15

Resolution of Grain Scale Interactions Using the Discrete Element Method

by

Scott Matthew Johnson

Submitted to the Department of Civil and Environmental Engineering
in partial fulfillment of the requirements for the degree of

Doctor of Philosophy in Civil and Environmental Engineering

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

February 2006

Author ....
Department of Civil and Environmental Engineering
December 15, 2005

Certified by ....
John R. Williams
Associate Professor of Civil and Environmental Engineering and
Associate Professor of Engineering Systems
Thesis Supervisor

Accepted by ....
Andrew J. Whittle
Chairman, Department Committee for Graduate Students

Resolution of Grain Scale Interactions Using the Discrete Element Method

by

Scott Matthew Johnson

Submitted to the Department of Civil and Environmental Engineering
on December 15, 2005, in partial fulfillment of the
requirements for the degree of
Doctor of Philosophy in the Field of Civil and Environmental Engineering

Abstract

Granular materials are an integral part of many engineering systems. Currently, a popular tool for numerically investigating granular systems is the Discrete Element Method (DEM). Nearly all implementations of the DEM, however, use spheres to represent particles despite mounting evidence showing that shape at multiple scales (sphericity, angularity, and friction) plays a role in granular material behavior.

This thesis contributes a new non-spherical representation to model particles as ellipsoidal bodies. This is validated and benchmarked against current representations and is shown to have attractive computational efficiency and numerical stability. A numerical study of the formation of heaps using spheres and ellipsoids both validates the ellipsoid representation and illustrates shape-induced behavioral differences.

Resolution of shape is extended by a new algorithm for a hierarchical, multi-scale representation of convex particle surface characteristics. Two applications are offered: (1) a micro-asperity model is used to demonstrate pair-wise interlocking, and (2) a surface-based cohesive contact law is validated using a series of virtual numerical pull-off tests, which agree well with experimental findings.

An explicit quadrature algorithm based on quaternion rotation is developed and shown to more accurately determine rotational orientation with less computational effort than other common algorithms for integrating finite rotations.

Finally, a contact resolution algorithm between discrete elements and a polyhedral boundary is developed and shown to scale in $O(M + N)$ versus common algorithms with scaling of $O(NM)$, where N is the number of discrete elements and M the number of faces on the polyhedral boundary. These developments are illustrated with numerical studies to simulate the blending kinetics of cohesive, micron-scale pharmaceutical powders in V-shaped and cylindrical bench-scale blenders.

Thesis Supervisor: John R. Williams
Title: Associate Professor of Civil and Environmental Engineering and Associate Professor of Engineering Systems

Acknowledgments

John Williams, thank you for being my advisor, for your persistence, and for enduring a sometimes truculent advisee. I look forward to working together more in the future. Prof. Connor, thank you for agreeing to serve on my committee at such a late hour. Your dedication to your students and to the discipline is something to which all of us should hope to aspire. My special thanks also go to Ruaidhri O'Connor and Charles Cooney for serving on my committee and helping to shape this work. Ben Cook, thank you for all of the advice, criticism, and support, which have helped me immeasurably to complete this thesis.

Sandia National Laboratories has been incredible, providing me with not only the financial support to me as a Sandia Fellow in Engineering Systems but also with a wealth of experience and excellent advising through summer internships at the laboratory. I especially appreciate the support from the geomechanics group (org.6117).

Prof. Charles Cooney and his lab were also instrumental in providing the extensive experimental work that underlies the pharmaceutical modeling part of this thesis. Reuben Domike began this work with us, and Samuel Ngai has been an incredible collaborator for the past few years. Samuel's tireless user testing of the 3-D DEM application from its earliest prototypes has been, well, superhuman. Lakshman, Yu, and Mridula, it has been great working with you on this, and I hope we can carry this work on to another group of students.

Joanie, you are great! Thanks for looking out for me and for all of us in the lab. For all of my colleagues in the (now former) IESL, we had a lot of fun in the waning days of the lab. Thanks for keeping everything copacetic: Bharath, Sudarshan, Stefan, Ying-Jui, Anand, Victor, and Julio.

Cynthia, James, and Jeanette, you have always been there to talk with and to help me to sort out all of the graduate policies. Donna, you are incredible; thanks for all of the help in years of untangling my fellowship support. Anthee, you have been great source at arranging everything: rooms, meetings, bbq's, power cords. Is there

anything you don't do?

I would like to thank my wonderful Monica for all of her support throughout my Ph.D. Monica, you are my fire and my life, and I am looking forward to all of the adventures we are going to have together!

I also owe an untold amount to my parents. Without their patience, insights, and wonderful value for education, I wonder whether my curiosity for learning would ever have emerged.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Why do we care about granular materials? From both a practical and academic persepctive granular materials hold a great deal of interest because of the pervasiveness and continuing mystery of large particle system interactions. This thesis attempts to provide a framework that can address problems that serve both the practical and academic dimensions.

In practical terms, there are over one trillion (10^{12}) kilograms of granular material produced each year in the U.S., and 61 billion kilograms are linked to granular material technology in the chemical industry alone. U.S. pharmaceutical companies, for instance, produced nearly US$220 billion in sales in 2002 [83]. The pharmaceutical industry is one of the most heavily regulated in the United States. Moreover, the industry is currently undergoing a regulatory transition, with guidelines for the uniformity of finished products stipulated in the Current Good Manufacturing Practices (CGMP) set forth in 21 CFR 211 [1]. To comply with these regulations, quality control departments dispose of an undisclosed value of pharmaceutical product every year. Informal estimates of the waste in pharmaceutical manufacturing range from 10-50% depending on the product line and product process maturity. This is mainly due to an inability of quality assurance departments to adequately predict the behavior of a new pharmaceutical during the manufacturing process, requiring the use of empirical and trial-and-error methods to calibrate the manufacturing process for each new pharmaceutical product. This set of regulations now not only requires manufacturers to ensure the quality of the outgoing product but to also explain why a batch fails.

Where do we stand currently in the ability to model granular materials? Modeling granular materials in general has seen maturation over the past decade, as computational methods for evaluating discontinuous systems have been developed and refined. Coupled with tremendous leaps in the availability of computational resources, many problems of industrial size (10^8 and greater numbers of particles) can now be addressed using particle methods such as the discrete element and related methods on readily available cluster computers. However, the drive to simulate ever larger numbers of particles has come at a price in the form of less detailed contact resolution and simplifications of the modeled geometry. The overwhelming majority of simulations use spheres as the primitive approximation of the grain and use the simplest contact laws possible to increase speed. This is the basis for the popular codes TRUBAL [20], PFC3D [21], DMC [115], and the granular dynamics extension to LAMMPS [98]. Despite mounting research suggesting the key role of particle shape, angularity, and friction in determining granular behavior [12, 124, 93, 99, 91, 89, 76, 74, 68, 42], sphere-based DEM remains, with a handful of exceptions, the dominant simulation method. Using systems of spheres, even agglomerates of spheres, often do not accurately capture the mechanisms operating in the modeled system. As Favier [28] shows, it is difficult to capture even the behavior of real systems of ellipsoids using the sphere cluster representation.

Though the fidelity of a sphere-based model is not ideal, few particle representations have been offered as an alternative. Arbitrary geometric representations, such as the discrete function representation (DFR) of O'Connor [130] as well as dilated sphere models [40, 56] and ellipsoidal models [66, 92, 127, 46] have been proposed. The largest obstacle is the computational intensity required by the resolution of contact for aspherical primitives. Until now, the number of particles that can be represented by smooth ellipsoids, for instance, is orders of magnitude less than that for spheres. Even for those formulations that can represent non-spherical primitives, there is little practical ability to capture sub grain scale physics, excepting micro-scale frictional interactions through the use of Coulombic laws.

This practical need to model several resolution scales of the grain surface (e.g., asphericity, angularity, and friction) in DEM is addressed in this thesis through the development of efficient algorithms for interacting ellipsoids and representation of surface asperities on convex bodies. A physical parameter separation of not only the scales of asphericity is given but also of the physical parameters influencing contact (e.g., surface normals, deviations of surface height, and cohesive properties). Though algorithms exist to address both of these issues (e.g., representations for analytical ellipsoidal representations [66, 122, 127], superquadric representations [5, 131], star convex/concave surface representations via DFR [130], or arbitrary polyhedral representations [78]), only arbitrary polyhedra, which are limited to a low number of faces per body (typically blocks of 6 faces), are in common use due to the computational requirements of the algorithms.

The algorithms introduced in this thesis are shown to significantly reduce the computational requirements, as demonstrated in Section 4.4, while presenting new ways of modeling both the primitive used (ellipsoidal) as well as modeling sub grain-scale physics over multiple scales using a simple hierarchical surface model. The practical applications of the sub grain scale model are illustrated through the development of a micro-asperity model for representing geometric interlocking and through the development of an experimentally validated cohesion model for use in describing the forces acting between a bonded pair of micron scale grains at the micro and meso length scales.

Broadening the application domain from single particle interactions to whole systems, this thesis presents two numerical applications based on different parts of the approach. These focus on two areas: the formation of granular heaps and pharmaceutical powder blending. In the first set of simulations, conclusions are drawn for the dynamics of heaps and how exclusion of asphericity can affect the validity of DEM results. The second set of experiments uses models to describe micro- and meso-length scale bonding effects between powder grains with conclusions on the effect of cohesion on the mixing behavior of pharmaceutical powders. This research study, conducted in cooperation with the experimental and computational facilities of collaborators in the Department of Chemical Engineering at MIT, demonstrates how this research fits into a larger coupled experimental and numerical approach for investigating the multi-scale behaviors of cohesive powder systems.

Before investigating the physical and engineering aspects of this work, however, the development of the software framework is discussed. Many academic software applications suffer a common fate. They were designed as academic code to functionally solve a particular problem for a particular researcher, so the resultant code is often cryptically written and optimized for speed rather than maintainability or extensibility. This thesis first attempts to show, through the analysis of an existing particle code, that a focus on maintainability and extensibility can result in a lower overall lifetime cost for a code and, in many cases, offer a faster time to solution. This concept is demonstrated by the resulting DEM3D application, which focuses on the abstraction of a particle method code, employing concepts of code organization and modulization to make the code as usable as possible to researchers, code integrators, and code developers.

In the pursuit of ever larger systems, a critical component of numerical analysis is often forgotten: analysis of the resultant data. Without critical analysis tools, much of the computational work performed during simulation results in unused data. The literature is fraught with examples of large simulation runs over multiple nodes and multiple days resulting only in macroscopic estimates of granular temperature or density. Particle trajectory, force chain formation, and internal stress states are often ignored for large-scale simulations. This thesis offers a platform for mining the data in DEM data sets using knowledge of the problem domain to provide researchers with meaningful statistics and visualizations. The usefulness of this capability is illustrated through the work on stress formation in granular heaps, which would not have been possible without such a facility.

1.1 Motivation and Contributions

The main motivation of this thesis is to add greater resolution to the discrete element method at the grain scale and to determine how this added resolution affects certain systems of engineering interest, including pharmaceutical powder blending kinetics and the development of stress chains in granular heaps during formation. A secondary motivation is to develop a DEM software application that not only implements the advancements detailed in this thesis but also is efficient, extensible, and maintainable by several classes of external developer.

The resulting thesis has answered several outstanding questions and provided a new way of probing systems composed of rough, aspherical particles. In particular, it contributes the following:

Shape Resolution

1. Formulation, validation, and benchmarking of a new representation for ellipsoidal particles including contact resolution is discussed in Chapter 4.

2. Formulation, validation, and benchmarking of a new contact detection algorithm for determining contact between discrete elements and an arbitrary polyhedral boundary element is discussed in Chapter 8, which is illustrated through the modeling of a V-mixer and a double helical ribbon blender in Section 9.2.2.

3. Development of a hierarchical, multi-scale surface representation for discrete elements is discussed in Chapter 7. Applications are discussed for the modeling of geometric interlocking between micro-scale surface asperities in Section 7.2 and of cohesive properties of individual surface points in Section 7.3.

Quadrature

1. Formulation, validation, and benchmarking of a new explicit integration algorithm for integrating the finite rotations of bodies under external forcing is discussed in Chapter 5.

Software

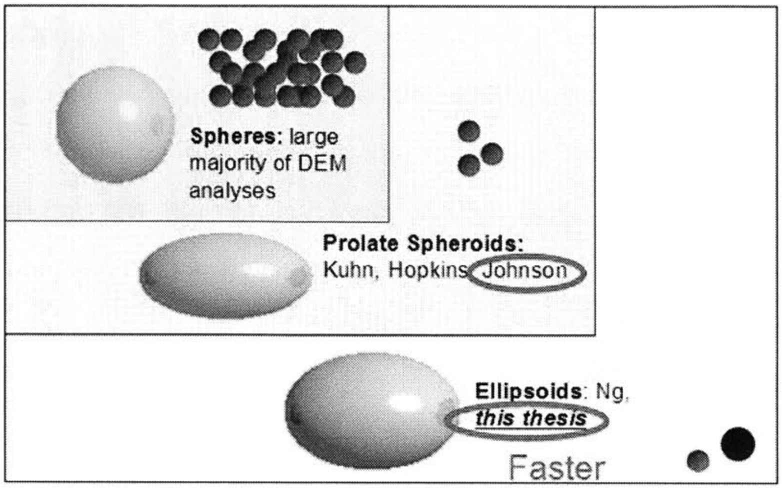

Spheres: large majority of DEM analyses
Prolate Spheroids: Kuhn, Hopkins, Johnson
Ellipsoids: Ng, this thesis
Faster

Figure 1-1: Contributions of this thesis in representing smooth, aspherical bodies.

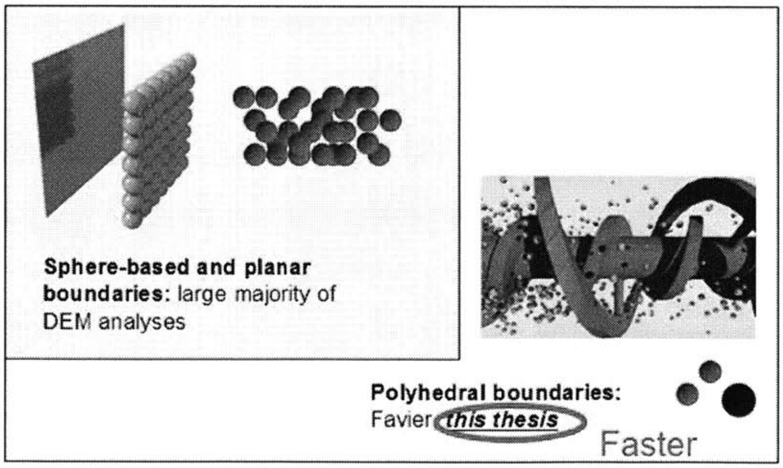

Sphere-based and planar boundaries: large majority of DEM analyses
Polyhedral boundaries: Favier, this thesis
Faster

Figure 1-2: Contributions of this thesis in representing arbitrary boundaries.

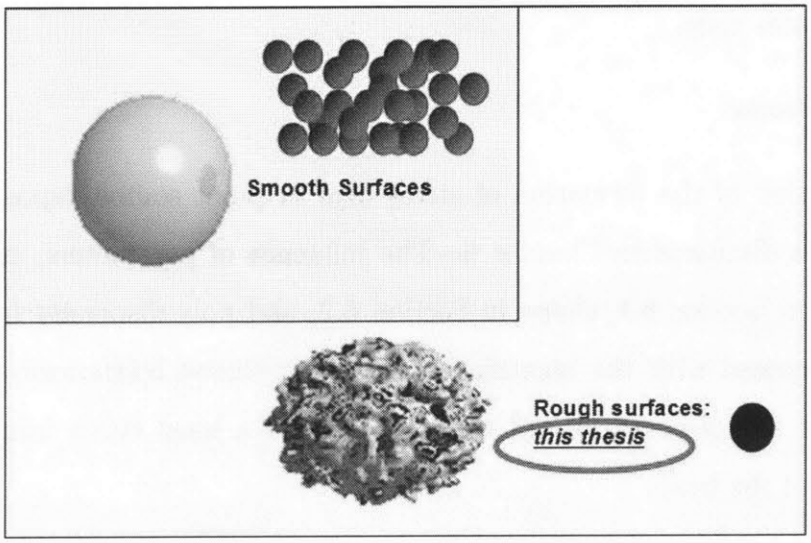


Figure 1-3: Contributions of this thesis to representing surface features at the subgranular scale.

1. Development of a consistent, object-oriented framework for performing DEM simulations is discussed in Chapter 3. This framework combines data analysis, data archiving, simulation, visualization, and user interface into separate modules. It also provides well-abstracted interfaces and a dll manager to allow for ease of extension and ease of maintenance.

2. Implementation of a class hierarchy to interface the DEM simulation with database technologies and provide data analysis/visualization is discussed in Chapter A.

3. Application of a testing (validation/verification) framework into a DEM simulator is discussed in Section 3.2.

4. Case study of LAMMPS, an open source molecular/granular dynamics simulator, is discussed in Section 3.1. This discussion illustrates the value proposition for using level 2.5 languages [1] and object-oriented design principles in the design

[1] The level of a programming language is a categorization of language abstraction. These loose categories (levels) are often subdivided according to the average number of lines of code (LOCs) required to define a function point (FP), or atomic function, as referenced in the tables of Jones [47].

of research code.

Applications

1. Simulation of the formation of stress dips in point source deposited granular heaps is discussed in Chapter 6. The influence of parameters, such as source height in Section 6.1, shape in Section 6.2, and poly-dispersity in Section 6.3, are discussed with the identification of a correlative relationship between the average age of contacts and the formation of a local stress minimum at the center of the heap.

2. Experimental and numerical studies of pharmaceutical powder processing is discussed in Chapter 9. These studies have been performed in collaboration with the laboratory of Prof. Charles Cooney, Department of Chemical Engineering, MIT, and have shown qualitative agreement between experimental results from a bench-scale pharmaceutical powder blender and numerical results using a stochastic cohesive contact law co-developed with Dr. Samuel Ngai.

Chapter 2

Discrete Element Modeling

2.1 Discontinua versus Continua

In a continuum approach, such as the finite difference method or finite element method (FEM), the microscopic behavior is effectively viewed as uncoupled from the macroscopic behavior. Representation element volumes (REV's) are used to define the minimum scale below which the grain-scale behavior cannot be decoupled using constitutive models. Empirically-based constitutive equations can then be used with a mesh of elements of the scale of the REV to determine the behavior of a larger continuum. Using this approach, many problems in large-scale granular materials can be addressed when careful attention is paid to the design of the consitutive equations. A typical application of FEM in soil mechanics is for the prediction of pile stress as illustrated in Figure 2.1.

The most popular continuum model is the finite element method (FEM). The continuum assumption is quite valid for many materials, and the fields of mechanical, material, and structural engineering have been fundamentally changed by FEM. There are certain systems, though, for which the length scale of interest is not significantly larger than the grain size or where the behavior of the system of interest is significantly affected by discontinuous grains that compose the system. At this point homogenization efforts break down with the REV becoming coincident with the individual particles. For instance, sanding in oil wells is a multi-billion dollar

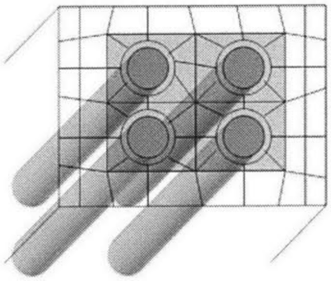


Figure 2-1: Illustration of a continuum-based analysis of building piles interacting with the underlying soil

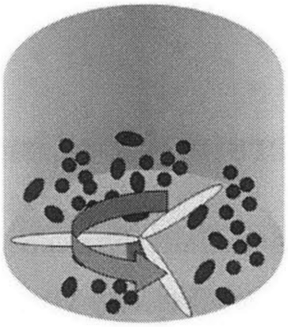


Figure 2-2: Illustration of a discretely modeled powder blender

problem for energy companies, and much research has been dedicated to designing new FEM models to predict the onset and severity of sanding. However, analytical and FEM models have been unable to satisfactorily predict well-bore stability in actual oil wells [132]. Continuum methods also can offer little insight into the material behavior of the bore hole, since the micromechanical behavior is smoothed into a continuum instead of the discontinuous media (sand, soil, etc.) of which well-bores consist.

Unlike continuum approaches, discrete approaches, such as discontinuous deformation analysis (DDA), the discrete element method (DEM), cellular automata (CA), smoothed particle hydrodynamics (SPH), and molecular dynamics (MD), model a statistically valid sample of discontinuous constituents of a physical system. The theory is that by properly modeling the behavior of the individual constituents and their interaction behavior at the microscale, the correct emergent behavior of the system

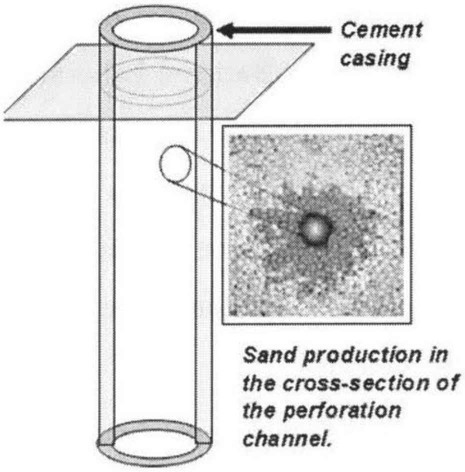

Cement casing
Sand production in the cross-section of the perforation channel.

Figure 2-3: Illustration of the near-borehole region modeled using the SandFlow2D DEM code; simulation image courtesy of B. Cook, Sandia National Laboratories

at the mesoscale can be accurately captured. Using the example of well-bore stability, DEM has been employed in studying the problem as illustrated in Figure 2.1. Though DEM has also been unable to quantitatively predict the onset of sanding, its microscopic resolution has been instrumental in uncovering several micromechanical mechanisms responsible for sanding, as shown by O'Connor [90], Cook [17], and Boutt [10]. The application focus of this thesis is on capturing the qualitative behaviors of granular materials in two specific areas: modeling the behavior of granular materials in mixers (see illustration in Figure 2.1) and investigating the stress fabric behavior in granular heaps.

2.2 Theory of DEM

Discrete element modeling (DEM) was first proposed in 1979 by Cundall and Strack [22], and the theoretical basis of the method was proven later by Williams, Hocking, and Mustoe [128]. Because of its capability to model physics at the grain scale, DEM has become an important computational method for modeling complex phenomena exhibited by particulate systems. The applications for DEM have ranged across several disciplines, including the simulation of the manufacture of pharmaceuticals, modeling of bore hole flow for the petrochemical industry (e.g., work by O'Connor [90] and by Cook and Boutt at Sandia National Laboratories [17, 19, 18, 10]), study of flow and packing in applied mathematics [15], granular rheology in applied physics [34], and prediction of pebble bed reactor residence time in nuclear engineering [50]. As computational resources have become more readily available to a wide range of researchers, DEM and other particle methods have begun to play a greater role in analyzing many engineering systems.

However, there remains a significant scale gap in the research of granular materials. Soil mechanics, for instance, has traditionally focused on continuum modeling of granular material, adapting microscale theory to the development of constitutive laws for bulk materials. For the area of interest of most soil mechanics research, which is of the order $10^0 - 10^3 m$, the continuum assumption has worked well. Because the valid representative element volume (REV) of interest for many of these problems is of a much coarser scale than the grain size, this research strategy has largely neglected meso-scale modeling, where inherent variability of the systems they are analyzing and the exceptional computational cost of modeling such systems have been barriers. However, attempts have been made to model localized behavior of soils (such as the local sand region around an oil well borehole) using continuum models with poor results. The lack of focus on mesoscale modeling is understandable, since sand, one of the largest particles of interest in soil mechanics, is on the order of $1mm^3$, and a $1m^3$ sample would contain on the order of 10^9 particles. This number of particles, even idealized as perfect spheres, which in many instances is unacceptable, would be extremely costly to evaluate computationally. Also, the coupling between the scale of interest and the grain length scale is often weak; i.e., the grain size $<<$ REV size.

Other researchers have focused on micromechanical response of the material at the scale of a particle pair. This has generally been divided into analytical research and, more recently, experimental work. Analytical studies of interparticle forces between spheres have been performed, such as the groundbreaking work on physical contact forces of Mindlin and Deresiewicz [72, 73]; adhesion and Van der Waals interaction via the Johnson, Kendall, and Roberts (JKR) model [44]; and the well-studied liquid bridging work of Lian and Thornton [63]. These solutions are used in mathematical dispersion models or continuum models to predict the behavior of particles under different conditions, such as chemical powder processing applications. However, there is a lack of qualitative agreement between experimental observations and the continuum model predictions based on the theoretical behavior of individual grains. The constitutive models typically must be redefined for each model based on system-specific experimental findings, which generally degrades their usefulness in investigating the behavior of the system at the meso-scale or for predicting the behavior of other systems.

The analytical solutions of micron-scale particle-to-particle interaction were not directly tested until recently with the invention of electron microscopy. Indeed, it has only been since 1986 when atomic force microscopy (AFM) was introduced that detailed force displacement curves could be generated. Because of the relative youth of experimental methods to verify the force models, understanding of the sensitivity of the different laws to the relaxation of their assumptions is still poorly understood. For instance, the widely implemented JKR model for adhesive contact is based on perfectly smooth spheres, yet it has been shown that the adhesion force is sensitive to the presence of surface asperities, as shown by Jones [49].

Through this discussion, we have provided a motivation to view meso-scale modeling for use in granular systems where microstructural grain interactions substantially influence behavior in the macroscale. The tool we use is discrete element modeling (DEM), the background of which is discussed in the next section.

2.3 Background

In the middle of the domains of micro- and macro- scale modeling is the domain of meso-scale modeling, which, compared to micro- and macro- scale modeling, is still in its infancy. The meso-scale is any length scale where the microscopic grain-to-grain interaction cannot be decoupled from the bulk behavior of the system to the desired tolerance of the model. The question of how interparticle interactions effect different granular system behaviors is a mostly open question, the answer to which likely lies in the meso-scale link. This must be qualified with the note that many macroscopic granular behaviors have been modeled qualitatively successfully, such as, the demonstration of shear band formation via the circulation cell mechanism by Williams and Rege [131] in triaxial tests, the characterization of the inner particle flow fields in simple spherical particle hopper flow, and the modeling of energy requirements for ball mill operations. Despite qualitative success, the discrete element method has yet to predict the behavior of many real world applications.

The first attempts to handle microscopic properties emergent into larger meso-scale systems came at the advent of the computer when massive systems of simultaneous equations could be evaluated. The idea of cellular automata, used for the simulation of evolutionary computing and development of Turing machines, was first developed by John Von Neumann [84]. Von Neumann postulated that one could generate a theoretical world populated by objects that responded to stimuli according to certain rules. Taken in large numbers over a period of discretized timesteps, these automata displayed not only some pedantic behavior (such as inactivity or constant activity) but also strikingly complex emergent behavior. The idea of emergent behavior from a system of simple primitives became the basis for the lattice gas method and molecular dynamics simulations.

The Lattice Gas method [36, 30, 23] is a specific type of cellular automata where the behavior of the individual cells is governed by the continuity equation and Navier-Stokes equation for fluids. Here, a grid in two dimensions or three dimensions is generated and particles are exchanged between the different cells. These cellular gas packets are constrained to move between adjacent or diagonal cells.

Cellular automata brought about a new way of thinking about the behavior of systems. The approach of using cells on a spatially and temporally discretized grid could simulate the atomistic level interactions of particles. This was the inspiration for molecular dynamics [2] formulations, which were made possible by the development of increased computing power. Explicit modeling of individual point potentials were modeled in the computer alongside point-to-point spring contacts to simulate atomistic level bonding, attraction, and repulsion. Unlike cellular automata, point potentials in molecular dynamics simulations extend in any direction for any computer representable distance.

The point potentials used in the molecular dynamics formulation generate a spherical potential field. For granular systems, however, geometry of the particle plays an important role in determining the behavior of the system. Compounded with the fundamental importance of friction (a tangentially acting force), the molecular dynamics formulation was inappropriate in its pure form for powder and grain modeling. A different methodology had to be employed.

The role of molecular dynamics in fundamental material behavior was realized soon after it was developed; however, it was viewed mainly as a means of investigating the atomistic and molecular scale of materials, such as, silicon for transistors. The emergent effects of single atoms (or even groups of atoms) on the bulk properties of a material are considered negligible for many systems of engineering interest (e.g., steel beams). For granular materials, this assumption has been shown to be invalid in a wide range of applications and engineering materials. Molecular dynamics and its modeling philosophy, however, has led directly to the development of the discrete element method.

2.3.1 2-D DEM

At the time of Cundall's and Strack's seminal work on DEM [22], analyzing problems in more than two dimensions was too computationally costly to perform by many researchers. The two-dimensional formulation was extended from discs to ellipses by Ting [118] and to angular shapes represented by polygons by several researchers. An innovative solution to representing arbitrary shapes was made by Williams and O'Connor [130] with the introduction of the discrete function representation (DFR).

Three-dimensional discrete element modeling poses several problems that are not encountered in two dimensional discrete element processing. Instead of 3 degrees of freedom, one must handle 6 degrees of freedom. Rotation is no longer in one dimension, instead one must integrate around an axis in three-dimensional space, adding additional complexity in integrating rotational motion. If Euler angles are used, this becomes an involved integration process about 3 independent axes. A quaternion-based integration scheme is introduced in Chapter 5 to solve this problem.

2.3.2 3-D DEM

Three-dimensional DEM simulation at scales on the order of typically performed field tests requires computational resources (in terms of CPU time and memory) that are far in excess of those available to most researchers. Again, the example of a $1m^3$ sample of $1mm$ diameter sand grains typically contains on the order of 10^9 grains.

To mitigate the effect of modeling so many particles, spheres are typically used in 3-D DEM formulations, since they are simple and compact and hence are fast computationally. This is the basis for the popular codes TRUBAL [20] and PFC3D [21] by Cundall and for DMC by Taylor and Preece at Sandia National Laboratories [115, 100]. The granular dynamics extension to the molecular dynamics code LAMMPS by Plimpton [98] also uses spherical potentials.

More recently, other geometric representations have been developed. Although less common, implementations of DEM codes with polyhedral elements, such as the combined finite-discrete element application described through a large body of work by Munjiza [79], exist. Also, ellipsoidal elements can also be found in the groundbreaking work of Lin and Ng [66]. Dilated shapes proposed in the works of Hopkins and Kuhn [40, 55] are also an elegant solution to modeling asphericity. In general, though, the research field for numerical investigation of particle shape on granular system behavior has been sparse.

To address the need for a computationally efficient representation of non-spherical geometries, this thesis proposes a prolate spheroid approximation intersection algorithm in Section 4.2 and a general tri-axial ellipsoid intersection in Section 4.3. Computational geometry at the sub-grain scale is addressed in Chapter 7.

Chapter 3

Software Architecture

The goal of the software architecture is to produce a framework from which other DEM extensions can be made simply, consistently, and by many different researchers. In high-level software development terms, this means that the architecture should be maintainable, extensible, and portable across platforms to as maximal an extent as possible while maintaining acceptable performance. This philosophy of simulation design is discussed by Perkins [94], which argues for the development of a more extensible and abstracted framework for DEM, citing the need for explicit support for core, extension, and application developers.

Using software development research, this chapter first provides a basis and base parameters around which the use of an architecture optimized framework provides the smallest total development time.

This chapter will then address the core developer's concerns in the context of an object-oriented approach. This is the highest level of programming and affects both the extension and application developers. Core development concerns the class hierarchies used and the interfaces necessary for extension developers to add functionality to the DEM framework in a consistent manner. Core development must also be concerned with cross-platform portability.

Next, the chapter will address several different common components and their interfaces in the DEM framework. By rigidly defining what methods different classes must implement, the interface can assure that certain methods will be available. It also exposes classes of the same type to the same battery of user and functional tests via the test module, allowing quality standards to be built in to the framework. The component "contract" defined by the class interface is described for each component and illustrated through currently implemented components.

3.1 Architecture versus Performance

Both the software development community (e.g., widespread adoption of the Capability Maturity Model (CMM)) and the computational mechanics community (e.g., the United States national labs' guidelines on software development) have converged on valuing software quality practices for improving the reliability of their codes. Code verification in the national labs and unit testing for the general software development community attempt to reduce the programmer-originating bugs in a code by enforcing a policy of testing individual components against requirements governing the intended operation. Code validation and functional testing are then used to test whether the application produces the expected behavior. Combined, these practices help to reduce many sources of error and, hence, uncertainty in the manufacturing of software products.

Unlike testing, however, there is a more controversial debate on the usefulness of architecture versus performance that is often neglected in the computational mechanics literature. As many software developers and some computational mechanics researchers have been working toward improving the architecture of their software at the expense of performance, many more insist on improving software performance through the sacrifice of architectural consistency. This thesis will not attempt to resolve this debate; however, this section will attempt to propose a coherent thought experiment in support of the pursuit of maintainability and extensibility for research DEM codes as well as to determine practical bounds on the usefulness of this approach in reducing the overall time to solution (TTS) for general problems.

A prime example of a DEM simulator optimized for speed while sacrificing extensibility and maintainability is the LAMMPS application available as an open source code from Sandia National Labs (http://www.cs.sandia.gov/ sjplimp/download.html) and is one of the only scalable granular dynamics simulators available open source. The application is written in the C language and uses a structure optimized for performing molecular dynamics. The LAMMPS application has been extended to include an implementation of granular dynamics, which is currently limited to modeling of spheres with common material properties, bonds, and axis-aligned planar boundary conditions and allows for two contact laws to be applied (simple spring dashpot and Hertzian, both with Coulombic friction).

Analyzing the architecture of the LAMMPS application, the simulation pipeline is hardwired into few procedures. That is, procedures are not logically separated between performing neighbor-sorting, contact resolution, and force resolution. This is optimal for speed; however, every time another contact law, particle shape primitive, or neighbor-sorting algorithm is added, the code required is increased by a multiple of the current lines of code (LOCs) involved in the sum of these operations, and 2 classes are affected. If a testing framework is applied, the number of testing methods must also be doubled, and the resolution of the testing methods is reduced (since parameters may not be as easily isolated with procedures which lump multiple functionalities). For the purposes of this example, only the direct costs of coding will be considered without the consideration of testing.

Since developers have been observed to produce similar numbers of LOCs independent of the language used [70], LOCs are often related linearly to the number of man-hours required for coding. Estimates of function point (FP) productivity per LOC are approximately 1/55 for a higher level language like C# versus 1/128 for a level 2.5 language like C [1].

By inspecting LAMMPS manually, approximately 150 LOCs are required for each permutation of neighbor-sorting algorithm, contact law, and geometry, which is currently 2 neighbor-sorting algorithms, 1 force resolution algorithm, and 2 contact laws resulting in 4 permutations, or approximately 600 LOCs. Using this as a basis of

[1] These numbers are adapted from "programming languages tables" [47, 119] which specify ranges of expected function points per lines of code. The tables differ by approximately 10 LOCs/FP. Also, it is notable that the variance is 51-66 LOCs/FP for C# versus 9-704 LOCs for C [119]

Application	LAMMPS	DEM3D
Efficiency (LOC/FP)	150	55
Component (FP)	2	1
Component (LOC)	300	55
Productivity (LOC/person-day)	100	100
Effort (person-day)	3	0.55
Error Rate (LOC/error)	133	133
Errors (person-day/error)	0.63	0.63
Error Effort (person-day)	1.42	0.35
Total Effort (person-day)	4.42	0.9

Table 3.1: Summary of Time Cost to Implement Hertzian Contact in LAMMPS versus DEM3D

comparison, implementing another contact law would require an extra 300 LOCs. As a check for our estimates, it is notable that the estimate of 128 LOCs/FP is fairly accurate here. In DEM3D, the number of LOCs required to add another contact law is constant. In the case of Hertz-Mindlin contact [73], for instance, 55 additional LOCs were required, which again agrees well with the estimate above.

Studies indicate that a developer is estimated to produce 10-100 LOCs/day [8, 33, 70]. If we assume that the programmer is ”good” (produces 100 LOCs/day), then we have an unadjusted 3 person-days in LAMMPS versus 55 LOCs (i.e., 1 FP estimated for C#), or 0.55 person-days for DEM3D.

To acquire a good estimate, however, the effect of defects must be considered. Assuming a constant error rate of 1 error per 133 LOCs [96] and that errors are caught at different points in the project development with an average correction cost of 6.3 hours/defect [33], the total time should be increased by 2.5 defects for LAMMPS (14.2 hours or 1.5 person-days) and 0.4 defects for DEM3D (2.5 hours or 0.25 person-days).

