uniform grid. This is where we break from Höllig's formulation in that our basis is defined over non-uniform non-vectors) whose support begins at the single-index grid location *i*. Höllig offers a more specific notational short-hand (Hoellig 2003). He writes a bi-variate tensor product B-Spline over uniform grid-width h, and degree n as

$$N^n_{k,h}(x) = N^{n_1}_{k_1,h}(x_1) N^{n_2}_{k_2,h}(x_2) \tag{3.14}$$

where k_1 and k_2 are the spline indices in each of the variable directions. x_1 and x_2 are the two independent spline variables (which span the support of b), and n_1 and n_2 are the univariate spline degrees. However, to keep things simple Höllig always uses the same spline degree in all directions. Finally, to approximate surfaces over x_1,x_2 with a uniform grid of width h, the spline in (3.14) must be scaled and translated. That is to say, each univariate spline must be scaled and translated according to

$$N^{n_\nu}_{k_\nu,h}(x_\nu) = N^{n_\nu}(x_\nu / h - k_\nu) \tag{3.15}$$

Thus, tensor-product spline $b^n_{k,h}$ has positive support on $kh + (0, n+1)^2 h$ and vanishes outside this square. Again, the index *k* on the left hand side of equation (3.15) is a scalar integer-numbering of grid locations (these usually represent the lower-left hand side of the support of B-Spline *k* in a 2D lattice). Finally, Höllig

dispenses with the subscript *h*, superscript *n*, and independent variable *x* in the left hand side of equation (3.15) when these are either irrelevant, or clear from the context. Thus, the appearance of equation (3.13). Although uniform splines, as defined in (3.15) are *not* used in this investigation, the notational conventions will remain largely unchanged. In particular, we refer to B-Splines according to the "lower left" index of their support, which in our case, correspond to a knot tensor product (more will be explained in Chapter 4). With these explanations out of the way, we introduce Höllig's weB-Splines, which effectively solve the stability problem.

3.2.3 weB-Splines

For an outer index $j \in J$ (a spline for whom the domain overlaps less than one full grid cell) let $I(j) = l + \{0,...,n\}^m \subset I$ be an m-dimensional array of inner indices (splines for whom at least one full grid cell is fully contained within the domain) closest to j, assuming that h is small enough so that such an array exists. Moreover, denote by

$$e_{i,j} = \prod_{\nu=1}^{m} \prod_{\substack{\mu=0 \\ l_\nu+\mu\neq\nu}}^{n} \frac{j_\nu - l_\nu - \mu}{i_\nu - l_\nu - \mu} \tag{3.16}$$

The values of the Lagrange polynomials associated with I(j) and by J(i) the set of all j with $i \in I(j)$. Then, the weB-Splines defined by

$$B_i = \frac{w}{w(x_i)}\left[b_i + \sum_{j\in J(i)} e_{i,j} b_j \right] \tag{3.17}$$

form a stable finite element basis for the domain *D*. The word "web" in weB-Splines is an acronym for "weighted, extended, B-Splines" – so named because

the second term in equation (3.17) represents an extension of the familiar (weighted) B-Spline basis. This formulation effectively "glues" outer B-Splines to inner B-Splines close to the boundary (the difference in their indices usually being no more than 1), thus extending the support of outer splines and ensuring a stable basis. A useful consequence of this is computational efficiency and performance. The effort that goes into calculating the second term is minimal and occurs during matrix assembly. This leaves a system of equations whose degrees of freedom span the index set *I* (inner B-Splines only), instead of *I*+*J*.

3.2.4 Blended Distance Functions

WeB-Splines are the chief innovation introduced by Höllig et al. Another one is the introduction of "blended distance functions" (Hoellig 2003) to serve as weight functions for composite parametric boundaries. Höllig uses splines to construct such functions (when the boundary is given as piecewise parametric functions, which is usually the case in Engineering applications) utilizing the following definition

$$\sum_k N_k^{'}(x) w_k(x) \propto dist(x,\Gamma), \quad \Gamma = \cup_l \Gamma_l \tag{3.18}$$

Where the subscript l ranges over each smooth piecewise boundary Γ_l that encloses the domain. Thus Γ is the union of all such smooth components. The B-Splines denoted by $N_k^{'}$ are all relevant uniform B-Splines of gird width h'. Here $dist(x,\Gamma)$ is a function which returns the distance from point x to the boundary Γ within a some boundary strip $\delta > 0$. The boundary strip should be chosen small enough so that it remains smooth (usually this means that it should be chosen

smaller than the minimal radius of curvature $1/\kappa$. On the other hand, the strip should not be so narrow that the derivatives of the weight function become excessively large. The w_k are continuous functions which, for some δ satisfy

$$w_k(x) \propto dist(x, \cup_{d_{k,l} \le \delta} \Gamma_l), \quad x \in \text{supp } N_k' \tag{3.19}$$

where $d_{k,l} = dist(\text{ supp } N_k', \Gamma_l)$. Here supp N_k' denotes the support of N_k'. The w_k can be defined separately for each B-Spline support $N_k' = kh' + [0, n+1]^m h'$. The apostrophe in all cases indicates that the basis (and associated grid) being used to construct the weight functions is not necessarily the same as the one being used to approximate the solution space. In fact, Höllig states "usually h'>>h" (Hoellig 2003)).

Chapter 4

A NOVEL MESH-FREE FE FRAMEWORK

The current work investigates the feasibility of implementing an extension of the mesh-free methodology introduced by Shapiro and Höllig for the analysis of mechanical contact problems. The primary thesis of this investigation is to prove that such a framework results in an analysis system with the following properties:

1. Increased CAD interoperability over traditional mesh-based systems. Although this feature of the framework will not be explicitly tested in this investigation, it should suffice to demonstrate that the finite element data structures closely conform to those of trimmed surfaces found in most CAD systems. All curves and surfaces (boundaries) have an exact representation in the system equations and there is no need for a mesh.
2. The resulting system of equations produces results with accuracy and stability comparable to that of mesh-based systems for given grid widths and polynomial orders, but utilizing much smaller solution sub-spaces.

To this end, a suite of code was written in Mathematica® to interpolate a solution field over simple 2-dimensional solid domains in plane stress with the aid of a non-conforming tensor B-Spline grid as shown in Figure 3.1. The system is restricted to two spatial dimensions for expediency, but the extension to 3 dimensions is straightforward and left for future work.

4.1 The B-Spline basis

Instead of the weB-Splines of Höllig et al, the following basis is introduced:

$$\mathrm{B}_h(\mathbf{x},\mathbf{t}) = \frac{wN_h(\mathbf{x},\mathbf{t})}{w_h \left\| N_h(\mathbf{t}) \right\|_\infty} \tag{4.1}$$

where $b_i(\mathbf{t},\mathbf{x})$ is the field of non-rational tensor-product B-Splines of dimension two given by:

$$\begin{aligned} & N_h(\mathbf{x},\mathbf{t}) = N_{i,p}(x,s) \otimes M_{j,o}(y,t) \\ & , \quad s \in \{s_0, s_1, ..., s_n\}, \quad t \in \{t_0, t_1, ..., t_m\} \end{aligned} \tag{4.2}$$

where

$$p, o \in \{i+1, i+d+1\}, \{j+1, j+d+1\}$$

Here, d is the B-Spline degree and h is a single index that maps to indices of the outer product $h \rightarrow \{i \otimes j\}$ where $N_{i,p}$ and $M_{j,o}$ are the unidimensional B-Splines for the x and y-directions, given in equation (2.5) (non-rational and with uniform weighting), and s and t are the knot vectors for each spatial direction, respectively. The list **t** corresponds to some contiguous re-ordering of the two-dimensional knot space implied by the outer product (i.e., $\{\{s_0,t_0\},\{s_0,t_1\},\{s_0,t_2\},...,\{s_n,t_0\},\{s_n,t_1\},\{s_n,t_2\},\ldots\}$.

Although not necessary for the construction of tensor-product B-Spline surfaces, this indexing scheme allows for flexibility in controlling various properties of the assembled system matrices (such as bandwidth). Thus, the full flexibility and approximation power of the Cox-DeBoor B-Spline formulation is

utilized (knot insertion, repetition, non-uniformity, etc…). Note that the tensor indices *i* and *j* are replaced with a single index *h*. This is a quite arbitrary numbering scheme, but must span all bases used in a given discretization.

In this investigation, the span $h \subset H$ refers to all bases whose support intersects the bounding box containing the domain (the minimum and maximum *X* and *Y* points), as in Figure 4.1a, as opposed to the smaller set *H'* which represents basis functions whose support only overlap the geometry itself (Figure 4.1b). In Figures 4.1a and 4.1b, small blue circles are placed at the lower left (the 'origin') of each basis support to more easily identify it (in a bi-quadratic B-Spline grid). Figure 4.2 shows a typical basis numbering scheme.

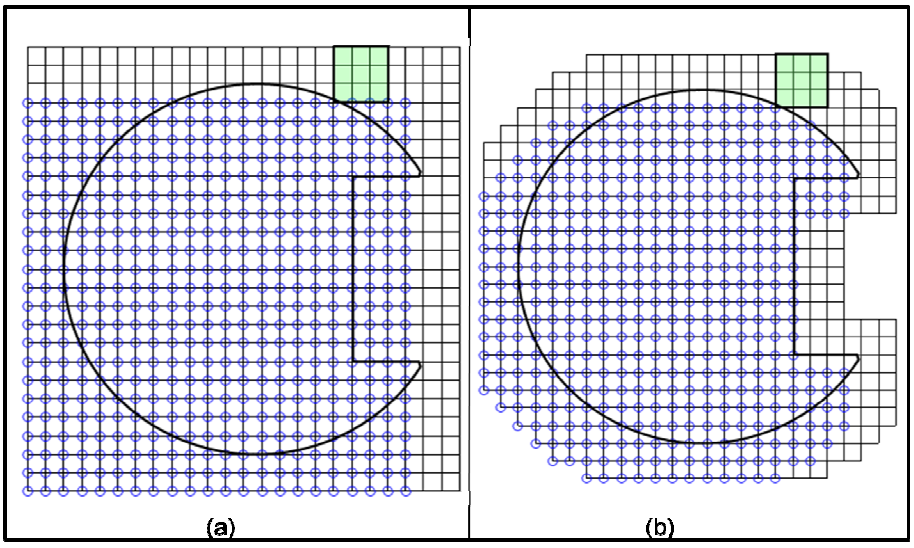


Figure 4.1. Bi-quadratic B-Spline gird over a domain showing two cases: (a) Basis spans entire domain bounding box, (b) Basis spans only points of domain which intersect basis. A single basis support is highlighted in light green

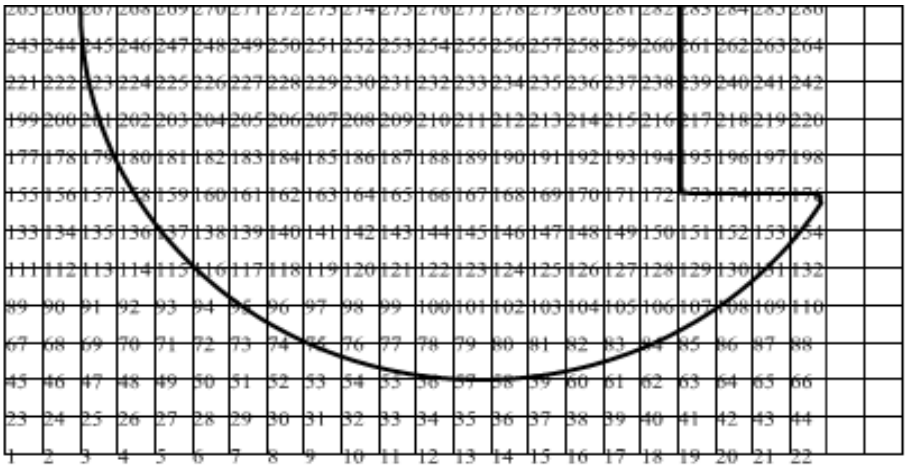


Figure 4.2. Typical bais numbering scheme. Numbers represent basis indices, *h*

Although *H>H'*, the larger set provides a computational convenience, as well as a mirror of the geometric trimmed-curve data structure. In the denominator, $\left\|N_i(\mathbf{t})\right\|_\infty$ represents the 'Infinity Norm' (the maximum absolute value of the B-Spline over the region in which its support overlaps the domain), and w_i is the extremum value the weighting function w takes over the same region.

Such a technique was suggested by Reif (Reif 2006), but no demonstration of it has been found in the literature. A general assessment of this technique to stabilize the FE basis (i.e.: reduce the range of possible condition number) is beyond the scope of this investigation, but the resulting condition number for all test cases will be reported. For a simple example of such a basis normalization in one dimension, see Appendix B.

4.2 Finite element formulation

All model problems in this investigation were 2D plane stress problems. The governing weak form for a state of plane stress is:

$$\int_{\Omega} \boldsymbol{\varepsilon}^{T} \boldsymbol{\sigma} dV = \int_{\Omega} \mathbf{u}^{T} \mathbf{r}_b dV + \int_{\Omega} \mathbf{u}^{T} \mathbf{r}_f dV \tag{4.3}$$

where:

$$\boldsymbol{\varepsilon} = \begin{Bmatrix} \varepsilon_{xx} \\ \varepsilon_{yy} \\ \varepsilon_{yx} \end{Bmatrix}; \boldsymbol{\sigma} = \begin{Bmatrix} \sigma_{xx} \\ \sigma_{yy} \\ \sigma_{yx} \end{Bmatrix} \tag{4.4}$$

are the vectors of planar stress and strain. The vector **d** represents a vector field of x and y planar displacements. Finally, the vector $\mathbf{r}_b$ represents planar body forces and $\mathbf{r}_f$ surface tractions.

Tensor-product B-Splines, B_i will be used to interpolate a displacement field in the plane:

$$\begin{aligned} u(x, y) &= \sum_{m} B_i \mathbf{u}_i \\ v(x, y) &= \sum_{m} B_i \mathbf{v}_i \end{aligned} \tag{4.5}$$

where *m* spans the squared order of the B-Splines (degree + 1)*(degree+1) of B-Splines. Thus, the interpolated displacement field, $\mathbf{u}^h$, can be written:

$$\mathbf{u}^h = \mathbf{B}^T \mathbf{d} \tag{4.6}$$

The superscript denotes a uniform grid of cell width *h*. The strain field is calculated according to:

$$\boldsymbol{\varepsilon}^h = \begin{Bmatrix} \dfrac{\partial u}{\partial x} \\ \dfrac{\partial v}{\partial y} \\ \dfrac{1}{2}\left(\dfrac{\partial u}{\partial y} + \dfrac{\partial v}{\partial x}\right) \end{Bmatrix} \tag{4.7}$$

Substituting (2.18) into (4.5) and (4.6) yields:

$$\boldsymbol{\varepsilon}^h = \mathbf{A}\mathbf{d} \tag{4.8}$$

Also, the isotropic constitutive law for plane stress is:

$$\begin{Bmatrix} \sigma_{xx} \\ \sigma_{yy} \\ \sigma_{yx} \end{Bmatrix} = \frac{E}{1-\nu^2}\begin{bmatrix} 1 & \nu & 0 \\ \nu & 1 & 0 \\ 0 & 0 & \dfrac{1-\nu}{2} \end{bmatrix}\begin{Bmatrix} \varepsilon_{xx} \\ \varepsilon_{yy} \\ \varepsilon_{yx} \end{Bmatrix} \tag{4.9}$$

or:

$$\boldsymbol{\sigma}^h = \mathbf{C}\boldsymbol{\varepsilon}^h \tag{4.10}$$

Substitution of (4.10), (4.8), and (4.6) back into (4.3) yields:

$$t\left(\int_\Omega \mathbf{A}^T\mathbf{C}\mathbf{A}\,dA\right)\mathbf{d} = t\int_\Omega \mathbf{B}^T\mathbf{r}_b + \int_\Gamma \mathbf{B}^T\mathbf{r}_f\,dS \tag{4.11}$$

where t is the planar thickness (taken to be unity). The first two terms of (4.11) are generated on an element basis. A typical element using bi-quadratic B-Spline basis functions is shown in Figure 4.3b. Figure 4.3a shows the element grid corresponding to the basis space depicted in Figure 4.1a. Elements highlighted in blue represent 'inner' elements (those whose cells intersect the domain completely). Elements highlighted in yellow represent 'outer' element (those whose cells intersect the domain only partially). Clear (or white) elements do not

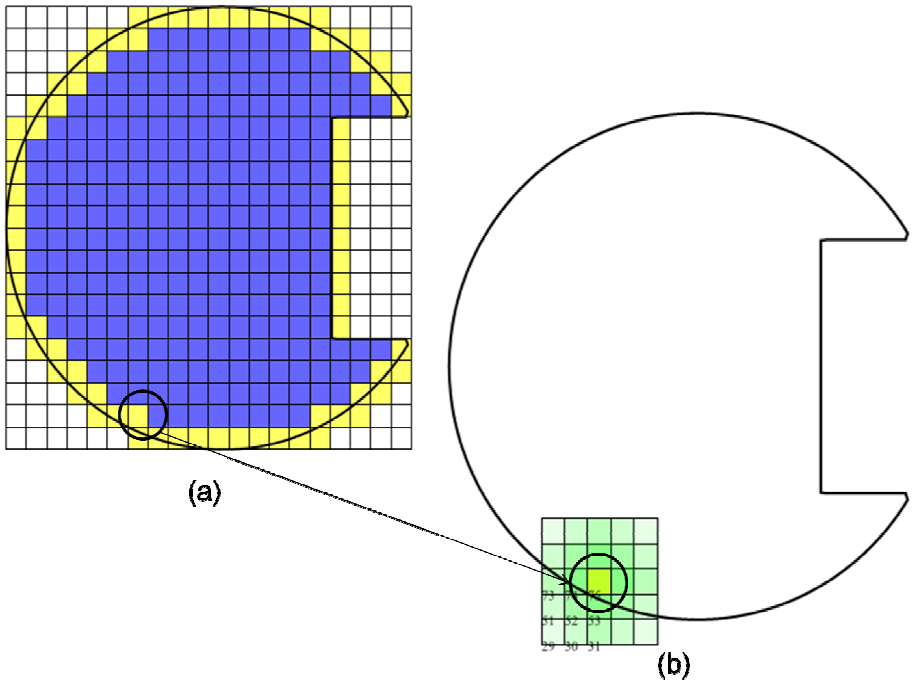


Figure 4.3. (a) Inner (blue), boundary (yellow), and outer (white) elements spanning the domain (element support not shown); (b) Element 27 showing basis support (green and numbered at lower left)

intersect the boundary at all. These elements would result in zero entries on the diagonal of the assembled stiffness matrix, and so are replaced with average stiffness values (or '1' if the matrix diagonal is normalized to 1), and corresponding force terms left zero. Figure 4.3b shows element number 27 (yellow) with its corresponding B-Spline support (numbered at lower left).

Note there are nine basis functions for each element in a bi-quadratic B-Spline grid. Note also that the basis support extends beyond the element. This illustrates the 'minimal support' property enjoyed by B-Splines. Equation (4.11)

may be written in the form of the familiar matrix equation (2.20) and assembled in the usual way.

4.3 Boundary Conditions

Boundary conditions are all contained within the solution structure (3.7), $u=\omega\Theta+\varphi$, where ω is a polynomial weighting function corresponding to a Boolean combination of one or more implicit surfaces, utilizing the R-function technique described in section 3.2.1, and φ is either zero, or constructed according to (3.8) for non-zero applied displacements. Only Dirichlet-type boundary conditions are considered (see section 3.3), and so the weighting functions are quite simple. An example showing two planar cylinders in contact is shown in Figure 4.4. The green arrows indicate directions of a zero displacement condition on the symmetry surfaces.