This line of reasoning, tabulated in Table 3.1, leads to an adjusted cost of 0.9 person-days to implement the additional functionality in DEM3D versus 4.42 person-days in LAMMPS. This estimate does not include the effort to augment a testing framework, scale the code necessary to extend the LAMMPS application, and the learning time for the developer, indicating that the costs are probably much higher.

Application	LAMMPS	DEM3D
Size (LOC)	191000	82000
Effort (person-month)	370	125
Time (month)	20	14
Salary ($/person-month)	8300	8300
Cost ($)	$3M	$1M

Table 3.2: Summary of Cost to Implement Full 3D DEM Application

Benchmarks run on the two codes place the efficiency of LAMMPS in serial at approximately twofold that of DEM3D, which uses a memory-managed language. In summary, we trade computational efficiency for a decrease in development time.

We showed earlier that the real LOCs for a function point in LAMMPS agreed remarkably well with the estimates produced in the studies [47, 119]. Unfortunately, there is no available source code for software that emulates the richness of the DEM3D application. If we use the estimates [47, 119], we can compute the approximate development time and team size necessary to complete each of the projects. DEM3D is composed of 2,453,460 Bytes of source code with an average of 30B/LOC, which yields approximately 82,000 LOC in managed code. Using a conversion of 128/55, this leads to an estimate of 191,000 LOCs in C.

Using data from a nominal schedule estimate table [70] and assuming that the project most closely resembles a "business product", the effort for an 82,000 LOC project is approximately 125 person-months and that for a 191,000 LOC project is 370 person-months. Using the calculations of optimal team size in the same table, we arrive at 14 months versus 20 months. If we agree with these estimates, the expected cost of a C-type application would be approximately 250 person-months greater than that for the C#-type application. This is a cost of 6 months of time and, assuming an average developer cost of US$100,000 per year (fully loaded), a financial cost of approximately US$2 million. This analysis is tabulated in Table 3.2. Assuming that the performance metrics are still valid in parallel, this indicates that the financial cost alone can justify the use of a C# type programming language to design a DEM simulator for problems requiring up to 5.7 teraflops [2], the price point of using an additional server. In summary for the current environment of software development: hardware is cheap, but developers and scientists are expensive.

This is a highly simplified analysis. In real situations, it may not be possible to allocate a team of 19 developers for 20 months or the burden of usability may be much lower than the effort estimation tables would indicate. For instance, a DEM simulator is not typically a deployable system but rather a "research" system built to operate for a limited user group (typically including the developer) and for a very specific hardware configuration. This often drastically reduces the software effort required. Also, developers for recent languages such as C# may not be available or may be less productive than the C programmers that can be assigned. This may restrict the team size and/or the average productivity of the team. Organizationally, salaried employees are typically more justifiable than equipment costs or the budget for employees is separate from that for equipment, which may alter the incentives for one approach over the other. Finally, applications running in parallel often exhibit non-linear scaling behavior versus problem size. This complex set of parameters is only a small subset of the real parameters affecting the analysis that should be performed in choosing a language for developing a system.

Though the issue is complex, the rough estimates indicate a good possibility that financial costs are lower for developing a DEM application in a level 2.5 type language (e.g., C++, Java, and C#) than a lower level language. Also, there are several issues that were not considered which would benefit the architecture-centric, managed code application, including:

- Less variance in development time given a similar set of function points
- Lower incremental time cost for extending the application
- Tighter coupling between experiment and numerical implementation during experimental validation of codes

[2] As of 9/16/05 the starting price of an IBM Blue Gene rack server rated at 5.7 teraflops was US$1.5 million [9].

- Greater flexibility to architectural change

- Low-level compiler optimizations from a supported and often updated compiler, which will result in increasing performance over time as the JIT compiler is upgraded by an external development entity

3.2 Testing: Verification and Validation Framework

Testing is a way of assuring the quality of an application throughout its lifecycle. Specifically, testing has two main roles in the software development process: to reduce the risk of errors propagating through the development of the application and to serve as a way of assuring the end user that the application will perform as specified. For an application developed by a single developer, this may seem to be a significant source of overhead. However, the resultant code comprises nearly 81,000 lines of C# code, which is under constant change.

During the development of any computer program it is important to assure the end user that it operates as specified. In the modeling and simulation (M&S) community, applications are often subjected to a rigorous testing methodology for assurance known as *validation* and *verification*. Distinctions have been made between testing of the output of functions and methods (*unit testing* in the computer science lexicon or *verification* in the parlance of the M&S community) as opposed to the accuracy of the output (*functional testing* or *validation* as defined in the computer science and M&S communities, respectively). This can be restated as a distinction between modeling/programming process and output simulation results/product.

Testing need not be used merely for assuring output or testing components in post development. It can become part of the software design methodology. This thesis has embraced testing both in the design and in the framework of the application.

In the development of DEM3D, there was ambiguity about the final requirements for the application. At the same time, there was pressure to provide evidence that the simulation was producing correct output within the assumptions of the DEM model. Throughout the software design, the current build was required to be correct and efficient. To cater to these pressures, a unique design methodology was adopted, which was a hybrid of a classical evolutionary prototyping development model with an adaptation of the test-driven development (TDD) methodology. The TDD component was used to reduce risk within individual iterations as well as to help guide requirements refinements and reduce coding errors. Though not explicitly part of the TDD methodology, the effect of this approach also helped to assure the end functionality and to institutionalize software refactoring during the design process.

A standard evolutionary prototyping development model was adopted using my own requirements as well as input from collaborators in the Department of Chemical Engineering to guide the initial requirements. A prototype with a minimal set of features was designed and deployed for user testing. The process was then restarted. Iterations were initially closely spaced to help identify major issues early in the design process; as the issues became fewer, iterations were allowed to be spread further and the use of tests within each iteration was increased. Also, as the maturity of the prototype increased the richness of the feature set also increased as did the number of tests required to assure the operation of the simulation.

The application has been developed as an agglomeration of simpler modules, each of which has a set of tests associated with it. Assemblies of modules are also assigned tests. Finally, the application is given a series of tests. Layered onto this hierarchical model of unit tests are benchmark tests. The hierarchical nature of the approach provides a reasonable degree of assurance that problems will be identified early in the development cycle and continue to be identified as the complexity of the software increases.

The result of this approach has been the integration of a testing framework based on the popular NUnit framework, a port for the .NET framework of the JUnit unit testing library for Java. The framework has served a significant number of roles throughout the project's lifecycle, providing a way to iterate through testing, optimization, and refactoring while providing metrics to demonstrate whether improvement was gained on the individual modules, as is suggested in the TDD methodology [32]. The resultant system has also yielded a self-validating code with built-in regression testing and built-in benchmarking capabilities. This provides not only rigorous verification and validation of an application that is considered "research code" but also provides a platform against which to test different algorithms that may be used throughout the simulation pipeline. For instance, when the NUnit application is invoked to run a benchmark against neighbor-sorting algorithms, reflection is used to automatically identify all existing algorithms and run a benchmark problem for which their performance is compared and Excel readable files are generated to produce illustrative graphics, several of which are included in this thesis.

3.3 Development Language Selection

From the preceding discussion, the order of effective computation per human-year spent on development is sufficient to justify the use of an architecture-centric approach, which sets performance criteria as secondary. It is also a reasonable justification after the fact for using a managed code framework versus an older unmanaged code approach. Managed code provides several benefits over unmanaged code, including:

- Optimized memory usage
- Garbage collection to prevent memory leaks
- Multi-level code optimization through CLR compilation and a secondary JIT compilation step

The particular chosen managed code framework is Microsoft's .NET framework. The .NET framework and the competing J2EE framework, an open source effort supported principally by Sun Microsystems, share a set of common attributes. For instance, both Java and C#, the main language of the .NET framework, have similar syntax, use JIT compilation techniques, have open standards, and are supported by freely available compilers. With the Mono project, the .NET framework (like J2EE) can also be used on the Linux operating system. The attractiveness of Java and C# are similar, and the final choice was based on the author's familiarity with the programming environment and language.

3.4 Visualization

Visualization is an important component in representing scientific information, especially from a simulation with several states recorded. In this work, visualization is abstracted into a set of software visualizers which handle both post-simulation production of graphics as well as real-time visualization. A mid-level API, DirectX9.0C from Microsoft, is commonly used in the programming of high-performance games and is used here to perform real-time visualization of DEM data as well as the matrix and vector math to support simulation calculations. When run in batch mode, visualization is typically handled as a post-processing activity to produce animations and other static visualizations. To handle these activities, the open source ray tracer, Persistence of Vision (POV-Ray), is wrapped using classes of abstraction to render DEM data. This allows the application to render either real-time using POV-Ray or DirectX or as a batch post-process using POV-Ray.

3.5 DEM3D Core

DEM3D is structured to function around a core kernel upon which other modules depend to add functionality to the core. The core kernel is responsible for simulation on the current application domain. This distinction is made, because in the anticipated extension of the DEM3D application to a parallel version, there would be multiple application domains not necessarily located on the same physical machine.

Several auxiliary packages are formed around the core kernel to provide additional required capabilities, as illustrated in Figure 3-1. The functional division is shown in Figure 3-2, indicating the different libraries that compose the full DEM3D application. These libraries need not all be used, and for different applications, the feature set can

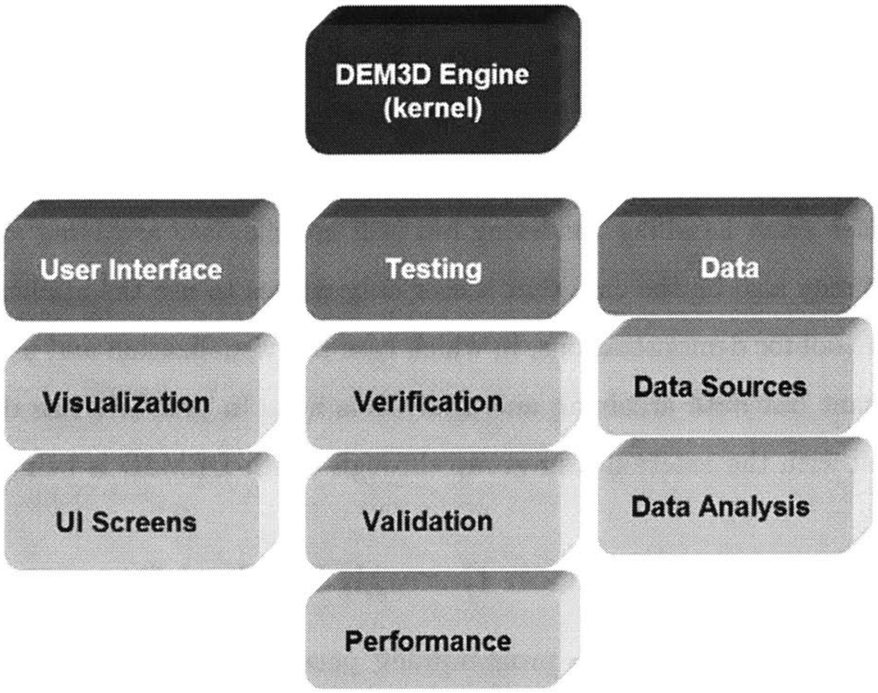


Figure 3-1: High-level diagram of the module configuration in DEM3D, which details the logical divisions between the kernel, visualization, data, and application testing modules.

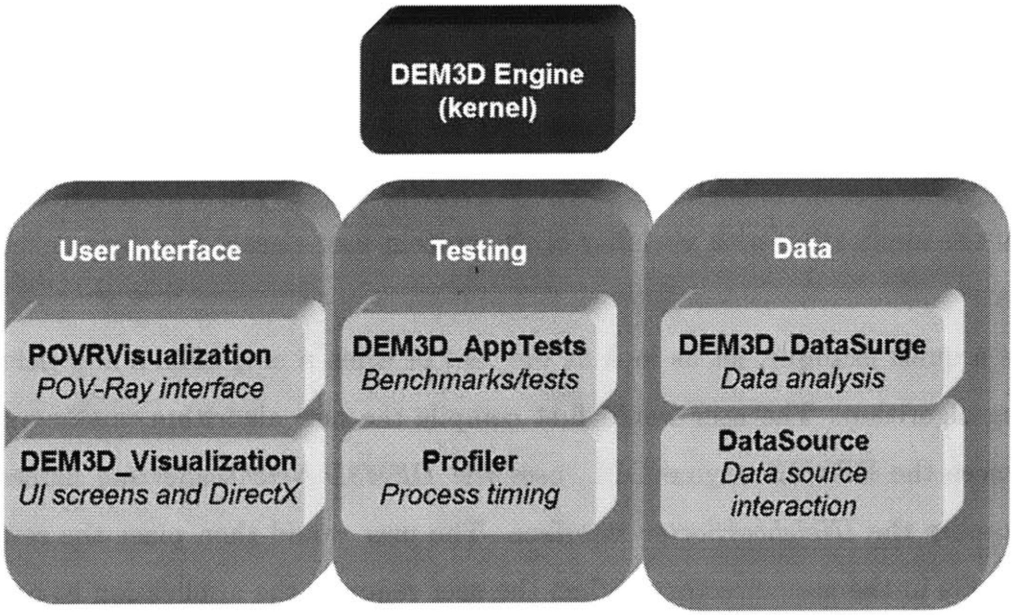


Figure 3-2: Diagram of the functional division of the application into assemblies (libraries).

be reduced to optimize the performance.

This separation of concepts is designed to allow a clean separation of modules and functionality according to different use cases. For instance, the end user may wish to run the simulation as a batch console application to reduce the overhead of user interface event handling/rendering but still use the data archiving and analysis module. It may also be the case that a user only wishes to use the application as an illustrative tool for demonstrations, in which case the visualization and user interface are important but data archiving and analysis is not. In practice, this division has worked well with the external user group through which DEM3D is being tested.

3.5.1 User Customization through Method Servers

DEM3D makes use of a simple programming paradigm that couples singletons in classes to access objects that inherit from that class. From first glance, this would seem to be a strange use of the class definition; however, when used with the *AssemblyManager*, it provides centralization of access to like objects and also allows for automatic discovery of new objects.

The user can add his/her own custom classes to the application by compiling classes and adding the resultant dynamically linked libraries (DLLs) to the user directory. At run-time, the *AssemblyManager* searches the user directory for relevant DLL files and loads them. This process occurs first in the application, providing the rest of the application with visibility of the custom user classes as soon as execution begins.

As a quick example, let us look at the case of when a user adds a new neighbor-sorting algorithm. The user would first compile the new algorithm, making sure it references the DEM3D engine DLL, uses the *DEM3D.NeighborSorting* namespace, and applies the *INeighborSorter* interface. The user would then place the resultant library file in the user directory. When the user requests the application to consume a simulation settings file, the file parser will come to the new user class under the neighbor sorting entry in the file. The application will first try to instantiate a class from its own library. If this fails, it will ask the *AssemblyManager* to provide

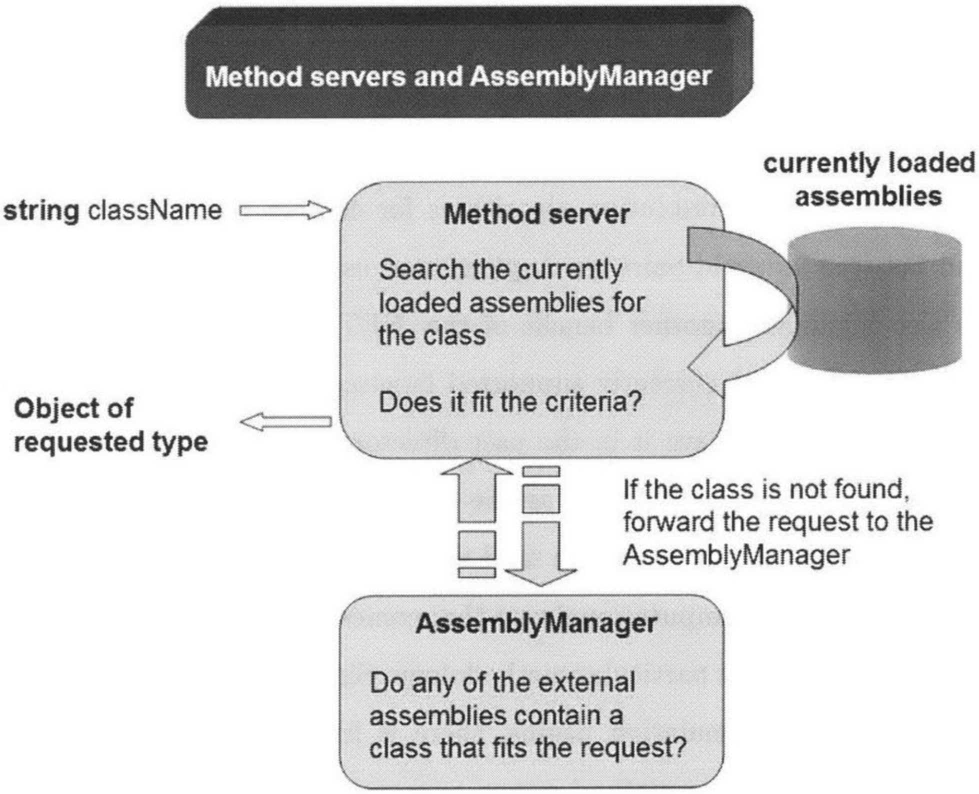

Method servers and AssemblyManager
currently loaded assemblies
string className
Method server
Search the currently loaded assemblies for the class
Does it fit the criteria?
Object of requested type
If the class is not found, forward the request to the AssemblyManager
AssemblyManager
Do any of the external assemblies contain a class that fits the request?

Figure 3-3: Diagram of the interaction of a method server and the *AssemblyManager* when a class of a certain type is requested for instantiation.

it with the class by providing the string-based name of the class (as provided in the simulation settings file). Using reflection, the *AssemblyManager* searches the assemblies it had discovered at initialization, and returns an appropriate object. The object is then referenced through a static variable in the *NeighborSorter* class as the current neighbor-sorting algorithm. This object is then used during the simulation. The process is shown in Figure 3-3.

This affords the user a great degree of transparency when adding functionality to the DEM3D application. The *NeighborSorter* (the base class for all neighbor-sorting algorithms), *ContactLaw* (the base class for all laws governing the physics of contact), *Shape* (the base class for the geometric definition of all particle types), and even specialized contact resolution algorithms for contact between pseudo-ellipsoid pairs and between ellipsoid pairs are implemented using this design methodology.

This also highlights another benefit of the .NET framework, in that, users can develop in any of the 25 currently supported languages and then compile that code into a DLL and simply leave it in the user directory. There is no need to port the code into another language; as long as the data abstractions that the code derives from are maintained, the code can be used without change.

There is typically a computational cost that comes with maintainability. However, this is not the case for this particular methodology. Since the overhead of instantiation is accrued before the simulation begins, there is little if any performance hit for this approach. In fact, the method server approach is often more computationally expedient than even the invocation of overridden abstract methods, as shown in Figure 3-4.

Figure 3-4 illustrates a benchmark problem of based on 100 trials of 10^6 iterations each. The trials consist of an evaluation of the square root of the product of 2 double precision numbers. The code used is as follows:

```
public class DelegateServer {
        public static DelegateServer current = new DelegateServer();
        public delegate double SqrtProductFcn(double a, double b);
        public SqrtProductFcn SqrtProduct;
```

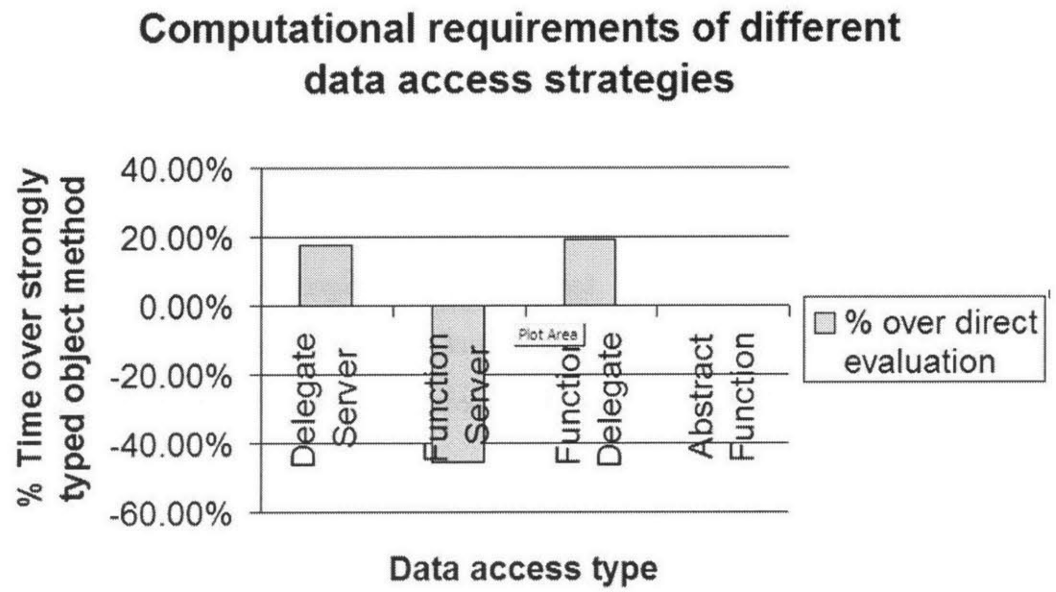


Figure 3-4: Comparison of the average run times for different object access strategies as a percentage of the runtime for the evaluation of the method on a strongly typed object.

```
    public DelegateServer()
    {
        FunFunction fcn = new FunFunction();
        this.SqrtProduct = new SqrtProductFcn(fcn.SqrtProduct);
    }
}
public abstract class FunctionServer {
    public static FunctionServer current = new FunFunction();
    public FunctionServer(){}
    public abstract double SqrtProduct(double a, double b);
}
public class FunFunction : FunctionServer {
    public FunFunction(){}
    public override double SqrtProduct(double a, double b)
    {
            return Math.Sqrt(a*b);
    }
}
```

The ”Abstract Function” test used a *FunctionServer* object to evaluate *SqrtProduct.* The ”Function Delegate” test evaluated a *SqrtProductFcn* delegate. The ”FunctionServer” test invoked the *SqrtProduct* method of a static variable of type *FunctionServer.* The ”DelegateServer” test invoked a static delegate variable of delegate type *SqrtProductFcn.*

3.6 Simulation Pipeline

The typical architecture of a simulation engine follows a particular execution pattern (sometimes with recursion):

- Input
- Simulation
- Output

The simulation pipeline in this work is structured around a simulation-driven environment. This idea emphasizes the centrality of the simulation in processing, slaving processes like visualization to the simulation process. The design also allows several custom interfaces for application developers to attach their own code and assemblies to the application. For extension developers, the simulation pipeline is considered inviolate outside of the custom interfaces provided.

Though constrained by the rigid nature of the simulation pipeline, extension developers are given access to a fairly extensive range of features to change, including: implementation of new neighbor-sorting algorithms, creation of contact laws, insertion of new geometric primitives for representing bodies, design of contact handling designs, and application of parallelization components. These components will be discussed in greater detail in the following sections of this chapter.

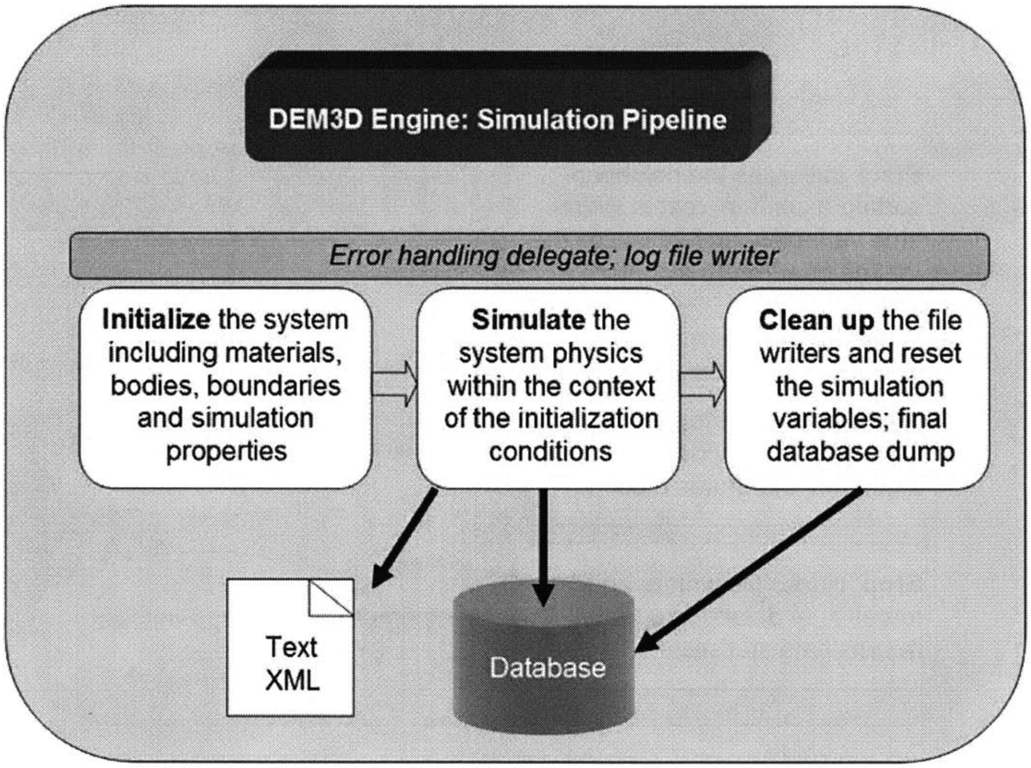

DEM3D Engine: Simulation Pipeline
Error handling delegate; log file writer
Initialize the system including materials, bodies, boundaries and simulation properties
Simulate the system physics within the context of the initialization conditions
Clean up the file writers and reset the simulation variables; final database dump
Text XML
Database

Figure 3-5: Diagram of the simulation pipeline.

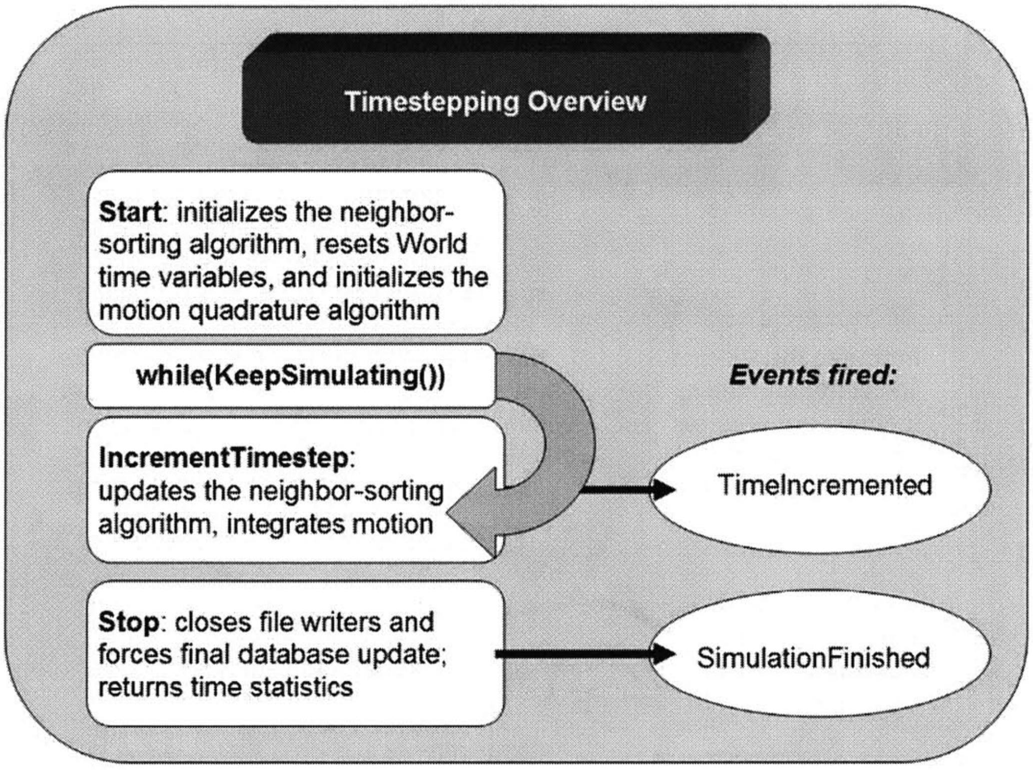

Timestepping Overview
Start: initializes the neighbor-sorting algorithm, resets World time variables, and initializes the motion quadrature algorithm
while(KeepSimulating())
IncrementTimestep: updates the neighbor-sorting algorithm, integrates motion
Stop: closes file writers and forces final database update; returns time statistics
Events fired:
TimeIncremented
SimulationFinished

Figure 3-6: Overview of the order of evaluation during time-stepping. External methods or actions can be attached to the timestep by attaching to the TimeIncremented and SimulationStopped events.

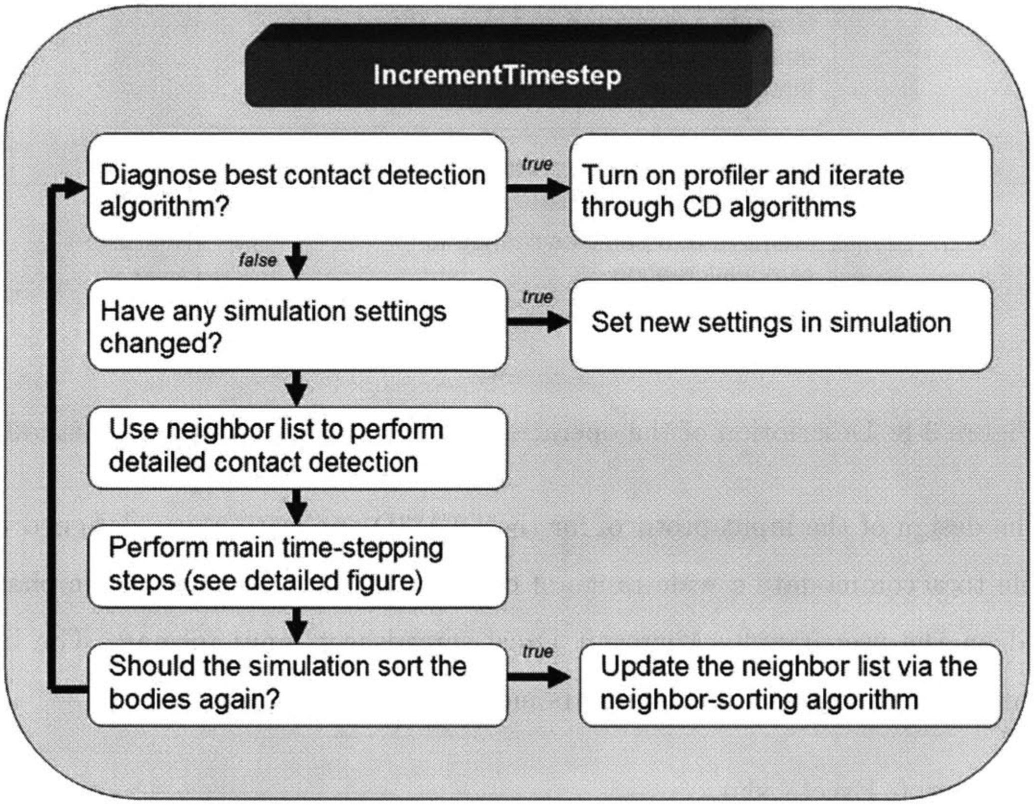

IncrementTimestep
Diagnose best contact detection algorithm?
true
Turn on profiler and iterate through CD algorithms
false
Have any simulation settings changed?
true
Set new settings in simulation
Use neighbor list to perform detailed contact detection
Perform main time-stepping steps (see detailed figure)
Should the simulation sort the bodies again?
true
Update the neighbor list via the neighbor-sorting algorithm

Figure 3-7: Description of the operations during the timestep update.

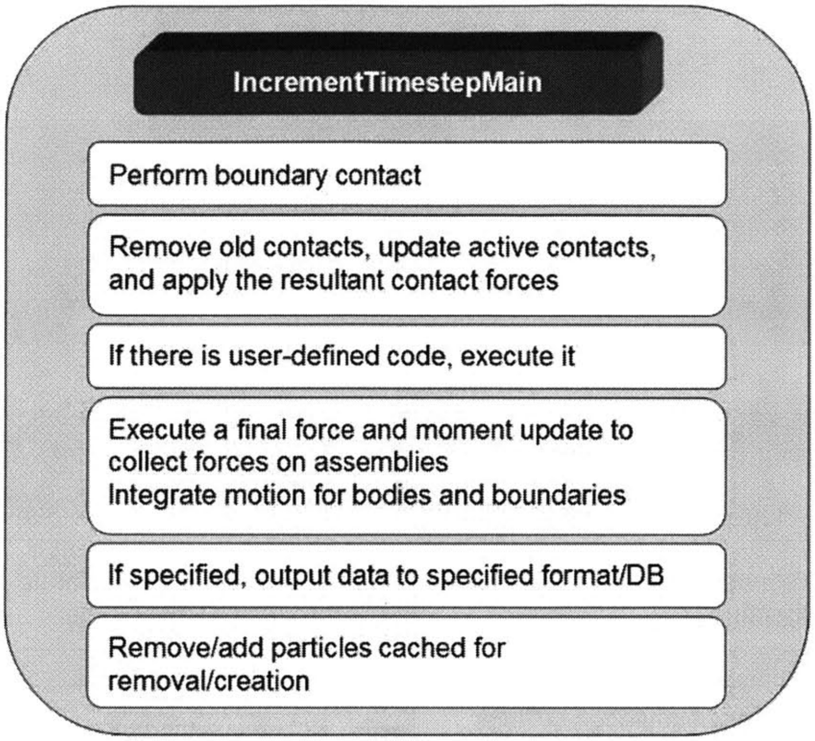

IncrementTimestepMain
Perform boundary contact
Remove old contacts, update active contacts, and apply the resultant contact forces
If there is user-defined code, execute it
Execute a final force and moment update to collect forces on assemblies
Integrate motion for bodies and boundaries
If specified, output data to specified format/DB
Remove/add particles cached for removal/creation

Figure 3-8: Description of the operations during the main timestep method.

The design of the input protocol for the DEM3D application was designed to be flexible to accommodate a wide range of common user needs; however, emphasis is placed on the user-friendly Microsoft Excel spreadsheet input format. The list of supported general system input formats include:

- Microsoft Excel (.xls)
- Matlab (via automation interface)
- Database (must adhere to specific DEM3D schema)

DEM3D also supports the importation of polyhedral boundary conditions, including the following formats:

- 3D Studio Max (.3ds)
- Virtual Reality Markup Language - VRML (.wrl)

- Proprietary text format: vertex number;vertex list [comma delimited];face number;face list [comma delimited]

The system inputs allow for the specification of geometric bodies, materials, and simulation properties via a structured format. Currently, a Microsoft Excel spreadsheet or an interface to the Mathworks Matlab application can provide system specification data. In Matlab, input must be specified in a structure with a particular name.

The interface to external applications is extensible by the architecture developer. The *Converter* class is designed to provide abstraction to inheriting classes to support new types of data-providing applications. The properties of the user interface elements are also not hard-wired and can be specified at run-time; that is, the mapping of particular members in the user interface to the internally represented elements can be specified through meta-data files in eXtensible Markup Language (XML), which adhere to a schema file (XSD).

Support is also provided to convert to and from the proprietary format used in LAMMPS [98], an open source molecular dynamics (MD) application, which has been modified to provide limited DEM simulation capabilities.

Data is returned from the simulation in 2 main ways:

- Database (discussed in a later chapter)
- Text-based format

The database support for output is highly desirable from an analysis perspective. This thesis focuses on the contributions made to analysis of DEM data through coupling with a database. The DEM3D architecture provides a rich set of abstraction to allow access to a variety of data sources that are not typically thought of as databases. The discussion of this part of the architecture is given in Appendix A.

The text-based formats include a flavor of XML developed for DEM as well as proprietary tab-delimited formats. The tab-delimited formats are divide data files by body ("ParticleBasedText") or by time slice ("TimeBasedText").

3.7 Contact Detection

Finding contacts within an n-body system is an $O(n^2)$ operation in the naive case. Several improvements for geometrically based systems have been developed. This section will investigate the different contact detection algorithms and then describe the contact detection algorithm implemented in this software package. Here "contact detection" is taken to mean the process of reducing the pairs checked for detailed intersection. In other texts this is known alternately as "spatial reasoning" or "neighbor sorting".