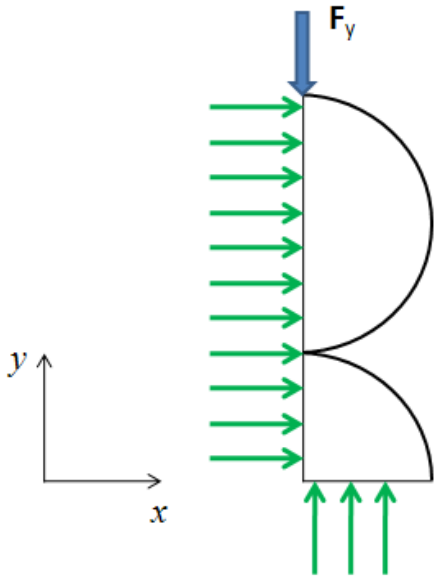


Figure 4.4. Two cylinders in contact and constrained by symmetry

The symmetry boundary conditions shown in Figure 4.4 can be easily represented with weighting functions:

$$w_x(x, y) = x$$
$$w_y(x, y) = y \tag{4.12}$$

with w_x representing a weighting for the x-displacement vector, and w_y representing a weighting function for the y-displacement vector. All but one of the validation problems in this investigation made use of a symmetry plane as in Figure 4.4, and so made use of simple R-function expressions of the form (4.12). R-functions for other boundaries were of the form:

$$w(x, y) = \sigma(x, y) \tag{4.13}$$

where $\sigma(x, y)$ is an implicit function describing the boundary curve on which the boundary condition is applied. The fourth validation problem, however, is not symmetric. That problem involved fixing the displacement of four circular domains (see Figure 4.21) in their interior. Since no boundary curve can be utilized to generate an expression like (4.13), two points are arbitrarily selected within the interiors of each circular domain to receive zero displacements. Thus the two singular 'boundaries', σ (in the sense of a boundary value differential equation) in each circular domain, *i* are defined as:

$$\sigma_i^1 = (x - P_{xi}^1)^2 + (y - P_{yi}^1)^2$$
$$\sigma_i^2 = (x - P_{xi}^2)^2 + (y - P_{yi}^2)^2 \tag{4.14}$$

These are then composed according to:

$$\omega_i = \sigma_i^1 \wedge_0 \sigma_i^2 \tag{4.15}$$

and the final function is obtain by composing ω for all domains:

$$w = w_x = w_y = (\omega_4 \wedge_0 (\omega_3 \wedge_0 (\omega_1 \wedge_0 \omega_2))) \tag{4.16}$$

A contour plot of this function is shown in Figure 4.5.

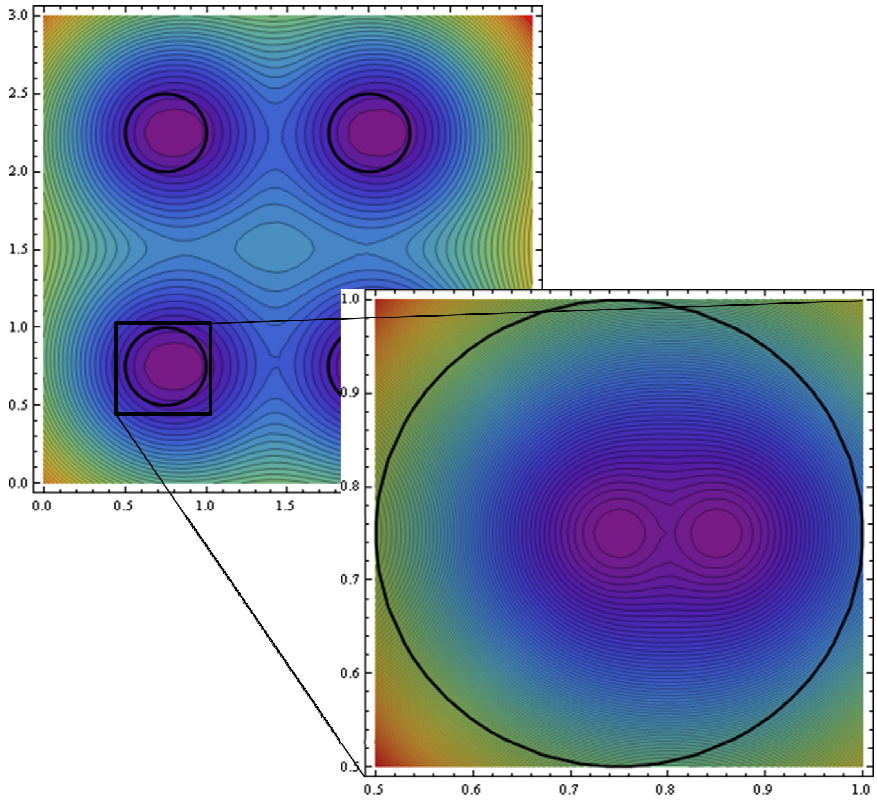


Figure 4.5. R-function contours for indeterminate beam problem

Evaluation of the system equations (4.10) involves a differentiation and an integration for every grid cell. As the model problems addressed in this investigation are relatively small (always less than ~2000 degrees of freedom), the differentiation steps are handled by Mathematica's robust symbolic engine (later studies may utilize these algorithms within a compiled program language like

C++, in which case the techniques of Automatic Differentiation described in section 3.2.1 would be utilized).

4.4 Grid cell classification and Gauss point manipulation

An algorithm was written to determine the intersections between the non-conforming grid and the domain. This is done in order to identify elements whose integration points will need to be moved (transformed), but also to identify bases which will require normalization. Figure 4.3a shows how a tensor-product B-Spline grid is partitioned into inner and outer cells. The cells highlighted in yellow represent outer cells. The blue cells are inner cells. Integration is carried out with standard quadrature rules (see (Bhatti 2006) or (Hoellig 2003) for further explanations). For instance, in two dimensions, an integral over an inner cell may be calculated as:

$$\int_Q f \approx h^2 \sum_{\nu,\mu} \gamma_\nu \gamma_\mu f(t_\nu, t_\mu) \tag{4.17}$$

where *f* depends on the basis functions, coefficients, and other parameters. Q represents the domain of a single grid cell. γ_ν and γ_μ are quadrature weights in each spatial dimension of the cell. $t' = lh = (t_\nu, t_\mu)h$, and *h* is the grid cell size. However, the outer cells intersect only a portion of the domain. So the quadrature points must be adjusted in some way to reflect this.

Höllig (Hoellig 2003) recommends a subdivision strategy, using information about the critical points of the intersection between the domain and outer grid cells ($Q' \cap D$). As an example, if three-point quadrature is used (as would be the case if 2rd-degree basis functions are used), point spacing is

prescribed as $t_\nu = (1/2 \pm 1/10\sqrt{15})h$. Thus the points shown in the lower right in Figure 4.6 are moved (transformed) to lie entirely within the domain, D in such a way that the spacing ratio remains constant while conforming to the domain. The transformed integral from a square grid as shown in the lower right of Figure 4.6 (inset) to its final configuration may be represented as:

$$\int_{Q'} \varphi \approx (\beta - \alpha) \sum_{\nu,\mu} \gamma_\nu \gamma_\mu (g(t_\nu') - f(t_\nu')\varphi(t_\nu', t_\mu') \tag{4.18}$$

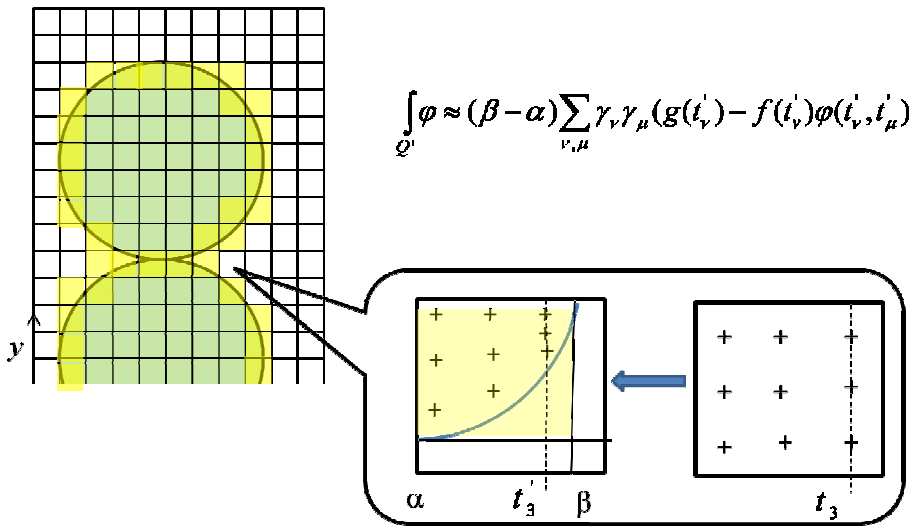


Figure 4.6. Quadrature spacing modified for bi-quadratic boundary element

4.5 Contact algorithm

The basic unilateral contact problem for two bodies, $\mathbf{\Omega}^1$ and $\mathbf{\Omega}^2$ as shown in Figure 4.7, may be described in the simplest terms as one in which the usual constitutive relations of elasticity theory are augmented by the constraint:

$$(\overline{\mathbf{x}}^2 - \mathbf{x}^1) \cdot \mathbf{n}^1 \geq 0 \tag{4.19}$$

where

$$\mathbf{x}^1, \mathbf{x}^2 \subset \Gamma^1, \Gamma^2$$

represent the current positions (original position plus displacement) on bodies 1 and 2, respectively that lie within adjacent boundary segments Γ^1, Γ^2.

The determination of these segments (surface segments that are potentially in contact) is the first non-trivial task in determining how the two bodies interact. The positions $\bar{\mathbf{x}}^2$ represent locations "closest to" corresponding positions $\mathbf{x}^1$. Thus, identifying locations of potential contact usually involves first identifying boundary segments Γ^1, Γ^2 that lie within some hypothetical volume of influence whose size is some fraction of volumes $\mathbf{\Omega}^1$ and $\mathbf{\Omega}^2$ and includes body bodies. Then locations $\mathbf{x}^1$ are chosen according to the numerical discretization of $\mathbf{\Omega}^1$ and $\mathbf{\Omega}^2$ (typically, integration points or nodes). Finally, points $\bar{\mathbf{x}}^2$ are calculated according to

$$\left\| \bar{\mathbf{x}}^2 - \mathbf{x}^1 \right\| = \min_{\mathbf{x}^2 \subseteq \Gamma^2} \left\| \mathbf{x}^2 - \mathbf{x}^1 \right\| \tag{4.20}$$

Following Wriggers (Wriggers 2002), an algorithm for performing general unilateral contact problems can be split into the following phases:

- Contact Search
- Contact Detection
- Application of Constraint (4.19)

In small strain problems, the first two tasks may be performed once, followed by an iterative solution (Newton-Raphson for example) in which

elements involved in contact (the elements detected in the second phase) are monitored to determine whether condition (4.19) is satisfied. If (4.19) is not satisfied, Penalty coefficients are added to the tangent stiffness matrix, another iteration is performed, and the condition is checked again. In large strain problems, at least the last two phases (and possibly all three) must be repeated after the displaced configuration has been determined for each load increment. The condition (4.19) has units of displacement and represents the relative normal displacement between $\mathbf{\Omega}^1$, and $\mathbf{\Omega}^2$. In fact, the "gap" between the two bodies is defined as

$$g = (\bar{\mathbf{x}}^2 - \mathbf{x}^1) \cdot \mathbf{n}^1 + g_0 \tag{4.21}$$

where g_0 is the initial gap between the two bodies expressed by

$$g_0 = (\bar{\mathbf{x}}_0^2 - \mathbf{x}_0^1) \cdot \mathbf{n}_0^1$$

Where the 0 subscript denotes the undeformed configuration. Thus the second and third phase of the algorithm is often referred to as "gap monitoring".

4.5.1 Mesh-free contact search and detection

Some of the tasks outlined above in constructing a unilateral contact solution are simplified within the mesh-free finite element framework investigated by Shapiro/Hoellig (Hoellig 2003; Shapiro and Tsukanov 1999). This is mainly due to the fixed solution grid, which almost immediately suggests an efficient means of identifying potential contact regions $\mathbf{\Gamma}^1$ and $\mathbf{\Gamma}^2$. In fact, one common means of determining the contact regions in traditional meshed approaches involves bucket sorting – or grid hashing, in which a uniform (or

Octree) grid is constructed, and points within $\mathbf{\Omega}^1$ and $\mathbf{\Omega}^2$ are hashed to determine whether they lie within the same cell. The entire collection of such points make up regions $\mathbf{\Gamma}^1$ and $\mathbf{\Gamma}^2$.

The tensor-product grid structure (the knot-span) of the B-Spline basis support provides a logical and convenient grid over which to perform such an operation. The hashing procedure works by evaluating a function, H(x,y) which maps domain coordinates (pairs of real numbers) to an integer representing those coordinates' position within some single continuous enumeration of the grid cells.

For the case of the mesh-free method of Shapiro/Hoellig, such a hashing procedure *is already performed* to identify grid cells which overlap the boundary in non-contact situations (the B-Spline grid points that overlap the domain are classified into "inner", "outer", and "boundary" cells)! Thus, with this existing architecture, a simple four-step procedure for identifying cells (elements) which are potentially in contact presents itself:

- Overlay a single B-Spline support grid over n domains. Use min/max bounding box over both domains to determine grid corners
- Determine inner, outer, boundary cells as usual
- Identify boundary cells $l \in C_n$ common to all domains (or which share an edge)
- Split the grid along bounding boxes defined by the boundary cells of each domain

Performing this procedure results in n grids which provide the support for n matrices (where n is the number of bodies in the analysis), which in turn populate the diagonal of the full assembled system of equations. They are connected only through penalty terms associated with the boundary cells identified in the third step above.

This algorithm is fast and efficient. Figure 4.7 below depicts results for the first three tasks above in a particular example, in which a single grid overlays two bodies potentially in contact. Boundary cells are highlighted in yellow while inner cells are colored blue (outer cells remain uncolored). Highlighted in green are three boundary cells common to both bodies $\mathbf{\Omega}^1$ and $\mathbf{\Omega}^2$. Figure 4.8 shows the fourth step in contact detection – the grid is split along the bounding boxes defined by the boundary cells of each domain. A noteworthy aspect is that potential contact surface determination (Γ^α) is tied to grid size. If a uniform grid size of characteristic length, h is used, all potential contact surface pairs will be separated by an approximate distance $\leq h$. This algorithm returns both the element numbers to be checked for contact, as well as the curves contained (by definition, all boundary elements contain boundary curves).

In all cases, there are only five possibilities for grid cells of different bodies to be considered for interaction: a) boundary cells overlap, b) boundary cells of different bodies come in adjacent pairs (sharing an edge), c) boundary cells on one body are adjacent to inner cells on another, d) inner cells on one body are adjacent to inner cells on another, and e) no potential contacting cells have been found. The first case is shown in Figure 4.7 and Figure 4.8, while an

example of the third case is shown in Figure 4.9. Thus, the third step in contact detection actually involves two steps: First determine if any two bodies share boundary cells. Then determine if any boundary or inner cells of one body are adjacent to either boundary or inner cells of another body.

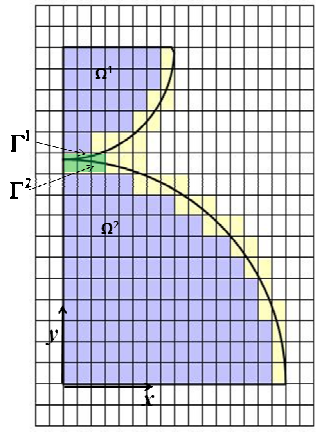


Figure 4.7. Results for the first three steps in contact detection with contact elements highlihgted

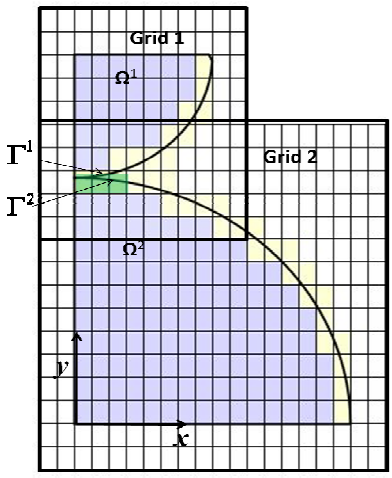


Figure 4.8. Fourth step in contact detection

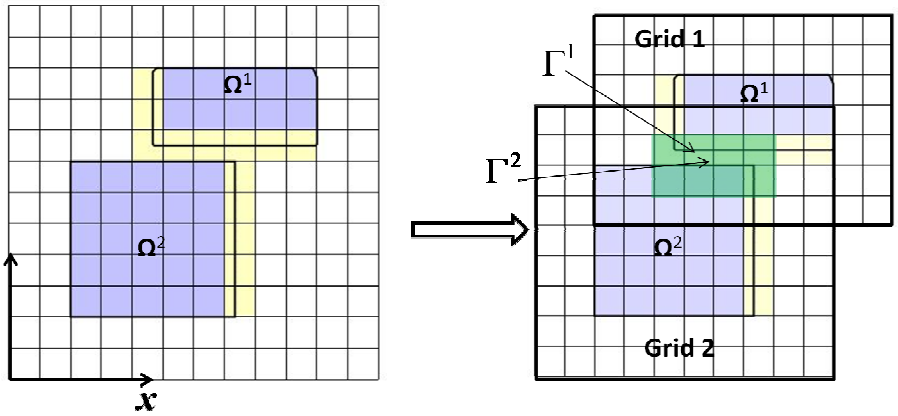


Figure 4.9. An example of steps 3 and 4 showing contact condition C

4.5.2 Application of constraints (gap monitoring and solution)

The familiar weak form of the governing equations of elasticity must be augmented by

$$\int_{\Gamma} \delta(g\mathbf{F}_n)d\Gamma = 0 \tag{4.22}$$

where g is defined by the left-hand side of the gap inequality given previously. Following equations 2.87 and 2.88, this results in the following discretized residual and tangent stiffness terms for the Penalty Method:

$$\mathbf{k}_l^c = \beta \begin{pmatrix} \int_{\Gamma_l^1} \mathbf{B}_l^1 \mathbf{n}_l (\mathbf{n}_l)^T (\mathbf{B}_l^1)^T dA & -\int_{\Gamma_l^1} \mathbf{B}_l^1 \mathbf{n}_l (\mathbf{n}_l)^T (\mathbf{B}_l^2)^T dA \\ -\int_{\Gamma_l^1} \mathbf{B}_l^1 \mathbf{n}_l (\mathbf{n}_l)^T (\mathbf{B}_l^2)^T dA & \int_{\Gamma_l^1} \mathbf{B}_l^2 \mathbf{n}_l (\mathbf{n}_l)^T (\mathbf{B}_l^2)^T dA \end{pmatrix} \tag{4.23}$$

$$\mathbf{F}_l^c = \beta \begin{pmatrix} \int_{\Gamma^1} g_{0l} \mathbf{B}_l^1 \mathbf{n}_l dA \\ -\int_{\Gamma^1} g_{0l} \mathbf{B}_l^2 \mathbf{n}_l dA \end{pmatrix} \tag{4.24}$$

For expediency, some liberties are taken with notation – namely the mixing of vector and indicial notation. For example, the term g_{0l} denotes the scalar 'initial gap' function, g_0 defined in (4.24) for element l. In the program, it is stored as a list of functions. Here $\mathbf{B}_l^n$ is the vector-valued basis for contact element l (as defined in equation (4.1)). Its transpose is given by:

$$(\mathbf{B}^n)_l^T = \begin{pmatrix} B_1^n(x,y), & 0 & , B_2^n(x,y), & 0 & , B_3^n(x,y), \ldots \\ 0 & , B_1^n(x,y), & 0 & , B_2^n(x,y), & 0 & , \ldots \end{pmatrix}_l$$

For two bodies in contact, $\mathbf{n}_l$ is a vector of vector-valued functions containing the outward normals on Γ^1 (the master surface) for contact element l (see equation (4.28).

The quantities $\mathbf{F}_l^c$ and $\mathbf{k}_l^c$ must be calculated for all $l \in C_n^1$ (here, C_n^1 is the subset of indices of contact elements C_n associated with body 1 – arbitrarily chosen as the 'master' set). They are assembled into the global system the usual way, provided that the gap condition (4.19) is satisfied. Thus, within each solution iteration, the gap for each element l is calculated and checked to see if the inequality holds. If not, $\mathbf{F}_l^c$ and $\mathbf{k}_l^c$ are introduced into the system equations.

To explain in more detail how equations (4.23) and (4.24) are calculated in the mesh-free context (in particular, how g_0 and $\mathbf{n}$ are evaluated in the integral), it is helpful to use an example. Consider the two-body problem depicted previously in Figure 4.7 and Figure 4.8. Body 1 is chosen arbitrarily as the 'master', so that elements $l \in \{1,2,3\} \subset \{\{1,2,3\},\{81,82,83\}\}$ are chosen for calculation of the penalty terms (see Figure 4.10). The term $Min(g_0)$ denotes the minimum value

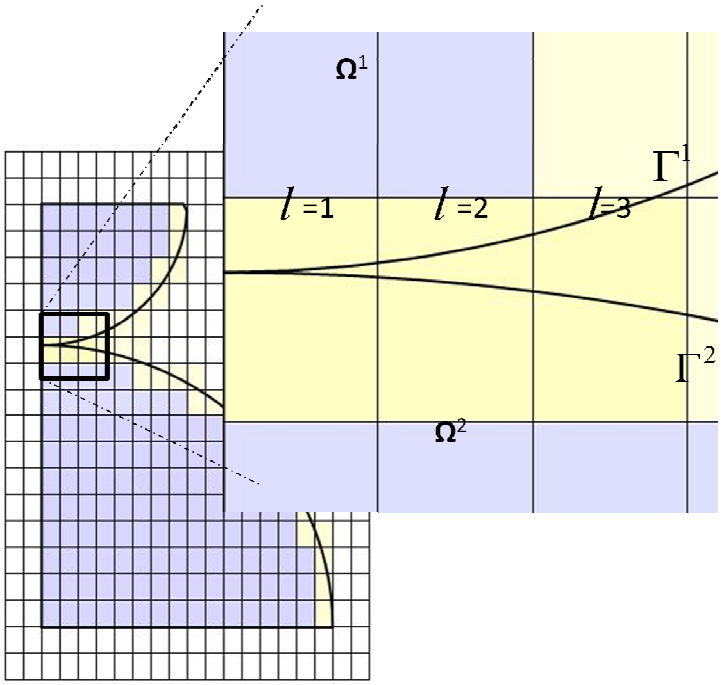


Figure 4.10. Two-body problem showing contact elements and surfaces

taken on by the geometric scalar gap function g_0 over all Γ^1, Γ^2. Bodies 1 and 2 are described by Boolean combinations of implicit functions:

$$\mathbf{\Omega}^1 : \sigma_1^1 <= 0 \cap \sigma_2^1 \geq 0 \cap \sigma_3^1 \leq 0 \tag{4.25}$$

where

$$\sigma_1^1 = x^2 + (y - (R^1 + R^2 + Min(g_0)))^2 - \left(R^1\right)^2$$
$$\sigma_2^1 = x$$
$$\sigma_3^1 = y - (R^1 + R^2 + Min(g_0))$$

and

$$\mathbf{\Omega}^2 : \sigma_1^2 \leq 0 \cap \sigma_2^2 \geq 0 \cap \sigma_3^2 \geq 0 \tag{4.26}$$

where

$$\sigma_1^2 = x^2 + y^2 - (R^2)^2$$
$$\sigma_2^2 = x$$
$$\sigma_3^2 = y$$

Here, R^1 and R^2 give the radii of the upper and lower cylinder represented by $\mathbf{\Omega}^1$ and $\mathbf{\Omega}^2$, respectively. The master and slave boundary curves are those that pass through the elements l (highlighted in green in Figure 4.7and Figure 4.8):

$$\begin{aligned} \Gamma^1 &= \sigma_1^1 = 0 \\ \Gamma^2 &= \sigma_1^2 = 0 \end{aligned} \tag{4.27}$$

The normal direction for each element, $\mathbf{n}_l^1$ is calculated according to:

$$\mathbf{n}_l^1(x, y) = \frac{\nabla\left(\Gamma_l^1(x, y)\right)}{\left\|\nabla\left(\Gamma_l^1(x, y)\right)\right\|} \tag{4.28}$$

An element may contain more than one boundary curve, and so in general Γ_l^1 represents a piecewise continuous implicit function passing through element l (with a minimum of C^0 continuity). An additional step is used to determine the sign of $\mathbf{n}_l^1$:

$$\mathbf{n}^1 = \begin{cases} -\mathbf{n}^1, & \mathbf{p} + \beta\mathbf{n}^1 \subset \mathbf{\Omega}^1 \\ \mathbf{n}^1, & \mathbf{p} + \beta\mathbf{n}^1 \not\subset \mathbf{\Omega}^1 \end{cases}$$

where $\mathbf{p}$ is any point on Γ^1. β is a very small positive number ($0 < \beta << 1$).

In order to use the implicit boundary inequalities, σ_j^α effectively in calculations (for example, in the calculation of contact stiffness, pressure, gap and normals), it is helpful to decompose them into implicit functions of a single variable only to yield piecewise boundary segments. This is done by choosing an

independent variable a priori (and quite arbitrarily, except for such segments that have only one variable). Boundary functions are then redefined as implicit functions of this independent variable over an explicit interval. Thus, for $\mathbf{\Omega}^1$, the entire boundary Γ^1 may be given as:

$$\Gamma^1 = \begin{cases} y + \sqrt{(R^1)^2 - x^2} & ,0 \le x \le R^1 \\ -(R^1 + R^2 + Min(g_0)) & \\ x & ,R^2 + Min(g_0) \le y \le R^1 + R^2 + Min(g_0) \\ y - (R^1 + R^2 + Min(g_0)) & ,0 \le x \le R^1 \end{cases} \tag{4.29}$$

And for $\mathbf{\Omega}^2$:

$$\Gamma^2 = \begin{cases} y - \sqrt{\left(R^2\right)^2 - x^2} & ,0 \le x \le R^2 \\ x & ,0 \le y \le R^2 \\ y & ,0 \le x \le R^2 \end{cases} \tag{4.30}$$

With this piecewise functional definition of boundary segments, the independent variable is the one over which the interval is prescribed, and the dependent one is easily extracted by setting the implicit function equal to zero. To see how this facilitates computations, consider how equation (4.27) is replaced with the much more specific:

$$\begin{aligned} \Gamma^1 = \partial\Omega_1^1 &= y + \sqrt{\left(R^1\right)^2 - x^2} - (R^1 + R^2 + Min(g_0)) = 0, \quad 0 \le x \le R^1 \\ \Gamma^2 = \partial\Omega_1^2 &= y - \sqrt{\left(R^2\right)^2 - x^2} = 0, \quad 0 \le x \le R^2 \end{aligned} \tag{4.31}$$

To be used in calculations, the pair $\{x,f(x)\}$ or $\{f(y),y\}$ must be calculated (the form being determined by which variable is the independent one). In the form given by equation (4.27), there is some ambiguity when doing this for implicit functions containing quadratic and higher terms, but no ambiguity exists

in the form given by (4.31). This functional decomposition is therefore carried out at an early phase of model construction and all subsequent calculations utilize the decomposed piecewise boundary definitions to eliminate such ambiguity. In this example, the independent variable for Γ^1 and Γ^2 is x, so for Γ^1:

$$\{x, f(x)\}_{\Gamma^1} = \{x, \sqrt{\left(R^1\right)^2 - x^2} - (R^1 + R^2 + Min(g_0))\} \tag{4.32}$$

And for Γ^2:

$$\{x, f(x)\}_{\Gamma^1} = \{x, \sqrt{\left(R^1\right)^2 - x^2}\} \tag{4.33}$$

Now expressions like those given in (4.23) and (4.24) can readily be evaluated. Consider, for example, the initial gap, g_0 and the normal $\mathbf{n}^1$. For convenience, an approximation is used to calculate g and g_0. Instead of calculating points $\mathbf{x}^1$ nearest their corresponding points $\bar{\mathbf{x}}^2$, the approximation:

$$\begin{aligned} g &= \{[(\mathbf{x}^2(\mathbf{p}) + \mathbf{u}^2(\mathbf{p})] - [\mathbf{x}^1(\mathbf{p}) + \mathbf{u}^1(\mathbf{p})]\} \cdot \mathbf{n}^1(\mathbf{p}) + g_0 \\ g_0 &= [\mathbf{x}^2(\mathbf{p}) - \mathbf{x}^1(\mathbf{p})] \cdot \mathbf{n}^1(\mathbf{p}) \end{aligned} \tag{4.34}$$

is used, where $\mathbf{p}(x,y)$ is the piecewise continuous set of points on the boundary given by expressions (4.32), and (4.33). In this investigation, these points are selected from the underlying element Gauss points. When these expressions are substituted for $\mathbf{p}$ in (4.34), the initial normal gap may be calculated anywhere along the interval $0 \le x \le R^1$, and this gap may be easily (and quite accurately) monitored as the solution progresses. Continuing with the example,

$$\mathbf{n}^1(x, y) = \frac{\nabla(y + \sqrt{\left(R^1\right)^2 - x^2} - (R^1 + R^2 + Min(g_0)))}{\left\|\nabla(y + \sqrt{\left(R^1\right)^2 - x^2} - (R^1 + R^2 + Min(g_0)))\right\|} \tag{4.35}$$

by substituting (4.31) into (4.27). Or

$$\mathbf{n}^1(x,y)=\left\{\begin{array}{c}\dfrac{x}{\sqrt{(1.-x^2)\left(1+\left(x/\sqrt{1-x^2}\right)^2\right)}}\\ -\dfrac{1}{\sqrt{1+\left(x/\sqrt{1-x^2}\right)^2}}\end{array}\right\}$$

and

$$g_0=\left\{\begin{array}{l}\{x,\sqrt{\left(R^1\right)^2-x^2}\}\\ -\{x,\sqrt{\left(R^1\right)^2-x^2}-(R^1+R^2+Min(g_0))\}\end{array}\right\}^T\cdot\mathbf{n}^1(x,y)$$

Similarly, the term involving the element basis $\mathbf{B}_l^n(x,y)$ in (4.23) is easily evaluated for any and all points along Γ^1 or Γ^2 by substituting the arguments (x,y) by $\{x_i, \Gamma(x_i)\}$ (equations (4.32) and (4.33)). Once $\mathbf{F}_l^c$ and $\mathbf{k}_l^c$ (equations (4.24) and (4.23)) are calculated for all l, they are added to the system equations to solve the penalty problem:

$$\left(\mathbf{K}+\mathbf{k}^c\right)\mathbf{d}=\mathbf{F}+\mathbf{F}^c \tag{4.36}$$

For the two-body problem used in this example,

$$\mathbf{K}=\begin{pmatrix}\mathbf{K}_1 & 0\\ 0 & \mathbf{K}_2\end{pmatrix} \tag{4.37}$$

where $\mathbf{K}_1$ and $\mathbf{K}_2$ are the stiffness matrices associated with $\mathbf{\Omega}^1$ and $\mathbf{\Omega}^2$, respectively after the grid splitting operation described previously. They are assembled in the usual way.

4.6 Validation studies

The mesh-free system under investigation was assessed both for accuracy and efficiency of known contact solutions, as well as robustness of the contact algorithm employed. The validation studies may therefore be categorized as those assessing solution, and those assessing the contact algorithm. General convergence studies were not performed, as Hoellig et al (Hoellig 2003) have already demonstrated B-Spline finite elements to possess the familiar (h^2) rate of convergence for displacements.

4.6.1 Accuracy comparisons

The accuracy of the mesh-free system in structural contact calculations was assessed at a prescribed level of refinement within the predicted Hertzian contact zone. This corresponded to the number of elements across the contact zone (calculated a priori) to achieve a contact pressure (maximum absolute value) within 5 percent of the Hertz solution in a traditionally meshed finite element model. Calculated contact pressure values were compared for all three results (traditional mesh-based solution, mesh-free solution, and analytical solution) for an identical level of refinement for the same problem. In both mesh-based as well as mesh-free solutions, a bi-quadratic basis was used. The commercial software ANSYS v12.1 was used to generate all mesh-based models and their results.

Three simple 2D geometric configurations were selected as model problems. These represent iconic Hertz-type problems in elasticity for which analytical solutions are known. The three configurations selected consist of: 1)

Contact between a rectangular punch and semi-infinite die , 2) Convex contact between cylinders of arbitrary radii, and 3) Convex-Concave contact between a cylinder and a cylindrical groove. These three cases are illustrated in Figure 4.11.

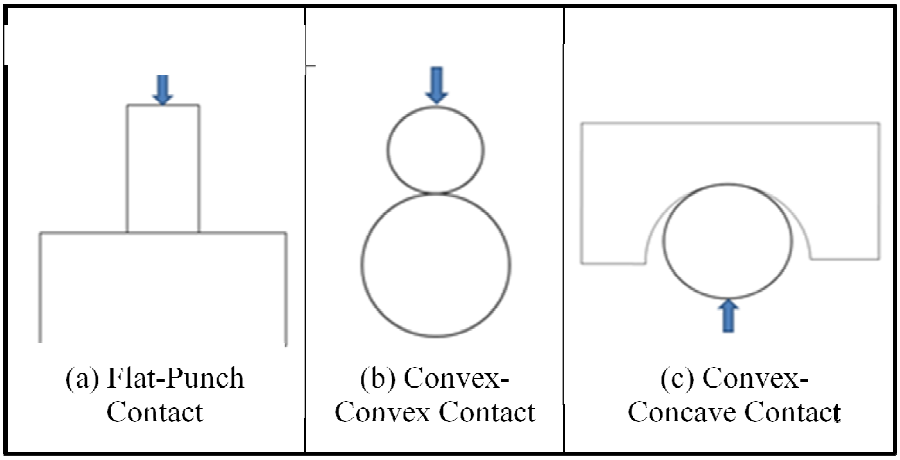


Figure 4.11. Simple validation cases for accuracy

For each test configuration, an ANSYS model was created and the mesh refined until the 5% convergence criterion was met. For all cases, this turned out to be ~five element divisions within the contact zone, a (h=a/5). In each case, the relative error norm $\eta=(\|(u_h - u)/u\|)$ for each solution was reported for the set of points corresponding to the nodes in the traditional mesh-based model. The quantity chosen for comparison is contact pressure, p, on the contact surface which is evaluated from both numerical approaches according to:

$$p = \mathbf{n} \cdot \boldsymbol{\sigma} \cdot \mathbf{n} \tag{4.38}$$

In the mesh-based solution, this pressure represents a "node-averaged" value, where stress values at integration points are extrapolated to nodes and then

averaged. A similar approach is used for the mesh-free solution. Since there are no "nodes" per se, stress values are obtained by "fitting" (In the least squares sense) a surface of degree *d*-1 (where *d* is the degree of the original B-Spline displacement solution) through stress values at integration points of order *p*-1 (where *p*=*d*+1), as well as cell corners. The stress values at such points (called "superconvergent" points in the literature – See (Cook, Malkus, and Plesha 1989)) are calculated according to (4.9) before being fit. Formally, the process of determining the least-squares stress may be stated as: The determination of a B-Spline surface of degree *d*-1, such that:

$$\mathrm{B}'\mathbf{C} = \boldsymbol{\sigma}_s \tag{4.39}$$

where $\boldsymbol{\sigma}_s$ are stresses at superconvergent points and B' is the first derivative of the B-Spline basis used to solve the problem. The coefficients C are calculated so as to solve the system:

$$(\mathbf{B}')^T \mathbf{B}'\mathbf{C} = (\mathbf{B}')^T \boldsymbol{\sigma}_s \tag{4.40}$$

Therefore, the stress values used in the calculation (4.38) for the surface stresses are in fact values on the fitted B-Spline surface (4.39). As a final step, the numerical contact pressure values are compared against the predicted analytical solution. Case B and C have simple closed form solutions characterized by equations (2.61b) and (2.62b). Case A, however, does not have a closed form solution for the case of two elastic blocks. The solution to this problem is discussed in the next sub-section.

4.6.1.1 Rectangular punch problem (Case A)

The first model problem used for accuracy comparison was that of a two-dimensional flat punch in contact with another flat surface as in Figure 4.12. A uniform pressure is applied to the top of the punch. The problem is modeled as symmetric about the *y*-axis. As mentioned previously, this problem does not have a closed-form solution for the case of two elastic bodies. However, if one considers the ratio of moduli, $k = E_1/E_2$ (see Figure 4.12), closed form solutions do in fact exist for the two extreme case $k=0$ and $k=\infty$. For arbitrary k ($k=1$ was used in this study), a power series approximation was discovered by Okubo (Okubo 1951). His solutions for the two limit cases, as well as for $k=1$, are summarized below.

- for the case $k = 0$, contact pressure = -p
- for the case $k = 1$, contact pressure = $p(-1/2+\sigma_y)$ where

  $$\sigma_y \approx \sum_{n=0}^{n=8} b_n x^{2n} \text{ and}$$

  $$b = \{-0.417, -0.152, 0.634, -2.006, 0.817, 2.062, -0.798, -0.916, -0.482\}$$

- for the case $k = \infty$, contact pressure = $\dfrac{-p}{\pi\sqrt{a^2 - x^2}}$

All three of these solutions are compared to the finite element solutions in the next chapter.