The first improvement to the most naive $O(n^2)$ algorithm is to realize that for any two objects o_1 and o_2, if o_1 is on contact with o_2 then o_2 must conversely also be in contact with o_1. This is a simple observation but leads to the reduction of pair checks by half (i.e., $O\left(\frac{n(n-1)}{2}\right)$). The more drastic improvements, however, have stemmed from the reduction of the search space for any given object.

Many algorithms have focused on subdividing the model space into cells oriented along the global Cartesian axes of the model space. Most simply this can be accomplished by using equal cells with dimensions equal to the largest dimension of any object in the space. If objects are assigned to a cell (otherwise known as a bin) based on the integer (*floor* operation) of the division of the geometric centroids by the maximum dimension, then it is a simple proof [3] to show that objects located more than one cell away in any direction cannot contact an object in the cell.

Therefore, an object in a certain cell can only be in contact with objects that are members of its own cell or its immediate neighbors. This results in a search of 9 cells in R^2 and 27 in R^3. This operation is sensitive to object size, but for equally-sized objects, the algorithm is of the order of $O(n)$. In the extreme of polydispersity in

[3] Let a body o_i be superscribed by a cubic box of side length l; then each cell is, similarly, of side length l. For simplicity, let every body be in the space region $R_{positive} \in R$, where $R_{positive}(x, y, z)$ defined on $[x > 0, y > 0, z > 0]$. Set the lower left corner of each bin as $[x_{lower}, y_{lower}, z_{lower}] = [l * i, l * j, l * k]$ for integer bins $[i, j, k]$. Taking the most conservative case where objects in non-adjacent bins lie on the boundaries of the bin that lie closest to each other and have axis-aligned dimensions of l, such that, the closest points are at $p_1 = l * (i + 1) - \delta$ and $p_2 = l * (i + 2 - 1)$, respectively, where $\delta \to 0$. The δ infinitesimal offset is required to reflect that the *floor* operator bins such that $floor(l * x) = l$ for the open region $x \in [0, 1)$. For the requirement of no contact $p_1 < p_2$, or $l * (i + 1) - \delta < l * (i + 2 - 0.5)$. This reduces to $i - \delta < i$, which is strictly true for all i.

size distribution, the algorithm may degrade to $O(n^2)$. This algorithm is known as the Cell-Verlet algorithm and is commonly used in many molecular dynamics (MD) implementations.

The NBS (no binary search) algorithm [80] expands on the Cell-Verlet algorithm by introducing an efficient linked list approach, to avoid checking between empty cells, and a reduced neighbor-checking mask, which results in a search of 5 cells in R^2 and 16 in R^3. The algorithm scales as $O(n)$ with a coefficient of $16/27 \approx 0.59$ that of the Cell-Verlet. Unfortunately, the NBS algorithm also suffers from similar degradation under polydispersity as the Cell-Verlet algorithm and may exhibit $O(n^2)$ scaling in the extreme.

To remove the size dependence of the contact detection algorithm's performance, a further refinement was proposed with the algorithm CGRID [94], which uses bins based on the smallest dimensions of a particle, calculates object bin based on the lowest extent of the object in each dimension, and uses an intelligent marching algorithm combined with a lowest ordinate starting point and a cataloging operation to retain objects larger than the bin over multiple cells. The effect of this is to allow objects to belong to multiple bins without the overhead associated with determining the addressable cells and storing the object in multiple bins. The bin length is adjustable to optimize efficiency for specific systems. The CGRID algorithm reduces the mask size to 4 cells in R^2 and 8 cells in R^3, which improves the performance of the algorithm at the boundaries and improves the scaling of the memory requirements as the algorithm is applied to dimensionally increasing problems. The CGRID algorithm scales as $O(n)$ irrespective of poly-dispersity; however, the constant coefficient is necessarily much greater than that of the NBS algorithm.

DEM3D allows for the implementation of arbitrary contact detection algorithms. Currently, the contact detection algorithms implemented in the base application are a naive $O(\frac{n(n-1)}{2})$ algorithm, DESS [95], NBS, and CGRID. In addition, the application offers an interface, *INeighborSorter*, which allows for users to develop their own algorithms. When classes are developed that implement *INeighborSorter* or that inherit from the class *DEM3D.NeighborSorter*, they can be compiled into a library

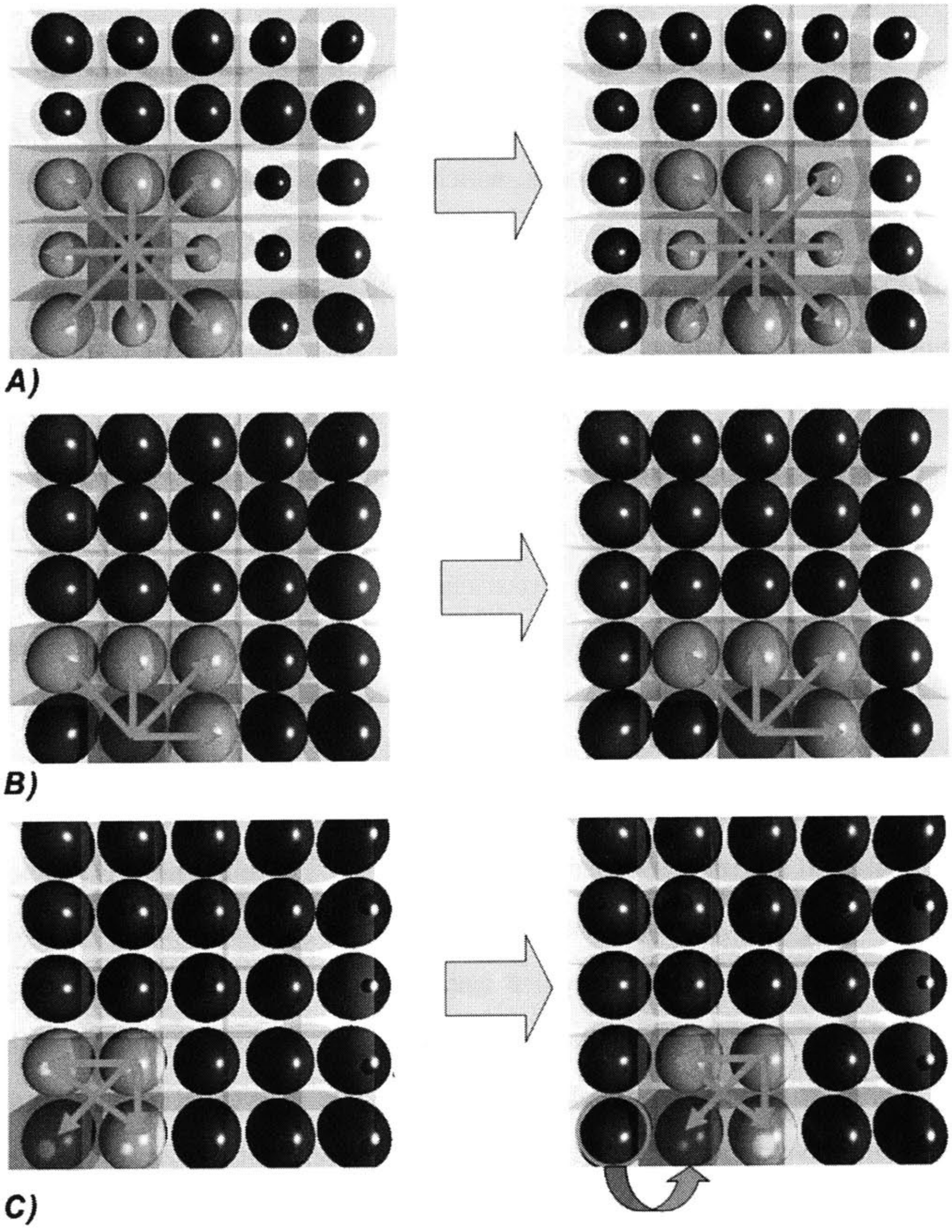


Figure 3-9: Illustration of masks and progression of algorithms along axis in the 2D case for A) unmodified Cell-Verlet algorithm, B) NBS algorithm, and C) CGRID. Note that in case C) the bins have a side length of the smallest body

and placed in the user directory. At run-time, the *AssemblyManager* object is tasked with discovering the user-created assemblies and exposing them to method servers in the application through a reflection-based query interface. This is portrayed as C# code as follows:

```
using System;
using System.Collections;
using DEM3D;
namespace DEM3D.NeighborSorting
{
    /// ⟨ summary⟩
    /// Summary description for INeighborSorter.
    /// ⟨ summary⟩
    public interface INeighborSorter
    {
        /// ⟨ summary⟩
        /// Initialize method is called before the neighbor sorter is used and
        /// is used to initialize the INeighborSorter class instance
        /// ⟨/summary⟩
        /// ⟨ param name=”particleList”⟩ Current bodies⟨/param⟩
        /// ⟨ param name=”numdims”⟩Dimensions to sort⟨/param⟩
        void Initialize(ArrayList particleList, int numdims);

        /// ⟨ summary⟩
        /// UpdateAndForm method is used to update particle extrema and then
        /// populate the neighbor sorter object.
        /// ⟨/summary⟩
        /// ⟨ param name=”particleList”⟩IContact objects⟨/param⟩
        /// ⟨ param name=”neighborList”⟩Neighbor list to populate⟨/param⟩
        void UpdateAndForm(ArrayList particleList, NeighborList neighborList);
        /// ⟨ summary⟩
        /// UpdateParticleNumber is used to update the neighbor sorting object
        /// if the number of particles in the system changes.
        /// ⟨/summary⟩
        /// ⟨ param name=”particleList”⟩IContact objects⟨/param⟩
        void UpdateParticleNumber(ArrayList particleList);
    }
}
```

From the above declaration, it should be noted that there are facilities for memory-based contact detection algorithms as well as initialization and updating methods. Since the requirements vary between algorithms, not all of these methods must provide functionality for every algorithm.

3.8 Particle Geometry

Representing computational geometry in a generic manner is a non-trivial task due to the varied nature of representations. Geometry can be viewed from several different perspectives, and it is the goal of the DEM3D shape definition to allow for the greatest variety of representations without sacrificing the efficiency of the simulation pipeline. To this end, a generic primitive base is allowed while an additional constructive geometry approach is taken for the formation of especially complex assemblies.

All primitives in the DEM3D application adhere to several interfaces based on requirements of different components in the simulation:

- INamed - for uniquely identifiable bodies
- IComparable - for determining a consistent ordering
- IFormattable - for string-based output formatting
- IWrite - for string-based output

- IEnergy - for determining tenergy metrics for the body

- IExtrema - for axis-aligned bounding box determinations as needed by contact detection algorithms and various contact resolution schemes

- IContact - for geometrically interacting bodies considered to be in physical contact

Because the abstract Shape class implements the above-mentioned interfaces and offers many of the methods required across general primitives, it is advantageous to derive new primitives from this class.

3.9 Geometric Primitives

Several primitives have been developed for use in DEM3D and are available in the release version:

- Sphere

- Ellipsoid: including contact resolution algorithms for ellipsoid-ellipsoid interaction which are detailed in Chapter 4

- Pseudo-ellipsoid: a prolate spheroid approximation using the draftsman's 3-arc approximation of an ellipse rotated about a central axis; this also includes contact resolution algorithms which are detailed in Section 4.2

- HollowCylinder: a boundary element formed from the geometric difference of a cylinder and R^3

- HollowSphere: a boundary element formed from the geometric difference of a cylinder and R^3

- Plane

- Polyhedron: a boundary element of triangular facets with edge and vertex connectedness, which is detailed in Chapter 8

3.10 Contact Resolution

In this thesis ”contact resolution” is taken to be the process of determining the detailed contact condition between two primitives. This is also known as ”intersection” and ”detailed contact” in other texts. For contact resolution, two methods are required by the *IContact* interface:
public bool InContact($<$ *ClassName* $>$ *otherBody)*
public bool DetailedContact($<$ *ClassName* $>$ *otherBody)*
As implied by the method names, these methods require that every *IContact* class implementation provide answers for whether another *IContact* class implementation is geometrically interfering (*InContact*) and what the detailed contact condition with another *IContact* class implementation is (*DetailedContact*).

Unlike much of the DEM3D architecture, the contact resolution framework has been significantly influenced by the choice of programming language. Because DEM3D is written in C# and C# is a ”type safe” language (i.e., it requires that every data object be strongly typed), every shape pair interaction type must be defined for every *IContact* implementing class. The minimum theoretical complexity of the definition of contact resolution irrespective of programming language is $O(n^2)$, where n is the number of interacting body types (i.e., *IContact* implementing classes). The implementation in this application requires twice the number of methods as the minimum theoretically required, though, this should be easily remedied with the forthcoming release of Visual Studio 2005. The extra difference between the minimum theoretical complexity is often very simply defined with a method of the following form, where the *shape* variable is of a type that has a definition for the pairwise contact resolution algorithm:

```
public bool DetailedContact(<IContact implementing type> shape)
{
    return shape.DetailedContact(this);
}
```

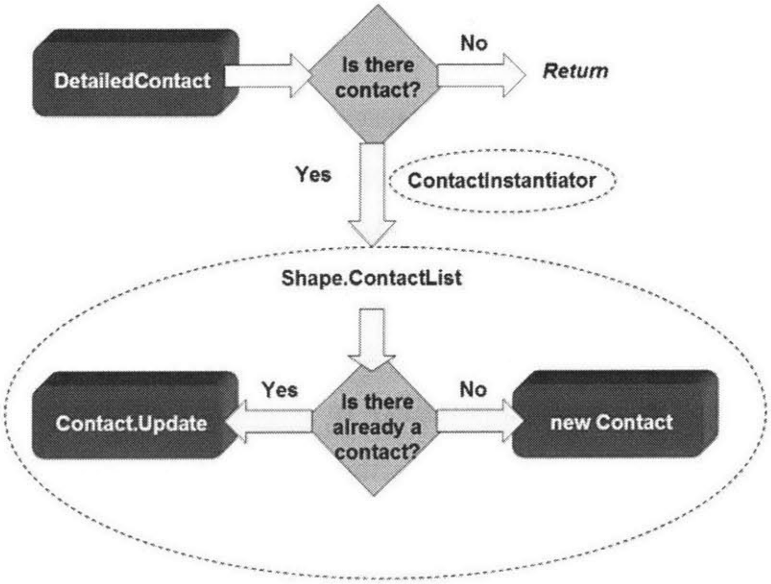


Figure 3-10: Contact handling protocol used in DEM3D

In the base *Shape* class, *DetailedContact* is implemented with a default contact handling protocol that uses a *ContactList* class to manage the contact pairs "owned" by the body. The *ContactList* object determines whether a contact detected at a timestep has been detected previously, in which case it instructs the extant *Contact* derived class instance to update based on the new contact information stored in a *ContactInstantiator* class instance. If the contact is completely new, a new *Contact* derived class instance is created and added into the *ContactList* instance's list of contacts. This internal list may or may not be sorted depending on the length of the owned contacts, which allows an internal algorithm to determine whether the cost of sorting is offset by the benefit of binary search. Also, the *Contact* derived class used to initialize new contacts can depend on the type of particle in the system; a smooth particle hydrodynamics (SPH) point will, for instance, manage SPH contacts, which are a fundamentally different type of "contact" than a physical contact between a pair of solid bodies. The protocol for handling contacts on a per timestep basis is shown in Figure 3-10.

3.11 Contact Hierarchy

The design goal for contacts has been to represent not simply a spring-dashpot contact between spheres but to provide a framework around which generalized interactions between compactly supported entities could be created. This allows the framework to approach general discrete (or particle) modeling approaches with little alteration to the code in general and no alteration to the core of the framework. It is also designed to assure backward compatibility to retain the efficiency and capabilities present in previous builds. Accommodating these requirements, however, is non-trivial, requiring an architecture that is extensible and efficient.

Realizing this design goal has centered to a large extent around the handling of contacts. It is generally accepted that contacts control the scaling of the simulation, since the number of contacts often far outnumbers the number of bodies. For the simulations described later under the pharmaceutical modeling (Chapter 9) and heap base stress distribution (Chapter 6) studies this is especially true, as both of these types of systems often involve dense granular materials with non-unary average coordination number.

Contacts are represented in DEM3D through a hierarchy of classes representing the different layers of abstraction of interest for DEM analysis. The generic contact types included currently in DEM3D are: point-to-point, multi-point, and potential interactions. Each of these generic types must also support 1 or more contact laws (e.g., a simple spring-dashpot interaction law for contacting solids).

3.12 Force Resolution

To resolve the forces present in an interaction (contact), a range of contact laws also needs to be accommodated. For this thesis we take the contact law to mean the algorithm used to accomplish force resolution. The contact laws currently include a stochastic implementation of the JKR model [44] of cohesion (which will be discussed in detail in Section 7.3), a simple spring-dashpot idealization for locally deforming elasto-plastic physical contacts, Hertz-Mindlin physical contact [72], Hertz-Mindlin-Deresewiecz [73] physical contact, and smoothed particle hydrodynamics (SPH) "contact" (though, more technically this is a type of quadrature between integration points evaluated for a compactly supported potential interaction).

The minimum information required to specify a contact are the two interacting entities. There are several properties of a contact, however, that are commonly computed to determine whether a contact should be created. These include the contact normal, the contact point, and the geometric overlap. These quantities are often required by a force resolution algorithm, and though these quantities are derivable from the two interacting entities, they are are typically computationally costly to determine and should not be calculated redundantly (i.e., both during contact resolution and force resolution). Because of this, the contact normal, contact point, and geometric overlap have been included in the contact law method signature.

Also, every degree of freedom must be captured in the contact law output of the contact law method. Additionally, because the contact law is typically an anisotropic response depending on the direction (tangential to the surface or normal), it is also useful to separate the tangential force components from the normal force components. Therefore, the output captures the normal force, tangent force, and moment vectors.

3.13 Contributions

This chapter offers an extensible, maintainable, user-configurable and verifiable framework for addressing a broad range of numerical problems in granular mechanics. Examples in capturing behavior of powders in arbitrarily-shaped blenders, as well as the representation of non-spherical elements are all equally possible in this framework. The framework also represents more general particle methods in the form of a smoothed particle hydrodynamics (SPH) extension to the original framework, which was added as an integration point primitive just as a spherical body would have been defined without alteration to the original framework. Finally, this thesis has offered a flexible testing framework in Section 3.2 for providing both user and (limited) functional testing of the application as well as a platform for benchmarking both the full application as well as components and algorithms within the application.

Chapter 4

Ellipsoidal Geometry

This chapter and Chapter 7 on multi-resolution surface representation address issues of representing particle geometry efficiently. This chapter provides two geometric formulations to represent smooth non-spherical particles as quadratic surfaces. The first is restricted to resolving contact between approximations of a prolate spheroid (an ellipsoid where the minor axis lengths are equal), and the second addresses general triaxial ellipsoids. These formulations are compared with other proposed formulations in the literature. An order of magnitude decrease in the computation time is achieved in resolving contact between both prolate spheroids and triaxial spheroids.

As indicated in the introduction, the main drawback to using non-sphere representations for modeling particle geometry is the additional computational requirement. Accordingly, the formulations in this chapter are specifically designed to reduce computational requirements as much as possible, both in terms of CPU cycles and memory.

4.1 Overview

It was mentioned in the introduction that sphere-based methods are typically used as a convenient idealization of particle shape. It is not certain, though, that discarding topological features is valid in the context of the analysis performed with the sphere assumption.

The argument has been set forth that sphere systems can be used to simulate geomechanical discontinua because it can capture the bulk density of the material. Mitchell [75] notes that while the occupation of interstitial spaces creates a typically higher density and lower void ratios, irregular particle shape tends to result in lower density and greater void ratios, and that these two effects combine to present densities and void ratios that are close to those of an equivalent mono-sized sphere system. Unfortunately, mono-sized sphere systems tend to organize into dense, ordered packings as the coefficient of friction decreases [35], which degenerates the validity of the model in many ways. For instance, the stress-strain anisotropy of the material, a fundamental property of soil, is lost, reducing the effectiveness of continuum methods for analysis.

Even for sphere systems that follow a random distribution of sizes, this problem can arise under dynamic excitation of the medium, such as in seismically excited soils and in mixed particulate matter found in the pharmaceutical and mining industries. It has been commonly observed that particles tend to segregate into mono-sized regions due to free surface segregation (in surface cascades), interparticle percolation (in failure zones), and particle migration (high strain rate regions) [11].

In general, spheres have several drawbacks that hinder their ability to capture key behaviors of real particle systems that contain non-spherical particles. The constant radial geometry constrains the center of curvature for every point on the surface of the sphere to be collocated with the centroid, which constrains all normal forces to pass through the centroid and precludes particle interlocking in the gross sense. Because of the normal forces always passing through the centroid, spheres cannot exert resistive couples, which can lead to excessive rolling when subjected to small perturbations. Excessive rolling results in a proportionally higher transfer of energy to rotational kinetic energy than would be observed in systems of aspherical particles. Some numerical implementations address this shortfall by introducing couples artificially by either perturbing the normal force direction so that it does not pass through the centroid (random asphericity) or by adding a "rolling friction" factor [121] to dampen pathological rolling.

Some of these problems have been overcome by clustering a number spheres together to form a non-spherical shape, as adopted in studies by Jensen [43], Favier [29], Vu-Quoc [122], and Mustoe [82]. Typically the discretization is coarse (7 spheres or less) in order to achieve fast execution times. However, the clustering is only C^0 continuous and, therefore, introduces discontinuous surface normals that lead to computational problems especially in contact resolution with polyhedral elements. Favier [28] also notes that resistive behavior induced by the C^0 continuity of the surface can reach as high as 40% of the normal force between such clustered sphere particles, restricting the types of materials such particles can accurately represent. In this manner, ellipsoids modeled are typically axi-symmetric, which creates an anisotropic friction that acts along the axial direction of the surface (as opposed to the transverse direction), adding further complications in adapting these shapes for use in modeling physical particles. Experiments have also been conducted with shallow cylindrical ellipsoidal metal punches in a narrow channel depth to compare the behavior of numerically simulated 7 disc ellipsoid approximations with ellipsoidal shapes and found poor agreement between the numerical and observed system behaviors [28].

This discussion is not meant to suggest that any use of spheres in modeling granular material is invalid. Sphere-based models have been used with great success by Vu-Quoc [122] to validate contact laws between spherical glass beads. Also, in powder angle of repose studies, disk elements have been used to accurately predict the internal angle of friction for lactose. For materials where the cohesive/adhesive interaction dominates the behavior (typically, this also indicates spheroidal material), a sphere-based DEM formulation is often used [117], as it is in this thesis to probe the behaviors of blended cohesive pharmaceutical powders.

The point of the previous discussion is to highlight the limitations of spheres in capturing some behaviors, such as granular flow from hoppers and mixing in blenders. It requires caution when assuming topological simplifications of a particle for use in a granular model. The following discussion will make this point more clear by developing a framework to determine the topological simplifications that are allowable for a given application.

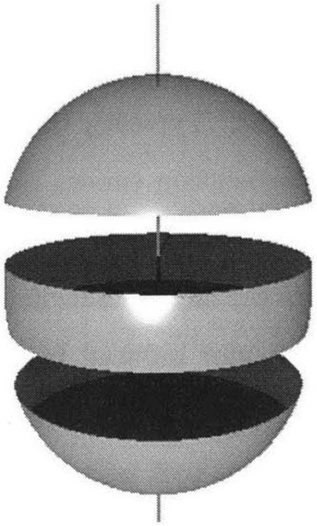


Figure 4-1: Decomposition of the prolate spheroid approximation into its 3 constituent parts (spherical caps and a clipped toroidal body).

4.2 Prolate Spheroids

The prolate spheroid approximation (PSA) is the rotation of the draftsmen's 3-arc approximation of the ellipse rotated about a central axis. Using this primitive in DEM was first proposed by Wang [127] and refined by Johnson [45, 46] and Kuhn [56]. The primitive is composed of 2 spherical caps fit onto the top and bottom of a clipped torus as illustrated in Figure 4-1. This thesis forwards a new method for determining the contact condition between two such bodies; the method is both more robust and more efficient than previously developed methods as shown in Section 4.4.

There are several lower level optimizations possible in refining the algorithm in [127] that are fairly trivial and have been implemented in the code associated with this thesis. For the benchmark comparisons in this thesis, the results are after the low-level optimizations have been implemented for the previous algorithm [127] and will not be discussed further here. The major difference, however, between the algorithms proposed above is in the determination of contact between the middle sections of two PSA's (as shown in the right hand part of Figure 4-2). Before addressing this point, determining which contact case occurs and resolving the other contact cases will be

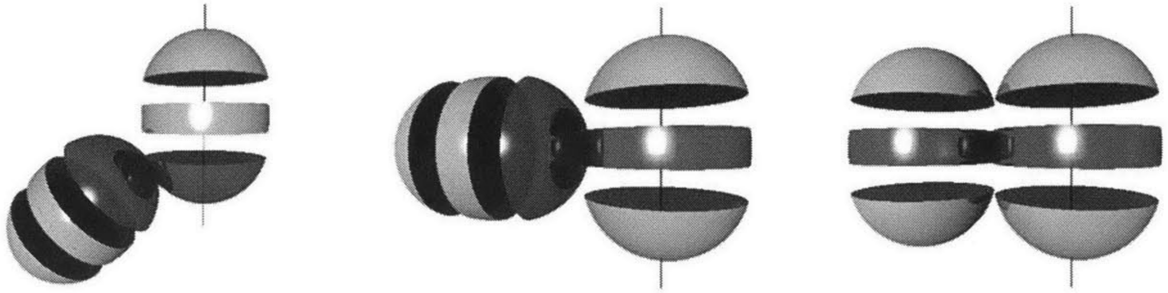


Figure 4-2: Cases of contact between PSA primitives: ends (left), end-middle (center), and middles (right).

reviewed.

To determine the contact case, the following equation is evaluated (note: this equation instead of that given in [127] allows us to evaluate a simple dot product instead of a transcendental function):

$$\frac{\vec{CoG_j} - \vec{CoG_i}}{|\vec{CoG_j} - \vec{CoG_i}|} \cdot -\vec{P_i} \geq \frac{CG\vec{middle}_i}{radius_{middle,i} - radius_{end,i}} \tag{4.1}$$

where $\vec{CoG_i}$ is the center of gravity of body i, $\vec{P_i}$ is the directional vector for body i, $CG\vec{middle}_i$ is the distance of the vertex of the middle section arc for body i from the major axis, $radius_{middle,i}$ is the radius of the middle section arc of body i, and $radius_{end,i}$ is the radius of the end cap of body i.

If the equation evaluates *false*, the end cap of body i is involved in any possible contact; otherwise, the middle section is involved. Evaluating the equation for both bodies in the contact pair determines which case in Figure 4-2 applies.

The contact cases may all be reduced to determining contact between 2 equivalent spheres, as shown in the following discussions of the contact cases. Because of this, the method offered here has been named the *Equivalent Spheres Method.*

If the contact involves the end cap for both bodies, then the problem reduces to the trivial determination of intersection between two spheres. If the following evaluates *false*, there is no contact; otherwise, contact occurs:

$$radius = radius_{end,i} + radius_{end,j} \tag{4.2}$$

$$\vec{CoG}_{end,j} = \vec{CoG}_j + \hat{R}_j^T[0, \pm CGend_j, 0] \tag{4.3}$$

$$\vec{distance} = \vec{CoG}_{end,j} - \vec{CoG}_{end,i} \tag{4.4}$$

$$radius^2 - \vec{distance} \cdot \vec{distance} \geq 0 \tag{4.5}$$

where $\hat{R}_i$ is the rotation transform matrix for body i.

If the contact involves the end cap and middle section, then the problem also reduces to the trivial evaluation of contact between spheres. That is, if the following evaluates *false*, there is no contact; otherwise, contact occurs (note: the form of the following equations are identical to the above equations for the last two equations; that is, this is sphere contact):

$$radius = radius_{end,i} + radius_{middle,j} \tag{4.6}$$

$$\vec{CoG}_{end,i} = \vec{CoG}_i + \hat{R}_i^T[0, \pm CGend_i, 0] \tag{4.7}$$

$$\vec{CoG}_{middle,j} = \vec{CoG}_j + \hat{R}_j^T(CGmiddle_j \cdot \vec{P}_j) \tag{4.8}$$

$$\vec{distance} = \vec{CoG}_{middle,j} - \vec{CoG}_{end,i} \tag{4.9}$$

$$radius^2 - \vec{distance} \cdot \vec{distance} \geq 0 \tag{4.10}$$

Finally, the case of middle section contact must be considered; this is analogous to the other cases in that two equivalent spheres are determined and then checked for intersection. If the following evaluates *false*, there is no contact; otherwise, contact occurs:

$$radius = radius_{middle,i} + radius_{middle,j} \tag{4.11}$$

$$\vec{CoG}_{middle,j} = \vec{CoG}_j + \hat{R}_j^T(CGmiddle_j \cdot \vec{P}_j) \tag{4.12}$$

$$\vec{distance} = \vec{CoG}_{middle,j} - \vec{CoG}_{middle,i} \tag{4.13}$$

$$radius^2 - \vec{distance} \cdot \vec{distance} \geq 0 \tag{4.14}$$

All of the cases from Figure 4-2 have been detailed above; however, there is a missing piece. We have all of the information we need except the value of the directional vector $\vec{P}_i$. This vector is necessary for not only determining which contact case candidate is possible but also for determining the position of the equivalent sphere for the middle section (if the middle section is involved in contact).

Determining this directional vector is handled differently by the different algorithms (i.e., [127, 45, 46, 56]). Wang [127] uses a simple sub-partitioning of the great circle about the centroid of the PSA perpendicular to the major axis. The algorithm then tries to determine the points at a maximal distance to determine the point of smallest potential as being on the opposite side of the circle. This does not, however, necessarily achieve the desired result. Interestingly, the error is subtle and is not exposed through the trivial cases of contact between co-planar and orthogonal bodies.

Why is this? From first glance, the approach seems reasonable, but the method does not guarantee that the contact vector pass through the axis of rotation, which is a requirement of any surface of rotation. For co-planar and orthogonal bodies, the method will coincidentally pass through the major axis of both bodies; however, for other rotations, the result will diverge from the true solution. In this thesis, we provide for guaranteeing that the contact vector pass through the major axis of both bodies. The approach here not only guarantees correct contact but performs as well as the previous algorithm, even though it evaluates a second constraint.

The general approach is illustrated in Figure 4-3. This same approach is used in other iterative approaches, and has been abstracted as a general process in the code

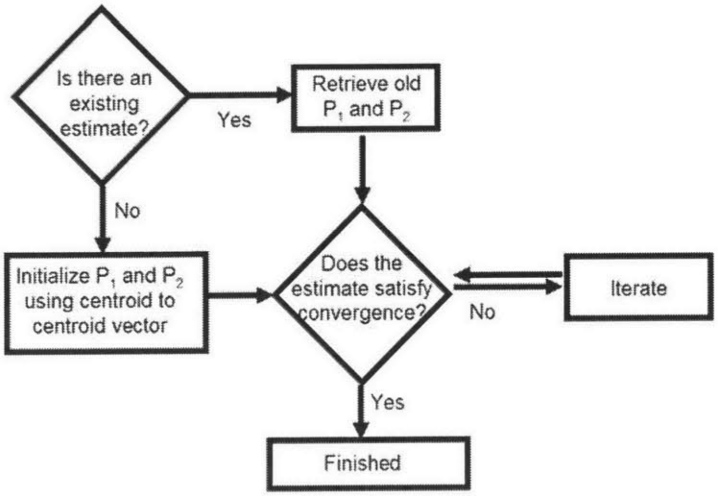

Is there an existing estimate?
Yes
Retrieve old P_1 and P_2
No
Initialize P_1 and P_2 using centroid to centroid vector
Does the estimate satisfy convergence?
No
Iterate
Yes
Finished

Figure 4-3: Flow diagram for determining the direction vectors $\vec{P_i}$ and $\vec{P_j}$.

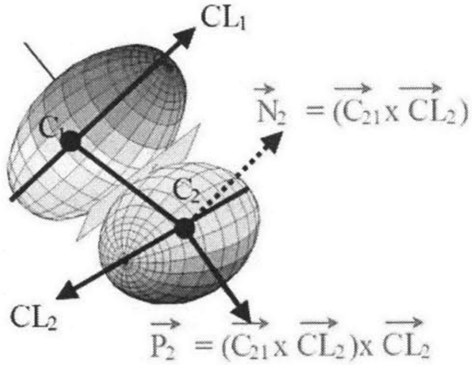

CL_1
$\overrightarrow{N_2} = (\overrightarrow{C_{21}} \times \overrightarrow{CL_2})$
C_1
C_2
CL_2
$\overrightarrow{P_2} = (\overrightarrow{C_{21}} \times \overrightarrow{CL_2}) \times \overrightarrow{CL_2}$

Figure 4-4: Illustration of the iteration method for determining geometric intersection of the middle sections for a pair of PSA's

using a concept of contact servers and contact objects, which can be stored for reuse in successive timesteps. This can reduce the computational requirement considerably.

The initialization of the values is derived from the component of the centroid-to-centroid vector orthogonal to the major axis as shown in Figure 4-4 and defined

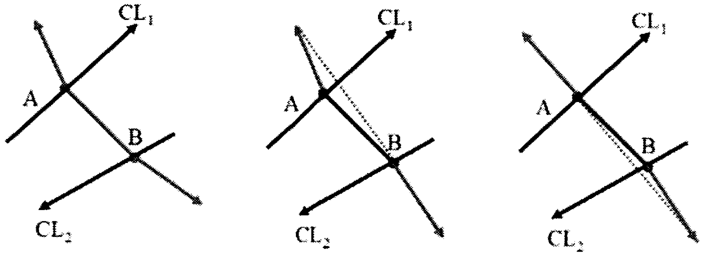


Figure 4-5: Illustration of the iteration method for determining geometric intersection of the middle sections for a pair of PSA's

below:

$$\vec{d} = C\vec{o}G_{middle,j} - C\vec{o}G_{middle,i} \tag{4.15}$$

$$\vec{d_i} = \hat{R}_i\vec{d} \times [1,0,0] \times [1,0,0] \tag{4.16}$$

$$\vec{d_j} = -\hat{R}_j\vec{d} \times [1,0,0] \times [1,0,0] \tag{4.17}$$

$$\vec{P_i} = \frac{\vec{d_i}}{|\vec{d_i}|} \tag{4.18}$$

$$\vec{P_j} = \frac{\vec{d_j}}{|\vec{d_j}|} \tag{4.19}$$

$$\tag{4.20}$$

Refinement is accomplished through a geometric iteration as shown in Figure 4-5 and defined below:

$$\vec{d} = C\vec{o}G_{middle,j} - C\vec{o}G_{middle,i} \tag{4.21}$$

$$\vec{d_i} = \hat{R}_i(\vec{d} + CGmiddle * \hat{R}_j^T\vec{P_j}) \times \vec{i_i} \times \vec{i_i} \tag{4.22}$$

$$\vec{d_j} = \hat{R}_i(-\vec{d} + CGmiddle * \hat{R}_i^T\vec{P_i}) \times \vec{i_j} \times \vec{i_j} \tag{4.23}$$

$$\vec{P_i} = \frac{\vec{d_i}}{|\vec{d_i}|} \tag{4.24}$$

$$\vec{P_j} = \frac{\vec{d_j}}{|\vec{d_j}|} \tag{4.25}$$

$$\tag{4.26}$$

The simplicity and convergence properties of this algorithm help to bring the computational requirements down. The quantitative description of this reduction will be covered in the last section of this chapter.

4.3 Triaxial Ellipsoids

Addressing the problem of resolving contact between general triaxial ellipsoids is a more difficult problem. The contact must no longer necessarily pass through the major axis of the body, which loses one of the constraints on our problem. However, there are other techniques that become more attractive as constraints on the solution space are removed. Before looking at a more appropriate algorithm, it is instructive to analyze previous algorithms.

Ouadfel [92] uses a somewhat complicated approach to address general quadrics. An example is given for triaxial ellipsoids where an initial constraint on the contact is obtained through the projection of the intersection of the bounding spheres of the ellipsoid pair onto the candidate ellipsoid. This is then used as a means of finding a candidate set of curves for determining the contact area. This approach is used when exact contact volumes are required and will not be reviewed further here.

Lin and Ng [65] rely on the minimization of the potential function to determine the intersection of two ellipsoids. This minimization results in a 6 degree polynomial, where the zeros of the polynomial are used to determine the point of contact. As with any solution of multiple degree polynomials, there are issues here not only with computational intensity but also with stability of the algorithm for pathological and ill-conditioned polynomials resultant from the approach.

This thesis adopts the idea of using the potential function as a means for determining a contact point between two ellipsoids. Borrowing concepts from work on the parameterization of the mapping of convex surfaces using a 2-variable parameterization, this section introduces a new *Directed Surface Walk* method for iterating toward a solution for the contact point.

The method is initialized by intersecting the centroid-to-centroid vector with the host surface and then restricting the searchable surface to the hemi-ellipsoid created

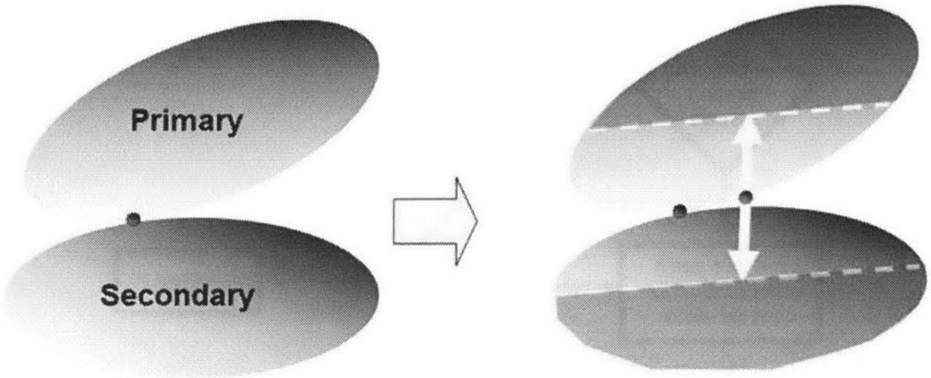


Figure 4-6: Initialization of the *Directed Surface Walk* algorithm for determining contact between general triaxial ellipsoids; the red dot represents the true contact point, the green dot indicates the initial guess, and the red shaded area is the part of the surface excluded from searching.

by the cutting plane perpendicular to the centroid-to-centroid vector, as shown in Figure 4-6.

The intersection with the surface determines an initial point from which to walk along the surface. An appropriate parameterization is to use a simple spherical coordinate system, setting the radius parameter as a function of the θ and ϕ angular parameters, which are used as the orthogonal directions along the surface of the ellipsoid. The equations for the θ and ϕ intersection of the centroid-to-centroid vector with the host surface is given in the following equation:

$$\vec{d} = \hat{R}_i(C\vec{o}G_j - C\vec{o}G_i) \tag{4.27}$$

$$\theta = tan^{-1}(\frac{\vec{d}\cdot\hat{j}}{\vec{d}\cdot\hat{i}}) \tag{4.28}$$

$$\phi = cos^{-1}(\frac{\vec{d}\cdot\hat{k}}{|\vec{d}|}) \tag{4.29}$$

Using the initial θ and ϕ parameter values, a search range is established over the hemi-ellipsoid centered at the point. The search range extends $\frac{\pi}{2}$ in each parameter direction, such that, the range becomes $[\alpha_0 - \frac{\pi}{2}, \alpha_0, \alpha_0 + \frac{\pi}{2}]$ where α_0 is the initial θ

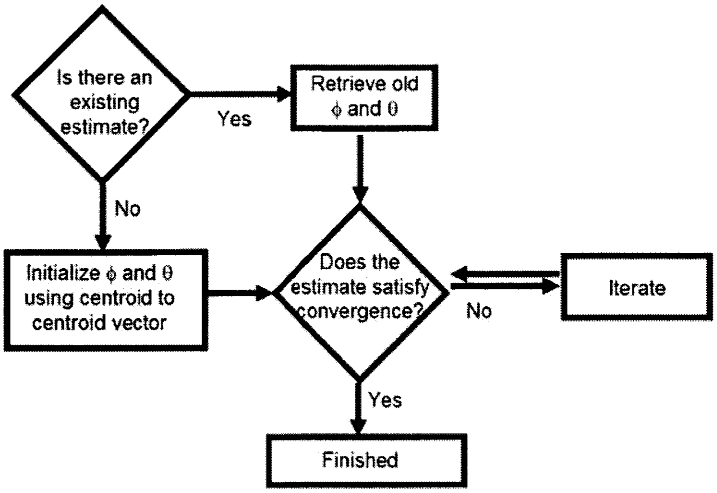

Is there an existing estimate?
Yes
Retrieve old ϕ and θ
No
Initialize ϕ and θ using centroid to centroid vector
Does the estimate satisfy convergence?
No
Iterate
Yes
Finished

Figure 4-7: Flow diagram for determining the θ and ϕ location of the contact point along the surface.

and ϕ parameter value. The step is illustrated in Figure 4-8.

The flow diagram for the iteration is illustrated in Figure 4-7. The *Directed Surface Walk* scheme follows a simple decision each iteration: walk or reduce range. The algorithm walks along the surface in the direction of least potential energy where the potential energy is calculated with respect to the potential of the non-host surface as shown in Figure 4-10. If the direction of least potential is at the current position, then the search range is reduced by half as illustrated in Figure 4-9. Through iteration, this algorithm converges to the potential minimum. Though it is unclear whether the potential minimum identifies the most appropriate point as the contact point, since it is not guaranteed that surface normals will match at this point. In practice, though, it has been found that the resultant point is nearly indistinguishable from the position where the surface normals are parallel [65].

To evaluate the potential with respect to the non-host surface, the non-host surface must first be represented as a polynomial form of 3 variables (Cartesian coordinates in the convective frame of the host surface). This can be obtained through the following procedure:

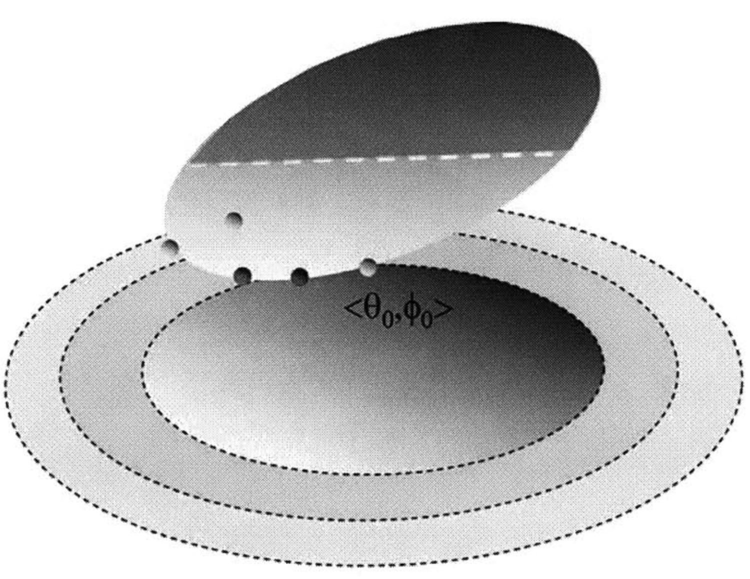


Figure 4-10: The walk decision for the *Directed Surface Walk* algorithm.

$$\hat{t} = \hat{R}_B\hat{R}_A^T \tag{4.30}$$

$$\vec{c} = \hat{R}_B(C\vec{o}G_A - C\vec{o}G_B) \tag{4.31}$$

$$\gamma_i = \frac{1}{{r^2}_i} \tag{4.32}$$

$$\vec{\alpha}_i = [\hat{t}[i,0], \hat{t}[i,1], \hat{t}[i,2], \vec{c}_i] \tag{4.33}$$

$$\hat{q}_i[i,j] = \alpha_i \cdot \alpha_j \tag{4.34}$$

$$\hat{q} = \sum \gamma_i\hat{q}_i \tag{4.35}$$

where r_i is the radius of the ith coordinate (i.e., $\hat{\imath}$ direction is 0, etc.), $\hat{t}$ is the rotational transform matrix to transform from the convective frame of body A to the convective frame of body B, $\vec{c}$ is the translational transform vector to transform from the convective frame of body A to the convective frame of body B, $\hat{q}_i$ is the polynomial matrix for the ith coordinate, and $\hat{q}$ is the polynomial matrix representation of body B in the convective frame of body A. The polynomial matrix $\hat{q}$ is the matrix necessary for the following to be true:

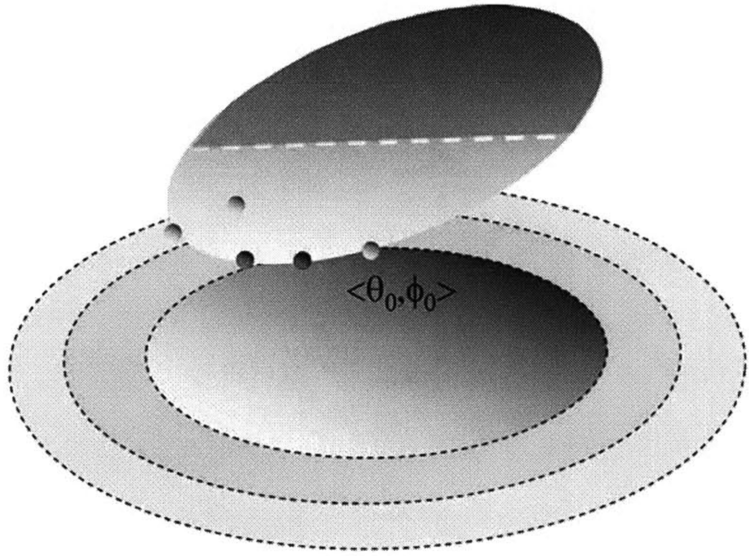


Figure 4-10: The walk decision for the *Directed Surface Walk* algorithm.

$$\hat{t} = \hat{R}_B \hat{R}_A^T \tag{4.30}$$

$$\vec{c} = \hat{R}_B(\vec{CoG}_A - \vec{CoG}_B) \tag{4.31}$$

$$\gamma_i = \frac{1}{{r^2}_i} \tag{4.32}$$

$$\vec{\alpha_i} = [\hat{t}[i,0], \hat{t}[i,1], \hat{t}[i,2], \vec{c_i}] \tag{4.33}$$

$$\hat{q}_i[i,j] = \alpha_i \cdot \alpha_j \tag{4.34}$$

$$\hat{q} = \sum \gamma_i \hat{q}_i \tag{4.35}$$

where r_i is the radius of the ith coordinate (i.e., $\hat{i}$ direction is 0, etc.), $\hat{t}$ is the rotational transform matrix to transform from the convective frame of body A to the convective frame of body B, $\vec{c}$ is the translational transform vector to transform from the convective frame of body A to the convective frame of body B, $\hat{q}_i$ is the polynomial matrix for the ith coordinate, and $\hat{q}$ is the polynomial matrix representation of body B in the convective frame of body A. The polynomial matrix $\hat{q}$ is the matrix necessary for the following to be true:

$$[x, y, z, 1]\hat{q}[x, y, z, 1]^T = \Gamma_{11} \tag{4.36}$$

At each iteration, the $[\alpha_0 - \frac{\pi}{2}, \alpha_0, \alpha_0 + \frac{\pi}{2}]$ range values are used to acquire the corresponding radius parameter and then the full spherical coordinate set is transformed to Cartesian coordinates $[x, y, z]$ using Equations 4.37, 4.38, 4.39, and 4.40. These coordinates are then used in Equation 4.36 to determine which of 4 orthogonal directions, $[\pm\phi,\ \pm\theta]$ or current position, $[\theta, \phi]$, has the lowest value of the potential function for the non-host body, Γ_{11}.

$$\rho^{-1} = \sqrt{\frac{cos^2(\theta)sin^2(\phi)}{{r^2}_x} + \frac{sin^2(\theta)sin^2(\phi)}{{r^2}_y} + \frac{cos^2(\phi)}{{r^2}_z}} \tag{4.37}$$

$$x = \rho \cdot sin(\phi) \cdot cos(\theta) \tag{4.38}$$

$$y = \rho \cdot sin(\phi) \cdot sin(\theta) \tag{4.39}$$

$$z = \rho \cdot cos(\phi) \tag{4.40}$$

4.4 Contributions

The suite of algorithms presented in the previous sections extend DEM to address the resolution of contact from the traditional sphere-based formulation to resolving contact between general quadratic surfaces. Figure 4-11 illustrates the relative performance of these different algorithms, using a log scale for time. As expected, the more general the shapes to intersect, the more computationally intensive the task. However, the developments in this thesis have helped to reduce the computational requirements by 1-2 orders of magnitude.

As illustrated in Figure 4-11, the resolution of contact between general triaxial ellipsoids as proposed by this thesis is an order of magnitude more efficient than that proposed previously [65]. If the quadratic surface is restricted to prolate spheroids (i.e., ellipsoids with minor axes of equivalent length), an algorithm has been offered that reduces the computational intensity by another order of magnitude. Figure 4-11

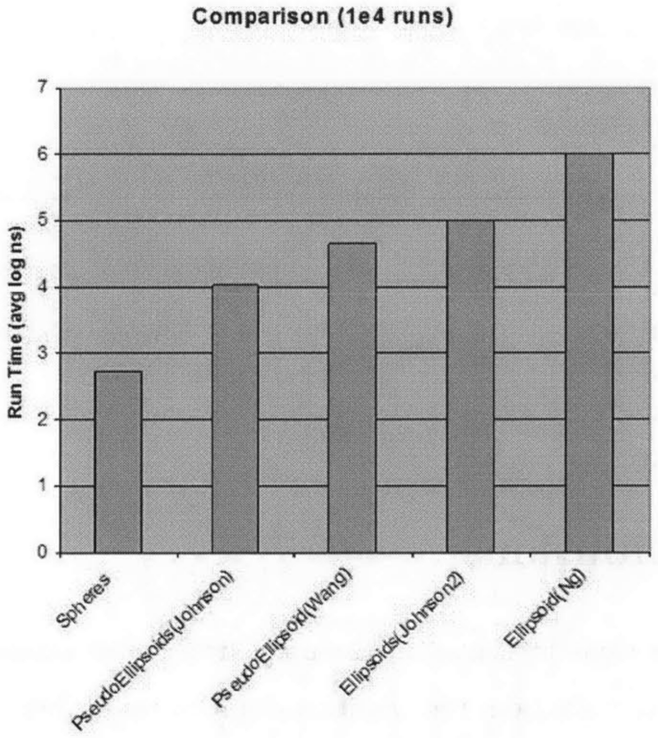


Figure 4-11: Comparison of the run-time per iteration for algorithms for resolving contact between quadratic surfaces aligned along the positive x-axis.

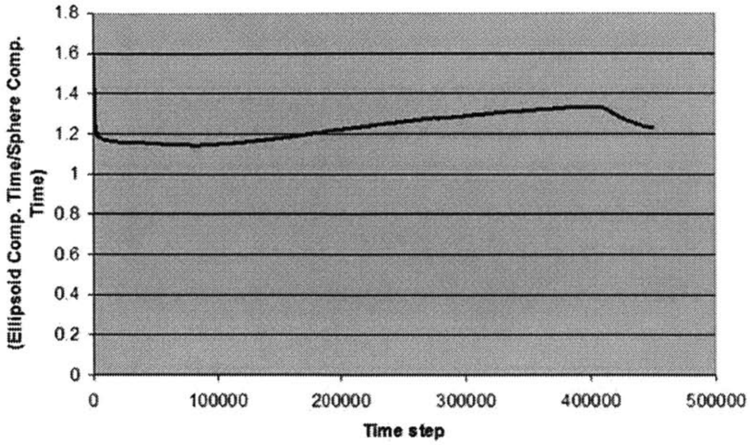


Figure 4-12: Comparison of the run-time ratio per time step for prolate spheroids versus spheres in a simulation of particle deposition from a point source.

also shows a four-fold performance gain by the prolate spheroid contact resolution algorithm here over that in [127] for axially-aligned contact. It has also been shown that [127] minimizes an incorrect energy function and will converge to increasingly incorrect values as aspect ratio of the primitive is increased.

Because of the relative computational intensity of detailed contact resolution and other parts of the simulation pipeline (e.g., results output, neighbor-sorting, etc.), in practice the differences between spheres and prolate spheroids are relatively minor. A simulation of the formation of granular heaps using spheres and prolate spheroids has been performed using the DEM3D batch application, and the relative performance of each simulation is shown in Figure 4-12, with a mean ratio of prolate spheroid system to sphere system runtimes of 1.23.

This new capability to address problems using aspherical geometries in a computationally tractable manner is an important step toward capturing the effect of shape on the behavior of granular systems. The first order effect of shape has been noted by many researchers as noted in the introduction to this chapter, and this is reinforced in the following section describing work performed analyzing the base stress of granular heaps.

Chapter 5

Integration of Body Rotation

Solving the underlying physical equations governing a system is a fundamental requirement of any simulation architecture. Modeling the physical behavior of a granular material problem hinges on the ability to solve the ordinary differential equations that govern the system.

In the case of discrete element modeling, for translational motion at each time step a system of decoupled ODE's of the form of Newton's Second Law must be solved:

$$m_i\ddot{u}_i(t) + c_i\dot{u}_i(t) + k_i u_i(t) = f_i(t) \tag{5.1}$$

which can be written as:

$$\ddot{u}_i(t) = \frac{f_i(t)}{m_i} - 2\zeta\omega\dot{u}_i(t) - \omega^2 u_i(t) \tag{5.2}$$

where the subscript i indicates the particle index, m is the particle mass, c is a damping coefficient, k is a stiffness, the fundamental frequency $\omega = \sqrt{\frac{k_i}{m_i}}$, and the mass-proportional damping coefficient $\zeta = \frac{c_i}{2\sqrt{k_i m_i}} = \frac{c_i}{2m_i\omega}$.

For rotational motion, the complication of maintaining consistency between local and global reference frames is added. This will be discussed further in the subsection on quadrature for rotational motion. Together, translational motion and rotational motion define the 6 degrees of freedom extant for each particle in a DEM simulation.

There is no constant definition of $f_i(t)$ given here, since the specific function is determined by the contact law applied. Contact models are described in the next section. Because of the assumption of Newtonian mechanics, there is also a scale (in the range where quantum effects can no longer be assumed negligible) at which this type of modeling will be invalid if not modified.

This second order ODE can be re-written in state space formulation as a set of first order ODE's, where we let $v = \dot{u}$:

$$\begin{bmatrix} \dot{v} \\ \dot{u} \end{bmatrix} = \begin{bmatrix} -2\zeta\omega & -\omega^2 \\ 1 & 0 \end{bmatrix} \begin{bmatrix} v \\ u \end{bmatrix} + \begin{bmatrix} \frac{f_i(t)}{m_i} \\ 0 \end{bmatrix} \tag{5.3}$$

The analytical solution to the case where $f(t) = 0$, is:

$$u(t) = e^{-\zeta\omega_n} \left[x_0 cos\omega_d t + \left(\frac{\dot{x}_0 - \zeta\omega_n x_0}{\omega_d} \right) sin\omega_d t \right] \tag{5.4}$$

where $\omega_d = \omega_n\sqrt{1-\zeta^2}$.

Unfortunately, the equations that need to be solved in DEM are not as simple as this. The forcing function $f(t)$ may not be $f = 0, \forall t$ and may contain spring and damping terms that are functions of time and/or other particle states. Also, stiffness and damping only act when two bodies are in contact, requiring a nonlinear stiffness and damping term (i.e., stiffness and damping terms for contact disappear when bodies are not in contact).

There are several numerical tools available to integrate first order ordinary differential equations. These can be divided into implicit and explicit algorithms. Previous research has shown that multistep algorithms (also known as implicit methods or, for specific algorithms, predictor-corrector algorithms) offer little gain in accuracy for discretized simulations and incur a substantial computational cost. These methods will not be discussed further here. Instead, we will investigate Taylor series expansion methods that belong to the set of explicit numerical methods, which include the common Euler and Runge-Kutta methods. These will be the focus of the discussion on numerical methods. For further reading any number of elementary numerical analysis texts may be consulted, such as Fausett [26].

5.1 Overview of Quadrature

As discussed in the previous section, explicit time integration schemes are preferable for a DEM system. This is due in large part to the large number of degrees of freedom with highly dynamic nodal connectivity that are present in a DEM system, which make implicit (multi-step or iterative) integration algorithms impractical to use. Moreover, the discontinuous nature of the ODE's being solved may lead to less accurate results using implicit schemes versus explicit.

The most common explicit time integration algorithms are of the Taylor series expansion class. In the Taylor series expansion, the derivative of the function is expanded into a summation of higher order derivatives. The general Taylor expansion is of the form:

$$x(t + \Delta t) = \sum x^{(n)}(t)\frac{(\Delta t)^n}{n!} \tag{5.5}$$

where $n \geq 0$

Substituting, $x(t) = \dot{u}(t)$, the equation becomes:

$$\dot{u}(t + \Delta t) = \sum u^{(n+1)}(t)\frac{(\Delta t)^n}{n!} \tag{5.6}$$

where $n \geq 0$ and $\ddot{u}(t) = f(u, \dot{u}) = \sum \frac{F_i}{m} - \omega_n^2 u(t) + 2\zeta\omega_n\dot{u}(t)$

The corresponding approximation is then:

$$\dot{u}(t + \Delta t) = \dot{u}(t) + \Delta t f(u(t), \dot{u}(t)) + h.o.t. \tag{5.7}$$

$$u(t + \Delta t) = u(t) + \Delta t\dot{u}(t + \Delta t) + h.o.t. \tag{5.8}$$

There is some value τ for $u(\tau)$ such that 5.7 and 5.8 can be written exactly. Using the Central Value Theorem, $t \leq \tau \leq (t + \Delta t)$, and 5.7 and 5.8 can be rewritten without neglecting the higher order terms (h.o.t.) as:

$$\dot{u}(t+\Delta t) = \dot{u}(t) + \Delta t f(u(t), \dot{u}(t)) + \frac{\Delta t^2}{2}\dot{f}(u(\tau), \dot{u}(\tau)) \tag{5.9}$$

$$u(t+\Delta t) = u(t) + \Delta t\dot{u}(t+\Delta t) + \frac{\Delta t^2}{2}\ddot{u}(\tau) \tag{5.10}$$

This can now be used to design a numerical approach to solving ordinary differential equations (ODE's). Since we do not know τ, we cannot know $\ddot{u}(\tau)$, and the $\ddot{u}(\tau)$ term can be written as $-\Theta(\Delta t)$, the local truncation error.

As an illustration, rewrite 5.10 as:

$$\dot{u}(t+\Delta t) = \frac{u(t+\Delta t) - u(t)}{\Delta t} + \Theta(\Delta t) \tag{5.11}$$

This is the form of the Forward Euler algorithm. The order of the Θ term is 1, indicating this as a first order algorithm.

5.1.1 Multiple Order

In the previous section, the Forward Euler algorithm (5.11) was developed as an illustration of designing a numerical integration algorithm from the Taylor series expansion. Because of the magnitude of the local truncation error, the first order methods perform poorly over several timesteps and are rarely used in simulation. For the DEM formulation, a higher order algorithm is needed. This section will investigate higher order numerical integration (quadrature) methods.

Rewriting 5.10 using staggered half steps, yields a more useful approximation of the derivative known as the central difference, or leapfrog, algorithm:

$$\dot{u}(t+\Delta t) = \frac{u(t+\frac{\Delta t}{2}) - u(t-\frac{\Delta t}{2})}{\Delta t} + \Theta(\Delta t^2) \tag{5.12}$$

or, rewritten for our purposes:

$$u(t+\frac{\Delta t}{2}) = u(t-\frac{\Delta t}{2}) + \Delta t \cdot \dot{u}(t+\Delta t) \tag{5.13}$$

To achieve higher orders of accuracy requires a higher order algorithm. The Runge-Kutta family of algorithms satisfies this requirement. Rewriting 5.10 using an extra higher order term yields:

$$u(t + \Delta t) = u(t) + (A + B)\Delta t\dot{u}(t + \Delta t) + BP\Delta t^2\ddot{u}(\tau) \tag{5.14}$$

which imposes the restrictions that $A + B = 1$ and $BP = \frac{1}{2}$. This can be rewritten to solve for $\dot{u}(t + \Delta t)$ in a 2nd Order Runge-Kutta algorithm as:

$$\dot{u}(t + P\Delta t) = \frac{1}{B}\left(\frac{u(t + \Delta t) - u(t)}{\Delta t} - A\dot{u}(t)\right) + \Theta(\Delta t^2) \tag{5.15}$$

Using the arbitrary coefficients $A = B = \frac{1}{2}$ leads to the modified Euler algorithm:

$$\dot{u}(t + \Delta t) = 2\frac{u(t + \Delta t) - u(t)}{\Delta t} - \frac{\dot{u}(t)}{2} + \Theta(\Delta t^2) \tag{5.16}$$

These methods are of sufficient order for the numerical integration of motion in DEM simulations. However, integrating rotational motion requires extra consideration in the formulation and greater precision. To accomplish this, the 4th Order Runge-Kutta algorithm will be introduced in the next section.

5.2 Quadrature for Rotational Motion

The integration of rigid body motion in 3-D space is a nontrivial task that is aggravated by the discrete temporal nature of most multi-body simulation approaches. Updating the rotation matrix over time through successive matrix multiplications can lead to error accumulation. Error accumulation, in turn, can lead to a loss of the orthogonality property of the rotation matrix. Matrix updates can also lead to singular matrices or to "gimbal lock" which results from the alignment of two of the three axes of rotation.

This is often solved through the use of quaternions, which offer a wealth of attractive attributes, including non-singularity under rotation, and ease of renormalization between timesteps to avoid orthogonality losses. Moreover, because there are only 4 degrees of freedom (instead of the 9 found in the rotation matrix), there are fewer operations to carry out to update the rotation of a body. Indeed, the quaternion requires only 26 flops to multiply, compared to 81 flops for matrix multiplication. Conversion from quaternion to orthogonal rotation matrix requires only 24 flops.

Much of the work in integrating the 3D orientation equations for rigid body systems has been performed to address problems where there are large magnitude rotations as in aerospace applications. For instance, a Hamiltonian formulation for the equations of motion has been proposed that uses quaternions (Euler parameters) to eliminate the problem of singularities [106],[114]. To take advantage of the attractive properties of the quaternion, Simo [108] has formulated a quaternion-based approach that utilizes an exponential map of the angle magnitude coupled with Newmark's numerical integration method to calculate the quaternion angle produced during the timestep. Utilization of the Newmark algorithm is also the basis for several other approaches in aerospace rigid body dynamics integration (e.g., [52] and [61]).

For discrete element modeling (DEM), on the other hand, there are typically no large angle displacements between time steps. The formulation of DEM typically uses fixed timesteps that must be relatively small to accurately resolve inter-body contact. Munjiza [81] introduces the assumptions of small angle displacement and small duration timestep into a 3D spatial orientation integration algorithm. These assumptions allow for computational efficiency to be gained by simplifying the expressions for trigonometric functions. However, matrix multiplication, not quaternions (or Euler parameters), are used in [81]. Though a method of preventing gimbal lock is offered, the operation is less efficient computationally and leads to stable, but incorrect, answers, as shown in this chapter.

is not only less efficient computationally than quaternion operations, but it can also lead to breach of the orthogonality constraint and the "gimbal lock" phenomenon.

This section unites the bodies of work in quaternion multiplication with the assumptions of small angle disturbance by offering a quaternion-based rotation integration algorithm based on the 4th order Runge-Kutta algorithm that assumes small rotation. The algorithm is also compared to the current methods for speed and accuracy as well as to an analytical solution of the simple precession of a prolate spheroid about its major axis.

5.3 Rigid Body Rotation Background

Let the body have a set of coordinates fixed to a particular orientation of that body called the convective frame with the coordinates:

$$\langle\ E_1 \quad E_2 \quad E_3\ \rangle \tag{5.17}$$

Let another frame of reference fixed to a constantly translating point in space be called the inertial frame with the coordinates:

$$\langle\ e_1 \quad e_2 \quad e_3\ \rangle \tag{5.18}$$

As the body rotates and translates with respect to the inertial frame, the convective frame and inertial frame can be related by a transformation matrix, Λ, such that:

$$e = \Lambda E + \varphi \tag{5.19}$$

where φ is the location of the origin of the convective frame relative to the inertial frame. If we take the convective and inertial frame origins to be collocated, $\varphi = \langle\ 0 \quad 0 \quad 0\ \rangle$, and:

$$e = \Lambda E \tag{5.20}$$

The transformation matrix, Λ, is typically referred to as the rotation matrix. It is an orthogonal matrix, such that, $\Lambda^T \Lambda = \hat{1}$. This necessarily implies that $\Lambda^T = \Lambda^{-1}$. The problem of determining the rotation matrix as a function of time, $\Lambda(t)$, is the subject of this chapter.

Angular and linear momentum are conserved in a physical system. This postulate forms the basis for determining the rotation and translation of a body.

Angular momentum is expressed by:

$$\vec{J}(t) = \vec{x}(t) \times \vec{p}(t) + \vec{\pi}(t) \tag{5.21}$$

where $\vec{x}(t)$ is the center of gravity in the inertial frame, $\vec{p}(t)$ is the total linear momentum, and $\vec{\pi}(t)$ is the total spatial angular momentum relative to the center of mass.

If the inertial frame is chosen such that the center of mass is at the origin of the inertial frame, then we can assume that $\vec{p}(t) = 0$, and the equation can be reduced to:

$$\vec{J}(t) = \vec{\pi}(t) \tag{5.22}$$

$$\vec{\pi}(t) = \hat{I}(t)\vec{\omega}(t) \tag{5.23}$$

where $\hat{I}(t)$ is the moment of inertia dyadic [1] in the inertial frame, and $\vec{\omega}(t)$ is the rotational velocity in the inertial frame.

$$\hat{I}(t) = \Lambda(t)\hat{J}\Lambda^T(t) \tag{5.24}$$

where $\hat{J}$ is the moment of inertia dyadic in the convective frame.

If we express the externally applied moments as $\vec{m}$, then the conservation of angular momentum principle can be expressed as:

$$\vec{m} = \frac{d}{dt}\vec{\pi} = \dot{\vec{\pi}} \tag{5.25}$$

Taking the derivative of rotational momentum with respect to time, we have:

$$\dot{\vec{\pi}} = \frac{d}{dt}\left(\hat{I}(t)\vec{\omega}(t)\right) = \hat{I}(t)\dot{\vec{\omega}}(t) + \vec{\omega}(t) \times \hat{I}(t)\vec{\omega}(t) \tag{5.26}$$

[1] The dyadic is a somewhat archaic term; such constructs are typically referred to by the general class to which they belong, the tensor. In the case of a dyadic, it can be represented by a tensor of type $(1, 1)$. The dyadic, in its defense, leaves little room for ambiguity, as it denotes a linear polynomial of dyads consisting of 9 components, that can be represented as a 3x3 matrix.

If we express the externally applied forces as $\vec{f}$, then the conservation of linear momentum can be expressed as:

$$\vec{f} = \frac{d}{dt}\vec{p} = \dot{\vec{p}} \tag{5.27}$$

$$\dot{\vec{p}} = \frac{d}{dt}\left(M\dot{\vec{\varphi}}(t)\right) = M\ddot{\vec{\varphi}}(t) \tag{5.28}$$

From this development, we can see that the angular momentum about the center of inertia can resolve the rotational component of motion about the center of inertia while the linear momentum equation describes the translational motion of the center of inertia.

5.4 Notation

To reduce the verbosity of the mathematics in the rest of the chapter, this section introduces symbols and operators. The convective frame of reference refers to the so-called "body" frame of reference; whereas, the inertial frame is the global frame of reference.

Rotational velocity in the convective frame:

$$\vec{W} = \Lambda^T\vec{\omega}(t) \tag{5.29}$$

Rotational acceleration in the inertial frame:

$$\vec{a}(t) = \frac{d}{dt}\vec{\omega}(t) \tag{5.30}$$

Rotational acceleration in the convective frame:

$$\vec{A}(t) = \Lambda^T\vec{a}(t) \tag{5.31}$$

Vector-to-Skew Symmetric Matrix Operator:

$$\hat{\chi} = \begin{bmatrix} 0 & -\chi_3 & \chi_2 \\ \chi_3 & 0 & -\chi_1 \\ -\chi_2 & \chi_1 & 0 \end{bmatrix} \tag{5.32}$$

Vector-to-Quaternion Operator:

$$E[\vec{\chi}] = \langle\ q_0 \quad \vec{q}\ \rangle \tag{5.33}$$

$$q_0 = cos\left(\frac{||\vec{\chi}||}{2}\right) \tag{5.34}$$

$$\vec{q} = \frac{1}{2}\left(\frac{sin\left(\frac{||\vec{\chi}||}{2}\right)}{\frac{||\vec{\chi}||}{2}}\right)\vec{\chi} \tag{5.35}$$

(if $||\vec{\chi}|| << 1$, then one must employ the Taylor series expansion):

$$\frac{sin(x)}{x} = 1 - \frac{x^2}{6} + \frac{x^4}{120} - \frac{x^6}{5040} + ... + (-1)^N \frac{x^{2N}}{(2N+1)!} \tag{5.36}$$

Note: $\left|\langle\ q_0 \quad \vec{q}\ \rangle\right| = \sqrt{cos\left(\frac{||\vec{\chi}||}{2}\right)^2 + sin\left(\frac{||\vec{\chi}||}{2}\right)^2\left(\frac{\vec{\chi}\cdot\vec{\chi}}{||\vec{\chi}||^2}\right)}$, so this is a unit quaternion.

Quaternion-to-Rotation Matrix Operator:

$$R(\langle\ q_0 \quad \vec{q}\ \rangle) = \Lambda \tag{5.37}$$

$$\Lambda = 2\begin{bmatrix} {q_0}^2 + {q_1}^2 - \frac{1}{2} & q_1q_2 - q_3q_0 & q_1q_3 + q_2q_0 \\ q_1q_2 + q_3q_0 & {q_0}^2 + {q_2}^2 - \frac{1}{2} & q_2q_3 - q_1q_0 \\ q_1q_3 + q_2q_0 & q_2q_3 - q_1q_0 & {q_0}^2 + {q_3}^2 - \frac{1}{2} \end{bmatrix} \tag{5.38}$$

Quaternion Product Operator:

$$\langle\ r_0 \quad \vec{r}\ \rangle = \langle\ q_0 \quad \vec{q}\ \rangle \circ \langle\ p_0 \quad \vec{p}\ \rangle \tag{5.39}$$

$$r_0 = p_0q_0 - \vec{p}\cdot\vec{q} \tag{5.40}$$

$$\vec{r} = p_0\vec{q} + q_0\vec{p} + \vec{p}\times\vec{q} \tag{5.41}$$

5.5 Overview of Current Approaches

The approach first used in DEM [22] and is adopted in other implementations [56] is a second order central difference algorithm of the form:

$$\omega_{t+\frac{1}{2}} = \omega_{t-\frac{1}{2}} + \Delta t \sum \frac{M_i}{I_i} \tag{5.42}$$

$$\theta_{t+1} = \theta_t + \Delta t \omega_{t+\frac{1}{2}} \tag{5.43}$$

Perkins [94] mentions using quaternions for DEM in resolving rotation and provides a definition of quaternions but provides no quadrature algorithm. A quadrature algorithm based on another rotational formula [38] is also used [66]:

$$(\omega_x)_{t+\frac{1}{2}} = (\omega_x)_{t-\frac{1}{2}} + \Delta t \sum \frac{M_{x,i}}{2 \cdot I_{x,i}} \tag{5.44}$$

$$(\omega_y)_{t+\frac{1}{2}} = \frac{1}{\Omega}\left(\xi \cdot I_z + \frac{\zeta I_t \omega_x \Delta t}{2}\right) \tag{5.45}$$

$$(\omega_z)_{t+\frac{1}{2}} = \frac{1}{\Omega}\left(\xi \cdot I_z - \frac{\zeta I_t \omega_x \Delta t}{2}\right) \tag{5.46}$$

where

$$I_t = I_x - I_z, I_z = I_y \tag{5.47}$$

$$\xi = I_z(\omega_y)_{t-\frac{1}{2}} + (I_t(\omega_x)_{t+\frac{1}{2}}\Delta t/2)(\omega_z)_{t-\frac{1}{2}} + M_y \Delta t \tag{5.48}$$

$$\zeta = I_z(\omega_z)_{t-\frac{1}{2}} - (I_t(\omega_x)_{t+\frac{1}{2}}\Delta t/2)(\omega_z)_{t-\frac{1}{2}} + M_z \Delta t \tag{5.49}$$

$$\Omega = {I_z}^2 + (I_t(\omega_x)_{t+\frac{1}{2}}\Delta t/2)^2 \tag{5.50}$$

Unfortunately, none of the aforementioned algorithms has been shown to preserve either energy or momentum. In aerospace applications, the quaternion-based Hamiltonian approach is common for the integration of motion. To illustrate, ALGO_C1[108] is an explicit integration scheme based on the Newmark Method. The algorithm is shown to preserve both the norm of angular momentum and energy.