The basic problem description is given in Figure 4.12. The mesh-free model is shown in Figure 4.13, while the corresponding ANSYS model is shown in Figure 4.14. Grid properties for both models are compared in Table 4.1

Table 4.1. Model properties comparison Case A

ANSYS	Body 1	Body 2	Total
Number of Elements	25	110	135
Degrees of Freedom	192	746	938
Mesh-Free			
Number of Elements	30	110	140
Degrees of Freedom	112	312	424

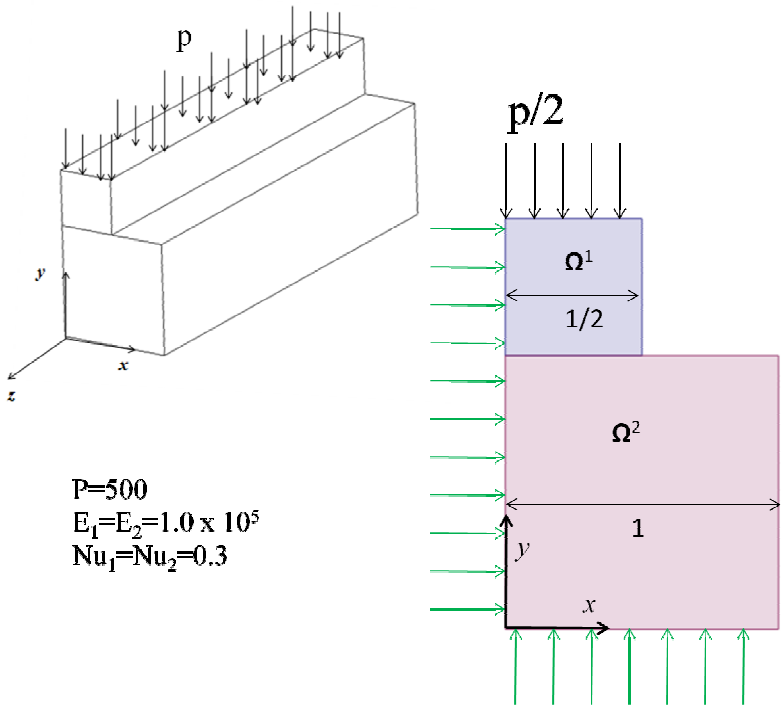


Figure 4.12. Case A problem description

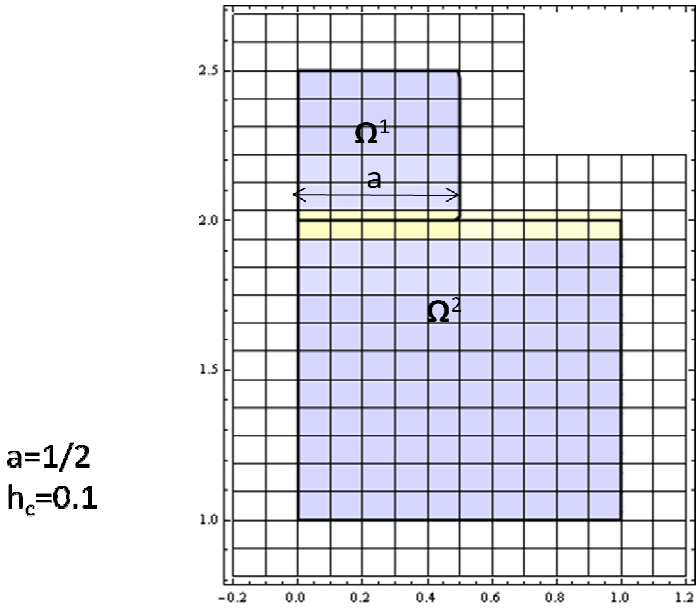


Figure 4.13 Mesh-free model for Case A showing both elements and support grid (inner elements highlighted in blue, outer elements highlighted in yellow)

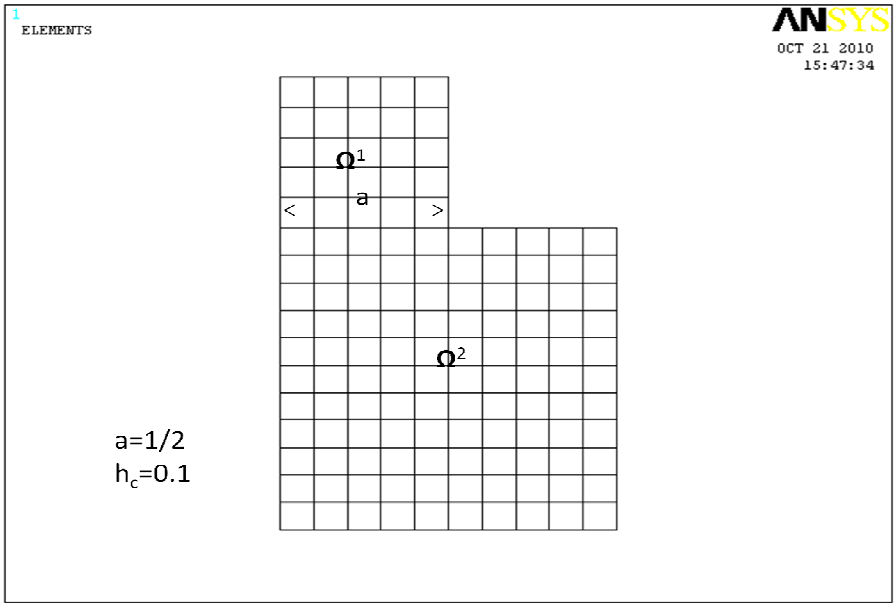


Figure 4.14. ANSYS model for Case A

4.6.1.2 Convex cylinder contact (Case B)

The second model used to test accuracy involved two convex cylinders coming into contact along their mutual axis (Figure 4.15). In both ANSYS and mesh-free models, non-uniform element spacing is used to achieve approximately five elements across the expected contact zone (a~0.1). Symmetry about the *y*-axis is used to model both cylinders. Body 2 (the slave) is additionally symmetric about the *x*-axis. The overall algebraic system is dramatically smaller in the mesh-free model, as can be seen in Table 4.2. Note, this is both because there are fewer elements (due to the fact that the grid refinement does not have to conform to the geometry), and because of the minimal support property of B-Spline basis functions in the mesh-free case. Again, bi-quadratic basis functions are used in both methods. Figure 4.15 gives a description of this problem. The mesh-free and ANSYS models, respectively, are shown in Figure 4.16 and Figure 4.17.

Table 4.2. Model properties comparison Case B

ANSYS	Body 1	Body 2	Total
Number of Elements	215	468	683
Degrees of Freedom	1408	3002	4410
Mesh-Free			
Number of Elements	144	228	372
Degrees of Freedom	396	588	984

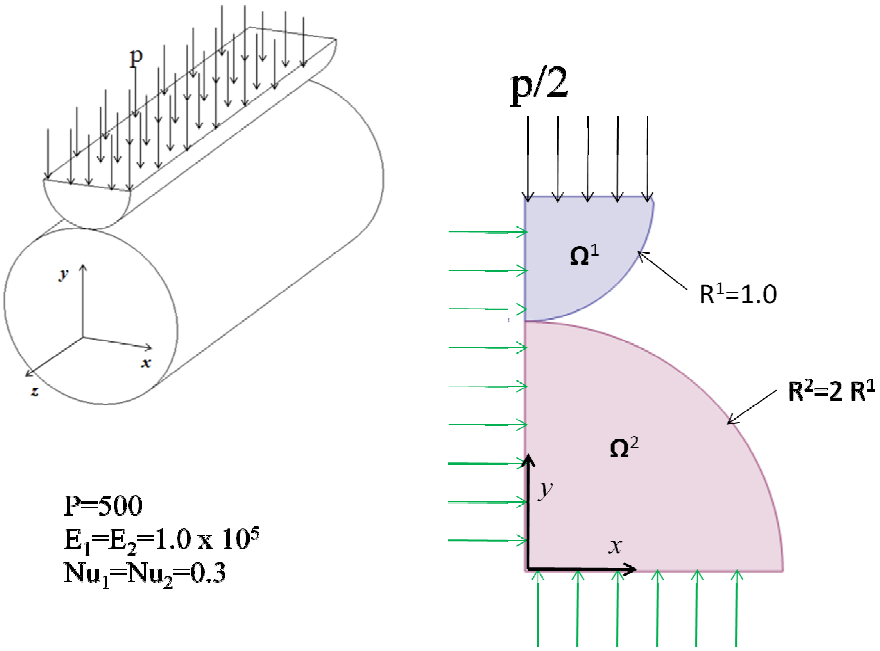


Figure 4.15. Case B problem description

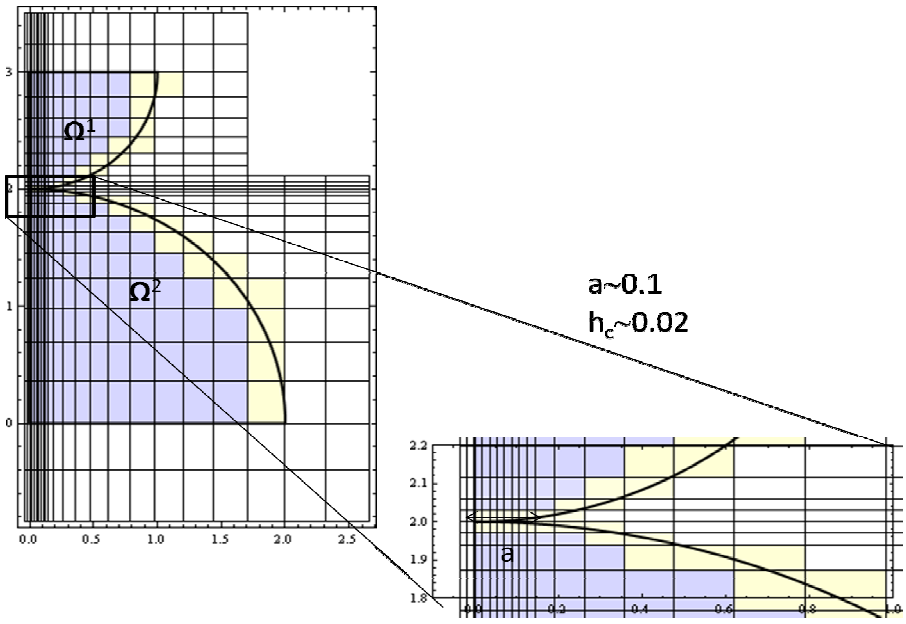


Figure 4.16. Mesh-free model for Case B showing both elements and support grid

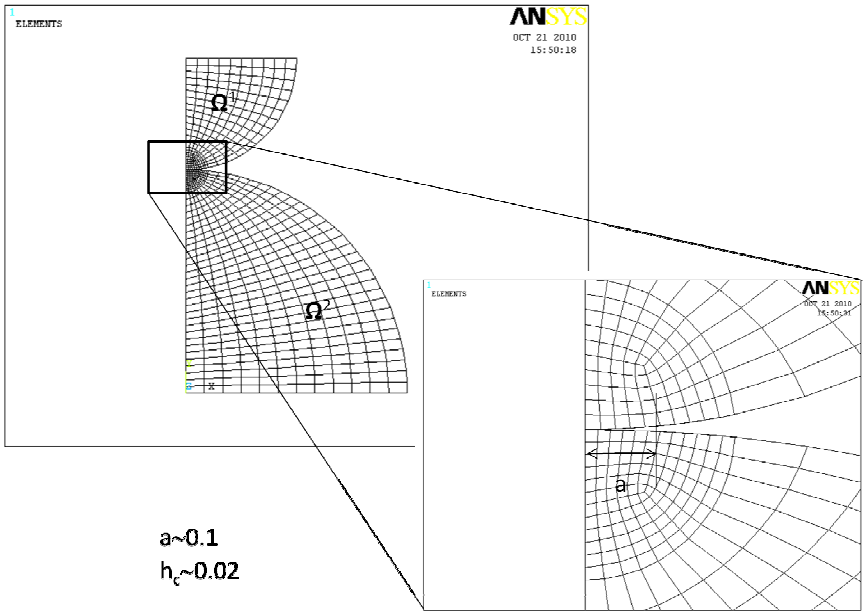


Figure 4.17. ANSYS model for Case B

4.6.1.3 Convex-concave cylindrical contact (Case C)

The third model used to assess accuracy in modeling contact involved a convex cylindrical surface coming into contact with a concave groove with coincident axes (Figure 4.18). This time, the total element count in both cases is similar (with only four fewer elements used in the mesh-free case than in the mesh model. See Table 4.3). The problem description is given in Figure 4.18. The mesh-free and ANSYS models, respectively, are shown in Figure 4.19 and Figure 4.20.

Table 4.3. Model properties comparison Case C

ANSYS	Body 1	Body 2	Total
Number of Elements	139	293	432
Degrees of Freedom	920	1892	2812
Mesh-Free			
Number of Elements	96	289	385
Degrees of Freedom	280	722	1002

4.6.2 Algorithm validation

Two test problems were selected to represent slightly more complicated assemblies in order to validate the contact algorithm. Since no closed-form solution to these problems exists, the solution will be compared only to a similar ANSYS model. No a priori predictions or assumptions are made concerning the contact zone in each case. Instead, a typical "coarse" mesh is constructed, and regions of potential contact are automatically calculated in the mesh-free case according to the methods outlined in section 4.5.1. In the ANSYS comparison,

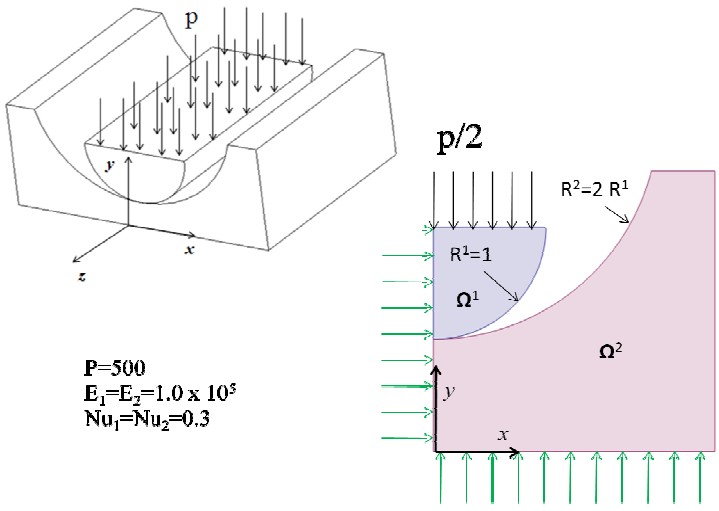


Figure 4.18. Case C problem description

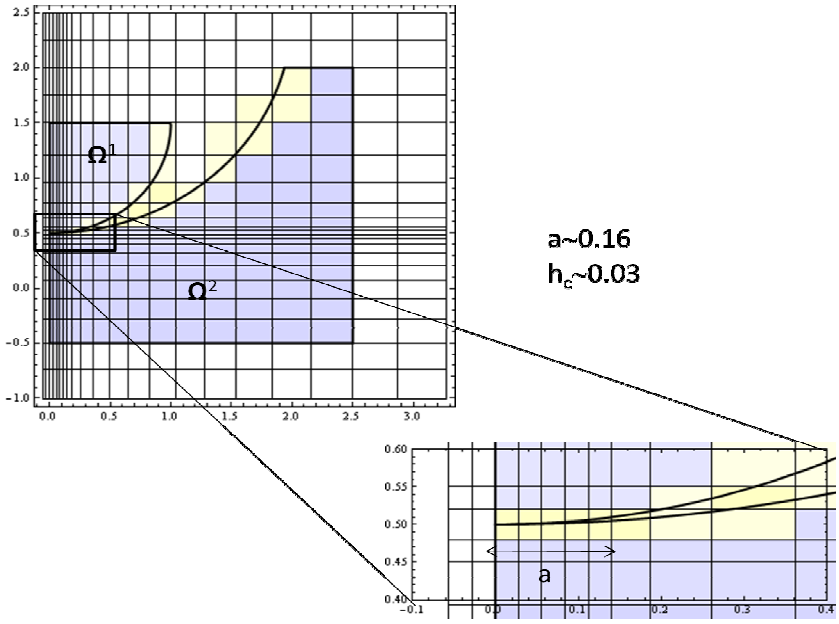


Figure 4.19. Mesh-free model for Case C showing both elements and support grid

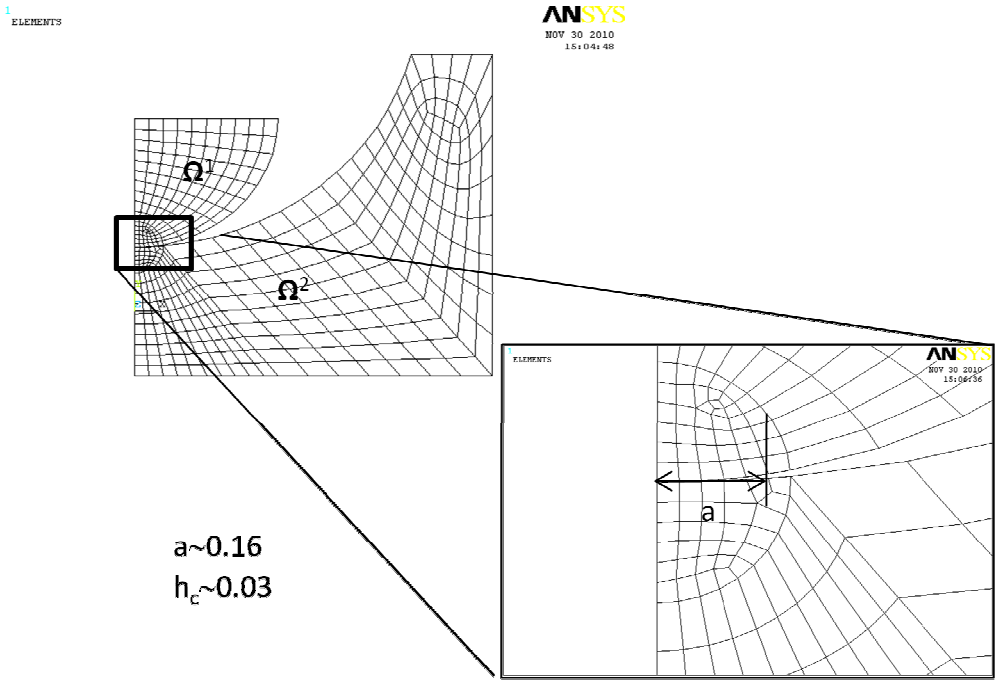


Figure 4.20. ANSYS model for Case C

contact elements are manually created on the same corresponding contact surfaces for comparison.

The point of these validation experiments was not to compare numerical quantities as was done in the previous studies (as stress results in a typical coarse grid solution cannot be expected to have converged), but as a qualitative assessment to demonstrate that the correct surfaces are automatically found for contact, and that for a similar grid size, the mesh-free solution gives a comparable solution. The first problem was used to assess hole deformation and stress patterns in a statically indeterminate pinned beam arrangement (see Figure 4.21). The second problem was inspired by a NAFEMS benchmark, specifically designed to test for proper convergence in a contact problem involving two materials with significantly different moduli. Specifically, the converged solution should not exhibit significant penetration when a penalty method is employed (as is done in the current investigation).

4.6.2.1 Statically indeterminate beam

The model problem for a 4-pinned beam is described in Figure 4.21. A grid width of 6 elements across the pin diameters was chosen simply to provide a reasonably continuous hole deformation pattern. The R-function composition (equation (4.16)) described in section 4.3 was used to fix the pins. The mesh-free grid is depicted in Figure 4.22. The ANSYS mesh is depicted in Figure 4.23. A comparison model properties is given in Table 4.4.

Table 4.4. Model properties comparison for beam problem

ANSYS	Body 1	Body 2	Total
Number of Elements	112	517	629
Degrees of Freedom	744	3356	4100
Mesh-Free			
Number of Elements	225	551	776
Degrees of Freedom	578	1302	1880

4.6.2.2 Vise mechanism

The model problem for the vise mechanism is described in Figure 4.24. Once again, no a priori predictions were made about the contact zone. However, the clamped material (body 1) was given 6 elements across its width (the

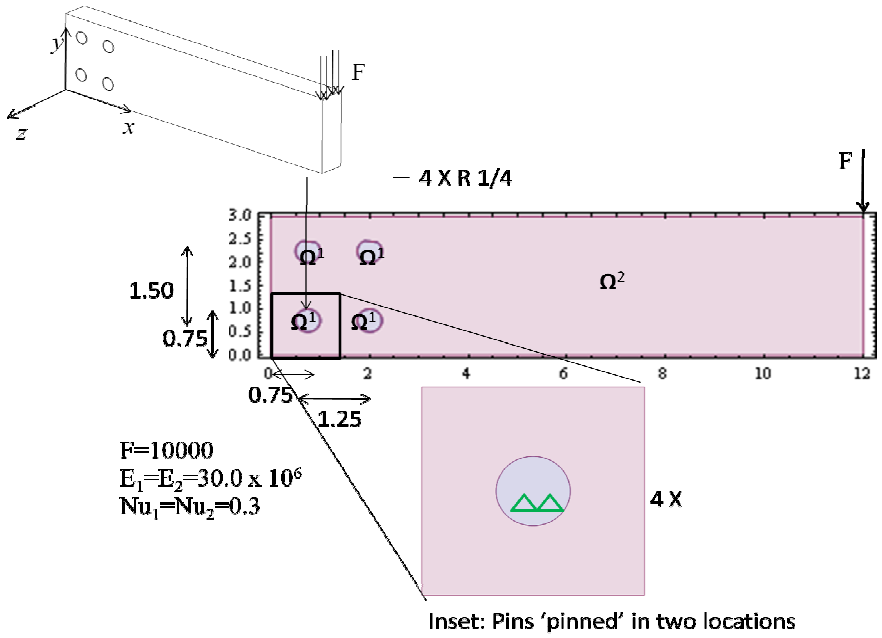


Figure 4.21. Statically indeterminate beam problem description

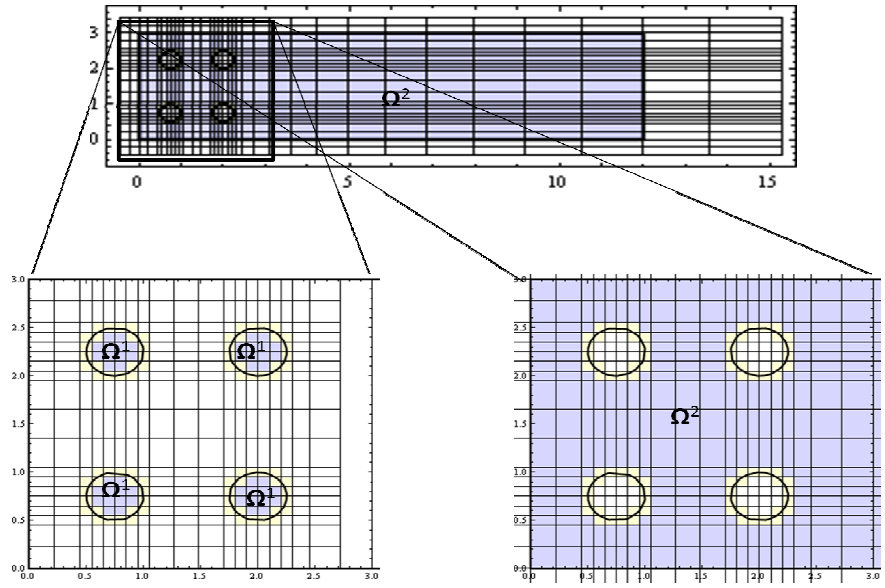


Figure 4.22. Statically indeterminate mesh-free beam model showing both elements and support grid

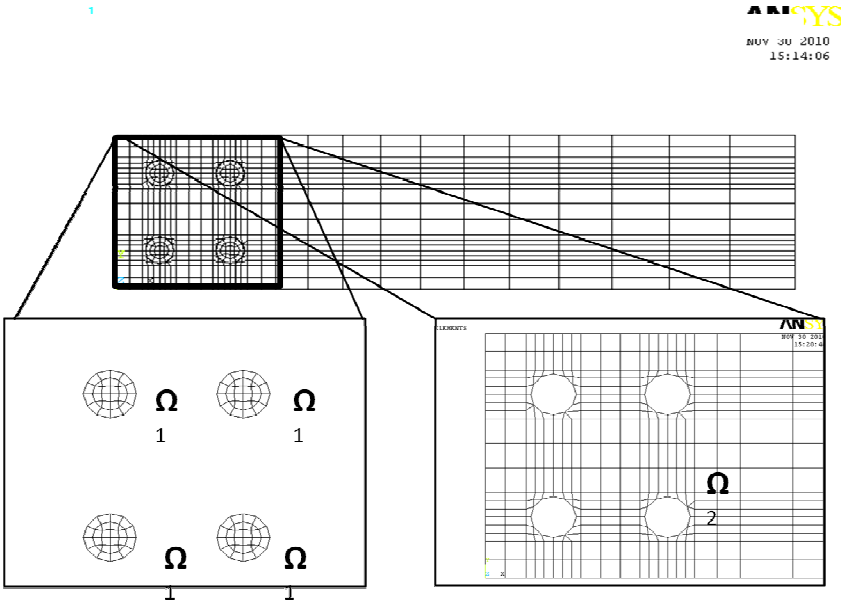


Figure 4.23. ANSYS Model for statically indeterminate beam

compression direction) in order to ensure sufficient resolution in the direction of compression. The mesh-free grid is depicted in Figure 4.25. The ANSYS mesh is depicted in Figure 4.26. A comparison of grid properties is given in Table 4.5. The displacement is applied via the solution structure (3.7), where the displacement function, φ was constructed according to (3.8). This resulted in: $\varphi = -15/2(d_0/2\ x)$.

Table 4.5. Model properties comparison for vise problem

ANSYS	Body 1	Body 2	Total
Number of Elements	66	149	215
Degrees of Freedom	466	1028	1494
Mesh-Free			
Number of Elements	66	320	386
Degrees of Freedom	208	792	1000

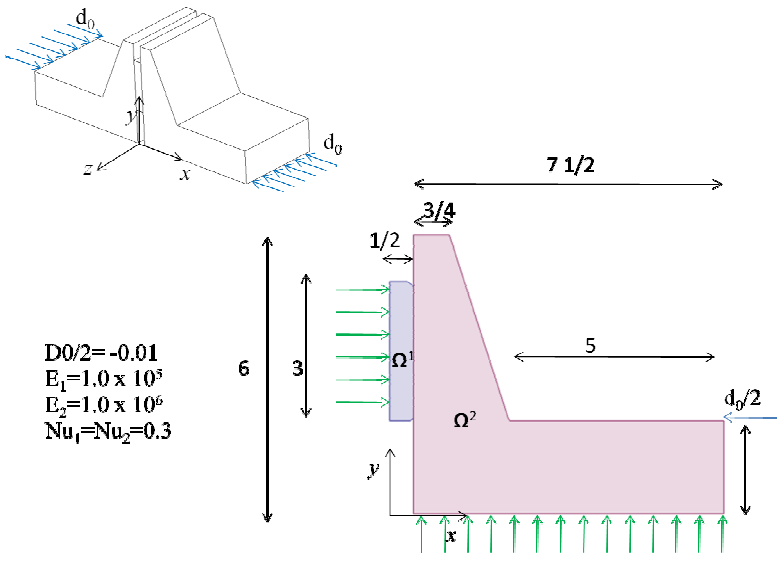


Figure 4.24. Vise problem description

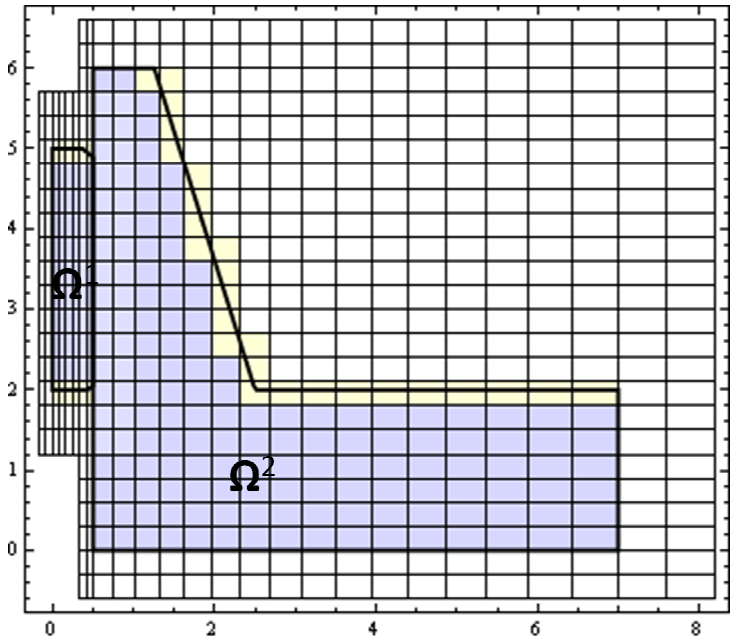


Figure 4.25. Vise mesh-free model showing both elements and support grid

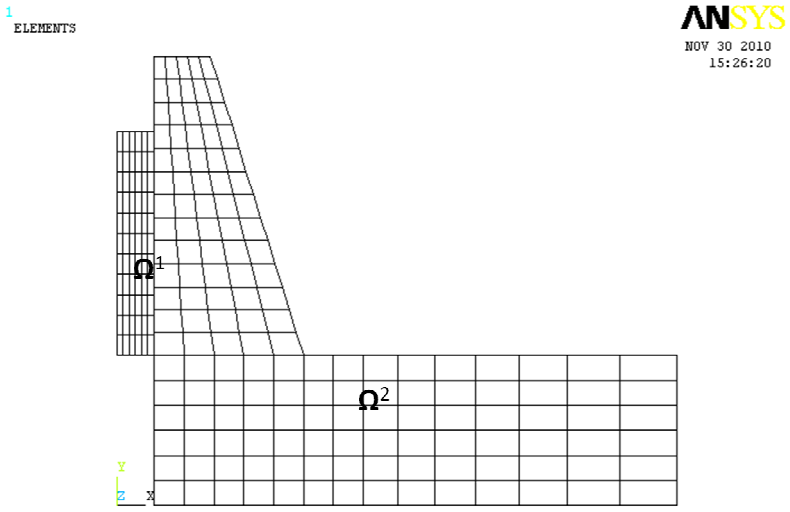


Figure 4.26. ANSYS model for vise problem

Chapter 5

VALIDATION RESULTS

The first part of this section is dedicated to comparing results for the three Hertz-type problems (case A, B, and C). In addition to a graphical comparison of global averaged stress contours, the averaged contact pressures are compared at meshed nodal locations within the contact zone (of which there are approximately five in all cases). The second part is devoted to assessing the validity of the general algorithm used in this mesh-free finite element framework on some more representative simple assemblies according to criteria outlined in Chapter 4.

5.1 Accuracy Validation

Stress results are first assessed graphically by comparing contour plots generated in ANSYS and the new mesh-free method (using tools available in Mathematica). In both plots, 10 contours are generated for each of stress component. Contact pressures are then calculated according to (4.38) on the master (body 1) surface and graphed. Relative error norms for the ANSYS and mesh-free contact surfaces are also reported.

5.1.1 Case A Results

Case A stress contours are compared in Figure 5.1. The contact pressures are shown in Figure 5.2. Both results show a relatively high error (0.152 for ANSYS, and 0.159 for the mesh-free method) due to the presence of a singularity at x=0.5. This is the only case for which the ANSYS error norm is lower than that of the mesh-free case. However, the mesh-free solution shows a greater sensitivity to the singularity. In Figure 5.2, only the first five points were used for

error estimation, while values at x=0.5 (the sixth point) are shown only to give a sense of how the two solutions reflect the presence of the singularity. The error norms for Case A are closer than in all other cases, possibly due to the lack of surface curvature and similarity of the two grids.

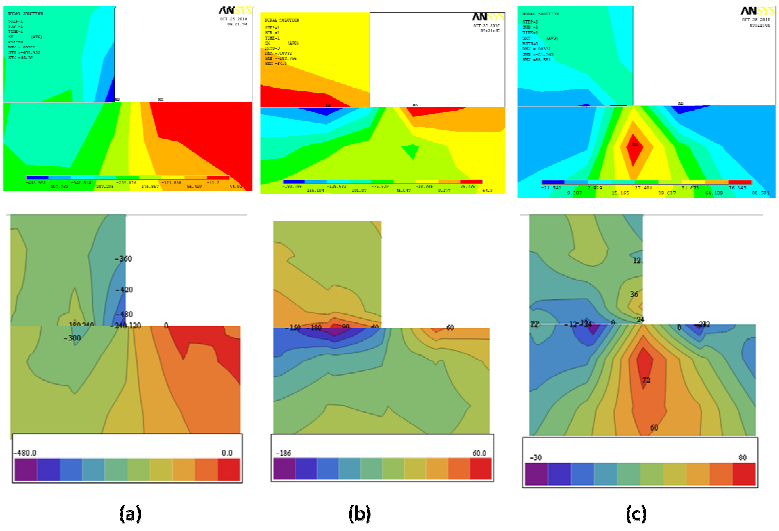


Figure 5.1. Stress contours Case A: ANSYS (top) vs. mesh-free for (a) σ_y, (b) σ_x, (c) σ_{xy}

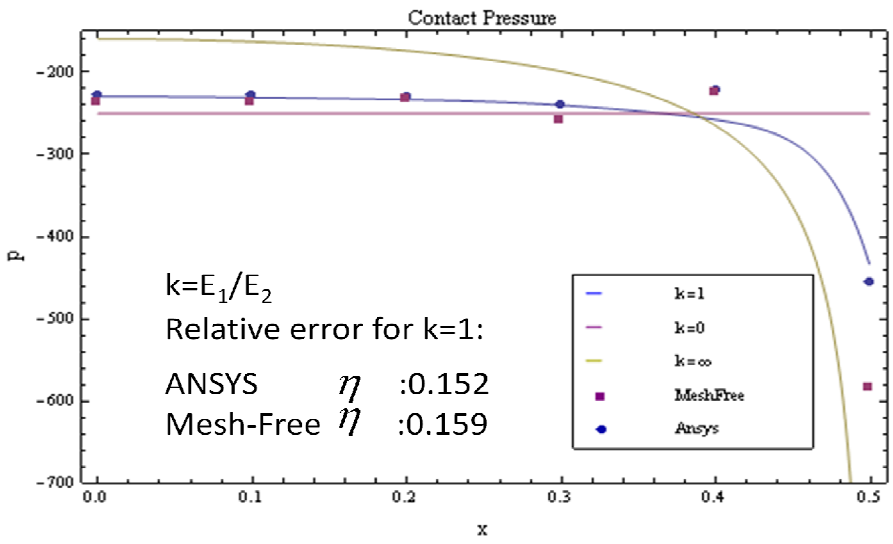


Figure 5.2. Contact pressure comparison Case A (ANSYS vs. mesh-free)

5.1.2 Case B Results

Case B stress contours are compared in Figure 5.3. The contact pressures are shown in Figure 5.4. The error norm for the mesh-free contact pressure is roughly ½ that of the ANSYS solution. The mesh-free solution is visibly closer to the Hertz solution than is the ANSYS solution.

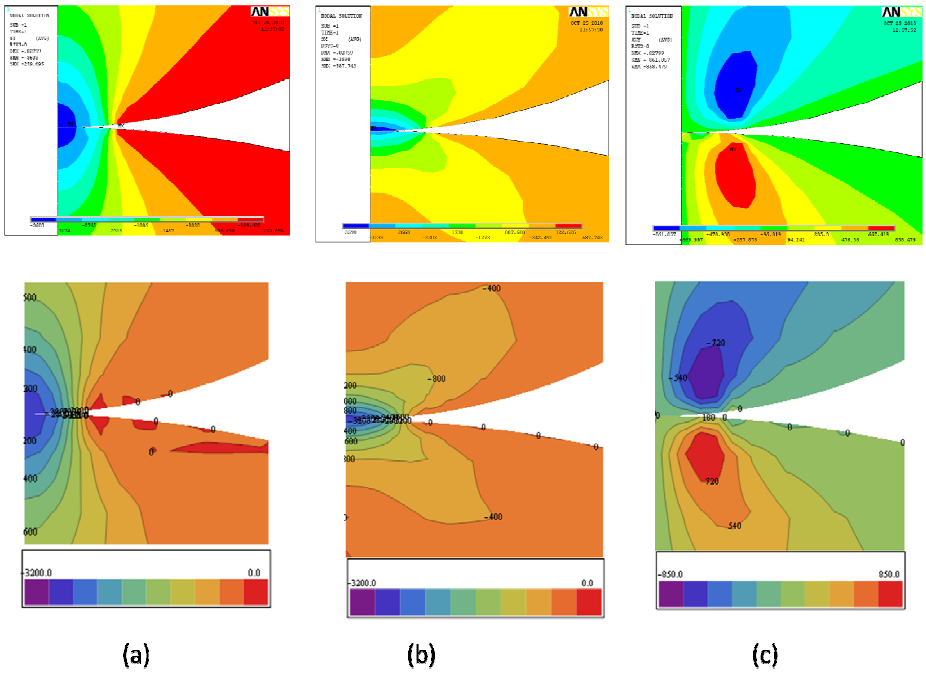


Figure 5.3. Stress contours Case B:ANSYS (top) vs. mesh-free for

(a) σ_y, (b) σ_x, (c) σ_{xy}

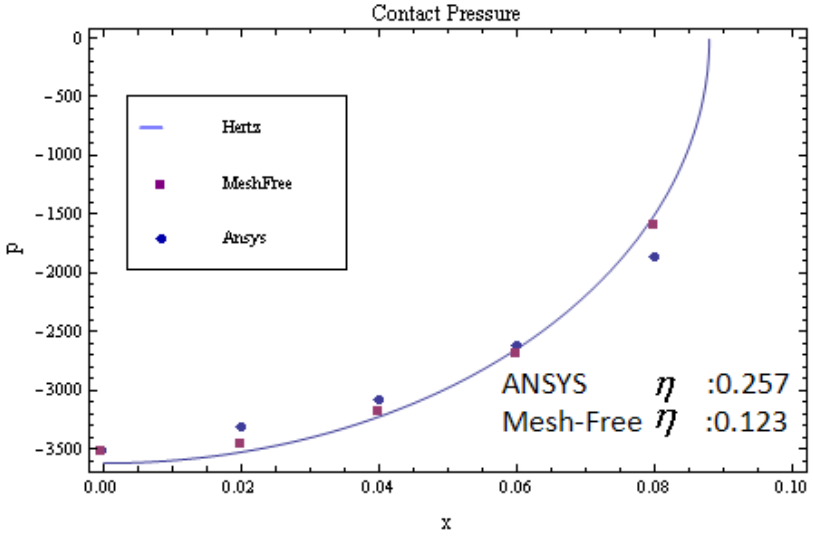


Figure 5.4. Contact pressure comparison Case B (ANSYS vs. mesh-free)

5.1.3 Case C Results

The Case C results were very similar to the case B results, both in terms of error norms and contours, with the mesh-free solution again trending closer to the Hertz solution. This time, both error norms were lower (possibly due to the fact that the contact surface curvatures are of opposite sign). The contours are shown in Figure 5.5. The contact surface pressure comparison is shown in Figure 5.6.

A summary of the results of the accuracy study is given in Table 5.1. It can be seen that, except for case A, all results showed smaller relative error in contact surface pressure and for a significantly smaller system of equations. It is believed that case A breaks this trend only because the flat contact surfaces offer no advantage to the mesh-free approach.