The approach for the primary iteration is fairly straightforward. A second-order approximation of the rotation due to the rotational acceleration and a first-order approximation of the rotation due to the rotational velocity are calculated. The resultant rotation matrix is calculated along with a value for the rotational momentum, where we let h be the duration of the timestep:

$$\vec{W}_{n+1} = \vec{W}_n + (1-\gamma)h\vec{A}_n \tag{5.51}$$

$$\vec{\Theta} = h\vec{W}_n + \left(\frac{1}{2} - \beta\right) h^2\vec{A}_n \tag{5.52}$$

$$\left\langle \begin{array}{cc} q_0 & \vec{q} \end{array} \right\rangle = E[\vec{\Theta}] \tag{5.53}$$

$$\left\langle \begin{array}{cc} \kappa_0 & \vec{\kappa} \end{array} \right\rangle_{n+1} = \left\langle \begin{array}{cc} \kappa_0 & \vec{\kappa} \end{array} \right\rangle_n \circ \left\langle \begin{array}{cc} q_0 & \vec{q} \end{array} \right\rangle \tag{5.54}$$

$$\Lambda_{n+1} = R\left(\left\langle \begin{array}{cc} \kappa_0 & \vec{\kappa} \end{array} \right\rangle_{n+1} \right) \tag{5.55}$$

$$\vec{\pi}_{n+1} = \Lambda_{n+1}\hat{J}\vec{W}_{n+1} \tag{5.56}$$

Using the calculated value for the rotational momentum and the applied moment, a value for the residual of the estimate can be calculated. If the residual lies beneath a certain threshold, the estimate is used and time is progressed by re-entering the primary iteration. Otherwise, the estimate is refined via a correction step.

$$\vec{R}_{n+1} = h\vec{m}_{n+\alpha} + \vec{\pi}_n - \vec{\pi}_{n+1} \tag{5.57}$$

where,

$$\vec{m}_{n+\alpha} = (1-\alpha)\vec{m}_n + \alpha m_{n+1} \tag{5.58}$$

and

$$\alpha = \left\{ \begin{array}{c} \beta/\gamma, \beta/\gamma \leq 1 \\ 1, otherwise \end{array} \right\} \tag{5.59}$$

if $\left|\vec{R}_{n+1}\right| \leq tol$,

$$\vec{A}_{n+1} = \frac{\vec{W}_{n+1} - \vec{W}_n}{\gamma h} + \left(1 - \frac{1}{\gamma}\right)\vec{A}_n \tag{5.60}$$

and begin a new timestep at n+1 with the "Primary Iteration".

else,

Go on to the "Correction Step"

The correction step uses a tangent matrix to apply a direction to the residual and adjusts the rotation matrix accordingly.

Let the tangent matrix be:

$$\hat{K}_{n+1} = \Lambda_{n+1}\left[\frac{\gamma}{\beta h}\hat{J}T\left(\vec{\Theta}\right)_{matrix}\right] \tag{5.61}$$

where the *matrix* subscript indicates that the parenthetical expression has the ”Vector-to-Skew Symmetric Matrix Operator" applied to it.

To calculate the linearization matrix, $T\left(\vec{\Theta}\right)$, we do the following:

$$T\left(\vec{\Theta}\right) = \frac{\vec{\Theta}}{\left|\vec{\Theta}\right|} \otimes \frac{\vec{\Theta}}{\left|\vec{\Theta}\right|} + \frac{\frac{\left|\vec{\Theta}\right|}{2}}{tan\left(\frac{\left|\vec{\Theta}\right|}{2}\right)}\left[1 - \frac{\vec{\Theta}}{\left|\vec{\Theta}\right|} \otimes \frac{\vec{\Theta}}{\left|\vec{\Theta}\right|}\right] + \vec{\Theta}/2 \tag{5.62}$$

where we use the following Taylor series expansion:

$$\frac{tan(x)}{x} = 1 + \frac{x^2}{3} + \frac{2x^4}{15} + \frac{17x^6}{315} + \frac{62x^8}{2835}\ldots \tag{5.63}$$

The incremented rotation vector is given by:

$$\Delta\vec{\Theta} = \left(\hat{K}_{n+1}\right)^{-1}\vec{R}_{n+1} \tag{5.64}$$

$$\left\langle\ \Delta q_0 \quad \Delta\vec{q}\ \right\rangle = E\left[\Delta\hat{\Theta}\right] \tag{5.65}$$

Now update the values calculated for timestep n+1 as:

$$\left\langle\ \Delta\kappa_0 \quad \Delta\vec{\kappa}\ \right\rangle_{n+1} = \left\langle\ \Delta\kappa_0 \quad \Delta\vec{\kappa}\ \right\rangle_{n+1} \circ \left\langle\ \Delta q_0 \quad \Delta\vec{q}\ \right\rangle \tag{5.66}$$

$$\left\langle\ q_0 \quad \vec{q}\ \right\rangle = \left\langle\ q_0 \quad \vec{q}\ \right\rangle \circ \left\langle\ \Delta q_0 \quad \Delta\vec{q}\ \right\rangle \tag{5.67}$$

$$\vec{\Theta}^{i+1} = E^{-1}\left[\left\langle\ \Delta q_0 \quad \Delta\vec{q}\ \right\rangle\right] \tag{5.68}$$

$$\vec{W}_{n+1} = \vec{W}_{n+1} + \frac{\gamma}{\beta h}\left(\vec{\Theta}^{(i+1)} + \vec{\Theta}^{(i)}\right) \tag{5.69}$$

$$\Lambda_{n+1} = R\left(\left\langle\ \kappa_0 \quad \vec{\kappa}\ \right\rangle\right) \tag{5.70}$$

$$\vec{\pi}_{n+1} = \Lambda_{n+1}\hat{J}\vec{W}_{n+1} \tag{5.71}$$

Finally, begin a new timestep at n+1 with the "Primary Iteration".

It is instructive to observe that the handling of orientation is performed through the operator $\Lambda x(t)\Lambda^T$. Momentum transfer is explicitly handled through the π terms. The similarity of the problem of rotational integration in aerospace applications and in DEM is apparent in the formulation discussed next for use in DEM.

5.5.1 Small Angle Assumption

The small angles explicit integration scheme [81] based on the 4th order Runge-Kutta Method. The method is materially similar in formulation of the physics of the problem as in [108], except here the authors make three additional assumptions:

1. The rotations are small
2. The timestep is small
3. Since the timestep is small, the moment applied during the timestep can be assumed to be an impulse (delta) loading applied at the beginning of the timestep.

If we let h be the duration of the timestep, then:

$$\vec{\Theta} = h\vec{\omega}_n = \left\{ \begin{array}{ccc} \Theta_1 & \Theta_2 & \Theta_3 \end{array} \right\} \tag{5.72}$$

and the incremental rotation matrix is such that:

$$\vec{\imath}_{t+h} = \hat{\Theta}\vec{\imath}_t \tag{5.73}$$

If we take the vector notation in [81] and convert it into a matrix notation, we have:

$$\hat{\Theta} = \tag{5.74}$$

$$\begin{bmatrix} \frac{\Theta_1\Theta_1}{|\Theta|^2} + cos(|\vec{\Theta}|)\left(1 - \frac{\Theta_1^2}{|\vec{\Theta}|^2}\right) & \frac{\Theta_1\Theta_2}{|\Theta|^2}\left(1 - cos(|\vec{\Theta}|)\right) - \frac{\sin(|\vec{\Theta}|)}{|\vec{\Theta}|}\vec{\Theta}_3 & \frac{\Theta_1\Theta_3}{|\Theta|^2}\left(1 - cos(|\vec{\Theta}|)\right) + \frac{\sin(|\vec{\Theta}|)}{|\vec{\Theta}|}\vec{\Theta}_2 \\ \frac{\Theta_2\Theta_1}{|\Theta|^2}\left(1 - cos(|\vec{\Theta}|)\right) + \frac{\sin(|\vec{\Theta}|)}{|\vec{\Theta}|}\vec{\Theta}_3 & \frac{\Theta_2\Theta_2}{|\Theta|^2} + cos(|\vec{\Theta}|)\left(1 - \frac{\Theta_2^2}{|\vec{\Theta}|^2}\right) & \frac{\Theta_2\Theta_3}{|\Theta|^2}\left(1 - cos(|\vec{\Theta}|)\right) - \frac{\sin(|\vec{\Theta}|)}{|\vec{\Theta}|}\vec{\Theta}_1 \\ \frac{\Theta_3\Theta_1}{|\Theta|^2}\left(1 - cos(|\vec{\Theta}|)\right) - \frac{\sin(|\vec{\Theta}|)}{|\vec{\Theta}|}\vec{\Theta}_2 & \frac{\Theta_3\Theta_2}{|\Theta|^2}\left(1 - cos(|\vec{\Theta}|)\right) + \frac{\sin(|\vec{\Theta}|)}{|\vec{\Theta}|}\vec{\Theta}_1 & \frac{\Theta_3\Theta_3}{|\Theta|^2} + cos(|\vec{\Theta}|)\left(1 - \frac{\Theta_3^2}{|\vec{\Theta}|^2}\right) \end{bmatrix}$$

It is interesting to compare this formulation to the commonly accepted incremental rotation matrix formulation extracted from the angle vector as given by [108]:

$$\hat{\Theta} = R\left[E(\vec{\Theta})\right] \tag{5.75}$$

If we go through the exercise of expanding this expression, and we use the more common quaternion to matrix conversion found in [107] rather than that identified in [108] (which is a scale of $\frac{1}{2}$ that of the former), it gives the equivalent matrix representation:

$$\hat{\Theta} = \tag{5.76}$$

$$2\begin{bmatrix} c(\frac{|\vec{\Theta}|}{2})^2 + \left(\frac{s\left(\frac{|\vec{\Theta}|}{2}\right)}{|\vec{\Theta}|}\Theta_1\right)^2 - \frac{1}{2} & \Theta_2\Theta_1\left(\frac{s\left(\frac{|\vec{\Theta}|}{2}\right)}{|\vec{\Theta}|}\right)^2 - \Theta_3\frac{c\left(\frac{|\vec{\Theta}|}{2}\right)s\left(\frac{|\vec{\Theta}|}{2}\right)}{|\vec{\Theta}|} & \Theta_3\Theta_1\left(\frac{s\left(\frac{|\vec{\Theta}|}{2}\right)}{|\vec{\Theta}|}\right)^2 + \Theta_2\frac{c(\frac{|\vec{\Theta}|}{2})s(\frac{|\vec{\Theta}|}{2})}{|\vec{\Theta}|} \\ \Theta_2\Theta_1\left(\frac{s(\frac{|\vec{\Theta}|}{2})}{|\vec{\Theta}|}\right)^2 + \Theta_3\frac{c(\frac{|\vec{\Theta}|}{2})s(\frac{|\vec{\Theta}|}{2})}{|\vec{\Theta}|} & c(\frac{|\vec{\Theta}|}{2})^2 + \left(\frac{s(\frac{|\vec{\Theta}|}{2})}{|\vec{\Theta}|}\Theta_2\right)^2 - \frac{1}{2} & \Theta_3\Theta_2\left(\frac{s(\frac{|\vec{\Theta}|}{2})}{|\vec{\Theta}|}\right)^2 - \Theta_1\frac{c(\frac{|\vec{\Theta}|}{2})s(\frac{|\vec{\Theta}|}{2})}{|\vec{\Theta}|} \\ \Theta_3\Theta_1\left(\frac{s(\frac{|\vec{\Theta}|}{2})}{|\vec{\Theta}|}\right)^2 + \Theta_2\frac{c(\frac{|\vec{\Theta}|}{2})s(\frac{|\vec{\Theta}|}{2})}{|\vec{\Theta}|} & \Theta_3\Theta_2\left(\frac{s(\frac{|\vec{\Theta}|}{2})}{|\vec{\Theta}|}\right)^2 - \Theta_1\frac{c(\frac{|\vec{\Theta}|}{2})s(\frac{|\vec{\Theta}|}{2})}{|\vec{\Theta}|} & c(\frac{|\vec{\Theta}|}{2})^2 + \left(\frac{s(\frac{|\vec{\Theta}|}{2})}{|\vec{\Theta}|}\Theta_3\right)^2 - \frac{1}{2} \end{bmatrix}$$

where $c = cos()$ and $s = sin()$

By using the following (Pythagorean, half-angle, and double-angle) trigonometric identities:

$$cos(\theta)^2 = 1 - sin(\theta)^2 \tag{5.77}$$

$$\frac{sin(\theta)}{2} = sin\left(\frac{\theta}{2}\right)cos\left(\frac{\theta}{2}\right) \tag{5.78}$$

$$sin\left(\frac{\theta}{2}\right)^2 = \frac{1 - cos(\theta)}{2} \tag{5.79}$$

$$cos\left(\frac{\theta}{2}\right)^2 = \frac{1 + cos(\theta)}{2} \tag{5.80}$$

The incremental rotation matrix becomes:

$$\hat{\Theta} = \tag{5.81}$$

$$\begin{bmatrix} \left(\frac{\Theta_1}{|\vec{\Theta}|}\right)^2 + cos(|\vec{\Theta}|)\left(1 - \left(\frac{\Theta_1}{|\vec{\Theta}|}\right)^2\right) & \frac{\Theta_2\Theta_1}{|\vec{\Theta}|^2}(1 - cos(|\vec{\Theta}|)) - \Theta_3\frac{sin(|\vec{\Theta}|)}{|\vec{\Theta}|} & \frac{\Theta_3\Theta_1}{|\vec{\Theta}|^2}(1 - cos(|\vec{\Theta}|)) + \Theta_2\frac{sin(|\vec{\Theta}|)}{|\vec{\Theta}|} \\ \frac{\Theta_2\Theta_1}{|\vec{\Theta}|^2}(1 - cos(|\vec{\Theta}|)) + \Theta_3\frac{sin(|\vec{\Theta}|)}{|\vec{\Theta}|} & \left(\frac{\Theta_2}{|\vec{\Theta}|}\right)^2 + cos(|\vec{\Theta}|)\left(1 - \left(\frac{\Theta_2}{|\vec{\Theta}|}\right)^2\right) & \frac{\Theta_3\Theta_2}{|\vec{\Theta}|^2}(1 - cos(|\vec{\Theta}|)) - \Theta_1\frac{sin(|\vec{\Theta}|)}{|\vec{\Theta}|} \\ \frac{\Theta_3\Theta_1}{|\vec{\Theta}|^2}(1 - cos(|\vec{\Theta}|)) + \Theta_2\frac{sin(|\vec{\Theta}|)}{|\vec{\Theta}|} & \frac{\Theta_3\Theta_2}{|\vec{\Theta}|^2}(1 - cos(|\vec{\Theta}|)) - \Theta_1\frac{sin(|\vec{\Theta}|)}{|\vec{\Theta}|} & \left(\frac{\Theta_3}{|\vec{\Theta}|}\right)^2 + cos(|\vec{\Theta}|)\left(1 - \left(\frac{\Theta_3}{|\vec{\Theta}|}\right)^2\right) \end{bmatrix}$$

Comparing this to 5.74, we note that the quaternion form is identical to that used in this paper.

Unlike the approach in [108], [81] takes the approach of assuming small rotations, approximating the trigonometric functions as:

$$sin(\theta) = \frac{\theta}{\sqrt{1+\theta^2}}, cos(\theta) = \frac{1}{\sqrt{1+\theta^2}} \tag{5.82}$$

This derives from the first term in the power series expansion of the cosine and sine functions combined with the normalization/trigonometric identity 5.77. The normalization step is necessary, in this case, to enforce orthogonality as the integration proceeds. This assumption yields the following rotation matrix:

$$\hat{\Theta} = \tag{5.83}$$

$$\begin{bmatrix} \left(\frac{\Theta_1}{|\vec{\Theta}|}\right)^2 + \frac{1}{\sqrt{1+|\vec{\Theta}|^2}}\left(1 - \left(\frac{\Theta_1}{|\vec{\Theta}|}\right)^2\right) & \frac{\Theta_2\Theta_1}{|\vec{\Theta}|^2}(1 - \frac{1}{\sqrt{1+|\vec{\Theta}|^2}}) - \frac{\Theta_3}{\sqrt{1+|\vec{\Theta}|^2}} & \frac{\Theta_3\Theta_1}{|\vec{\Theta}|^2}(1 - \frac{1}{\sqrt{1+|\vec{\Theta}|^2}}) + \frac{\Theta_2}{\sqrt{1+|\vec{\Theta}|^2}} \\ \frac{\Theta_2\Theta_1}{|\vec{\Theta}|^2}(1 - \frac{1}{\sqrt{1+|\vec{\Theta}|^2}}) + \frac{\Theta_3}{\sqrt{1+|\vec{\Theta}|^2}} & \left(\frac{\Theta_2}{|\vec{\Theta}|}\right)^2 + \frac{1}{\sqrt{1+|\vec{\Theta}|^2}}\left(1 - \left(\frac{\Theta_2}{|\vec{\Theta}|}\right)^2\right) & \frac{\Theta_3\Theta_2}{|\vec{\Theta}|^2}(1 - \frac{1}{\sqrt{1+|\vec{\Theta}|^2}}) - \frac{\Theta_1}{\sqrt{1+|\vec{\Theta}|^2}} \\ \frac{\Theta_3\Theta_1}{|\vec{\Theta}|^2}(1 - \frac{1}{\sqrt{1+|\vec{\Theta}|^2}}) + \frac{\Theta_2}{\sqrt{1+|\vec{\Theta}|^2}} & \frac{\Theta_3\Theta_2}{|\vec{\Theta}|^2}(1 - \frac{1}{\sqrt{1+|\vec{\Theta}|^2}}) - \frac{\Theta_1}{\sqrt{1+|\vec{\Theta}|^2}} & \left(\frac{\Theta_3}{|\vec{\Theta}|}\right)^2 + \frac{1}{\sqrt{1+|\vec{\Theta}|^2}}\left(1 - \left(\frac{\Theta_3}{|\vec{\Theta}|}\right)^2\right) \end{bmatrix}$$

$$\vec{\pi}_{n+1} = \vec{\pi}_n + \vec{m}h \tag{5.84}$$

Because of the assumption that an impulse is instantaneously applied at the beginning of the timestep and that this is applied as an impulse loading implies that:

$$\vec{\pi}_{n+1} = \vec{\pi}_n = \left(\Lambda_{n+1}\hat{J}\Lambda_{n+1}^T\right)\vec{\omega}_{n+1} \tag{5.85}$$

When inverted, this yields the formula for $\vec{\omega}_{n+1}$ as:

$$\vec{\omega}_{n+1} = \left(\Lambda_{n+1}\hat{J}^{-1}\Lambda_{n+1}^T\right)\vec{\pi}_n \tag{5.86}$$

4th Order Runge-Kutta Method:

To integrate the rotation, [81] uses the standard 4th order Runge-Kutta algorithm. For the first approximation, assume that:

$$ {}_{(1)}\vec{\Theta}_{t+h/2} = \frac{h}{2}\vec{\omega}_n \tag{5.87}$$

Evaluating 5.83 with 5.87 yields the first approximation of $\hat{\Theta}$, noted as ${}_{(1)}\hat{\Theta}$, where:

$$ {}_{(1)}\vec{\imath}_{t+h/2} ={}_{(1)} \hat{\Theta}_{t+h/2}\vec{\imath}_t \tag{5.88}$$

The approximation of the angular velocity, $\vec{\omega}$, is updated as ${}_{(2)}\vec{\omega}$:

$$ {}_{(2)}\vec{\omega}_{t+h/2} = \left(\left({}_{(1)}\hat{\Theta}_{t+h/2}\Lambda_t\right)\hat{J}^{-1}\left({}_{(1)}\hat{\Theta}_{t+h/2}\Lambda_t\right)^T\right) \times \left(\Lambda_t\hat{J}\Lambda_t^T\vec{\omega}_t + \vec{m}h\right) \tag{5.89}$$

The second approximation becomes:

$$ {}_{(2)}\vec{\Theta}_{t+h/2} = \frac{h}{2}{}_{(2)}\vec{\omega}_{t+h/2} \tag{5.90}$$

Evaluating 5.83 with 5.90 yields ${}_{(2)}\hat{\Theta}_{t+h/2}$, and the $\vec{\omega}$ is updated as:

$$ {}_{(3)}\vec{\omega}_{t+h/2} = \left(\left({}_{(2)}\hat{\Theta}_{t+h/2}\Lambda_t\right)\hat{J}^{-1}\left({}_{(2)}\hat{\Theta}_{t+h/2}\Lambda_t\right)^T\right) \times \left(\Lambda_t\hat{J}\Lambda_t^T\vec{\omega}_t + \vec{m}h\right) \tag{5.91}$$

The third approximation becomes:

$$ {}_{(3)}\vec{\Theta}_{t+h/2} = h_{(3)}\vec{\omega}_{t+h/2} \tag{5.92}$$

Evaluating 5.83 with 5.92 yields ${}_{(3)}\hat{\Theta}_{t+h/2}$, and the $\vec{\omega}$ is updated as:

$$ {}_{(4)}\vec{\omega}_{t+h/2} = \left(\left({}_{(3)}\hat{\Theta}_{t+h/2}\Lambda_t\right)\hat{J}^{-1}\left({}_{(3)}\hat{\Theta}_{t+h/2}\Lambda_t\right)^T\right) \times \left(\Lambda_t\hat{J}\Lambda_t^T\vec{\omega}_t + \vec{m}h\right) \tag{5.93}$$

We calculate the average angular velocity as:

$$\vec{\omega} = \frac{1}{6}\left({}_{(1)}\vec{\omega} + 2{}_{(2)}\vec{\omega} + 2{}_{(3)}\vec{\omega} + {}_{(4)}\vec{\omega}\right) \tag{5.94}$$

The incremental rotation matrix approximation becomes:

$$\vec{\Theta} = h\vec{\omega} \tag{5.95}$$

This sets the rotation matrix's new orientations as:

$$\Lambda_{t+h} =_{(4)} \hat{\Theta}\Lambda_t \tag{5.96}$$

5.6 Improved Method

Using the renormalization of quaternions in conjunction with a 4th Order RK method yields a more robust method for determining angles. Because we are using quaternions where the renormalization between timesteps is trivial, we are not constrained to defining the cosine and sine approximations as being implicitly normalized. Instead, we can use appropriate Taylor Series expansions of the trigonometric functions (which do not guarantee orthogonality or that the resultant approximation will satisfy 5.77) and then renormalize the quaternion.

First, we assume small angular displacements and give a modified vector-to-quaternion operator to be defined as: $\tilde{E}\left[\vec{\chi}\right] = \frac{1}{|q|}\left\langle\ q_0 \quad \hat{q}\ \right\rangle$, such that

$$q_0 = \frac{1}{2\sqrt{1 + |\chi|^2/4}} \tag{5.97}$$

and

$$\vec{q} = \left(\frac{1}{2\sqrt{1 + |\chi|^2/4}}\right)\vec{\chi} \tag{5.98}$$

where we have assumed that a reasonable approximation (though not the only approximation we could use) for the sine and cosine functions of small angles are:

$$cos\left(\theta\right) = 1 - \frac{\theta^2}{2}, sin\left(\theta\right) = \theta - \frac{\theta^3}{6} \tag{5.99}$$

First, define the angular velocity as a quaternion:

$$\left\langle \; \omega_0 \quad \hat{\bar{\omega}} \; \right\rangle = \tilde{E}\left(\vec{\omega}\right) \tag{5.100}$$

To integrate the rotation, we use the standard 4th order Runge-Kutta algorithm. For the first approximation, assume that:

$$_{(1)}\left\langle \; q_0 \quad \hat{q} \; \right\rangle = \frac{h}{2}{}_{(1)}\left\langle \; \omega_0 \quad \hat{\bar{\omega}} \; \right\rangle \tag{5.101}$$

The approximation of the angular velocity, $\left\langle \; \omega_0 \quad \hat{\bar{\omega}} \; \right\rangle$, is updated as:

$$\begin{gathered} _{(2)}\left\langle \; \omega_0 \quad \hat{\bar{\omega}} \; \right\rangle = \\ \tilde{E}(\left(R\left({}_{(1)}\left\langle \; q_0 \quad \vec{q} \; \right\rangle \circ \left\langle \; r_0 \quad \vec{r} \; \right\rangle\right)\hat{J}^{-1}R\left({}_{(1)}\left\langle \; q_0 \quad \vec{q} \; \right\rangle \circ \left\langle \; r_0 \quad \vec{r} \; \right\rangle\right)^T\right) \times \\ \left(\left(R\left(\left\langle \; r_0 \quad \vec{r} \; \right\rangle\right)\hat{J}R\left(\left\langle \; r_0 \quad \vec{r} \; \right\rangle\right)^T\right)\hat{\omega}_t + \bar{m}h\right)) \end{gathered} \tag{5.102}$$

The second approximation becomes:

$$_{(2)}\left\langle \; q_0 \quad \hat{q} \; \right\rangle = \frac{h}{2}{}_{(2)}\left\langle \; \omega_0 \quad \hat{\bar{\omega}} \; \right\rangle \tag{5.103}$$

The approximation of the angular velocity, $\left\langle \; \omega_0 \quad \hat{\bar{\omega}} \; \right\rangle$, is updated as:

$$\begin{gathered} _{(3)}\left\langle \; \omega_0 \quad \hat{\bar{\omega}} \; \right\rangle = \\ \tilde{E}(\left(R\left({}_{(2)}\left\langle \; q_0 \quad \vec{q} \; \right\rangle \circ \left\langle \; r_0 \quad \vec{r} \; \right\rangle\right)\hat{J}^{-1}R\left({}_{(2)}\left\langle \; q_0 \quad \vec{q} \; \right\rangle \circ \left\langle \; r_0 \quad \vec{r} \; \right\rangle\right)^T\right) \times \\ \left(\left(R\left(\left\langle \; r_0 \quad \vec{r} \; \right\rangle\right)\hat{J}R\left(\left\langle \; r_0 \quad \vec{r} \; \right\rangle\right)^T\right)\hat{\omega}_t + \bar{m}h\right)) \end{gathered} \tag{5.104}$$

The third approximation becomes:

$$ {}_{(3)}\left\langle\ q_0 \quad \hat{q}\ \right\rangle = \frac{h}{2}{}_{(3)}\left\langle\ \omega_0 \quad \hat{\tilde{\omega}}\ \right\rangle \tag{5.105} $$

The approximation of the angular velocity, $\left\langle\ \omega_0 \quad \hat{\tilde{\omega}}\ \right\rangle$, is updated as:

$$ {}_{(4)}\left\langle\ \omega_0 \quad \hat{\tilde{\omega}}\ \right\rangle = $$

$$ \tilde{E}(\left(R\left({}_{(3)}\left\langle\ q_0 \quad \vec{q}\ \right\rangle \circ \left\langle\ r_0 \quad \vec{r}\ \right\rangle\right)\hat{J}^{-1}R\left({}_{(3)}\left\langle\ q_0 \quad \vec{q}\ \right\rangle \circ \left\langle\ r_0 \quad \vec{r}\ \right\rangle\right)^T\right) \times $$

$$ \left(\left(R\left(\left\langle\ r_0 \quad \vec{r}\ \right\rangle\right)\hat{J}R\left(\left\langle\ r_0 \quad \vec{r}\ \right\rangle\right)^T\right)\hat{\omega}_t + \bar{m}h\right)) \tag{5.106} $$

The fourth approximation becomes:

$$ {}_{(4)}\left\langle\ q_0 \quad \hat{q}\ \right\rangle = h_{(4)}\left\langle\ \omega_0 \quad \hat{\tilde{\omega}}\ \right\rangle \tag{5.107} $$

This sets the rotation matrix's new orientations as:

$$ \left\langle\ q_0 \quad \vec{q}\ \right\rangle = \frac{1}{6}\left({}_{(1)}\left\langle\ q_0 \quad \vec{q}\ \right\rangle + 2_{(2)}\left\langle\ q_0 \quad \vec{q}\ \right\rangle + 2_{(3)}\left\langle\ q_0 \quad \vec{q}\ \right\rangle +_{(4)}\left\langle\ q_0 \quad \vec{q}\ \right\rangle\right) \tag{5.108} $$

$$ \left\langle\ r_0 \quad \hat{r}\ \right\rangle_{t+h} = \left\langle\ q_0 \quad \hat{q}\ \right\rangle \circ \left\langle\ r_0 \quad \hat{r}\ \right\rangle_t \tag{5.109} $$

Or, in matrix notation:

$$ \Lambda_{t+h} = R\left(\left\langle\ r_0 \quad \hat{r}\ \right\rangle_{t+h}\right) \tag{5.110} $$

5.6.1 Comparisons

To determine the effectiveness of the proposed algorithm, the algorithm was compared to that found in Munjiza ([81]) with the test problem given in that paper. The results of the test problem are compared using the angular rotation of the local major axis.

The test problem used is as follows:

A body with a convective inertia dyadic of $\hat{J} = \begin{bmatrix} 2 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$ and initial rotational transformation matrix of $\hat{R} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$ is given a rotational velocity in the inertial reference frame of $\hat{w} = \begin{bmatrix} 100 \\ 1 \\ 0 \end{bmatrix}$. Given this problem, the angular rotation of the major body axis, $\hat{x} = \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix}$, in the inertial frame is given by the following equations:

Let the axes in the inertial frame of reference be given by: $\hat{\imath} = \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix}$, $\hat{\jmath} = \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix}$, and $\hat{k} = \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}$,

Let the vector perpendicular to the rotational velocity vector be given by:

$$\psi = \frac{\vec{w}}{||\vec{w}||} \times \hat{k} \tag{5.111}$$

Let the projection of the major body unit vector onto the rotational velocity unit vector be given by:

$$\vec{w}_{origin} = \left(\frac{\vec{w}}{||\vec{w}||} \cdot \hat{\imath} \right) \cdot \frac{\vec{w}}{||\vec{w}||} \tag{5.112}$$

Let the radius of rotation of the axis be given by the distance from the unit vector along the major body axis to the unit vector along the principal inertial axis as:

$$r = \sqrt{1 - \vec{w}_{origin} \cdot \hat{\imath}} \tag{5.113}$$

Then the analytically derived equations of rotational motion can be expressed in

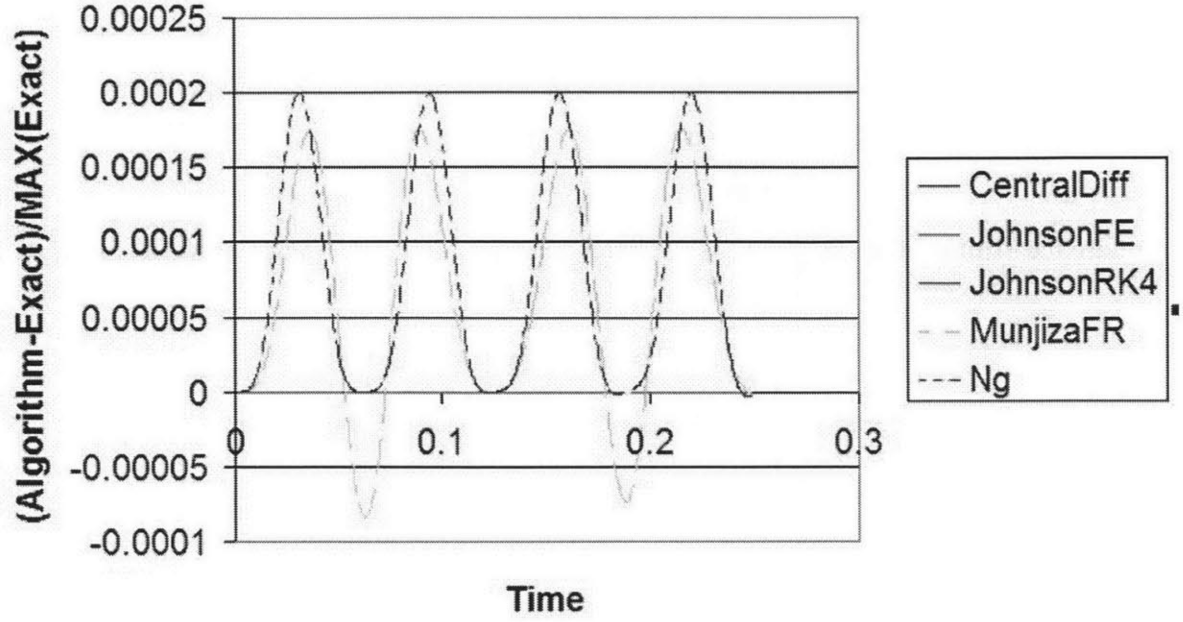


Figure 5-1: Translational position error (Algorithm-Exact)/MAX(Exact) of principal body axis in the inertial frame with h = 0.0005 s along the major x-axis. Note that units of time are seconds.

inertial coordinates as:

$$\vec{x}(t) = \vec{w}_{origin} + r \cdot cos\left(\|w\|\, t\right)\left(\hat{i}\cdot\left(\psi\cdot\hat{i}\right) + \hat{j}\cdot\left(\psi\cdot\hat{j}\right)\right) + r \cdot sin\left(\|w\|\, t\right)\hat{k} \quad (5.114)$$

Comparing the analytical solution to algorithms proposed by Cundall [22] and Ng [65], the quaternion-based approaches exhibit superior performance, and shown in Figure5-1, Figure5-2, and Figure5-3. Comparing the computational time for the 3 algorithms evaluated over 10000 timesteps, as shown in Figure5-4 and Figure5-5, we see that though the quaternion-based algorithms significantly outperform the method of finite rotations and are only moderately more computationally intensive than the method described in [65].

If the finite rotations algorithm [81] is renormalized after each time step, as is suggested in the article, then comparing the analytical solution to the results of the proposed algorithm yields the following for the time step h = 0.005 s:

The abbreviations in the legends above indicate the following:

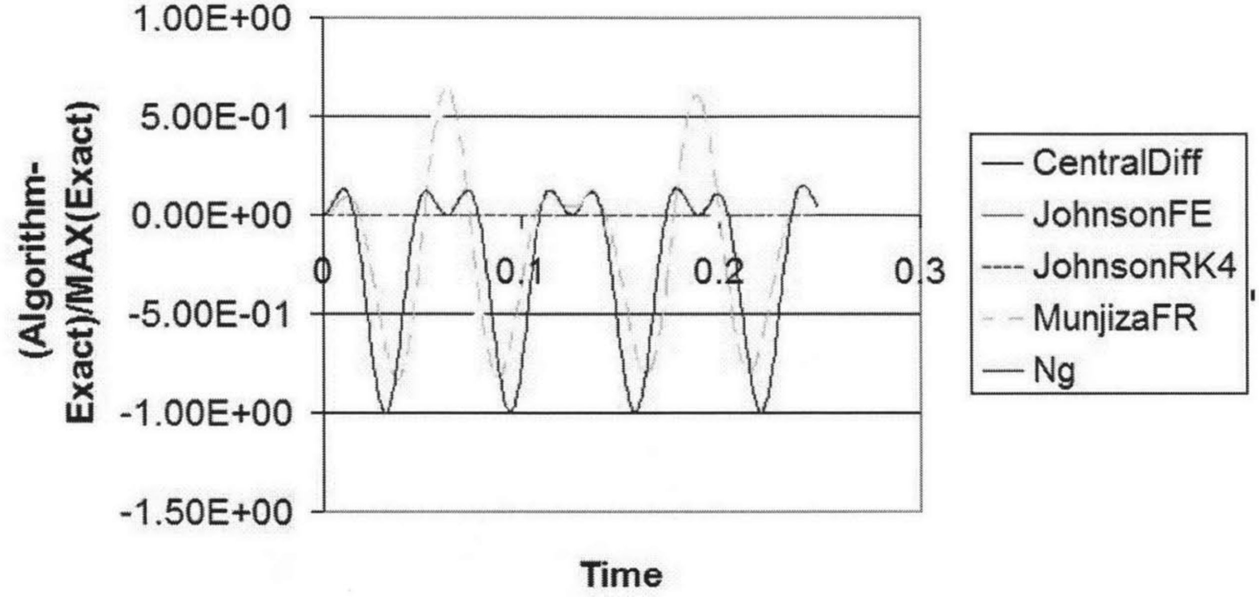


Figure 5-2: Translational position error (Algorithm-Exact)/MAX(Exact) of principal body axis in the inertial frame with h = 0.0005 s along the minor y-axis. Note that units of time are seconds.