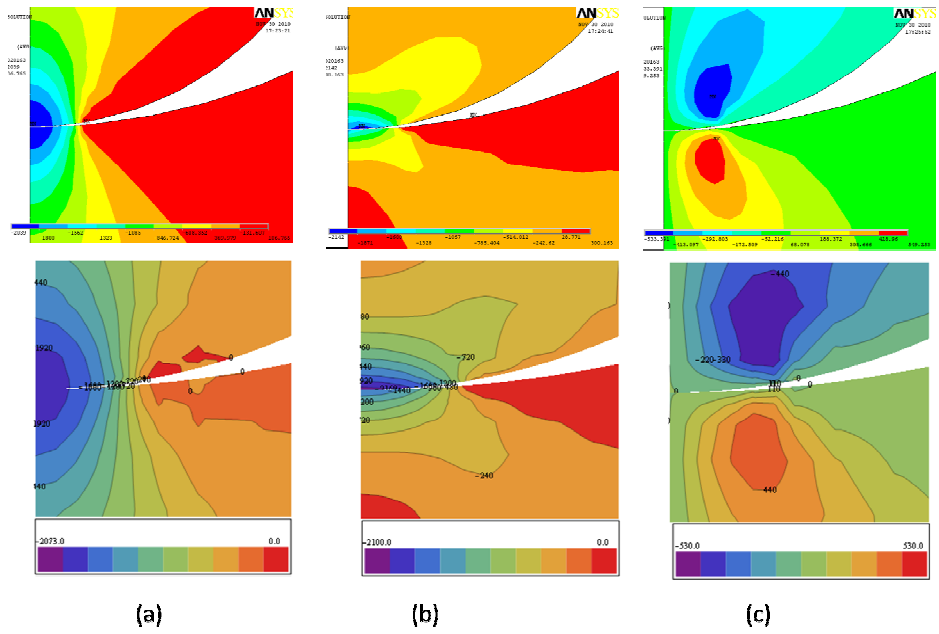


Figure 5.5. Stress contours Case C:ANSYS (top) vs. mesh-free for (a) σ_y, (b) σ_x, (c) σ_{xy}

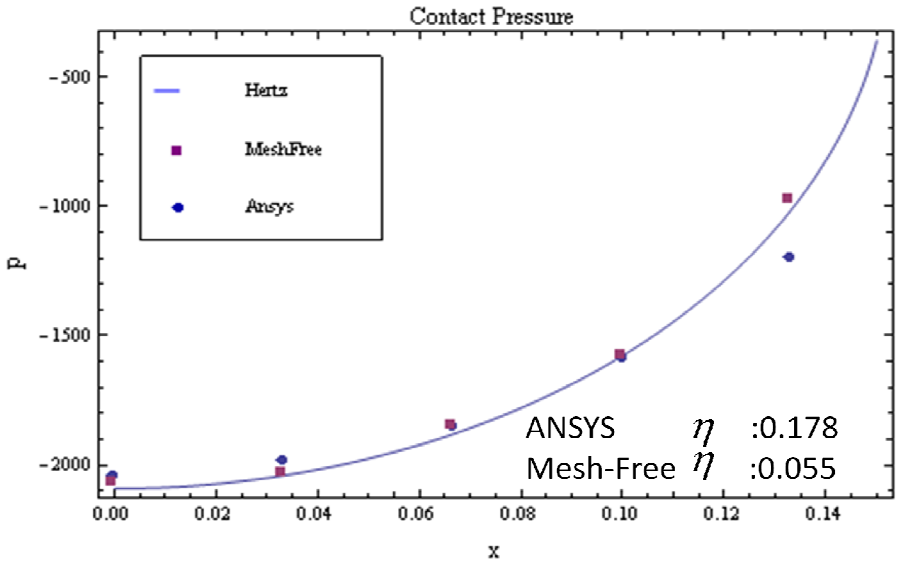


Figure 5.6. Contact pressure comparison Case C (ANSYS vs. mesh-free)

Table 5.1. Summary of accuracy study results

Problem	ANSYS Max. |p|	Mesh-Free Max. |P|	ANSYS Relative Error η	Mesh-Free Relative Error η	ANSYS No. DOF	Mesh-Free No. DOF
Case A	611	453	0.152	0.159	938	424
Case B	3513	3511	0.257	0.123	4410	768
Case C	2048	2060	0.186	0.055	2812	1002

5.2 Algorithm Validation

For the statically indeterminate beam model and the vise model, deformation contour plots and stress components were compared on a qualitative basis. In addition, deformed shape contours were compared to confirm common trends. Once again for all contour plots, 10 contours were generated for each of stress component and each deformation. The statically indeterminate beam was assessed first, followed by the vise.

5.2.1 Statically indeterminate beam

The overall transverse (in the direction of the applied load) deflection of the ANSYS and mesh-free models is shown in Figure 5.7. The X and Y component deflections around the pin locations are shown in Figure 5.8. A comparison of the hole distortion patterns is depicted in Figure 5.9. Finally, each stress component is compared around the pin locations in Figure 5.10.

The overall trends in the mesh-free case were similar to those calculated by ANSYS. However, the maximum beam deflection was calculated to be -0.080 by the ANSYS model, whereas the Mesh-Free solution showed a deflection of -0.075 (a difference of ~6%.). It should be emphasized that these benchmarks

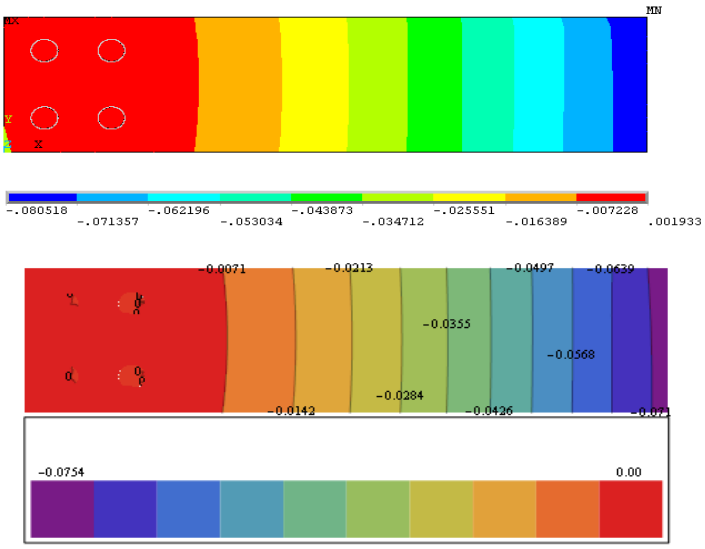


Figure 5.7. Y-component deflection contours ANSYS (top) vs. mesh-free

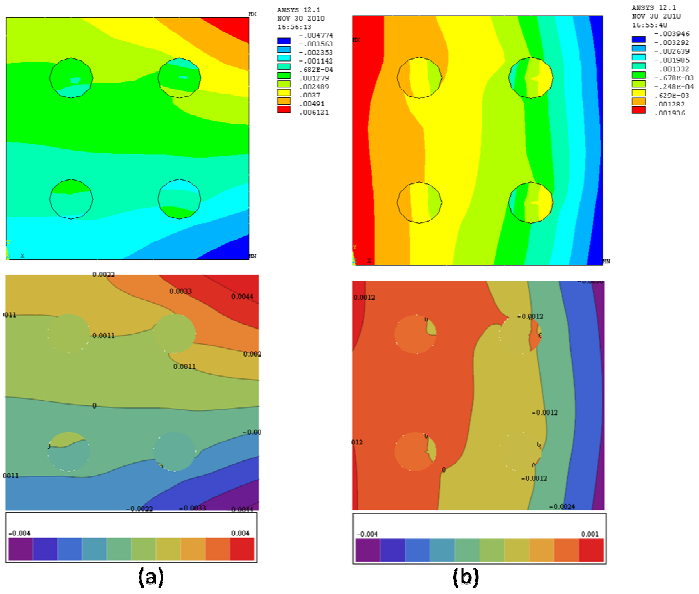


Figure 5.8. (a) X-component deflection contours around holes for ANSYS (top) vs. mesh-free, and (b) Y-component-deflection

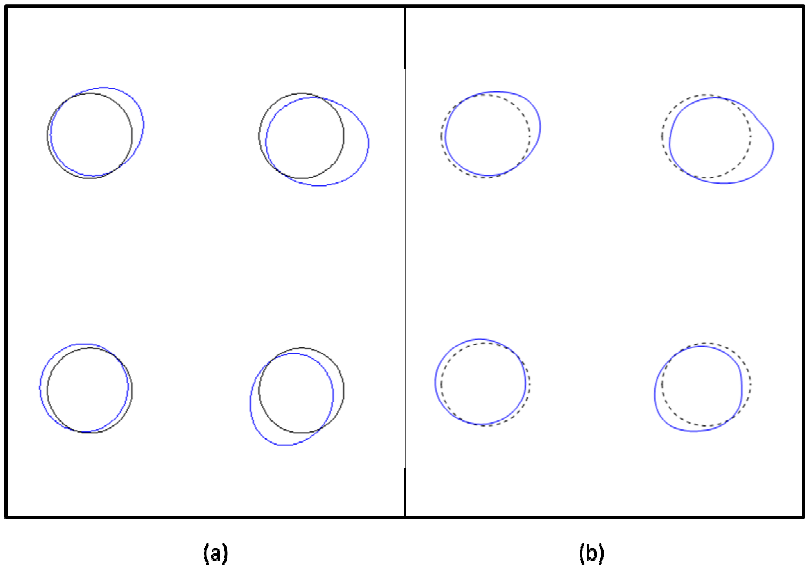


Figure 5.9. (a) ANSYS hole deflection magnified 50 x, (b) mesh-free hole deflection magnified 50 x

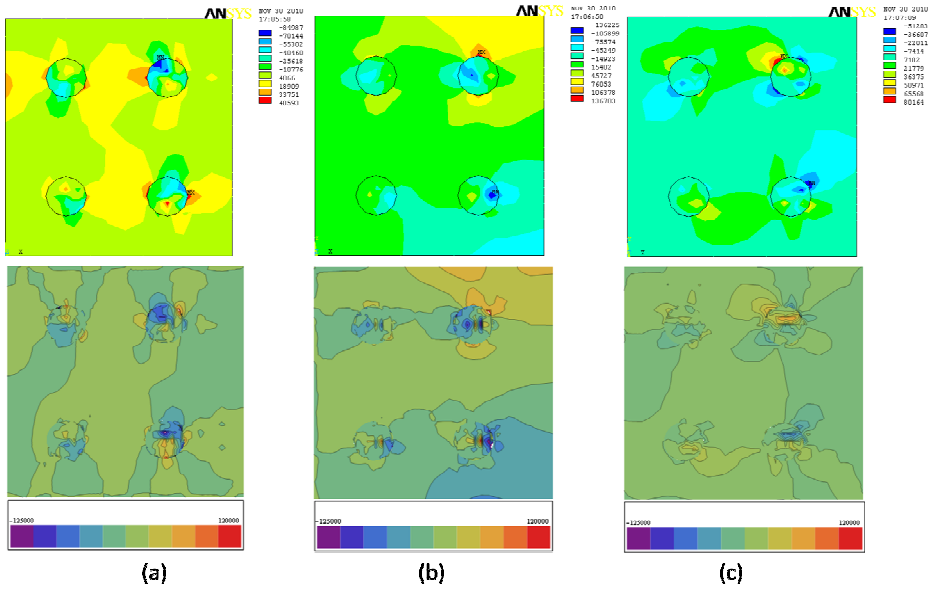


Figure 5.10. Stress contours comparing ANSYS results (top) vs. mesh-free for

(a) σ_y, (b) σ_x, (c) σ_{xy}

utilize coarse meshes and so do not anticipate a converged solution. Nevertheless, closer agreement in deflection was expected. Since deflection in the Y-direction at the holes gets magnified at the beam tip (through its action as a moment arm), the increased tip deflection in the ANSYS model may simply be due to coarse mesh differences around the holes. All peak stress values around holes in the Mesh-Free case were higher than those seen in the ANSYS model (in some cases, by roughly a factor of 2, but these are all associated with the fixed point-displacement singularities inside the pins).

5.2.2 Vise mechanism

The X and Y-component deflections calculated in the ANSYS and Mesh-Free models is shown in Figure 5.11. The distorted domain shapes are shown in Figure 5.12, and the stress component comparisons are depicted in Figure 5.13.

The main goal of the vise mechanism study was to verify that a) no excessive penetration occurred due to the material modulus mismatch between the two contacting bodies, and b) that the upper region of contact separates as necessary by the physics of the problem. An inspection of the Figure 5.11 reveals that the latter does indeed happen in both models (with maximum separation approximately 0.002 in the X-direction).

The concern over penetration is motivated by the differing elastic moduli of the two bodies. The master (body 1) had a modulus of 1.0×10^5, while body 2 had a modulus of 1.0×10^6. The penalty parameter was chosen to be 5.0×10^7, or 50 times the modulus of the stiffer body. This parameter value was used in both the

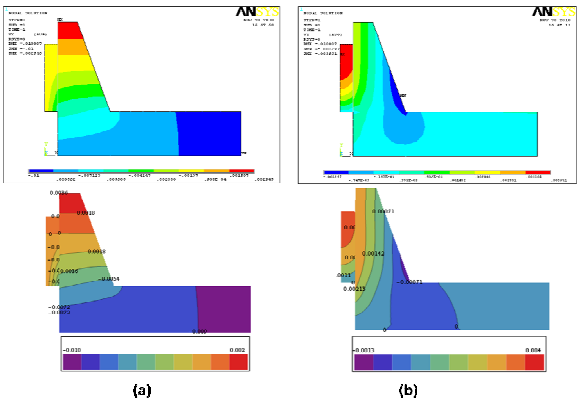
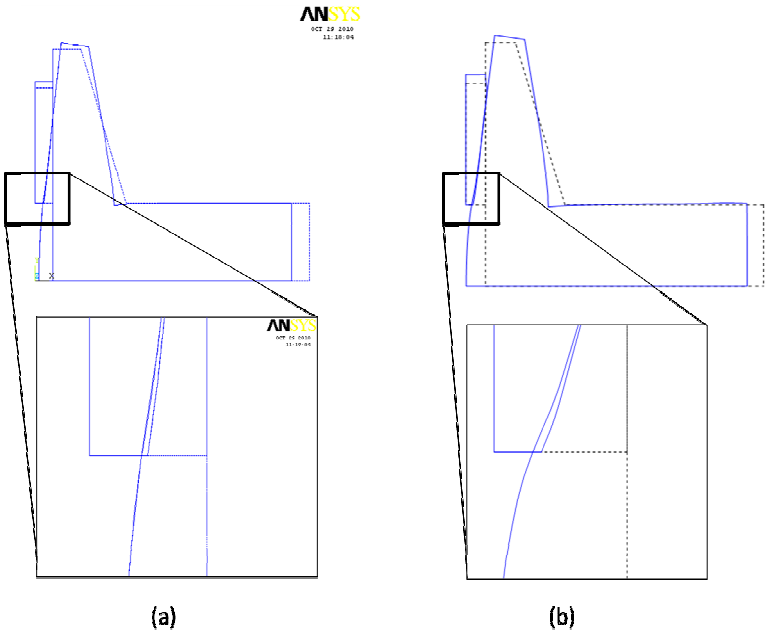


Figure 5.11. (a) X-component deflection contours for ANSYS (top) vs. mesh-free, and (b) Y-component deflections



5.12. (a) ANSYS deflection mangified 50 x, and (b) mesh-free deflection magnified 50 x

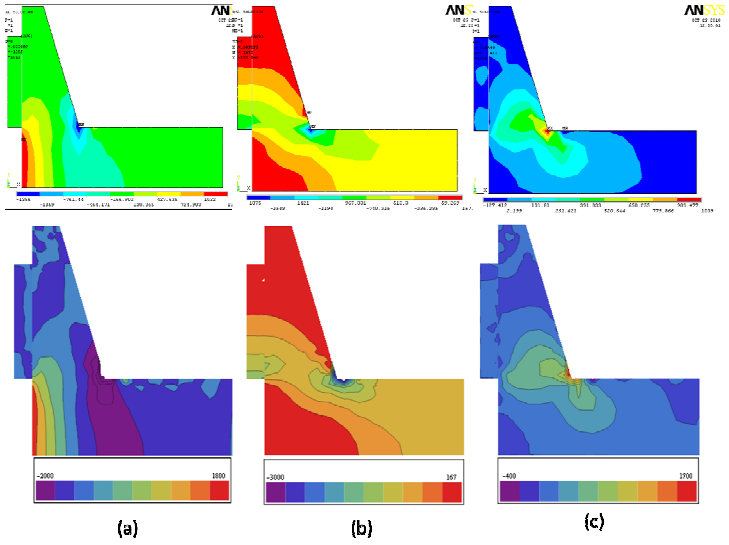


Figure 5.13. Stress controus for ANSYS (top) vs. mesh-free contours for (a) σ_y, (b) σ_x, (c) σ_{xy}

ANSYS and Mesh-Free models. This resulted in maximum contact penetration values of 2.12×10^{-5} for the ANSYS model, and 3.0×10^{-5} in the Mesh-Free models (this penetration can be seen in Figure 5-12). Although the Mesh-Free solution does exhibit more penetration, both values are typical for a penalty formulation and considered acceptable.

The deflection and stress contours matched well, as expected, as there was only one geometric singularity associated with the angled concave corner in addition to the singularities at the ends of the contact zone. In fact, this was the only location where stress values differed substantially. For all stress components, the Mesh-Free solution showed a roughly 2x (magnitude) increase in value over that seen in the ANSYS solution in this one region.

Table 5.2 compares the matrix condition numbers of the ANSYS models to their mesh-free counterparts. Even though the mesh-free condition numbers are generally higher, they appear to be stable. It should be noted that these are un-normalized condition numbers. In general, these numbers can be further reduced by normalizing the diagonal to 1 (a diagonal pre-conditioner).

Table 5.2. Summary of stiffness matrix condition numbers

	ANSYS	mesh-free
Case A	7.086E+05	8.992E+05
Case B	3.480E+05	3.163E+06
Case C	2.783E+05	4.416E+06
Vise problem	2.705E+05	2.815E+06
Beam problem	2.356E+06	4.862E+06

Chapter 6

CONCLUSIONS

This research is motivated by the desire to reduce the gulf between CAD and CAE. Assembly design is the very crux of engineering design, yet CAD and simulation tools (CAE) are not particularly compatible with assembly level thinking. In FEA, difficulties in modeling contact interfaces accurately forces users to perform simulation mostly at the part level. That requires the analyst to estimate load magnitudes and distributions at mating regions, which in most cases, is difficult to do accurately. Additionally, loads might be statically indeterminate. Regardless of part or assembly level design, conventional FEA requires meshing. The quality of the mesh is critical for good results. Even experienced analysts spend considerable effort in obtaining 'good' meshes. Particularly in contact and large deformation problems, maintaining high mesh quality is difficult and a significantly finer mesh is needed in contact regions. Expertise needed in element selection, meshing and simulation of boundary conditions makes designers dependent on specialists for use of CAE tools.

This research has successfully demonstrated that it is possible to perform FEA on mechanical assemblies without meshing. This is achieved by overlaying a uniform grid on the bounding boxes of all parts in an assembly, including contact regions. The grid is independent of the boundary shape of parts, i.e. is non-conforming. Further development of this method is expected to lead to integrated CAD-CAE systems envisioned in Chapter 1.

The FE formulation developed in this work uses B-Spline basis functions. The domain to be analyzed is integrated with the use of transformed Gauss points within the domain, and boundary conditions are applied via distance functions (R-functions). However, the basis is stabilized through a novel selective normalization procedure. In addition, a novel contact algorithm is developed in which the B-Spline support grid is re-used for contact detection. Finally, a modified Penalty Method is demonstrated for connecting elements with incompatible bases.

The unilateral structural contact algorithm has been designed specifically to exploit the properties of this formulation. Criteria for this selection were: 1) Optimal algebraic system size for given mesh size *h*, and basis degree *d*, and 2) Accuracy and robustness in solving general multiple-part, unilateral frictionless contact problems. In the last chapter, the last goal was demonstrated, while evidence was given for the first. The smaller algebraic system sizes seen in the current mesh-free framework are directly related to *the minimal support* property of B-Splines. This property can be found in (Farin 2002), from which the following quote comes: "...if a piecewise polynomial with the same smoothness properties over the same knot vector has less support than N_i^n, it must be the zero function." This property of B-Splines has long served as the central motivation for using them as finite element bases, but until recently, the implementation of B-Splines in a finite element framework has been dogged by insufficient computing power and the inability to apply the tensor-product basis to problems with non-rectangular domains.