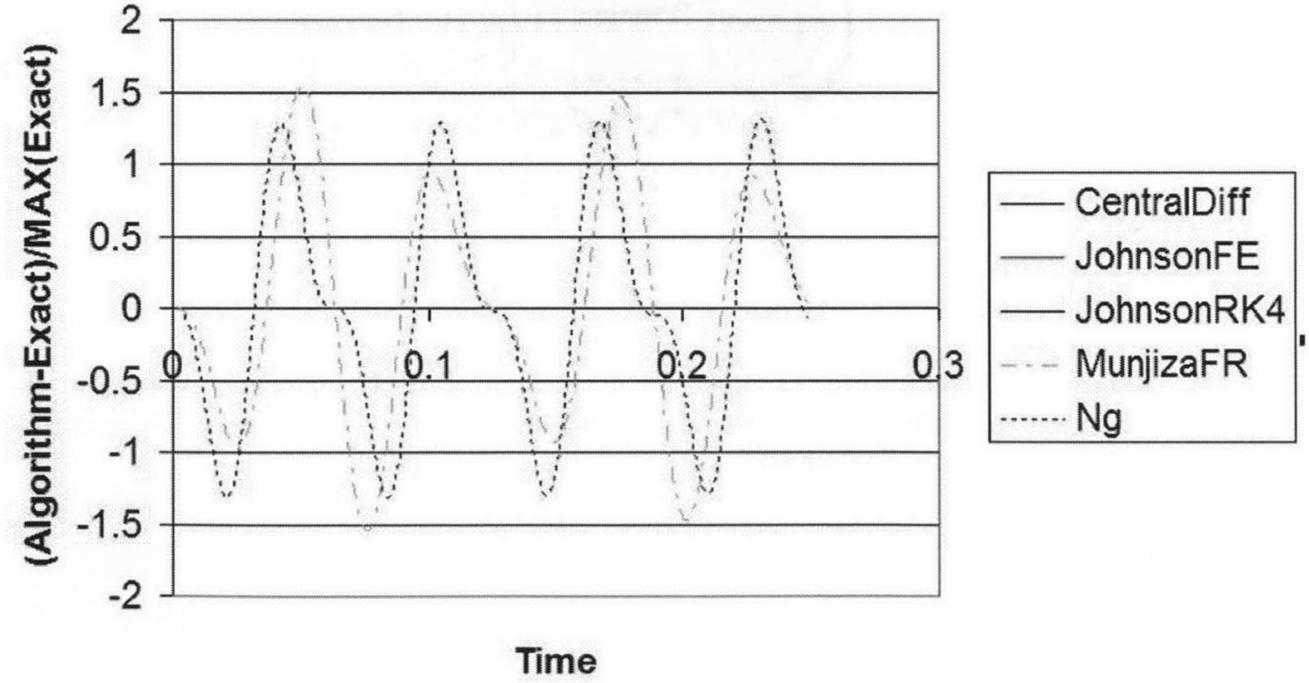


Figure 5-3: Translational position error (Algorithm-Exact)/MAX(Exact) of principal body axis in the inertial frame with h = 0.0005 s along the minor z-axis. Note that units of time are seconds.

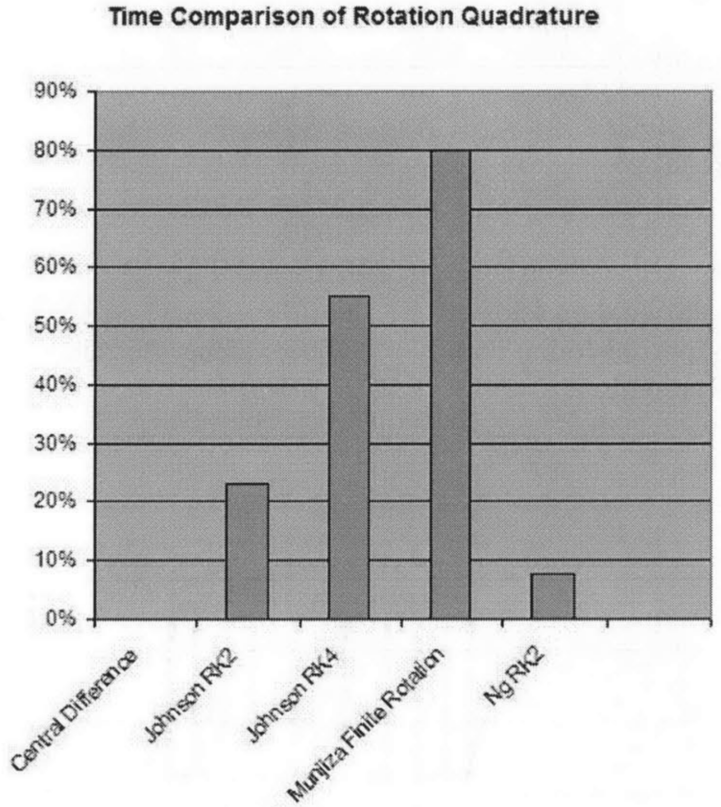


Figure 5-4: Comparison of times for evaluation of 10e4 timesteps. Values in the y-axis indicate the percentage greater evaluation time over the least time-intensive algorithm (Central Difference)

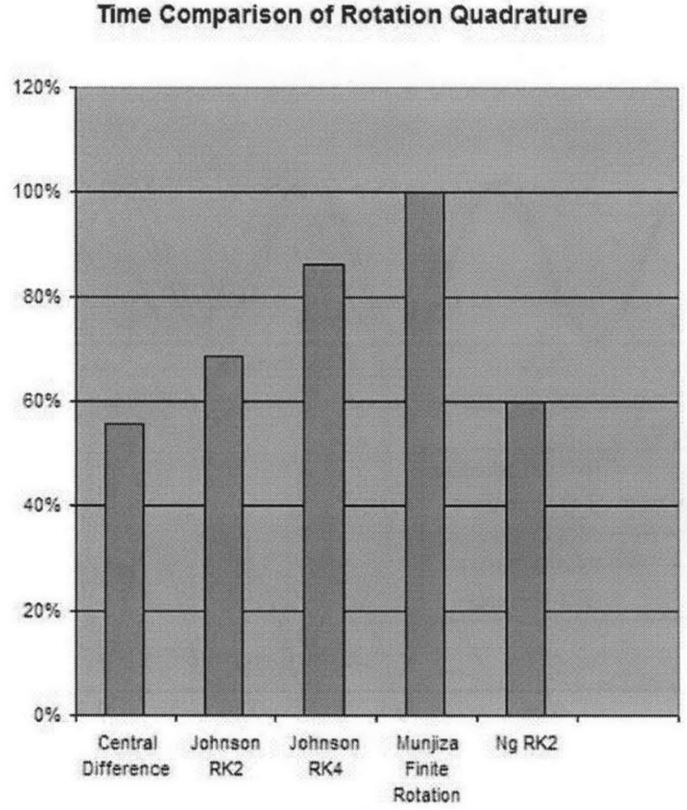


Figure 5-5: Comparison of times for evaluation of 10e4 timesteps normalized according to the most time-intensive algorithm [81]

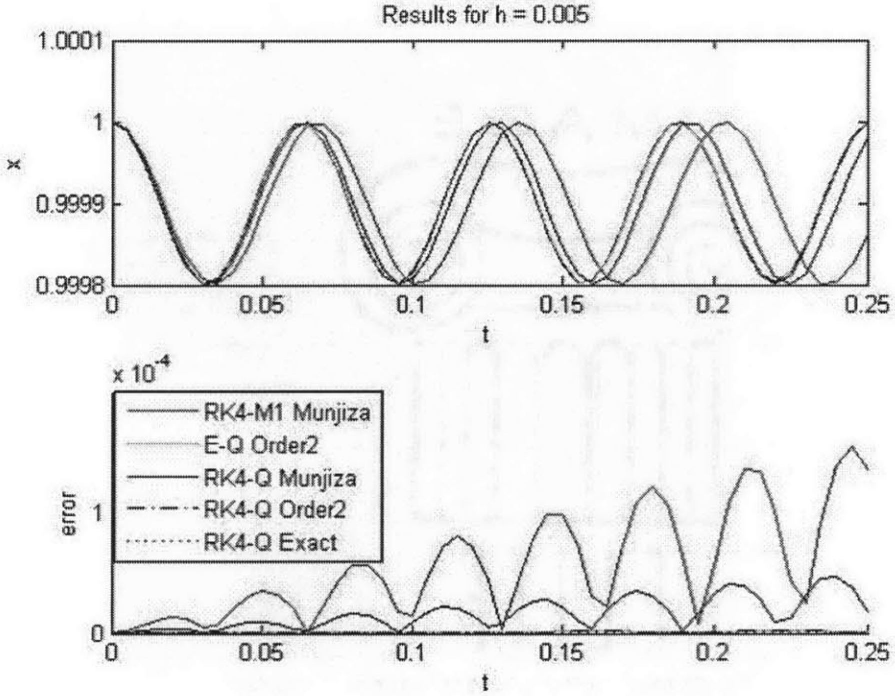


Figure 5-6: Translational position of principal body axis in the inertial frame with h = 0.005 along the principal inertial axis (x-axis).

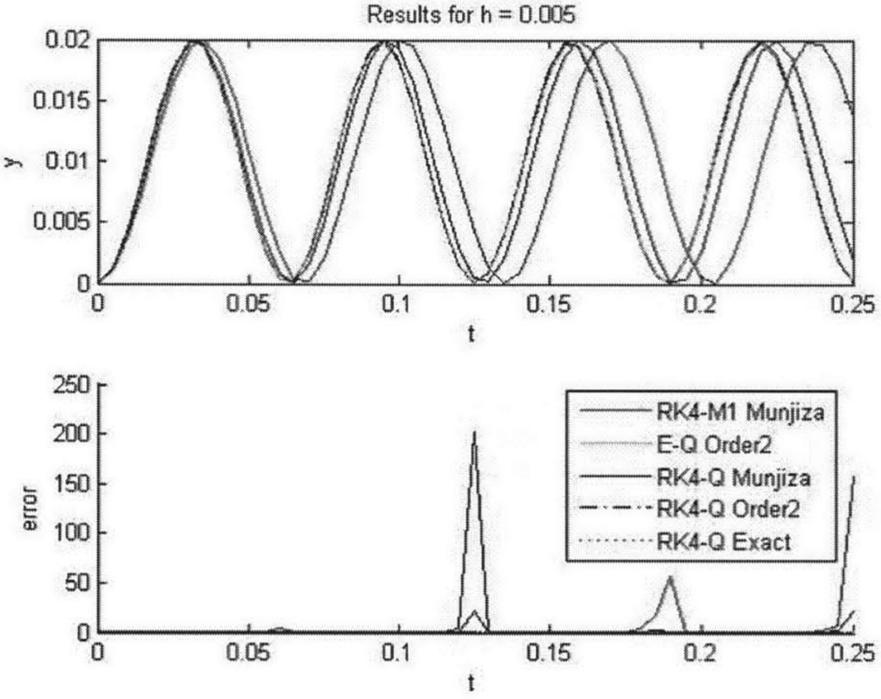


Figure 5-7: Translational position of principal body axis in the inertial frame with h = 0.005 along the first minor inertial axis (y-axis).

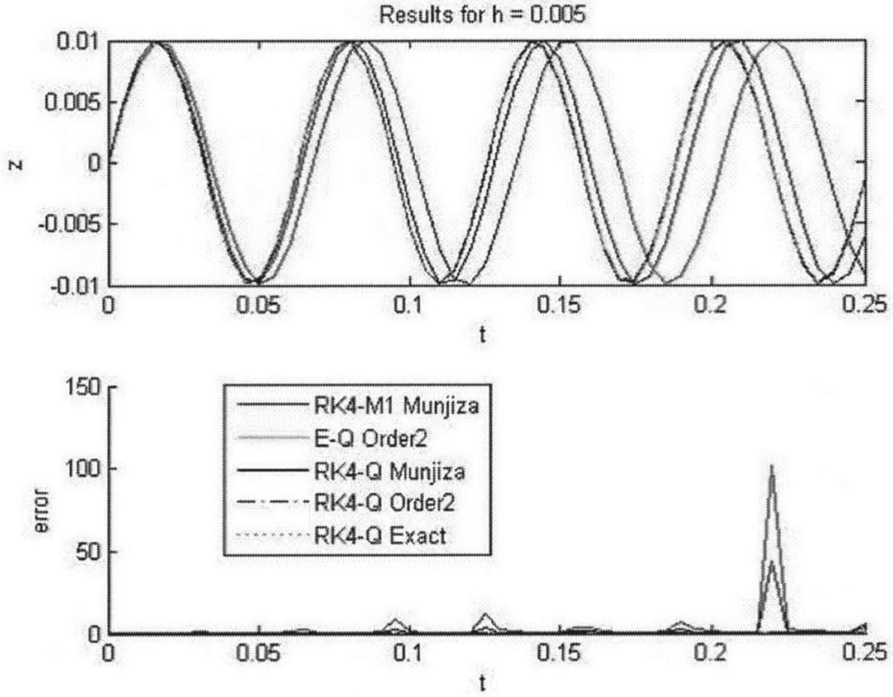


Figure 5-8: Translational position of principal body axis in the inertial frame with h = 0.005 along the second minor inertial axis (z-axis).

- RK4-M1 Munjiza - 4th Order Runge-Kutta numerical integration algorithm using matrix multiplication for updates of the rotation transformation matrix and using the Munjiza approximations for the sine and cosine of small angles.

- E-Q Order2 - Euler (1st order) numerical integration algorithm using quaternion multiplication for updates of the rotation transformation matrix (through the underlying rotation quaternion) and using the first two terms of the power series expansion as approximations for the sine and cosine of small angles.

- RK4-Q Munjiza - 4th Order Runge-Kutta numerical integration algorithm using quaternion multiplication for updates of the rotation transformation matrix (through the underlying rotation quaternion) and using the Munjiza approximations for the sine and cosine of small angles.

- RK4-Q Order2 - 4th Order Runge-Kutta numerical integration algorithm using quaternion multiplication for updates of the rotation transformation matrix (through the underlying rotation quaternion) and using the first two terms of the power series expansion as approximations for the sine and cosine of small angles.

- RK4-Q Exact - 4th Order Runge-Kutta numerical integration algorithm using quaternion multiplication for updates of the rotation transformation matrix (through the underlying rotation quaternion) and using the FDLIBM algorithm (a 13-degree polynomial) as implemented in Matlab to determine sine and cosine.

The results above show four interesting results.

The first is that the use of quaternions coupled with a Runge-Kutta algorithm outperforms a similar implementation using matrix multiplication. Moreover, because there are fewer operations involved in multiplying quaternions (28 floating point operations) versus multiplying dyadics (45 floating point operations), the computational efficiency is also greater. Quaternions implicitly guarantee orthogonality and reduce numerical errors through ease of periodic renormalization and reduced operations per time step.

The second is that approximating the cosine and sine through the first two terms of their power series expansion significantly reduces the error of the algorithm over the approximation introduced in [81]. Moreover, using the two-term approximation is significantly more computationally efficient than the Munjiza approximation (4 flops versus 2 flops and 1 square root evaluation).

The third is that by using an Euler-based first order integration algorithm, it was shown that the increased accuracy of the quaternion based algorithm coupled with a two-term approximation of the sine and cosine compensates for the low order of the numerical algorithm. This algorithm is shown to yield comparable accuracy to [81] algorithm.

Finally, the timing benchmark for the algorithm shows relatively attractive behavior for the quaternion-based algorithms versus the finite rotations algorithm. Compared with the finite difference method of Ng [65], the quaternion-based method yields a significantly more accurate result with only a moderate relative increase in the computational effort required.

5.7 Contributions

The preceding chapter has demonstrated a new quadrature algorithm based on quaternion arithmetic for reducing the errors inherent in multiplying rotation dyadics. The method is shown to not only more accurately capture the rotational motion of a rigid body but to also do so with fewer computational cycles than a common algorithm in DEM. Results to determine accuracy were performed against the exact solution to a test problem.

Chapter 6

Application: Granular Heap Base Stress

The problem of determining the stress distribution at the base of a static heap has been a well-studied phenomenon. Since at least the 18th Century, the emergence of a local stress minimum at the middle of granular heaps had been noted but was first studied formally with the work of Hummel and Finnan [41]. Until the experiments of Novosad, the phenomenon had been a controversial topic of granular materials researchers, who had been divided on whether the local stress minimum at the middle of the heap was caused by arching mechanisms resultant from deflection of the center of the base plate on which the pile rested or whether it was a fundamental property of the granular material that was independent of the boundary conditions. The experiments of Smid and Novosad [110] used a stiff base plate with embedded stress sensors to show conclusively that the stress dip seen in a granular heap often cannot be accounted for solely by the deflection of the base plate. Several experiments, such as those of Vanel [120], have since verified the initial findings of Smid and Novosad.

Researchers have attempted (e.g., Cates [13] and Vanel [120] as shown in Figure 6), through a variety of techniques (including analytical, experimental, and numerical), to uncover the reasons for the stress distribution beneath heaps. The occurrence of a local stress minimum at the middle of a heap is dependent on several factors. Some of the observations the have endured the scrutiny of experimentation indicate that

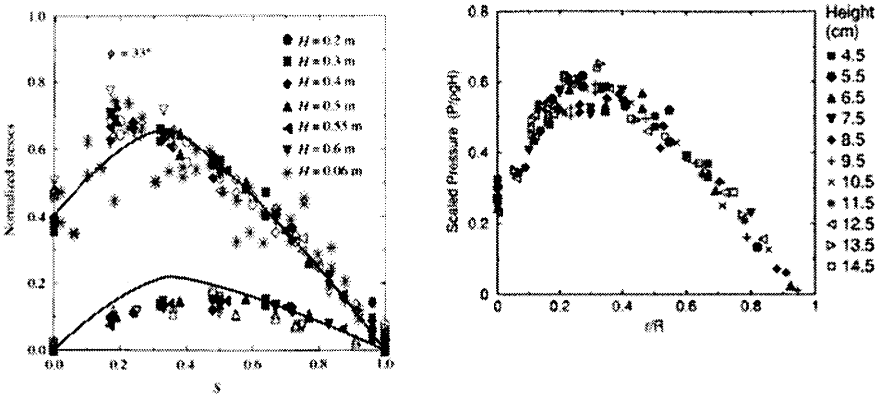


Figure 6-1: Graphs of non-dimensional stress (vertical) and normalized radius (horizontal) from researchers Cates [13](left) and Vanel [120](right).

the formation of a local stress minimum and the relative magnitude of this minimum depends on the asphericity of the constituent particles [69, 68]; the poly-dispersity [64]; and the technique used to create the pile [120, 69], which is generally divided between a distributed source (aka "raining method") or a point source.

Explanations for this phenomenon have come from several studies. Cates [13] uses an analytical model based on local rules of stress propagation with the assumptions that (1) force chains within the granular material support the geometric structure and (2) this network is "fragile" based on a metric that they define. Mattutis [68] alludes to a possible mechanism for a flattened stress distribution that emerges from the high impact velocity and re-ordering of contacts near the impact vicinity. Vanel [120] concludes that preferential grain orientations when a heap is built from a point are locked in at construction time and lead to the local stress minimum (presumably from an arching effect). Interestingly, this conclusion is presented along with data suggesting that the local stress minimum magnitude is directly proportional to the height of the point source, which is counter-intuitive, since height increases the average energy of the particles leading to greater dispersion of the particles upon striking the base or the pile and, accordingly, increased rearrangement of orientation. An explanation for the mechanism behind this seeming paradox will be proposed later in this section.

From the perspective of analyzing stress chains, there has also been a large body of work performed. Ng performed numerical investigations of fabric stresses in uniaxial compression of ellipsoids of different aspect ratios [85] as well as fabric stresses for ellipsoids of two sizes in triaxial drained and undrained tests [86]. In 2-D, studies have been performed to specifically investigate the role of particle geometry (polygonal geometry) in the stress distribution beneath granular heaps [69] with the conclusion that geometry (angularity) does not significantly influence the form of the stress distribution at the base.

The 3-D studies detailed in this chapter have been performed using the DEM3D framework, the development of which has been detailed in previous chapters. All trials have been performed in serial on a Pentium 4 3.0GHz Xeon (dual-threaded) processor with 1GB memory. The numerical experiments are set up as a deposition of particles from a point source at a height of 10 particle diameters (12 particle diameters for the higher deposition trials) onto a plate of fixed mono-disperse spheres configured in a regular grid pattern. Each simulation is run for 4.5e5 timesteps (approximately 90 seconds of real time) and allowed to settle.

The particle properties are held constant across trials and are given in Table 6.1. The contact model used is a simple linear spring-dashpot model. As noted by Mattutis [69], the choice of contact model (linear or Hertzian) and friction coefficient (tested by Mattutis for coefficients of friction of 0, 0.3, and 0.6, respectively) have negligible effect on the emergence of a local minimum of the base stress at the center of the heap. It should be noted that this same statement does not hold for the determination of the angle of repose, where friction is a first order effect, as noted by Zhou [133]. The parameters in Table 6.1 have been selected arbitrarily here for computational expediency and do not reflect properties for real materials. All simulations for granular heaps have been conducted on "dry" materials, or materials with no cohesive potential.

Stiffness (linear)	1e4
Density (constant)	1e-1
Cohesion	0
Friction Coefficient	0.3
Contact Damping	0.3

Table 6.1: Material Properties for 3-D Numerical Experiments Analyzing the Occurrence of Local Stress Minima at the Center of Granular Heaps

6.1 Effects of Source Height

As the relative height of the particle source above the free surface of the heap, it has been observed by several experiments that the magnitude of the stress dip at the base of the heap also increases.

To determine whether the method could reproduce this phenomenon, a numerical system was designed to deposit spherical particles at a height of 10 particle diameters above the base. Particles were created using a uniform random distribution of radii in the range $[0.5, 0.6]$. Approximately $1e3$ particles are deposited on a base plate constituted of fixed spherical elements for a model time of 90 seconds. The source is then removed, and the heap is then allowed to reach steady state (6 seconds model time). To vary the source height, a second set of trials is conducted with the same setup as just discussed except with the height of the deposition source at 12 particle diameters. In both sets of trials, the resultant number of unconstrained particles is $1.2e3$.

The normalized stress results of the trials are derived using a series of operations on the database used to store the data from the simulations. The forces on the base are first binned along the radial direction from the center of the heap, summed, and then divided by the area of the containing annulus of the base to determine the resultant stress state. These values are normalized by the density, gravity, and heap height ($stress_{norm} = \frac{stress}{\rho g H}$).

For the lower point source, the results are charted in Figure 6-2. The normalized stress has a local normalized stress minimum at the center of the heap of 48.85, which extends for a length of 15.5 radially to a maximum normalized stress of 77.08.

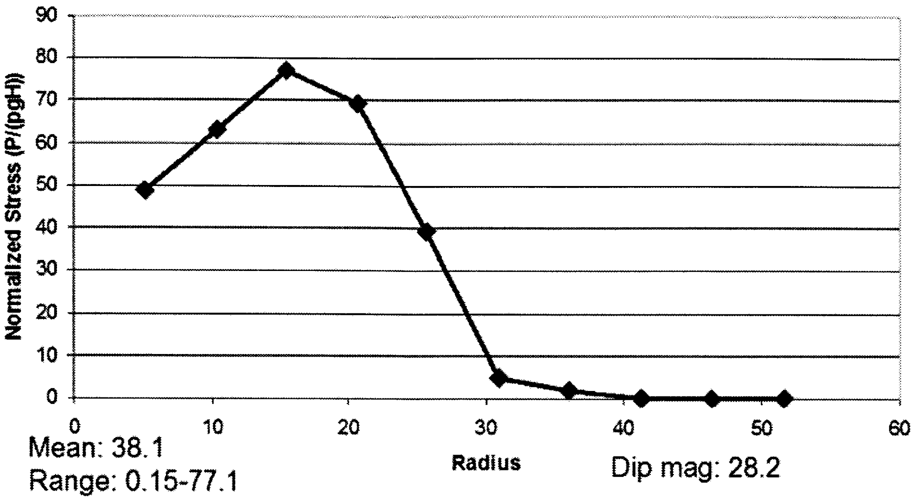


Figure 6-2: Chart of the stress in the annulus of the base with the inner boundary lying at the radial position in the abscissa for a poly-disperse spherical body source at a height of 10 particle diameters.

Using the end of the monotonically increasing curve as the extent of the local stress minimum shows a magnitude of the stress dip as 28.23 in normalized stress units.

For the 20% higher point source, the results are charted in Figure 6-3. The normalized stress has a local normalized stress minimum at the center of the heap of 52.1, which extends for a length of 10.3 radially to a maximum normalized stress of 84.44. Using the end of the monotonically increasing curve as the extent of the local stress minimum shows a magnitude of the stress dip as 32.34 in normalized stress units.

It is worth noting that, as predicted by other experiments, the stress dip magnitude is greater for the increased height point source. That is, the stress dip in the higher source is 14.6% greater than that in the lower source.

Another interesting result that can be derived from the data is the profile of the average age of the contacts as radial distance from the center of the heap is increased. In this thesis, the average contact age is named the *contact residence*, as it is a metric for the percentage of the total time that a particular contact pair stays in contact. Binning the contacts according to radial distance from the center of the heap, it can be seen from Figure 6-5 that the profile of the average residence time for

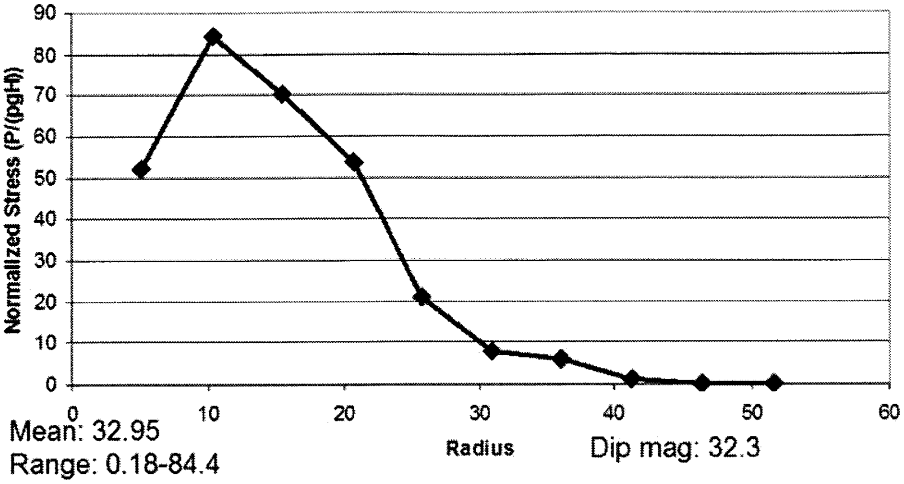


Figure 6-3: Chart of the stress in the annulus of the base with the inner boundary lying at the radial position in the abscissa for a poly-disperse spherical body source at a height of 12 particle diameters.

contacts increases approximately linearly with radial distance from 52.25 (11.6%) to 394 (87.6%) out of a total of 450 samples for the higher source. The profile of the average residence time for the lower source is shown in Figure 6-4, where the average residence time for contacts increases approximately linearly with radial distance, as well, but only increases from 75 (16.7%) to 400 (88.9%) out of a total of 450 samples.

These results suggest that the contacts in the center of the heap are younger for the heap formed from the higher source than for that formed from the lower source. Charting the contacts and coloring them according to the normalized residence (increasing residence with increasing color wavelength) , Figure 6.1 illustrates that a region of younger contacts indeed forms in the center of the heap, eliciting a correlative relationship between the age of the contacts and the formation of a local stress minimum at the center of a heap.

The age of the contacts along the "crater" edge and beyond are older (87.6%-88.9% of the simulation duration) on average and correlate well with the locations of the stability points seen in Figure 6.1. At the edge of the heap are the oldest average contacts, corresponding to the initial impact curtain. The correlative relationship between contact age and formation of a local stress minimum will be revisited

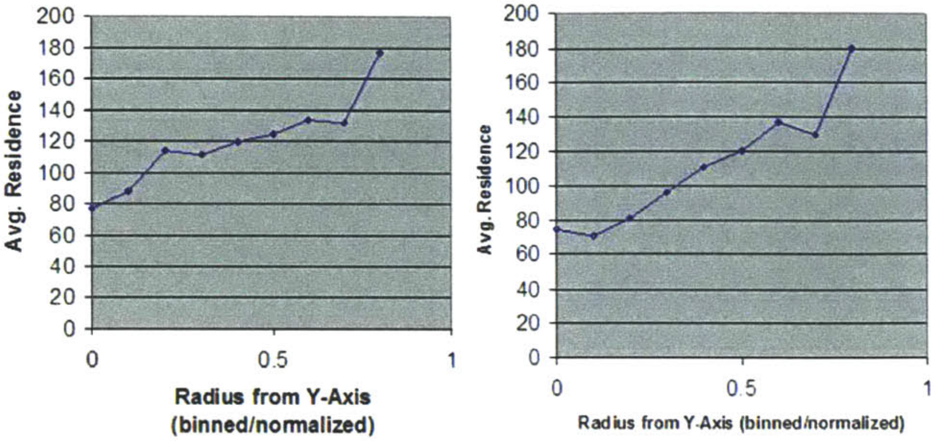


Figure 6-4: Chart of the residence (average active contact pair age) taken radially across the base for a poly-disperse spherical body source at a height of 10 particle diameters.

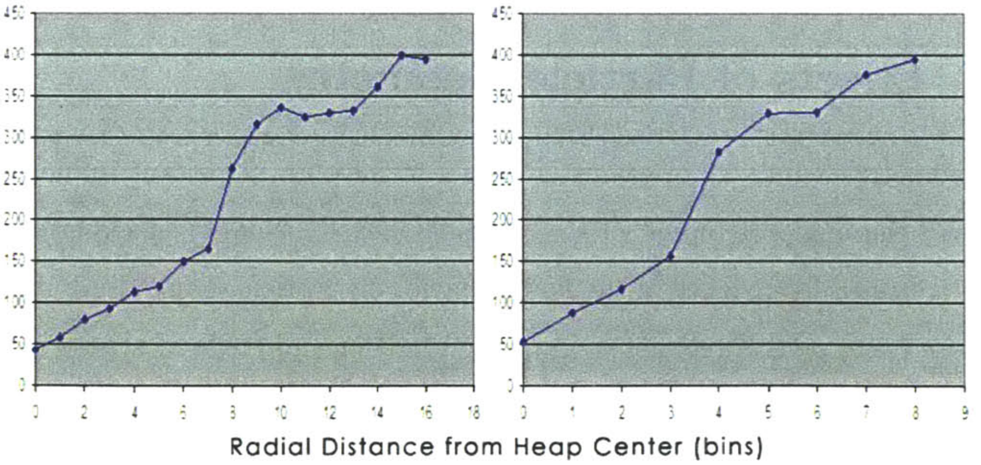


Figure 6-5: Chart of the residence (average active contact pair age) taken radially across the base for a poly-disperse spherical body source at a height of 12 particle diameters.

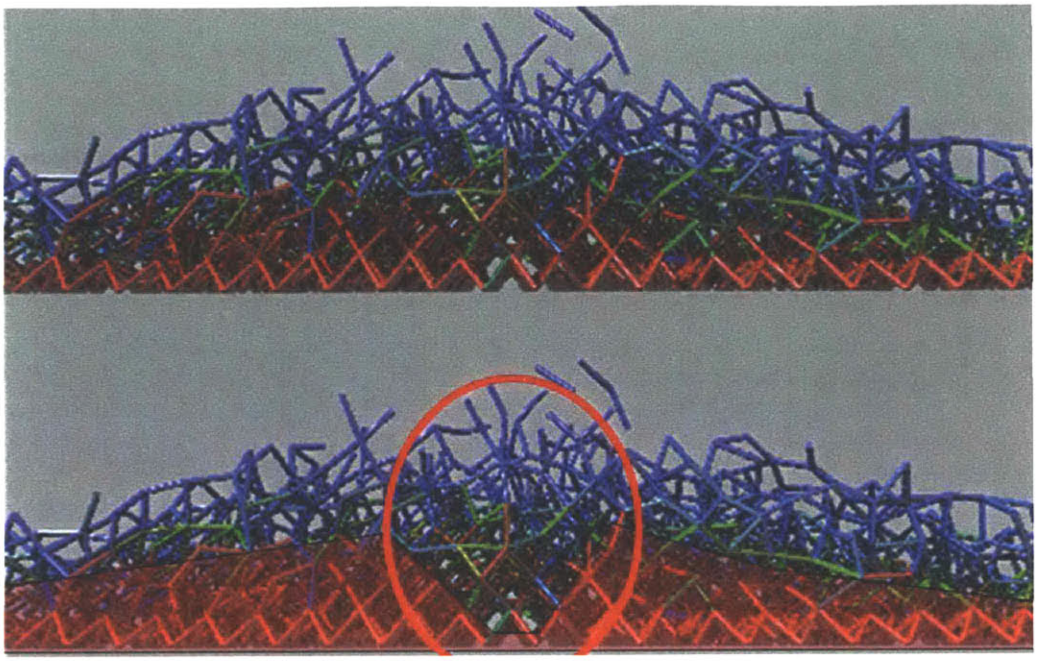


Figure 6-6: Illustration of fabric damage observed during the initial phase of heap creation during deposition of poly-disperse, spherical discrete elements from a localized source at a fixed height.

throughout the chapter, as it is exhibited in all of the studies.

6.2 Effects of Particle Geometry

The geometry of individual grains is known to have an important influence on, for instance, the angle of repose of dry granular heaps. Numerical experiments have shown a strong dependence of the flow behavior on particle shape. In dynamic angle of repose tests, several regimes of granular behavior and angle of repose in excess of 31-degrees are shown to require non-spherical primitives [124].

From previous experiments by the author, as shown in Figure 6-8 and Figure 6-10, morphology has a pronounced effect on the behavior of a granular heap created by deposition in a silo then later removal of the silo sides. In Figure 6-9, there is emergent organization of the granular material when allowed to slump after removal of the containment walls constraining the pile. Because of the low energy of the individual grains in the pile as well as the intrinsic tendency for organization of mono-disperse

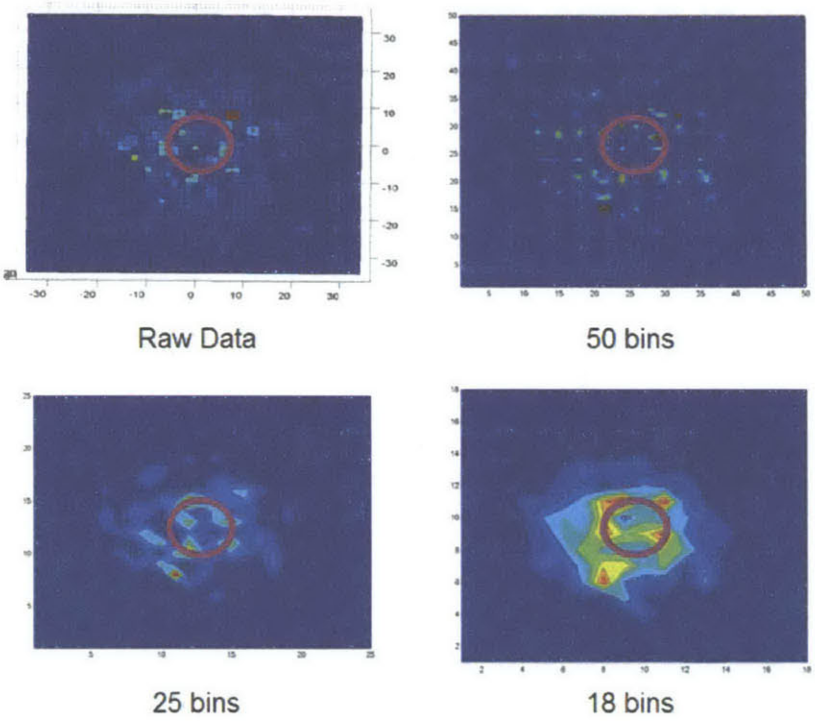


Figure 6-7: Illustration of base stress; bin numbers indicate the resolution of the homogenization technique used to convert point contacts to effective base stress.