This approach is superior to mesh-free particle-based methods in that the FE formulation developed here and its implementation have been validated both by comparison to the idealized contact (Hertzian) equations found in the Theory of Elasticity and to the results from a leading commercial FEA package for classical contact (planar-planar, convex-convex, convex-concave) cases. It has also been demonstrated that not only does this method yield accurate contact pressures and deformation plots, it also results in a smaller set of system equations.

Current limitations of the solution offered in this dissertation are that it is only implemented in two dimensions and only incorporates boundary nonlinearities. Material and geometric (large displacement) effects are not considered. Despite these limitations it should be applicable to a large number of common mechanical assemblies, such as those in automotive power trains, static structures with fasteners, and many others.

Chapter 7

ORIGINAL CONTRIBUTIONS

Shapiro et al demonstrated the feasibility of using weighted B-Spline basis functions defined over a non-conforming structured grid within a finite element framework (Shapiro and Tsukanov 1999b). Höllig et al. then extended this work by discovering a stabilized basis to address the inherent instability of the B-Spline basis in two or more dimensions, as well as introducing new weighting schemes to satisfy boundary conditions (K. Hoellig, Reif, U., and Wipper 2001). Through these efforts, this approach is now on a firm mathematical footing. However, to our knowledge, no researchers have demonstrated the applicability or usefulness of this method within a general engineering framework.

Most research today in mesh-free finite element technology focuses on particle-based methods, which have at best specialized applications (T. Belytschko et al. 1996). In assessing the feasibility of this technique in solving unilateral contact problems, this investigation marks the first effort in exploring the feasibility of the B-Spline approach within a general structural engineering framework (specifically oriented toward assembly design). In doing so, a Penalty formulation for connecting incompatible bases was developed which we have not seen in the literature (however, it should be pointed out that this formulation seems to be identical to the stabilization term found in the Nitsche Method (Wriggers 2002).)

Because the analysis of large-scale assemblies is becoming increasingly commonplace, and because a large portion of such analyses involve unilateral

contact, it is felt that any numerical approach that can solve such problems accurately, efficiently, and with minimal user intervention (compared with the current state of the art) represents a great leap forward toward the future of engineering design.

Beyond this, it should be noted that this investigation marks the first demonstration of selective B-Spline normalization (as opposed to the weB-Spline method of Höllig et al) as a means to achieve reasonable matrix condition numbers. This possibility was mentioned by Reif (Reif 2006), but no demonstration of it has been found in the literature. In addition, the contact algorithm design for this purpose is novel. It is hoped that the implementation of this algorithm has demonstrated that the unilateral contact problem actually becomes somewhat simpler in this mesh-free setting, as opposed to the use of surface-to-surface contact segments in the traditional approach.

Chapter 8

FUTURE WORK

Toward the goal of applying the mesh-free B-Spline method of Shapiro/Höllig to a generalized FEA/Design application, several questions remain. For example, can the non-conforming tensor-product approach (trimmed solution spaces) be used in conjunction with reduced-order (non-manifold) elements such as beams and shells formulated in the same way? Can such a methodology be applied to material and geometrically nonlinear problems in a robust and efficient manner?

In Appendix A, it is shown how the technique can be applied to small deflection beam theory, but a typical finite element model may have solids, beams, and shells – all with arbitrary spatial orientations. This is one reason why mesh-based finite elements use an element coordinate system (often iso-parametric). An obvious solution suggests itself: Extend the current methodology to parametric B-Spline spaces. Several researchers (Hughes, Cotrell, and Bazilevs 2005) have discovered NURBS, or Bezier finite elements, but all these interpolate element boundaries, and are therefore not mesh-free. It would be advantageous to assess the feasibility of extending the current methodology to parametric weighted B-Spline spaces in which boundaries are described just as they are in B-rep CAD models, with the solution space superimposed directly on B-Spline control points. If this is feasible, we believe it

would offer an excellent framework for geometrically nonlinear problems, as well as reduced-order element types.

Beyond extending the mesh-free formulation introduced in this study, future work should seek to investigate the design/analysis interoperability opportunities offered by this framework. For example, the Generic Functional Interfaces introduced in Chapter 1 could interact directly with a mesh-free contact data-structure. In fact, it is conceivable that a mesh-free contact algorithm might even be capable of constructing such interfaces.

REFERENCES

Anonymous. n.d. Structural analysis - Wikipedia, the free encyclopedia. http://en.wikipedia.org/wiki/Structural_analysis.

Antes, H. 1974. Bicubic fundamental splines in plate bending. *International Journal for Numerical Methods in Engineering* 8: 503-511.

Babuska, I. 1973. The Finite Element Method with Lagrange multipliers. *Numerische Mathematik* 20: 179-192.

Babuska, I, and J.M. Melenk. 1996. *The Partition of Unity Finite Element Method: Basic Theory and Applications*. Research Report. The University of Texas at Austin.

Ball, S. 1900. *A Treatise on the Theory of Screws*. Cambridge: Cambridge University Press.

Bathe, K.J., and P.A. Bouzinov. n.d. On the Constraint Function Method for Contact Problems. *Computers and Structures* 64, no. 5: 1069-1085.

Baumgart, B.G. 1974. *Geometric Modeling for Computer Vision*. Stanford Artificial Intelligence Laboratory: Stanford University.

Belytschko, T, Y.Y Lu, and L. Gu. 1994. A New Implementation of the Element-Free Galerkin Method. *Computer Methods in Applied Mechanics and Engineering* 113, no. 3: 397-414.

Belytschko, T., and M. Fleming. 1999. Smoothing, enrichment and contact in the element-free
Galerkin method. *Computers and Structures* 71: 173-195.

Belytschko, T., L. Gu, and Y.Y. Lu. 1994. Fracture and Crack Growth by Element-Free Galerkin Methods. *Modeling and Simulation in Materials Science and Engineering* 2, no. 3: 519-534.

Belytschko, T., Y. Krongauz, D. Organ, M. Fleming, and P. Krysl. 1996. Meshless Methods: An Overview and Recent Developments. *Computer Methods in Applied Mechanics and Engineering* 139: 3-47.

Belytschko, T., Y.Y. Lu, and L. Gu. 1994. Element-Free Galerkin Methods. *International Journal for Numerical Methods in Engineering* 37, no. 2: 229-256.

Bhatti, M.A. 2006. *Advanced Topics in Finite Element Analysis of Structures*. John Wiley & Sons.

Borgedoni, M., and U. Cugini. 1997. Feature-Based Assembly Design: Concepts and Design Environment. In *Proceedings of DETC'97*. Sacramento, CA.

Bourjault, A. 1984. Contribution à une Approche Méthodologique de l'Assemblage Automatisé: Elaboration Automatique des Séquences Opératoires. l'Université de Franche-Compté.

Brunetti, G., and B. Golob. 2000. A feature-based approach towards an integrated product model including conceptual design information 32, no. 14: 877-887.

Cescotto, S., and R. Charilier. 1992. Frictional Contact Finite Elements Based on Mixed Variational Principles. *International Journal for Numerical Methods in Engineering* 36: 1681-1701.

Chen, J.S., and H.P. Wang. 2000. New Boundary Condition Treatments in Meshfree Computation of Contact Problems. *Computer Methods in Applied Mechanics and Engineering* 187, no. 3: 441-468.

Chu, Y.A., and B. Moran. 1995. A Computational Model for Nucleation of Solid-Solid Phase Transformations. *Modeling and Simulation in Materials Science and Engineering* 3, no. 4: 455-471.

Cleveland, W.S. 1993. *Visualizing Data*. Research Report. AT&T Bell Labs.

Cook, R.D., D.S. Malkus, and M.E. Plesha. 1989. *Concepts and Applications of Finite Element Analysis*. 3rd ed. New York, NY, USA: John Wiley & Sons.

Courant, R. 1943. Variational Methods for the Solution of Problems of Equilibrium and Vibrations. *Bulletin of the American Mathematical Society* 49: 1-23.

Davidson, J., and K. Hunt. 2005. *Robot and Screw Theory*. Oxford University Press.

Davies, A.M. 1978. The use of the galerkin method with a basis of B-Splines for the solution of the one-dimensional primitive equations. *Journal of Computational Physics* 27, no. 1: 123-137.

De Boor, C.E. 1966. The Method of Projections as applied to the Numerical Solutions of Two Point Boundary Value Problems using Cubic Splines. Doctoral dissertation in mathematics, Ann Arbor, MI: University of Michigan.

De Fazio, T.L., A.C. Edsall, R.E. Gustavson, P.M. Hernandez, and H. Hutchins. 1991. A Prototype of Feature-based Design for Assembly. *Lecture Notes in Computer Science* 492: 369-392.

Delaunay, B.N. 1934. Sur La Sphere Vide. *Izvestia Akademii Nauk SSSR, Otdelenie Matematicheskii i Estestvennyka Nauk (Bulletin of Academy of Sciences of theUSSR),* 7: 793-800.

Duarte, C.A., and J.T. Oden. 1995. *A Meshless Method to Solve Boundary Value Problems*. Technical Report. The University of Texas at Austin: Institute for Computational and Applied Mathematics.

Duncan, W.J., and A.R. Collar. 1934. A Method for the Solution of Oscillation Problems By Matrices. *Philosophical Magazine Series 7* 17, no. 115: 865-909.

———. 1935. Matrices Applied to the Motions of Damped Systems. *Philosophical Magazine Series 7* 19, no. 125: 197-219.

Dyka, C.T., and R.P. Ingel. 1995. An approach for tension instability in smoothed particle hydrodynamics (SPH). *Computers and Structures* 57, no. 4: 573-580.

Farin, G. 2002. *Curves and Surfaces for CAGD: A Practical Guide*. 5th ed. Morgan Kauffman.

Felippa, C.A. 2001. A historical outline of matrix structural analysis: a play in three acts. *Computers and Structures* 79, no. 14: 1313-1324.

Fernandez-Mendez, S., S., and A. Huerta. 2004. Imposing essential boundary conditions in mesh-free methods. *Computer Methods in Applied Mechanics and Engineering* 193, no. 12: 1257-1275.

Frazer, R.A., W.J. Duncan, and A.R. Collar. 1963. *Elementary Matrices and some Applications to Dynamics and Differential Equations*. 7th ed. Cambridge University Press.

Guangyao, L., and T. Belytschko. 2001. Element-free Galerkin method for contact problems in metal forming analysis 18, no. 1: 62-78.

Gupta, S.K., C.J.J. Paredis, and R. Sinha. 2001. Intelligent Assembly Modeling and Simulation. *Assembly Automation* 21, no. 3: 215-235.

Harris, T.A. 1991. *Rolling Bearing Analysis*. 3rd ed. Wiley-Interscience.

Hartmann, E. 1998. Numerical Implicitization for Intersection and G"-Continuous Blending Surfaces. *Computer Aided Geometric Design* 15: 377-397.

Heinbockel, J.H. 2006. *Introduction to the Variational Calculus*. Trafford Publishing

.

Hertz, H. 1896. On the Contact of Elastic Solids. In *Miscellaneous Papers*, 1:146-163. London: Macmillan.

Hoellig, K. 2003. *Finite Element Methods with B-Splines*. SIAM.

Hughes, T.J.R., J.R. Cotrell, and Y. Bazilevs. 2005. Isogeometric Analysis: CAD, Finite Elements, NURBS, Exact Geometry and Mesh Refinement. *Computer Methods in Applied Mechanics and Engineering* 194: 4135-4195.

Johnson, C. 1992. *Numerical Solution of Partial Differential Equations by the Finite Element Method*. Cambridge University Press.

Johnson, G.R., and S.R. Beissel. 1996. Normalized Smoothing Functions for SPH Impact Computations. *International Journal for Numerical Methods in Engineering* 39, no. 16: 2725-2741.

K. Hoellig, Reif, U., and J. Wipper. 2001. Weighted Extended B-Spline Approximation of Dirichlet Problems. *SIAM Journal On Numerical Analysis* 39, no. 2: 442-462.

Kantorovich, L.V., and V.I. Krylov. 1958. *Approximate Methods of Higher Analysis*. Interscience Publishers.

Krongauz, Y., and T. Belytschko. 1996. Enforcement of Essential Boundary Conditions in Meshless Approximations using Finite Elements. *Computer Methods in Applied Mechanics and Engineering* 131, no. 1: 133-145.

Lancaster, P., and K. Salkauskas. 1981. Surfaces Generated by Moving Least Squares Methods. *Mathematics of Computation* 37, no. 155: 141-158.

Lee, K., and G. Andrews. 1985. Inference of the positions of components in an assembly: part 2. *Computer-Aided Design* 17, no. 1: 20-24.

Lee, K., and D.C. Gossard. 1985. A hierarchical data structure for representing assemblies: part I. *Computer-Aided Design* 17, no. 1: 15-19.

Li, S., D. Qian, W.K. Liu, and T. Belytschko. 2001. A meshfree contact-detection algorithm. *Computer Methods in Applied Mechanics and Engineering* 190, no. 24: 3271-3292.

Liang, V.C., and C. Paredis. 2003. A Port Ontology for Automated Model Composition. In *Proceedings of the 2003 Winter Simulation Conference*, 1:613-622.

Liu, W.K., S. Jun, and Y.F. Zhang. 1995. Reproducing Kernel Particle Methods. *International Journal for Numerical Methods in Fluids* 20: 1081-1106.

Lo, S.H. 1991a. Volume Discretization into Tetrahedra -I. Verification and Orientation of Boundary Surfaces. *Computers and Structures* 39: 493-500.

———. 1991b. Volume Discretization into Tetrahedra - II. 3D Triangulation by Advancing Front Approach. *Computers and Structures* 39: 493-500.

Lohner, R. 1996. Progress in Grid Generation via the Advancing Front Technique. *Engineering with Computers* 12, no. 3: 186-210.

Lohner, R., P. Parikh, and C. Gumbert. 1988. Interactive Generation of Unstructured Grid for Three Dimensional Problems. In *Proceedings of the Numerical Grid Generation in Computational Fluid Mechanics '88*, 687-697. Pineridge Press.

Lucy, L.B. 1977. A Numerical Approach to the Testing of the Fission Hypothesis. *The Astronomical Journal* 82, no. 12: 1013-1024.

Mahbub, M. 2008. Development of TechSpec Model for Legacy Systems Engineering. Master's thesis, Tempe, AZ, USA: Arizona State University.

Marcheix, D., and G. Pierra. n.d. A survey of the persistent naming problem. In *Proceedings of the seventh ACM symposium on Solid modeling and applications*, 3-22. New York, NY, USA: ACM.

Monaghan, J.J. 1982. Why Particle Methods Work. *SIAM Journal of Scientific and Statistical Computing* 3, no. 4: 422-433.

———. 1988. An introduction to SPH. *Computer Physics Communications* 48, no. 1: 89-96.

———. 1992. Smoothed Particle Hydrodynamics. *Annual Review of Astronomy and Astrophysics* 30: 543-574.

Nayroles, B., G. Touzot, and P. Villon. 1992. Generalizing the Finite Element Method: Diffuse Approximation and Diffuse Elements. *Computational Mechanics* 10, no. 5: 307-318.

Nevins, J.L., and D.E. Whitney. 1989. *Concurrent Design of Products and Processes*. McGraw-Hill.

Noort, A., G.F.M Hoek, and W.F. Bronsvoort. 2002. Integrating Part and Assembly Modeling. *Computer-Aided Design* 34: 899-912.

Okubo, H. 1951. On the two-dimensional problem of a semi-infinite elastic body compressed by an elastic plane. *Quarterly Journal of Mechanics and Applied Mathermatics* 4, no. 3: 260-270.

Owen, S. n.d. A Survey of Unstructured Mesh Generation Technology. Research Survey. http://www.andrew.cmu.edu/user/sowen/survey/index.html.

Pahl, G., and W. Beitz. 1997. *Engineering Design: A Systematic Approach*. 2nd ed. Springer-Verlag.

Przemieniecki, J.S. 1968. *Theory of Matrix Structural Analysis*. Dover.

Rachuri, S. 2005. Information Models for Product Representation: Core and Assembly Models. *International Journal of Product Development* 2, no. 3: 207-235.

Rachuri, S., Young-Hyun Han, S.C. Feng, Fujun Wang, R.D. Sriram, and U. Roy. 2003. Object-

Rayleigh, J.W. 1870. In Finding the Correction for the Open End of an Organ-Pipe. *Philosophical Transactions* 161: 77.

Reif, U. 2006. B-Splines as Finite Elements. In . Century City, CA, USA.

Requicha, A.A.G. 1980. Representations for Rigid Solids: Theory, Methods, and Systems. *ACM Computing Surveys* 12, no. 4: 437-464.

Requicha, A.A.G., and H.B. Voelcker. 1982. Solid Modeling: A Historical Summary and Contemporary Assessment. *IEEE Computer Graphics and Applications* 2, no. 2: 9-24.
———. 1983. Solid Modeling: Current Status and Research Directions. *IEEE Computer Graphics and Applications* 3, no. 7: 25-37.

Ritz, W. 1908. Ueber eine neue Methode zur loesung gewisser Variationsprobleme der mathematischen Physik. *Zeitschrift fuer Angewandte Mathematik und Mechanik* 135, no. 1: 1-61.

Rocheleau, D.N., and K. Lee. 1987. System for interactive assembly modelling. *Computer-Aided Design* 19, no. 2: 65-72.

Rvachev, V.L. 1975. Method of R-functions in boundary-value problems. *International Applied Mechanics* 11, no. 4: 345-354.

Rvachev, V.L., T.I. Sheiko, V. Shapiro, and I. Tsukanov. 2000. On completeness of RFM solution structures. *Computational Mechanics* 25, no. 2: 305-317.

Sabin, M.A. 1997. Spline Finite Elements. Doctoral Thesis, Leeds, England: University of Leeds.

Saffari, H., and R. Tabatabaei. 2007. A Finite Circular Arch Element Based on Trigonometric Shape Functions. *Mathematical Problems in Engineering* 78507.

Shah, J. 2009. Mesh Free Simulation-Based Design of Mechanical Assemblies. NSF Proposal.

Shah, J., and M. Mantyla. 1995. *Parametric and Feature-Based CAD/CAM*. John Wiley & Sons.

Shapiro, V. 2001. Solid Modeling. In *Handboold of Computer Aided Geometric Design*, by G. Farin, J. Hoschek, and M.S. Kim, 473-518. Elsevier.

Shapiro, V., and I. Tsukanov. 1999a. Implicit functions with guaranteed differential properties. In *Proceedings of the fifth ACM symposium on Solid modeling and applications*, 258-269. Ann Arbor, Michigan: ACM new York.

———. 1999b. meshfree simulation of deforming domains. *Computer-Aided Design* 31, no. 7: 459-471.

Shepard, D. 1968. A Two-Dimensional Interpolation Function for Irregularly Spaced Points. In *Proceedings of the 1968 23rd ACM National Conference*, 517-524. New York, NY, USA: ACM.

Shepard, M.S., and K.G. Marcel. 1991. Three Dimensional Mesh Generation by Finite Octree Technique. *International Journal of Numerical Methods in Engineering* 32: 709-749.

Sodhi, R., and J.U. Turner. 1991. Representing tolerance and assembly information in a feature-based design environment. In *Proceedings of the ASME Design Automation Conference*, 32-1:101-108.