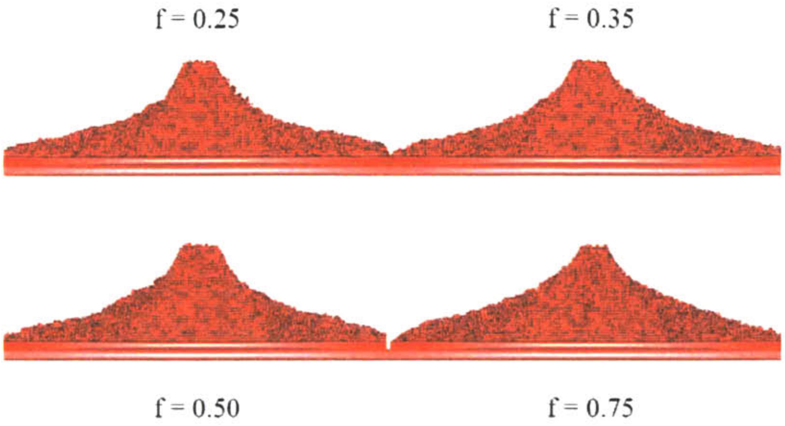


Figure 6-8: Trials for angle of repose of 2-D disk discrete elements under a friction coefficient (from left to right, top to bottom): 0.1, 0.25, 0.35, 0.5.

disks into a highly stable packing configuration (hexagonal), stable structures occur in the middle of the heap. This organization is also noted in work by Mattutis [69] where the packing is essentially ordered at initialization and allowed to sag under its own weight.

Under the same numerical experimentation conditions, a granular material consisting of angular, mono-dispense 2-D ellipses with an aspect ratio of 1.5 will not exhibit these stable structures. In fact, very little self-organization is seen in these marginally aspherical systems, as can be seen in Figure 6-10.

In this study, geometry is introduced through the use of prolate spheroid primitives to represent particles. The prolate spheroids are given a constant aspect ratio of 1.2. The major axis length is a uniform random distribution of values in the range [1, 1.2]. The deposition source is the same as that for the poly-disperse trials discussed in the previous section.

The average stress distribution as the point of observation is moved radially outward from the center of the heap displays a homogeneous normalized stress distribution (33.6) up to a radial distance of 15.5 with a monotonic stress decrease as radial distance increases. That is, the stress distribution displays a flattened stress distribution at the center instead of a local minimum. The resultant stress distribution is shown in Figure 6-11. Contrasting this with the stress distribution for the case of

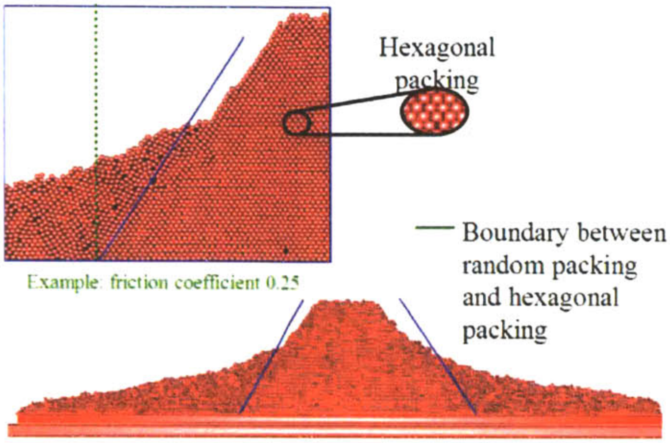

Hexagonal packing
Boundary between random packing and hexagonal packing
Example: friction coefficient 0.25

Figure 6-9: Detail of organization of 2-D disk discrete elements deposited using a raining technique into a silo with the walls removed after deposition to allow slump of the pile.

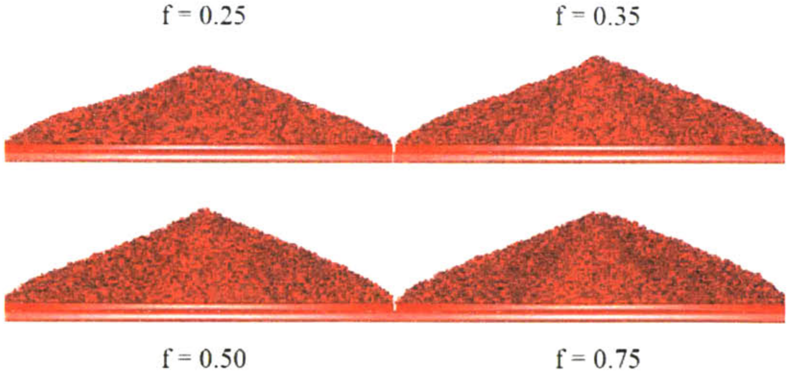


Figure 6-10: Trials for angle of repose of 2-D elliptical discrete elements with a constant aspect ratio of 1.5 under a friction coefficient (from left to right, top to bottom): 0.1, 0.25, 0.35, 0.5.

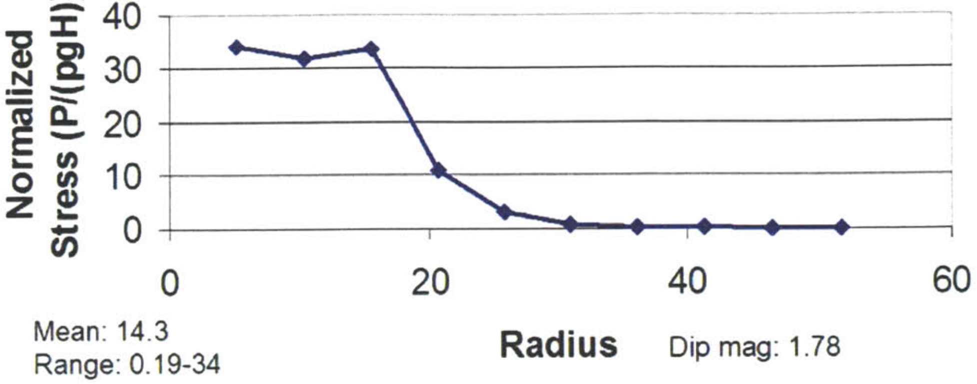


Figure 6-11: Stress at the base of a granular heap as radial distance from the center is increased.

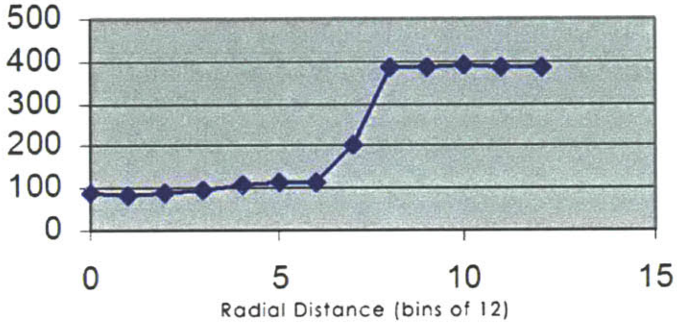


Figure 6-12: Average residence time (of 450 samples) as radial distance from the center is increased.

spherical bodies, it can be observed that the stress dip decreases in magnitude with asphericity from 28.2 for poly-disperse spherical bodies to no dip for poly-disperse prolate spheroidal bodies.

As shown in Figure 6-12, the average residence is homogeneous across the center of the pile at an average age of 100 (22.2%) up to a radius of 13.75, correlating well with the radial distance along which the base stress is homogeneous.

6.3 Effects of Poly-Dispersity

It is also interesting to investigate the effect of particle size distribution on the stress distribution at the base. A similar numerical procedure to that discussed in previous

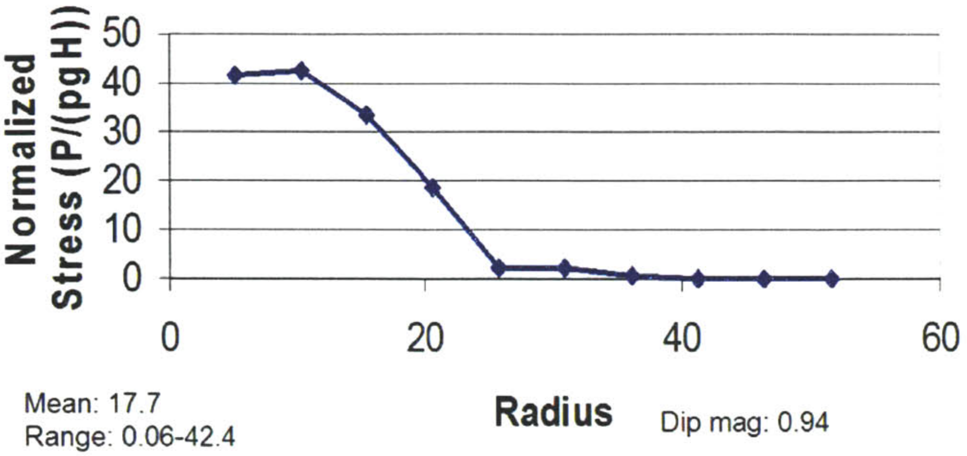


Figure 6-13: Stress at the base of a granular heap as radial distance from the center is increased.

sections is adopted such that the only variable is the size distribution. Approximately $1e3$ prolate spheroids of a constant aspect ratio of 1.2 and a uniform major axis length of 1.2 are deposited from an aggregate point source (i.e., 5 point sources clustered together) at a height of 10 particle diameters onto a rough base plate consisting of fixed spheres.

The average stress distribution as the point of observation is moved radially outward from the center of the heap displays a monotonically decreasing stress distribution with a maximum normalized stress (41.5) at the center. That is, the stress distribution is parabolic as would be expected The resultant stress distribution is shown in Figure 6-13. This can be contrasted with the case of poly-dispersity where the stress distribution at the center is flat for a radial distance of 15.5.

As shown in Figure 6-14, the average residence is homogeneous across the center of the pile at an average age of 92.5 (20.6%) up to a radius of 18.5 (contrasted with an average age of 100 up to a radius of 15.5 in the poly-disperse case), indicating a relatively homogeneous contact age across the cross-section of the heap. That is, the residence implies the same contact age as a "raining" deposition pattern, and the effect of the deposition method is irrelevent. Again, we see a correlation between contact age and stress homogeneity.

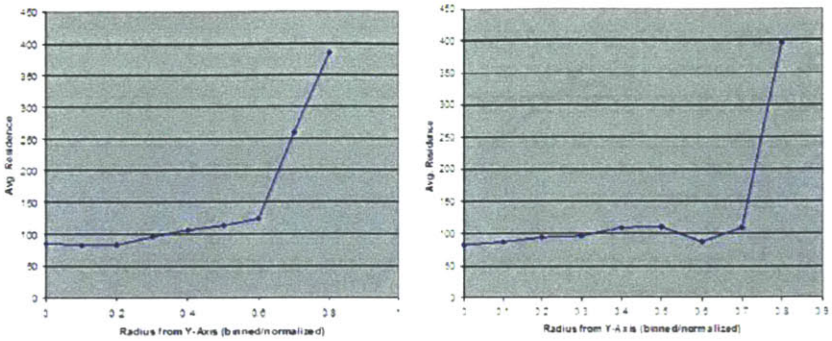


Figure 6-14: Average residence time (of 450 samples) as radial distance from the center is increased.

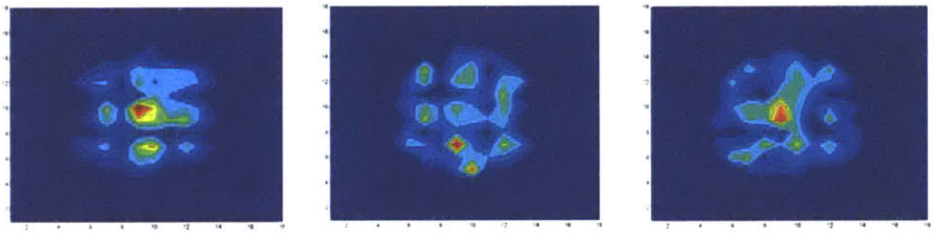


Figure 6-15: The stress at the base of a heap composed of mono-disperse prolate spheroids of aspect ratio 1.2 and deposited from a point source located above the origin onto a base of fixed spheres configured in a regular grid pattern.

From Figure 6-15 it can be seen from inspection that the base stress distribution is also qualitatively different from that observed in poly-disperse sphere systems. The distribution of stress, for instance, is not focused into 2-4 major support chains as seen in the case of poly-disperse spheres.

As a contribution over previous studies, this work suggests a mechanism for the lack of formation of a local stress minimum, as shown through the contact ages, which remain homogeneous across a range of radial distances from the heap center. This suggests that the contacts remain relatively stable (that is, a contact pair remains in contact longer) even near the point of impact of the deposited particles, and that energy dissipation through non-frictional mechanisms (e.g., contact damping) is an important mechanism in the formation or lack of formation of a local stress minimum. This further suggests that impact and energy transmission between high energy particles and stationary (or low-energy) particles drives the formation of the local stress minimum at the center by damaging the stress fabric and preventing the formation of stable arches in the center of the heap. Instead, stable arches are formed from bases at the edge of the "crater" region, as suggested by the contour plots of base stress.

6.4 Contributions

Numerical studies have been performed, which reproduce the stress distribution behavior of granular heaps deposited by a point source. A series of discrete element analyses have been performed on point source deposited systems of poly-disperse spheres, poly-disperse prolate spheroids, and mono-disperse prolate spheroids. The results for the trials are in agreement with analytical and experimental work [120, 69] in predicting the emergence of or lack of a local stress minimum at the center of the heap.

A correlative relationship has been demonstrated between the contact residence profile and the stress distribution beneath a granular heap. The hypothesis that this implies is that damage to the stress fabric occurs during deposition into the center of the pile, preventing stable configurations from forming in center. Geometric connectivity therefore becomes more stable outside of the "crater" region. Particles that become trapped in the "crater" fill this region and become stably supported by the older structures that lie radially distant from the heap center. That is, an effective arching mechanism is created via the older "crater" region.

Interestingly, this theory offers an explanation for the observation that the magnitude of the local stress minimum is proportional with the height of the point source used to deposit granular material [120], as was reproduced in the first section of this chapter. As the average kinetic energy of the particles impacting the heap increases, the rearrangement of the particles increases and the stability of the heap is moved more radially distant from the point of impact. The stability of the heap is enabled through localized support areas, where large relative magnitude stress chains appear and remain resident throughout the simulation. These supports provide the arching behavior which emerges in the center.

As point source height is increased for poly-disperse spheres, an increase in the magnitude of the stress dip is observed in agreement with experiment. The slope of the curve of residence versus radial distance increases with point source height, indicating that stable contact chains form first at radially more distant points before forming at the heap center as the point source increases.

A local stress minimum also develops at the middle of a heap constructed from poly-disperse prolate spheroids with a corresponding monotonically increasing contact age with radial distance, which again is indicative of a significant cratering phenomenon. No local stress minimum develops at the center of the heap for a system of mono-disperse prolate spheroids, which also correlates well with a homogeneous average contact age with radial distance (up to the initial crater skirt) and implies a layered deposition construction history. The characteristics of the deposition history suggest that cratering plays a critical role in determining the appearance and magnitude of the local stress minimum at the center of a granular heap.

Chapter 7

Development of Multi-physics and Multi-resolution Contact

Soil mechanics oftens uses terms of sphericity, angularity, and roughness to describe the surface geometry of granular material, and gravel, sand, silt, and clay to describe the total particle size [75]. A very deep literature base exists that elucidates the subject. Sphericity refers to the general shape of the particle; most soils range from roughly spherical to highly elongated or plate-like; this is a mainly qualitative distinction based on silhouette figures. Angularity refers to the deviations from the general shape; for instance many particles have protrusions from the surface of a fraction of the smallest gross dimension. Finally, roughness refers to the asperities on the surface that are considerably smaller than the angularity deviations, and which can be captured by replacing with a Coulombic friction term (or alternate constitutive contact law).

These calculated quantities are then used to model the material with laws that are a function of the sphericity, angularity, and roughness. However, the boundaries between sphericity, angularity, and roughness are not standardized. The determination of when an angular deviation should be considered as part of the general topology (sphericity term) or when a small asperity should be included in the angularity term is currently subjective, though, objective measures have been established to try to address this. Moreover, even with objective metrics, the classifications are depen-

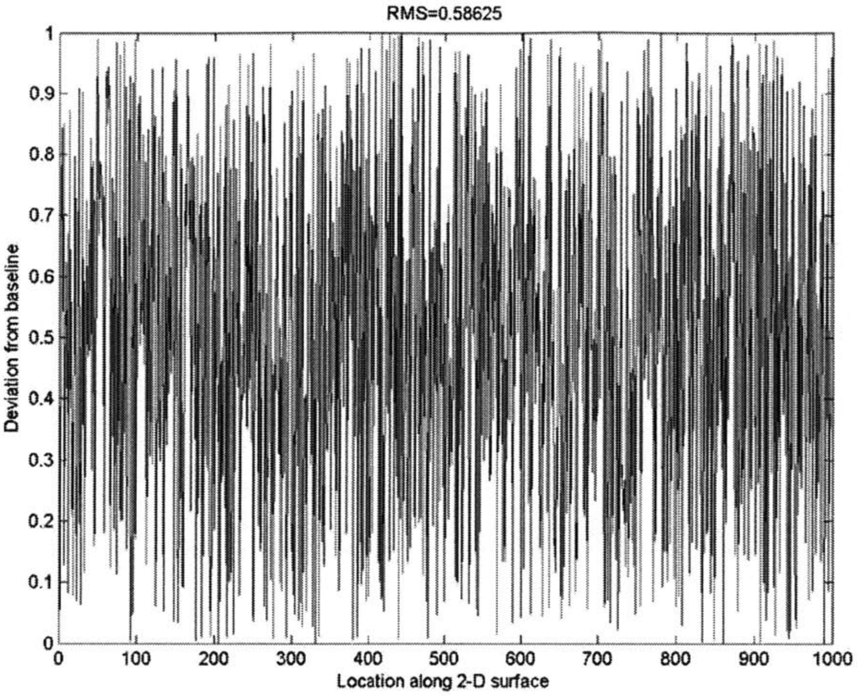


Figure 7-1: RMS Measurement for a random rough surface.

dent on the application. It can be readily seen that an asperity of 1% the height of the smallest dimension of the particle may be included in the friction parameters (roughness) for a compressed dense fabric analysis, but that same asperity may need to be explicitly modeled if the particle is being measured for angle of incidence after collision.

Chemical powder processing researchers have been interested in particle topology, as well and have developed a systematic method to characterize material shape quantitatively. Generally a coarse convex geometry is assumed and asperities are then characterized by statistical relations based on the deviation of the surface from the coarse approximation. Commonly used metrics are the root mean square (RMS) deviations of the surface [71] based on the finest observed resolution data. This is the first moment of the distribution (the variance) of the asperities. The mean as well as higher order moments (skewness and kurtosis) are also sometimes used to characterize a surface [71]. The drawback to this type of analysis is the loss of po-

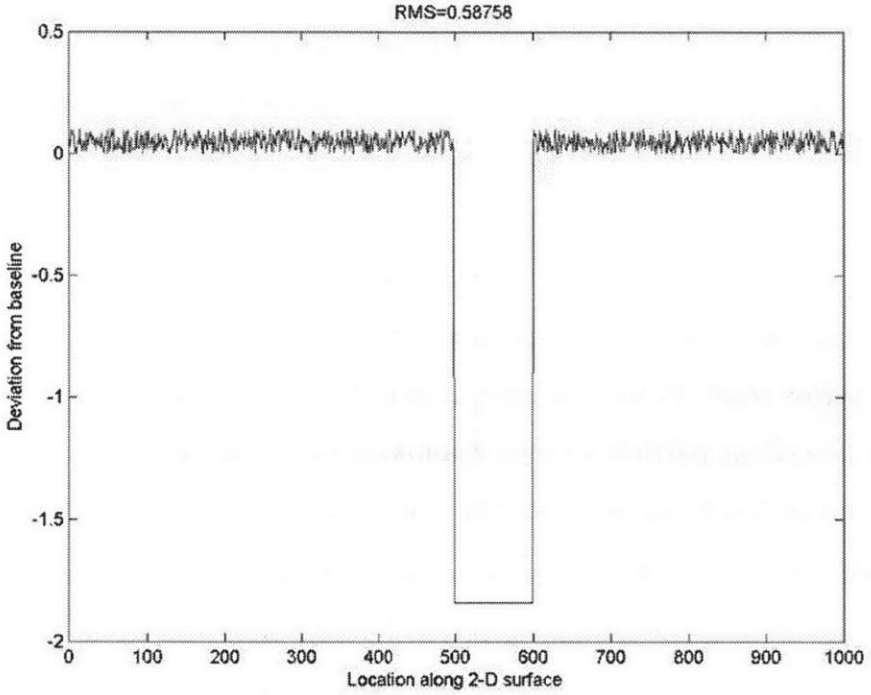


Figure 7-2: RMS Measurement for a random smoother surface with a single high asperity.

tentially important information. For instance, as Figures 7-1 and 7-2 show, the most common measure, RMS, will yield a metric that has the same result for a smooth surface with a few large asperities and a rougher surface of smaller asperities. For contact mechanics and interaction force theories, these two types of surfaces yield drastically different behavior. A similar effect was theorized [102], and it was shown that stochastic surface profiles are sensitive to the scanning length (analysis window) and the frequency.

The previous discussion highlights that there is a lack of a common framework to measure particle topology and differing resolution requirements between different applications, which has generally led to the adoption of the coarsest model. For instance Katsuaki [53] uses uniformly sized hemispherical asperities to model surface angularity, which disregards the multi-scale nature of surfaces. The confounding problem seems to be that geometric features are not separable into distinct categories but are continuous across scale. A proper framework would integrate the property of continuity across scale. From this logic, it would be preferable to have a hierarchical method for describing particle geometry, allowing the researcher to only include shape resolution to the depth necessary for the problem.

The ability to model object resolution in a hierarchical framework is not a new idea. Demand from game designers, recent interest from the agent-based computing/visualization community, and scientific visualization researchers have advanced the study of hierarchical geometry representation over the past decade. These have included voxelization [111], spherical harmonics, radial basis functions [62], and surface wavelets. For roughly spherical particles, for instance, spherical wavelets [103] can be used to capture hierarchically more detailed resolutions of a general topology. Schroeder and Sweldens further refined the spherical wavelet to explicitly represent graphical texture mapping [104].

Though excellent at representing geometries efficiently for graphics rendering, this work had generally been poor at capturing surface normals and edge features that are fundamental to accurately resolving contact in discrete element modeling. As an extension of the work in computer graphics, researchers adapted some of these methods for characterizing real-world granular materials. The idea of using spherical harmonics for capturing particle shape in a hierarchical manner was forwarded [31]. A more general method of capturing surface morphology using surface wavelets was formulated in the context of discrete element modeling [129] and further developed for general surface integral evaluations [3], [4].

These approaches are excellent for categorization of particles and provide a more precise characterization of granular materials over the classic categories of sphericity, angularity, and roughness. They also have the flexibility of capturing surface normals required for resolving contact. However, the hierarchical approaches have a drawback for application in DEM, in that the generality of the methods disregards the existence of an implicit convex hull for the geometry. Simple convex hulls are necessary for a contact detection algorithm in DEM to work efficiently. By applying destructive geometry to a simple gross topology, wavelets or simple surface maps can be used to characterize the angularity and roughness that describes the difference between a particle and the simple primitive used to approximate it.

The leap to hierarchical asperity mapping from graphics mapping is very small. The characteristics of each level of resolution can be captured and characterized independently of other resolution levels. Statistics and assumptions of fractal behavior can further be exploited via such a representation to reduce the memory requirements. The energy, for instance, of each level could be used as a more appropriate metric, showing the ratio of different sizes of asperities. In this thesis, the theory will not be extended into wavelets, though, it might be an interesting extension of the research. Instead, the idea of using a simple hierarchical, regular grid for layering data will be explored and demonstrated.

7.1 Development

The ability to layer information on the surface of a body can be useful from several different viewpoints. Layering of data for surface properties is achieved through a procedure akin to texture mapping in computer graphics. A 2-dimensional patch is

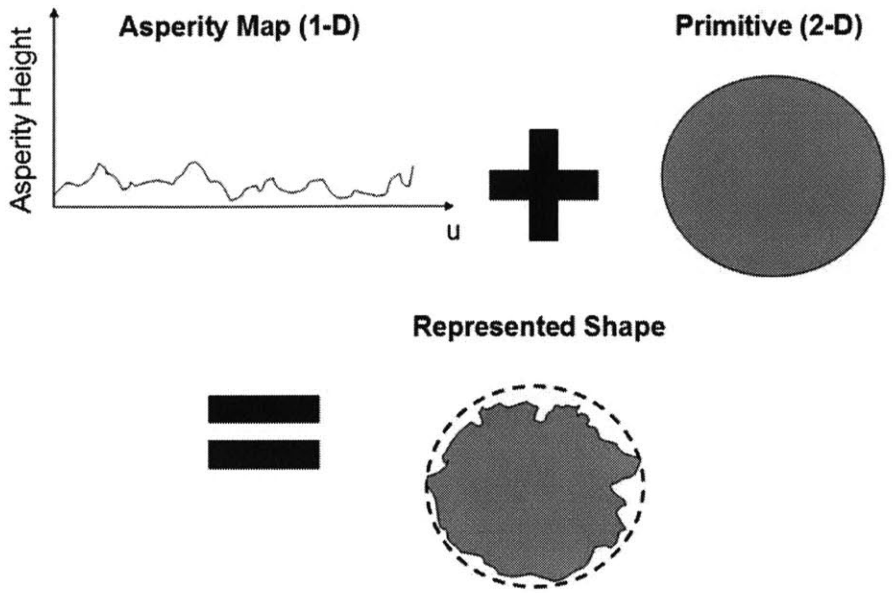


Figure 7-3: Illustration of data layering for a 1-D map and 2-D primitive: an unsigned 1-D function is used to map multi-physics information (e.g., asperity depth data) onto the surface.

mapped over the surface of a convex body using U-V coordinates corresponding to local latitudes and longitudes on the particle surface. If the data being layered is, for instance, asperity height data, it can be represented in the map much like digital elevation map data on a regular grid, except the baseline is the geometric boundary of the primitive and the height data represents the depth of indentations on the surface, allowing the primitive to become a convex hull.

As long as the topology being modeled is convex and relatively spherical, layering information on a surface via a 2-D mapping works well. Roughness can be defined in terms of the unsigned deviation of the true radius from the gross geometry assumed. The deviation will be taken as unsigned so as to enforce the constraint that the gross geometry be a convex hull of the particle, which is amenable to use in contact resolution schemes.

In Figure 7-3, the data layering approach that will be described in this section is used to capture asperities.

The example in Figure 7-3 is in 2-D to illustrate the details of applying the method;

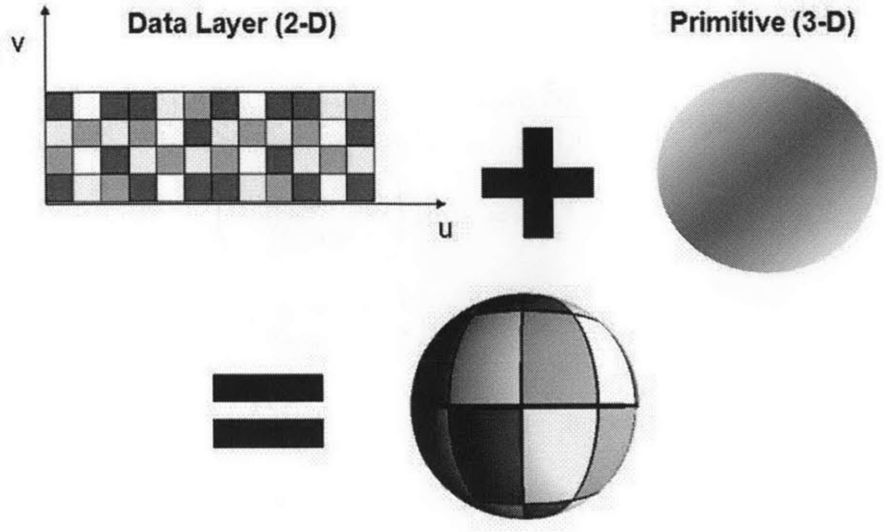


Figure 7-4: Illustration of data layering for a 2-D map and 3-D primitive: an unsigned 2-D function is used to map asperity depth data onto the surface.

however, the extension to 3-D naturally follows as shown in Figure 7-4.

The map is created using an array of dimension 3, where the dimensional discretization is of size $[num_u, num_v, num_p]$. num_u indicates the discretization along the latitudinal direction, num_v the discretization along the longitudinal direction, and num_p indicates the number of layers of information (extra parameters) being modeled. For instance, a map containing normal skews and asperity height data would be $num_p = 3$ for u_{skew}, v_{skew}, and asperity height.

To map a point in the global reference frame to the map, which is based in the local reference frame of the particle of interest, follow the following procedure.

$$\vec{c}_{global} = \vec{cp} - \vec{cog} \tag{7.1}$$

$$\vec{c}_{local} = R \cdot \vec{c}_{global} \tag{7.2}$$

$$\vec{c}_{normal} = \frac{\vec{c}_{local}}{|\vec{c}_{local}|} \tag{7.3}$$

$$\phi = cos^{-1}(\vec{c}_{normal} \cdot \hat{k}) \tag{7.4}$$

where $\vec{cp}$ is the global coordinate of the point and $\vec{cog}$ is the centroid of the particle.

If $sin(\phi) = 0$, then $\theta = 0$. Otherwise, the θ angle is determined through the following procedure:

$$\vec{c}_{normal,ij} = \frac{(\vec{c}_{normal} \cdot \hat{\imath})\hat{\imath} + (\vec{c}_{normal} \cdot \hat{\jmath})\hat{\jmath}}{|(\vec{c}_{normal} \cdot \hat{\imath})\hat{\imath} + (\vec{c}_{normal} \cdot \hat{\jmath})\hat{\jmath}|} \tag{7.5}$$

$$v_1 = sign\left(\frac{\vec{c}_{normal,ij} \cdot \hat{\imath}}{sin(\phi)}\right) \tag{7.6}$$

$$v_2 = sign\left(\frac{\vec{c}_{normal,ij} \cdot \hat{\jmath}}{sin(\phi)}\right) \tag{7.7}$$

$$\theta_1 = \left|cos^{-1}\left(\frac{\vec{c}_{normal,ij} \cdot \hat{\imath}}{sin(\phi)}\right)\right| \tag{7.8}$$

$$\theta_2 = 0, v_1 > 0, v_2 \geq 0 \tag{7.9}$$

$$\theta_2 = \frac{\pi}{2}, v_1 \leq 0, v_2 > 0 \tag{7.10}$$

$$\theta_2 = \pi, v_1 < 0, v_2 \leq 0 \tag{7.11}$$

$$\theta_2 = \frac{3\pi}{2}, v_1 \geq 0, v_2 < 0 \tag{7.12}$$

$$\theta = \theta_1 + \theta_2 \tag{7.13}$$

Note that the variables θ and ϕ are constrained to be in:

$$0 \leq \theta < 2\pi \tag{7.14}$$

$$0 \leq \phi < \pi \tag{7.15}$$

For a map of size $[num_u, num_v]$, the bins corresponding to the point $\vec{cp}$ can then be found as:

$$bin_u = floor\left(num_u \frac{\theta}{2\pi}\right) \tag{7.16}$$

$$bin_v = floor\left(num_v \frac{\theta}{\pi}\right) \tag{7.17}$$

If the data layer, for example, holds asperity height data, simply scale the value in $map[bin_u, bin_v]$ appropriately and subtract it from the calculated normal at point $\vec{cp}$. Note that $map[bin_u, bin_v]$ denotes the value of the data layer array corresponding to

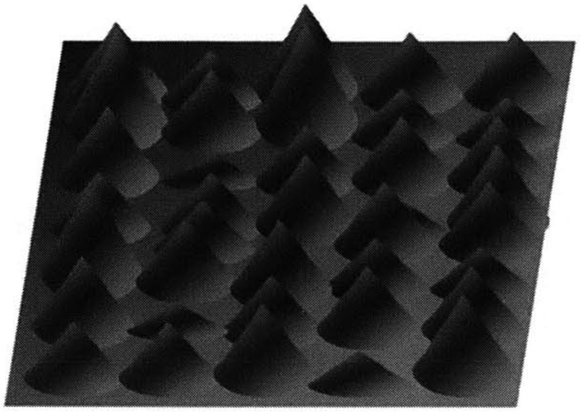


Figure 7-5: Illustration of the conical model of micro asperities used in modeling the perturbation of surface normals.

(bin_u, bin_v). This general procedure can be used for any scalar data:

$$r_{perturbed} = r_{unperturbed} - \beta_0 \cdot map[bin_u, bin_v] \tag{7.18}$$

where β_0 is the scaling factor at level 0.

7.2 Geometric Interlocking

A useful application of this concept of capturing surface resolution is in the application to modeling microscopic asperities on the surface of the particle. If the asperities are assumed to be relatively small compared to the dimensions of the primitive onto which they are superimposed, then their effect on the detection of contact between 2 primitives is considered negligible. Thus, the problem of determining whether two rough primitives interfere can be reduced to the problem of whether two smooth primitives interfere, and the computational intensity of determining detailed contact resolution can be reduced accordingly.

To model the surface angularity, this section presents a conical representation of the micro asperities on the particle surface. To achieve this, the data layers are used to hold the normal perturbation data. The perturbation data must be appropriately projected orthogonal to the plane defined by the normal at the contact point, $\vec{cp}$. Because there are an infinite number of unit vectors orthogonal to the unit normal vector, it is advantageous, in the case of modeling geometric interlocking, to set the u direction as lying parallel to the tangential velocity vector of the contact point. This allows us to explicitly model the asperity as a conical asperity, as shown in Figure 7-5, where the direction of normal perturbation necessarily opposes the direction of velocity.

$$\hat{n_1} = -\frac{\vec{v_2} + \vec{\omega_2} \times \vec{r_2} - \vec{v_1} - \vec{\omega_1} \times \vec{r_1}}{|\vec{v_2} + \vec{\omega_2} \times \vec{r_2} - \vec{v_1} - \vec{\omega_1} \times \vec{r_1}|} \tag{7.19}$$

$$\hat{n}_2 = \frac{\vec{c}_{local} \times \hat{n}_1}{|\vec{c}_{local} \times \hat{n}_1|} \tag{7.20}$$

$$\vec{\rho} = \beta_0 \left(\hat{n}_1 |\rho_u| + \hat{n}_2 \rho_v \right) \tag{7.21}$$

where ρ_u is the skew associated with bin_u and ρ_v is the skew associated with bin_v. This makes the adjusted normal:

$$\vec{c}_{local,new} = \frac{\vec{c} + \vec{\rho}}{|\vec{c} + \vec{\rho}|} \tag{7.22}$$

To validate this approach to modeling geometric interlocking, a simple numerical experiment is used. A pair of spherical particles of the same radius ($r = 1$) are brought into geometric contact such that the contact deformation is 1% of the radius (i.e., $\alpha = 0.01$. Both particles are fixed in every degree of freedom except for the rotational degree of freedom about the x-axis in the convective frame of reference. The first particle is specified to rotate at a constant rate of $1(rad/s)$ whereas the other particle is allowed to rotate freely. The contact is idealized by a simple linear contact law (spring contact) with no friction applied; therefore, rotational motion of the freely rotating particle will be initiated only through the mechanism of interlocking microasperities. The setup of this numerical experiment is illustrated in Figure 7-6.