Swegle, J.W., D.L. Hicks, and S.W. Attaway. 1995. Smoothed Particle Hydrodynamics Stability Analysis. *Journal of Computational Physics* 116: 123-134.

Timoshenko, S.P. 1983. *History of the Strength of Materials*. Dover.

Turner, M.J. 1959. The direct stiffness method of structural analysis. AGARD Meeting. In . Aachen, Germany.

Turner, M.J., R.W. Cough, H.C. Martin, and L.J. Topp. 1956. Stiffness and Deflection Analysis of Complex Structures. *Journal of Aeronautical Sciences* 23, no. 9: 805-823.

Voelcker, H.B. 1974. *An Introduction to PADL: Characteristics, Status, and Rationale*. Technical Memo. Production Automation Project. University of Rochester.

———. 1978. The PADL-1.0/2 System for Defining and Displaying Solid Objects. In *Proceedings of Sisgraph '78*, 12:257 - 263. NewYork, NY, USA: ACM.

Whitehead, T. 1954. *Design of Instruments & Accurate Measurements*. Dover.

Whitney, D.E. 2004. *Mechanical Assemblies*. Oxford University Press.

Woo, M., J. Neider, T. Davis, and D. Shreiner. 1999. *OpenGL Programming Guide: The Official Guide to Learning OpenGL Version 1.2*. 3rd ed. Addison-Wesley.

Wriggers, P. 2002. *Computational Contact Mechanics*. John Wiley & Sons.

Wriggers, P., and Panagiotopoulos, P.D. 1999. *New Developments in Contact Problems*. Springer-Verlag.

Yerry, M.A., and M.S. Shepard. 1984. Three-Dimensional Mesh Generation by Modified Octree Technique. *International Journal for Numerical Methods in Engineering* 20: 1965-1990.

Zienkiewicz, O.C. 1995. Origins, Milestones and Directions of the Finite Element Method – A Personal View. *Archives of Computational Methods in Engineering* 2, no. 1: 1-48.

Zienkiewicz, O.C., and R.I. Taylor. 2000. *The Finite Element Method*. 5th ed. Oxford University Press.

APPENDIX A

A MESH-FREE B-SPLINE BEAM ELEMENT

One of the biggest differences between the mesh-free finite element formulation of Shapiro/Höllig and the traditional mesh-based formulation is the support of the basis functions. Traditional finite element bases are defined strictly within an element domain. In the mesh-free method, by contrast, the basis functions are defined over a global support grid which need not conform to the domain geometry. Shapiro referred to this as the "Solution Space", as opposed to the "Domain Space" (Shapiro and Tsukanov 1999a). In this sense, the structured grid over which the B-Splines are defined is analogous to an Eulerian grid. In terms of the differential equation, the boundary conditions and solution are referred to this space. The approach is easily demonstrated with a 1-dimensional beam solution.

The differential equation for the Euler-Bernoulli beam is:

$$EI\frac{\partial^4 v}{\partial x^4} = p(x) \tag{A.1}$$

Where v is the transverse deflection, E is Young's Modulus, I is the cross-sectional moment of inertia, and p is the load per unit length of the beam. The weak form of this equation is:

$$\int_L a\frac{\partial^2 f(x)}{\partial x^2}\frac{\partial^2 f(x)}{\partial x^2}dx = \int_L p(x)f(x)dx \tag{A.2}$$

Where f is the trial or shape function. We will use Cardinal B-Splines as trial functions (following Höllig). This beam will be interpolated over a 1-dimensional grid with cell dimension h. Next, we define scaled, translated Cardinal B-Splines over this grid:

$$N^n_{k,h}(x) = N^n(x/h - k) \tag{A.3}$$

The support of a any particular B-Spline , N^n_k is:

$$Support\ N^n_k = [k, k+n+1]h \tag{A.4}$$

By way of example, suppose we have a beam of length *L*. We select the simplest problem from small deflection beam theory , as shown in Figure A-1.

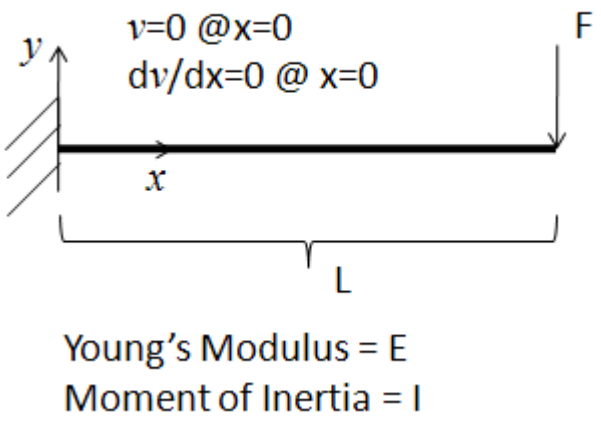


Figure A.1. Cantilever Beam Problem

If we want the beam to span 1 element, we set *h*=*L*. Let us also arbitrarily decide that the beam extends from *L* to 2*L* in the grid system. The point to be made here is that a fixed B-Spline grid will be used to interpolate the beam, but the beam may be located anywhere within this grid. By equation (A.4), if one chooses B-Splines of degree 3, the support of one basis function N^3_k will span [*k*,*k*+4]*L* (we choose a 3rd degree B-Spline because we know in advance that the solution is of degree 3. Note also that the index *k* refers to the "left end", or beginning, of the support). Thus, there will be exactly *n*+1=4 basis functions whose support overlaps the beam domain. In other words, there will be 4 basis

functions which are capable of interpolating this domain, and the combined support of all 4 basis functions spans the grid points [*k*-*n*,*k*+1+*n*]*L*. For this beam example, therefore, the support is [-2,5]L. Now, the bases are calculated according to the recurrence relation for B-Splines (Hoellig 2003):

$$N^n(x) = \frac{x}{n} N^{n-1}(x) + \frac{n+1-x}{n} N^{n-1}(x-1) \tag{A.5}$$

$$N^n(0) = \begin{cases} 1 & 0 < x \le 1 \\ 0 & x = 0 \end{cases}$$

combined with the grid transformation (A.3). The four cubic B-Splines that overlap the beam's domain are shown graphically in Figure A-2.

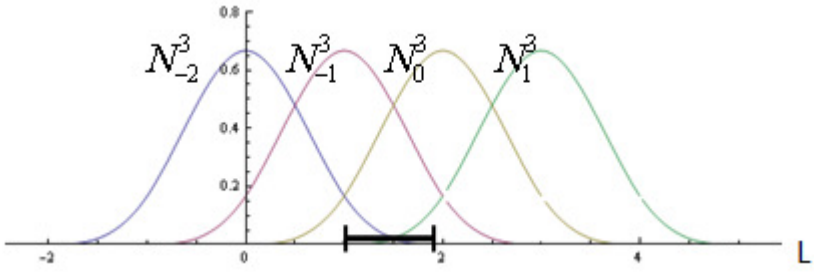


Figure A.2. Mesh-Free Beam Showing Basis Support

It is important to note that the beam's basis support extends beyond the physical domain of the beam [*L*,*2L*]. It should also be mentioned that, unlike the case of mesh-based bases, the basis coefficients do not correspond to any particular fixed location. These facts are reflected in the violation of the Kronecker delta property of bases discussed in section 2.4. One direct consequence of this is that one cannot immediately satisfy the requirement of finite element bases that $f=\varphi$ (the prescribed boundary condition) on the boundary. A simple solution to this dilemma for homogeneous equations is to modify the

bases by multiplying them with a smooth function that goes to zero on the boundary. The modified trial functions must be of the form (will now remove the superscript denoting degree of the trial functions, as well as the subscript, h denoting grid cell size):

$$f_i^{'} = wN_i + \phi \tag{A.6}$$

where the b_i are the unweighted basis functions (in our case, cardinal B-Splines), and φ is a function which takes on the value φ_0 at the boundary (this allows us to handle non-homogeneous problems). The function w must go to zero on the boundary. For the problem stated in Figure A-1, the two essential boundary conditions require that $f'=0$ at $x=L$ and $df'/dx=0$ at $x=L$ (in global grid coordinates). A simple weighting function, w that satisfies this requirement is:

$$w = \begin{matrix} \frac{x^2+1}{2} - x \\ 0 \end{matrix} \begin{cases} L \le x \le 2L \\ elsewhere \end{cases} \tag{A.7}$$

with φ=0 (this problems happens to be homogeneous). The new weighting functions, f' are shown in over the domain [*L*,2*L*] in Figure A-3.

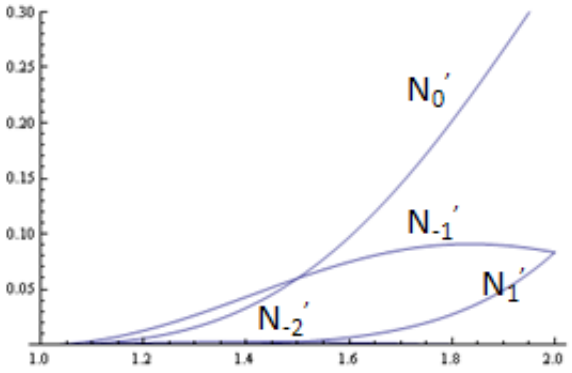


Figure A.3. Weighted Basis Functions over Beam Length

We now discretize equation (A.2) by replacing the trial functions with the weighted basis as in section 2.4.1:

$$\left(\int_{T_i} \frac{\partial^2}{\partial x^2} (\omega \mathbf{N})^T \frac{\partial^2}{\partial x^2} (\omega \mathbf{N}) \, dT_i \right) \mathbf{\Phi} = \int_{T_i} p(x) \omega \mathbf{N} \, dT_i \tag{A.8}$$

Which produced the familiar algebraic system:

$$\mathbf{K\Phi} = \mathbf{F} \tag{A.9}$$

Where:

$$\mathbf{K} = \left(\int_{T_i} \frac{\partial^2}{\partial x^2} (\omega \mathbf{N})^T \frac{\partial^2}{\partial x^2} (\omega \mathbf{N}) \, dT_i \right) \tag{A.10}$$

And, for the problem outlined in Figure A-1, the distributed load p(x) is replaced by a point load F, resulting in the load vector:

$$\mathbf{F} = \int_{T_i} F \delta(x - x_0) \omega \mathbf{N} \, dT_i \tag{A.11}$$

Where δ is the Dirac Delta function.

Performing the derivatives and integrals indicated in (A.10) and (A.11) in Mathematica®, gives the following 4 x 4 system:

$$\frac{EI}{L^3} \begin{bmatrix} 1/420 & 1/840 & -1/105 & 1/168 \\ 1/840 & 68/315 & -257/2520 & -25/126 \\ -1/105 & -257/2520 & 713/1260 & 107/504 \\ 1/168 & -25/126 & 107/504 & 25/63 \end{bmatrix} \begin{Bmatrix} \Phi_1 \\ \Phi_2 \\ \Phi_3 \\ \Phi_4 \end{Bmatrix} = - \begin{Bmatrix} 0 \\ 1/12 \\ 1/3 \\ 1/12 \end{Bmatrix} F \tag{A.12}$$

Solving this system yields:

$$\begin{Bmatrix} \Phi_1 \\ \Phi_2 \\ \Phi_3 \\ \Phi_4 \end{Bmatrix} = \begin{Bmatrix} -4/3 \\ -1 \\ -2/3 \\ -1/3 \end{Bmatrix} \tag{A.13}$$

The normalized solution $\mathbf{N}^T(x)\mathbf{\Phi}$ is plotted in Figure A-4

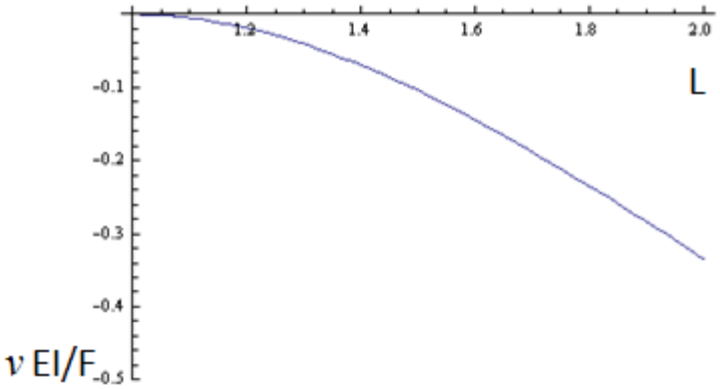


Figure A.4. Beam Solution – Normalized Deflection vs. Length

This is identical to the exact analytical solution:

$$\frac{EI}{F}v = \frac{1}{L}\left(\frac{-x^3}{L^2 6} + \frac{x^2}{L} - \frac{3}{2}x + \frac{2L}{3}\right) \quad L \le x \le 2L \tag{A.14}$$

APPENDIX B

MATRIX STABILIZATION BY BASIS NORMALIZATION

One problem with using the weighted B-Spline space $f_i^{'} = wN_i + \phi$ is that B-Spline values go to zero at the extremes of their supports. Therefore, if a domain just barely overlaps a B-Spline's support, the B-Spline contribution to the solution may be very small within the overlap region, leading to ill-conditioned systems. Simply ignoring such contributions could lead to poor accuracy (this is especially important in the case of the current research effort because contact surfaces will by definition be influenced by outer splines). Höllig et al (Hoellig 2003) has apparently devised a robust solution to this dilemma with the invention of WeB-Splines (see section 3.2.3). We will not use this method, however. Instead, we propose using a process of selective basis normalization. The process will work as follows:

1. Identify inner and outer grid cells
2. Identify B-Splines whose support overlaps an outer cell at just one corner
3. Normalize B-Splines identified in 2.

And the normalization itself is performed according to:

$$N_i^{'} = \frac{N_i}{\left\| N_i \right\|_{\infty, Q' \cap D}} \tag{B.1}$$

Where $\left\| N_i \right\|_{\infty, Q' \cap D}$ is the infinity norm (or simply the maximum absolute value) of N_i over the region of intersection between the outer grid cell (Q') and the domain (D).

As an illustration, we return to the mesh-free beam of Appendix A, and shift the domain by a small amount (x-0.1)h. The length L is the same, and we

keep the same grid width $h=L$. The beam's domain now spans the interval $[.9,1.9]L$. Shifting the domain in this way has picked up a new overlapping basis function (this is now effectively a two-beam solution) as shown in Figure B-1. The support now spans $[-3,5]L$. The new basis basis function (N_{-3}), however, only covers the domain over the interval $[.9,1]L$. The B-Spline value at $0.9L$ is very small (approximately 1/1000 the maximum value). When we follow the solution procedure described in Appendix A, the new matrix, K is:

$$\mathbf{K} = \begin{bmatrix} 2.381e-10 & 7.619e-9 & -1.595e-8 & 8.095e-9 & 0 \\ 7.619e-9 & 0.0046 & 0.002 & -0.015 & 0.007 \\ -1.595e-8 & 0.002 & 0.273 & -0.156 & -0.191 \\ 8.095e-9 & -0.015 & -0.156 & 0.669 & 0.252 \\ 0 & 0.007 & -0.191 & 0.252 & 0.256 \end{bmatrix} \tag{B.2}$$

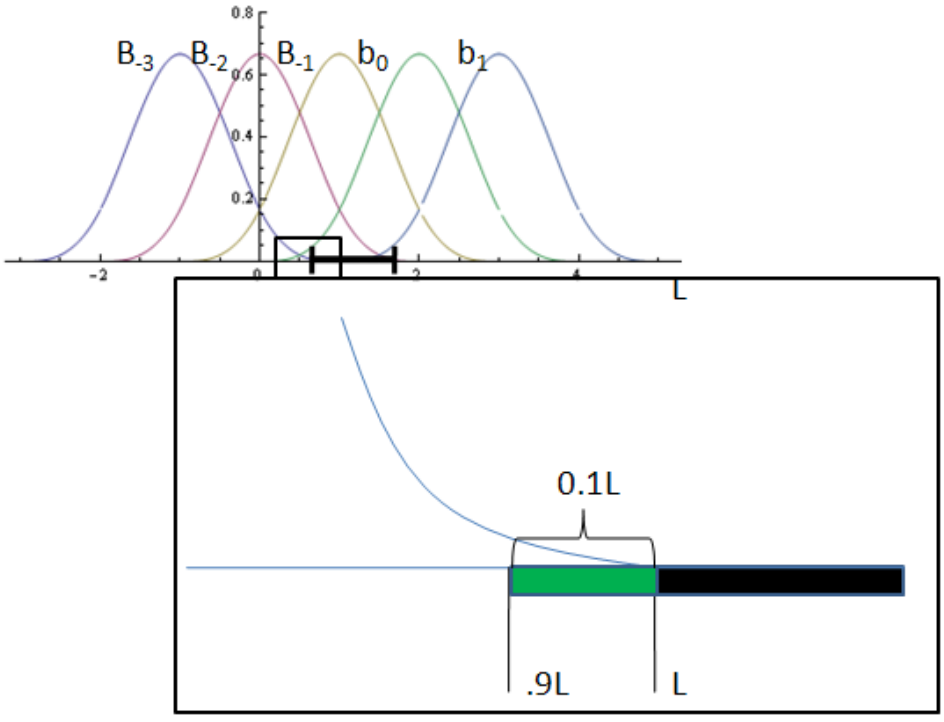


Figure B.1. Shifted Beam showing Basis Functions

which results in a condition number of

c=4.586e9

Such a large condition number suggests an ill-conditioned matrix. To fix the problem, we apply procedure 1 through 3 from above:

1. Identify inner and outer grid cells. According to our strict definition, we have two outer grid cells and no inner cells. They are on the intervals $[0,1]L$, and $[1,2]L$.
2. Identify B-Splines whose support overlaps an outer cell at just one corner. The B-Spline which overlaps the cell $[0,1]L$ at only one "corner" is b_{-3}. The B-Spline which overlaps the cell $[1,2]L$ at only one corner is b_1.
3. Normalize B-Splines identified in 2.

The maximum value of b_{-3} over the interval $[0.9,1]L$ is 0.00067. We thus normalize b_{-3} according to:

$$N'_{-3} = \frac{N_{-3}}{0.00067} = 6000\, N_{-3} \tag{B.3}$$

The maximum value of b1 over the interval $[1.9,2]L$ is 0.1215. Correspondingly:

$$N'_{1} = \frac{N_{1}}{0.1215} = 8.23\, b_{1} \tag{B.4}$$

Once equation (B.3) and (B.4) are applied, we calculate K again:

$$\mathbf{K} = \begin{bmatrix} 0.009 & 4.562e-5 & -9.552e-5 & 4.847e-5 & 0 \\ 4.563e-5 & 0.0046 & 0.002 & -0.015 & 0.007 \\ -9.552e-5 & 0.002 & 0.273 & -0.156 & -0.191 \\ 4.847e-5 & -0.015 & -0.156 & 0.669 & 0.252 \\ 0 & 0.007 & -0.191 & 0.252 & 0.256 \end{bmatrix} \tag{B.4}$$

The condition number of this matrix is:

c=511.35

which is generally acceptable for single precision arithmetic (it results in minimum of 6-digit accuracy). Solving this system results in a deflection plot over the interval [0.9,1.9]L which is indistinguishable from that in Appendix A.

BIOGRAPHICAL SKETCH

Mr. Grishin has been working as a Mechanical Engineering Analyst and Consulting Engineer for an engineering services firm in Tempe, AZ for the past 12 years. He holds a Bachelor of Science and Master's Degree in Mechanical Engineering from North Carolina State University.

He is currently single and lives alone. In addition to Engineering Science, his interests include pure and applied mathematics, computer science, physics, philosophy, history, literature, poetry, and film. He enjoys traveling, camping, hiking, and moutain biking – all of which he intends to pursue more intensely in the future.