The results shown in Figure 7-7 indicate that after an initial slip phase of contact, the rotational contacts stick. The counter rotation of the two spheres shows that the

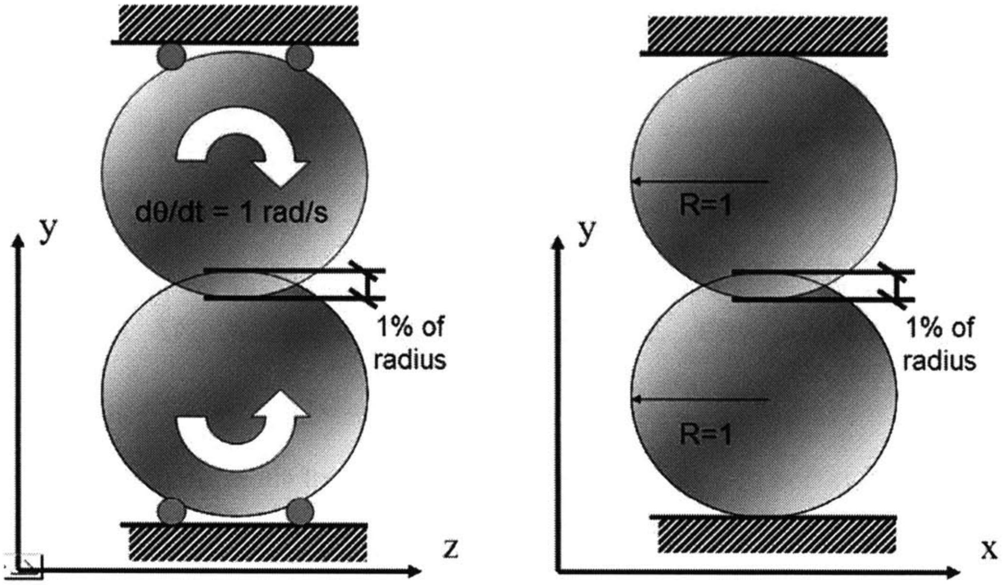


Figure 7-6: Illustration of a simple numerical validation of the conical asperity model.

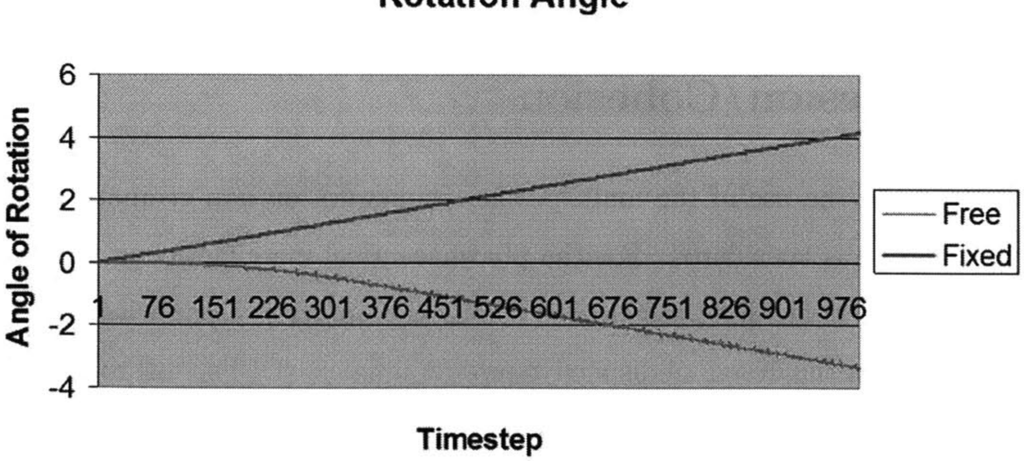


Figure 7-7: Illustration of a simple numerical validation of the conical asperity model.

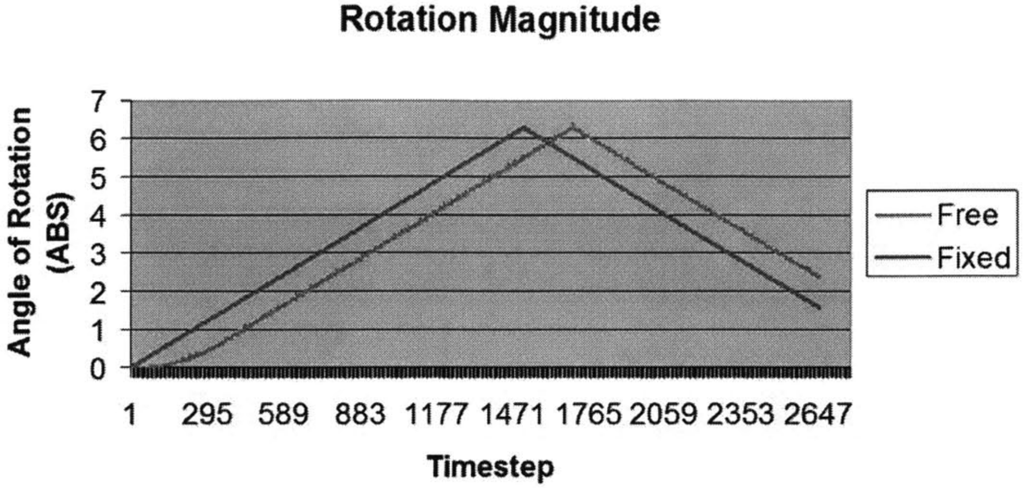


Figure 7-8: Illustration of a simple numerical validation of the conical asperity model.

shapes are rotating as expected of a cog-tooth type system. From Figure 7-8 one can see that the magnitudes of the rotations are exactly matched after an initial slip region and that this remains valid throughout the simulation. Together, these results show that the model operates as expected for a simple pair of spheres. Also, since this captures the important mechanisms to be captured by this model, the results can be extrapolated as validations for all systems of interest.

7.3 Adhesion/Cohesion

To demonstrate the use of the multi-physics framework on non-geometric data, the case of cohesion is considered. For the pharmaceutical manufacturing industry, understanding and controlling the surface cohesion properties of particles is extremely important. From the design of dispersive powder inhalers (DPI's) [14] to the selection and development of powder blenders and the selection of blending excipients, surface cohesion plays a key role in the industrial production of micron-scale powders.

Several researchers have noted the correlation between relative humidity and powder adhesion (e.g., see a small survey in [16]), noting that one of the dominant mechanisms in adhesion is the formation of liquid bridges at relative humidities similar to those found in the typical operating environment of pharmaceutical powder blenders. In relatively dry environments, an electrostatic potential may develop at the surface of micron-sized particles, providing a different high magnitude adhesive force. In both cases the cases of electrostatic potential [113] and surface geometry [24, 88], the surface roughness of the particle has a significant correlation with the magnitude of force developed in the interaction.

For blended cohesive pharmaceutical powders, the crushing and surface deformation of individual powder grains is low and the surface morphology relatively constant; therefore, surface chemistry remains relatively invariant during the process of blending. This invariance can be exploited to create a new model of cohesion that is tied to the microscopic surface of the powder grains.

Capturing cohesion in DEM models is not new. Several researchers have investigated cementatious material bonds using simple spring elements and more complicated beam elements such as Timoshenko beams. These elements have also been tied to specific surface points to add contact moments. In the 2D DEM simulations described in [88] cohesion was simulated in this manner; particles in contact were given point-to-point spring contacts with a failure force at a particular constant force.

Approaches to modeling adhesion/cohesion are numerous and mainly depend on the interactions of interest. Contact models capturing only Van der Waals forces [77] or only the explicit modeling of liquid bridge formation [116] have been used in combination with discrete element modeling. Others have used a contact model which explicitly treats the JKR model of adhesion [44] for perfectly smooth spheres using self-consistent micro-mechanical contact models coupled with a Hertzian contact condition for normal contact forces and a Hertz-Mindlin model for tangential tractions [126]. There has been other work on the representation of cohesive contacts [51] as well as the modeling a torque-bearing sintering necks for capturing moment-resisting contacts between particles [6].

In this thesis, however, cohesion is modeled as a stochastic process. The model is developed first as an independent stochastic process and then refined as a location-dependent stochastically distributed value across the surface of a convex body. That

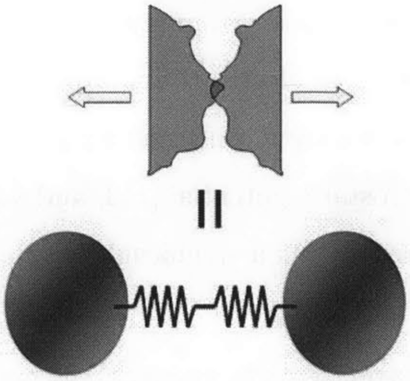


Figure 7-9: Illustration of surface level cohesion model for interacting cohesive powders.

is, a size-independent cohesion factor is applied at each point on the surface. Each value on the surface remains constant throughout the simulation, providing consistency between theory and model.

Several AFM cohesion maps and corresponding surface asperity maps were recorded by Ngai [88] for characteristic samples of micro-crystalline cellulose (MCC), lactose, and caffeine, and a log-normal distribution of adhesion pull-off forces were obtained. Using the model of adhesion developed by Johnson [44] (see the following equation), we were able to reconstruct a stochastic surface energy parameter that fits the log-normal distribution found from experiment.

The JKR model yields a maximum force before failure (pull-off) of:

$$F = -\frac{3}{2}\pi\gamma\left(\frac{R_1R_2}{R_1+R_2}\right) \tag{7.23}$$

$$\gamma = -\frac{2F(R_1+R_2)}{3\pi R_1R_2} \tag{7.24}$$

Using the results of Ngai [88], the cohesion model can be rewritten as:

$$\gamma(X) = -\frac{2F(R_1+R_2)}{3\pi R_1R_2} \tag{7.25}$$

where $\gamma(X)$ is a random variable that fits the experimentally determined log-normal distribution in [88].

This is effectively a constitutive law that helps to avoid modeling the complexity of particle surfaces while capturing the force distribution between particles. For caffeine powder grains, the distribution of surface energies was derived to be the a log normal distribution characterized by the standard log normal parameters $\mu = 42\frac{mJ}{m^2}$ and $\sigma = 21$. In code, this is implemented as two matrices of values: one of log normal means and one of log normal standard deviations which are then queried based on the element at the row corresponding to the material of the first particle and the column corresponding to the material of the second particle. It can immediately be seen that this constitutive relation avoids describing individual surfaces and instead describes the average pull-off force between pairs.

Experimental evidence suggests that the cohesive interaction is based on where on the surface a cohesive contact occurs, so it would be preferable to decouple the pair-wise cohesive parameter into parameters tied to each surface in contact. That is, if we decouple this stochastic law, we can get closer to the theoretical mechanisms of contact by connecting surface energy to specific surface points. This mirrors the effective properties of other surface parameters and improves of the lumped stochastic cohesion model that we developed (detailed in [88]). The model used here is shown in Figure 7-9, where the interaction is taken to be a series effect of the contributions of the two surfaces in contact. This is common for most effective parameters of interacting surfaces and is dimensionally consistent. The model can be specified as:

$$\gamma_{eff}(X) = \frac{\gamma_1(X)\gamma_2(X)}{\gamma_1(X) + \gamma_2(X)} \tag{7.26}$$

The first problem with decoupling the effect is to determine $\gamma_1(X)$ and $\gamma_2(X)$. Determining the constituent distributions that, after operation, form a log normal distribution is not a trivial task and determining the mean and variance of sums of log-normally distributed variables is an active research area [39, 97, 105, 109]. Using the assumption in [105] that the resultant distribution of the sum of two log-normally distributed variables can be approximated to the first order by another log-normally distributed variable and that the distribution of the surface energies for two variables

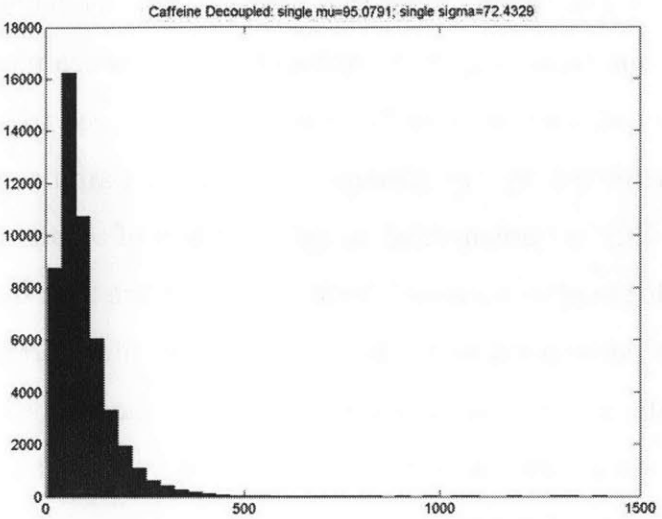


Figure 7-10: Constituent distribution of caffeine surface energies for γ_i from the model in 7.27.

are equal (i.e., from 7.26 $\gamma_1(X) = \gamma_2(X)$) for particles 1 and 2 composed of the same material, a model can be formulated.

It is first postulated that the model will result in a log-normally distributed $\gamma_1(X)$ given that $\gamma_1(X)$ and $\gamma_2(X)$ are also log-normally distributed, where the variables are related as described in 7.26. This will be verified later. Because of the underlying complexity of the model, research into power sums is of limited use here. It would be advantageous to identify a simpler relationship to estimate the parameters of the constituent distributions. Using a brute force approach fitted with experimental data from caffeine cohesion experiments, a relationship between the statistics of $\gamma_{eff}(X)$ and those of $\gamma_1(X)$ and $\gamma_2(X)$ is developed:

$$\mu_i = log(2.15) + \mu \tag{7.27}$$

$$\sigma_i = \sqrt{1.95} \cdot \sigma \tag{7.28}$$

Evaluating 7.27 using the parameters of the experimentally determined caffeine-caffeine cohesion surface energy distribution yields the log-normal distribution illus-

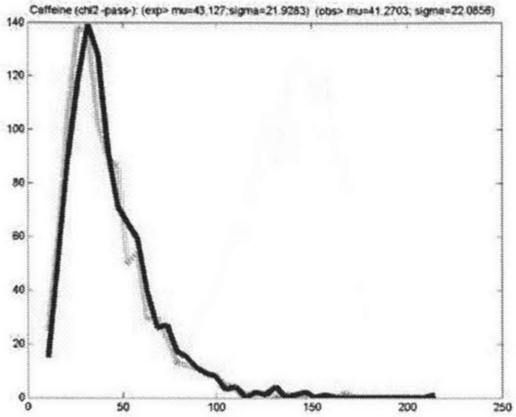


Figure 7-11: Comparison of the distribution of surface energies of interacting caffeine grains for those determined from the experiments of [88] (blue) and those predicted by the model presented here (red). Note that the prediction passes the Pearson goodness-of-fit test.

trated in Figure 7-10. Using this distribution in the model 7.26, the distribution shown in Figure 7-11 emerges and compares well with the distribution of the experimental data. The emergent distribution of caffeine surface energies passes the Pearson goodness-of-fit test, a test which indicates how close a sample is to a theoretical statistical distribution.

The model was calibrated by fitting data to the model, so it is not valid to claim success based on the results agreeing well with the results from which it is calibrated. To verify that this works for other materials the same procedure is performed on MCC using the model and coefficients specified in 7.27. The emergent distribution for MCC also compares well with experiment as shown in Figure 7-12, and it likewise passes the Pearson goodness-of-fit test.

This development shows that the model is valid across materials, but it does not indicate whether the model is unique or whether the model is even sensitive to the values of the coefficients. In other words, how is this formulation unique or innovative? To test this, it is useful to perturb the values of 7.27 and see whether the resultant distribution is similar to the experimentally determined distribution using the coefficients described in 7.29. The results illustrated in Figure 7-13 show that

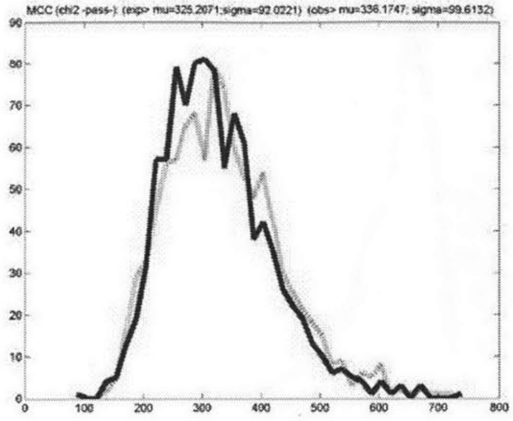


Figure 7-12: Comparison of the distribution of surface energies of interacting MCC grains for those determined from the experiments of [88] (blue) and those predicted by the model presented here (red). Note that the prediction passes the Pearson goodness-of-fit test.

this small perturbation creates a distribution that does not fit the experimental data to the degree necessary to pass the Pearson goodness-of-fit test.

$$\mu_i = log(2) + \mu \tag{7.29}$$

$$\sigma_i = \sqrt{2} \cdot \sigma \tag{7.30}$$

Figure 7-13 illustrates the sensitivity to coefficients; however, the emergent distributions are also sensitive to the model adopted. If a different model is contrived and the coefficients of the model are fit to the caffeine experimental data, the equation in 7.31 can be derived. The results of using this model to determine the statistics of the constituent distributions is reasonable for caffeine; however, extension to MCC results in a very poor fit (qualitatively or quantitatively), as shown in Figure 7-14.

$$\mu_i = \sqrt{1.6} \cdot \mu \tag{7.31}$$

$$\sigma_i = log(2) + \sigma \tag{7.32}$$

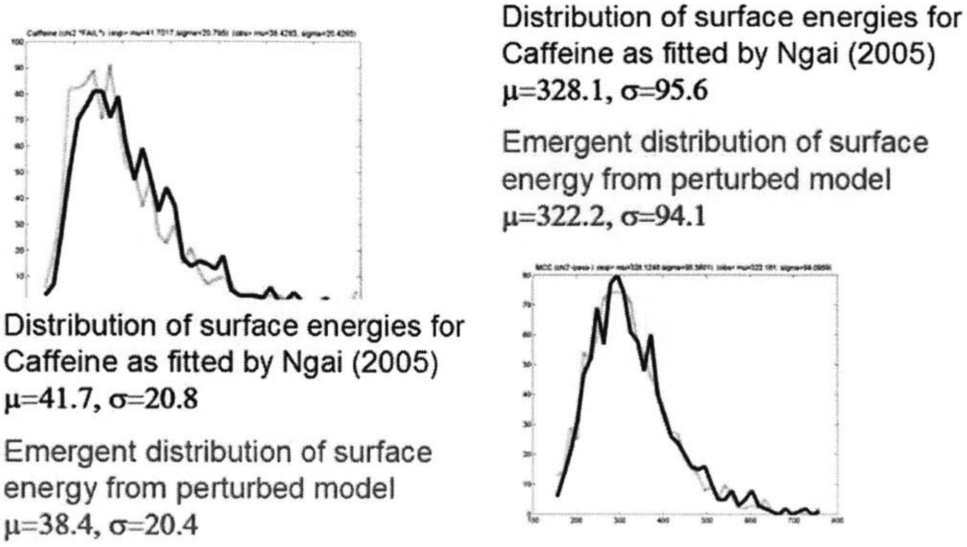


Figure 7-13: Effect of perturbing the model coefficients.

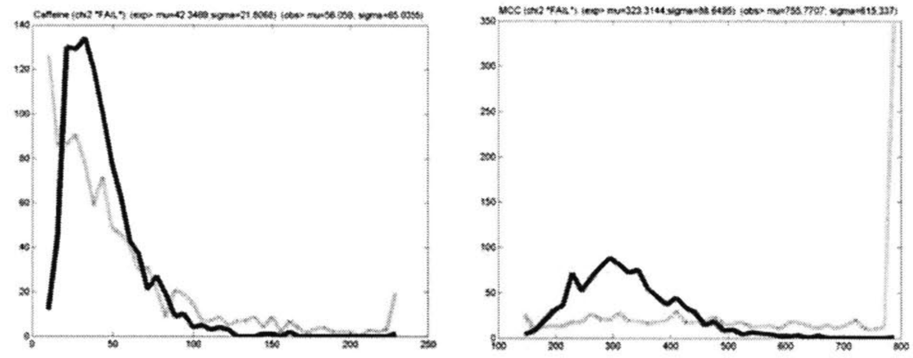


Figure 7-14: Effect of perturbing the form of the model.

Through the previous discussion both the model and the coefficients adopted for the model have been shown to be unique. This model has performed well in predicting the constituent distributions necessary to create an emergent surface energy distribution that agrees well with experimental data. With the theoretical work completed, it is attractive to try to use these constituent distributions to define the cohesive properties of specific points on the surfaces of bodies. This is accomplished here by using the multi-scale surface characterization approach that is the subject of this chapter. The graphical result of a typical mapping with surface energy values shown as height difference from the particle surface is shown in Figure 7-15.

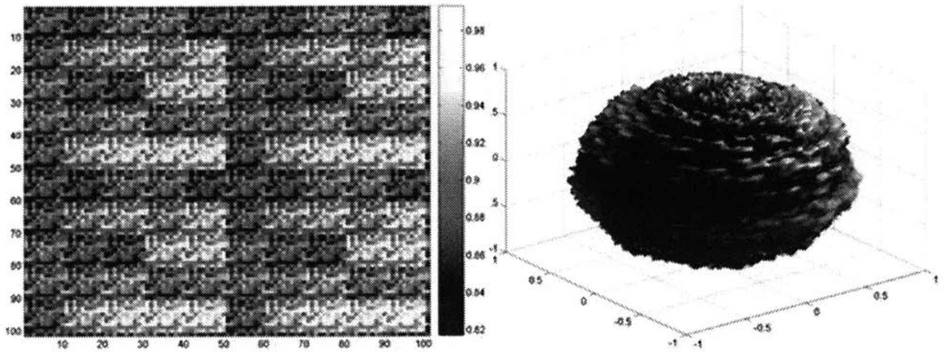


Figure 7-15: Shown by a height/color map, the surface energy map (left) and resultant visualization of a spherical particle (right) are illustrated.

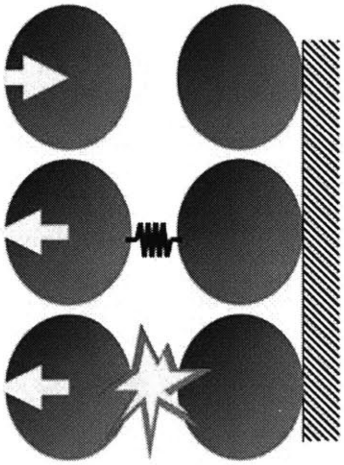


Figure 7-16: Procedure for performing numerical pull-off tests: (top) particles are brought together (middle) once particles bond, they are pulled apart (bottom) bond breaks and max force recorded

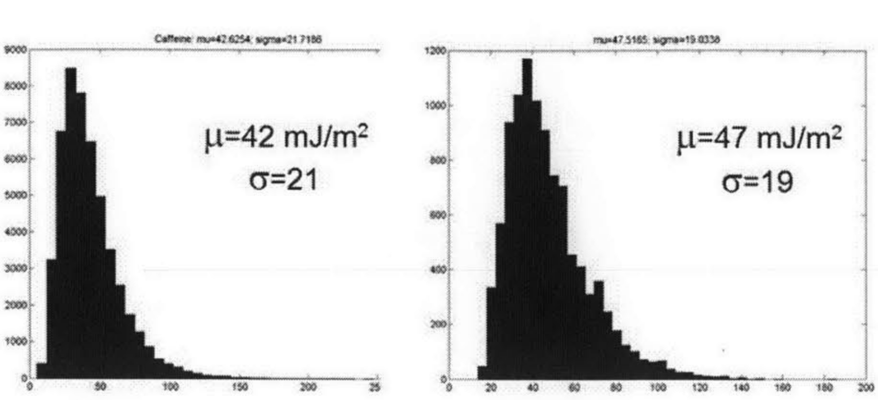


Figure 7-17: Comparison of the distribution of surface energies of interacting caffeine grains for those determined from the experiments of [88] (left) and those that emerge from numerical pull-of tests implementing a surface map of distributed surface energy values (right).

The results described in the previous paragraph indicate that the model presented in 7.27 produces similar distributions to those observed experimentally. It does not, however, indicate that by applying this distribution to the discretized multi-physics surface map described in this chapter that the distributions will be the same.

To test this hypothesis, a numerical experiment was structured in the DEM3D framework. First, a set of maps are assigned to each particle in a pair of particles; the distribution of values on each map is drawn from the derived log normal distributions of values. Second, 50000 trials are run where a random rotational orientation is assigned to each particle and the particles are pulled apart as illustrated in Figure 7-16. The maximum force is determined by the model in 7.27. The maximum force is recorded for each trial, the surface energy recorded, and the resulting distribution of surface energies plotted. The results, illustrated in Figure 7-17, show that the experimentally-derived distribution and the emergent distribution from the numerical pull-off tests agree well.

With the determination of the surface energy and maximum pull-off force specified, the force-displacement relationship for the JKR model can be written as follows, where a is the radius of the contact area, $F(a)$ is the adhesive force as a function of contact area radius, and α is the separation (displacement) of the particles:

$$F(a) = \frac{4E^*a^3(R_1+R_2)}{3R_1R_2} - \sqrt{8\pi\gamma_{eff}(X)E^*a^3} \tag{7.33}$$

$$E^* = \frac{(1-\nu_1{}^2)(1-\nu_2{}^2)}{E_1(1-\nu_2{}^2)+E_2(1-\nu_1{}^2)} \tag{7.34}$$

$$\alpha = \frac{a^2(R_1+R_2)}{R_1R_2} - \sqrt{\frac{2\pi\gamma_{eff}(X)a}{E^*}} \tag{7.35}$$

As fluidization of particle flow increases, the transmission of stress waves becomes a secondary effect to inertial effects. Similar behavior has been observed in gas-fluidized beds using cohesive particulate matter [77]. Tests were performed on 2-D rotating cylinders of cohesive particles using identical initial conditions but with the stiffness parameter an order of magnitude different between trials. For the variation of blend homogeneity as a function of time, there is little difference observed between the trials, indicating the relative insensitivity of time-varying blend homogeneity to the contact stiffness model used. A thorough sensitivity analysis for the stiffness parameter in particle-particle interactions for a DEM simulation of a V-mixer was conducted by Kuo [57]; their findings corroborate ours with a conclusion that "no difference could be distinguished" between simulations with different stiffnesses. Because of the second order nature of this parameter, a simple linearization of the Hertzian contact model can be used, which is designed to minimize the error in the stored energy over the range $[0, 5\% * radius]$.

From the work of Hertz [37], a force-displacement relationship for spherical bodies was derived:

$$F_n = k_n\alpha^{1.5} \tag{7.36}$$

$$k_n = \frac{4}{3}E^*\sqrt{R^*} \tag{7.37}$$

$$E^* = \frac{E_1E_2}{E_1(1-\nu_2{}^2)+E_2(1-\nu_1{}^2)} \tag{7.38}$$

$$R^* = \frac{R_1R_2}{R_1+R_2} \tag{7.39}$$

$$\tag{7.40}$$

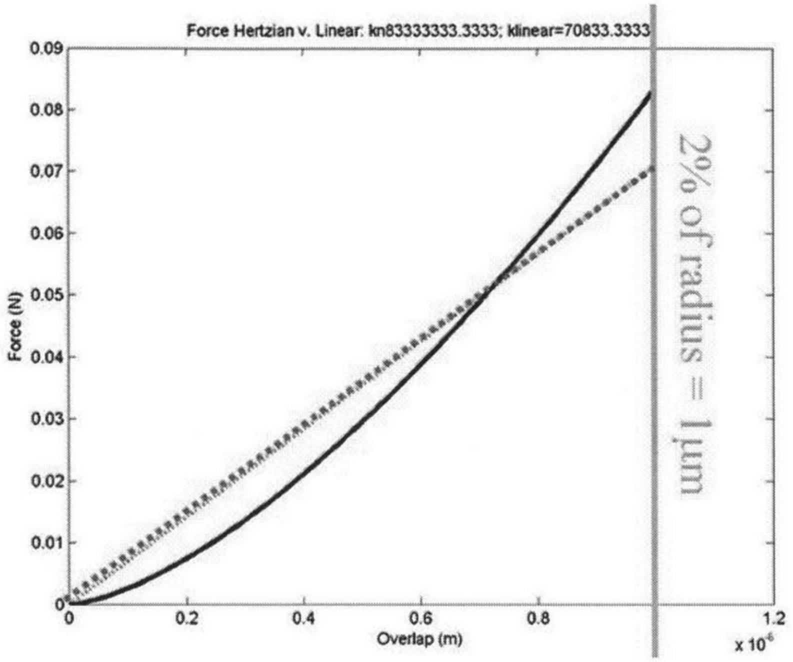


Figure 7-18: Force-displacement comparison over the range of the separation, $\alpha = [0, 2\% * radius]$. Red (dashed) is the linearized case and blue (solid) is the Hertzian case.

$$F_n = k\alpha \tag{7.41}$$

Taking the values for MCC from [25] and using the mean particle diameter for our material sample as $100\mu m$, we can determine that $k_n = 8.3e7\frac{N}{m}$. If the error is minimized for the stored energy over the range $\alpha = [0, 5\% * radius]$, k, the linearized stiffness, can be specified as $\frac{k}{k_n} \approx 0.00085$. The resultant force-displacement relationship compared with Hertzian model is shown in Figure 7-18, and the energy-displacement relationship is shown in Figure 7-19, which demonstrates the accuracy of the approximation with respect to energy.

Finally, the stochastic model of cohesion can be coupled with a model to resist shear, moment, and torque. One self-consistent method of treating frictional resistance is by taking the combined approach of [125, 126], which assumes a dependence of Coulombic friction on the contact normal force only (that is, without the component of tensile cohesion). The self-consistency of this formulation can be seen when

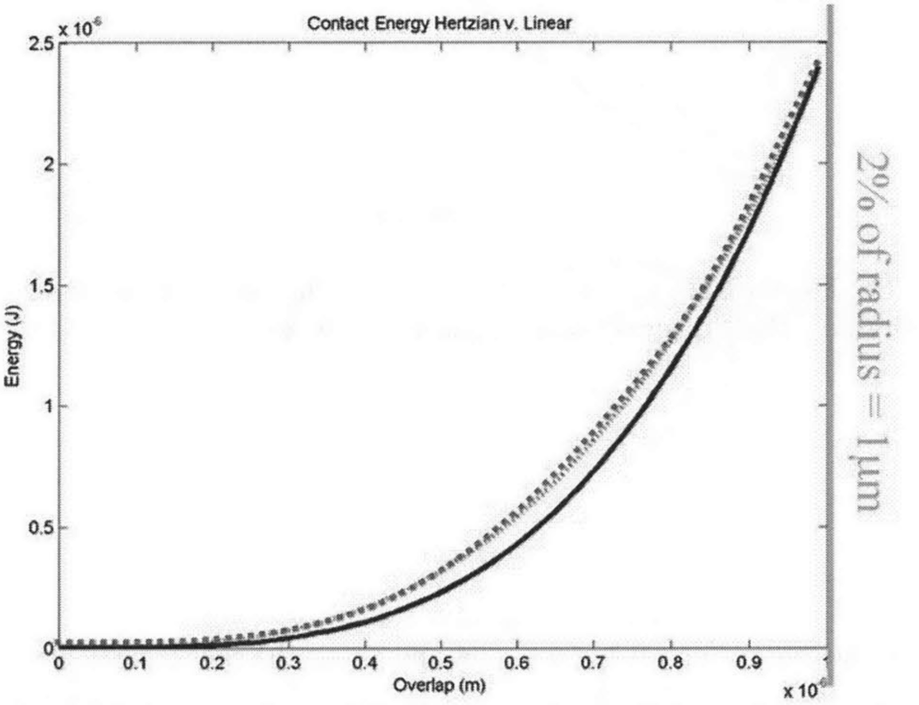


Figure 7-19: Energy-displacement comparison over the range of the separation, $\alpha = [0, 2\% * radius]$. Red (dashed) is the linearized case and blue (solid) is the Hertzian case.

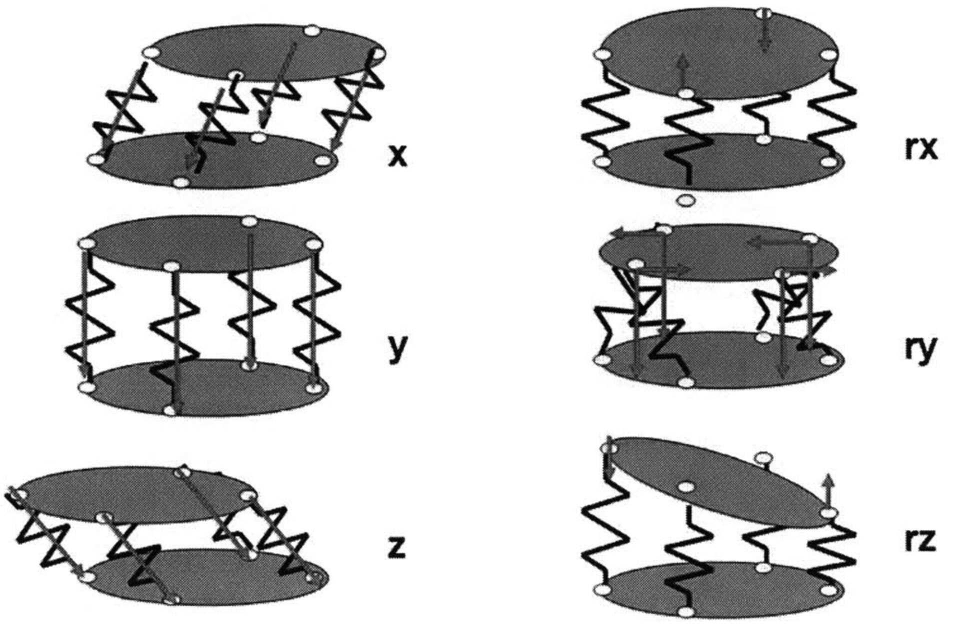


Figure 7-20: Illustration of constraints for degrees of freedom for multipoint contact based on a quadrant division of the contact area circumference.

one takes the limit case of static equilibrium between normal contact compressive force and cohesive tensile force (i.e., tensile cohesive force and contact compressive force are equal in magnitude and opposite in direction). From this, it is simple to see by inspection that the shear resistance is not zero and that frictional resistance would remain proportional to contact area, which is related to the contact traction.

A model for moment and torque resistance based on plastic yielding of the grains about the contact area has been proposed by [126]. In this work, moment and torque resistance are enforced by dividing the cohesive contact into quadrants spaced at equal intervals about the periphery of the initial contact area. Though not as theoretically accurate, the resultant behavior captures the important 6 degrees of resistance as shown in Figure 7-20.

7.4 Extension to Multi-resolution

The further power of this method comes from its extension across multiple scales. The easiest way of extension is by subdividing each grid cell into a new grid of

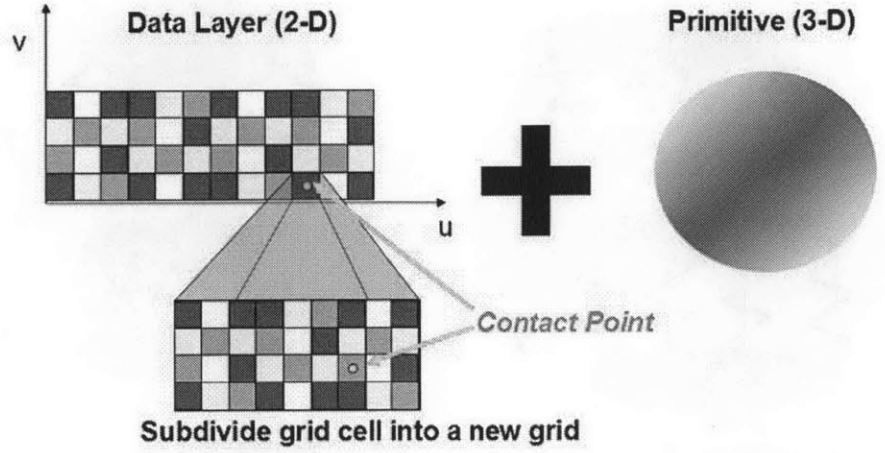


Figure 7-21: Illustration of grid subdivision for a 3-D primitive.

$num_u \times num_v$ cells. The part of the contact point $\vec{cp}$ resolved at level $n-1$ is removed for the analysis at level n and the remainder is mapped onto the data layer and scaled by β_i where $i = [0, n]$. The general concept is illustrated in Figure 7-21

The component of θ and ϕ at level $n-1$ can be determined by the following equations:

$$\theta_i = 2\pi \sum \left(\frac{bin_{u,i}}{num_u^{i+1}} \right) \tag{7.42}$$

$$\phi_i = \pi \sum \left(\frac{bin_{v,i}}{num_v^{i+1}} \right) \tag{7.43}$$

$$\tag{7.44}$$

Note that if $i < 0$, $\theta_i = \phi_i = 0$.

The mapped bins corresponding to these values can then be determined by:

$$bin_{u,i} = floor\left({num_u}^{i+1} \frac{\theta - \theta_{i-1}}{2\pi} \right) \tag{7.45}$$

$$bin_{v,i} = floor\left({num_v}^{i+1} \frac{\phi - \phi_{i-1}}{\pi} \right) \tag{7.46}$$

The scaled vector skew is then calculated:

$$\vec{\rho_i} = \beta_i \cdot (\hat{n}_1 \rho_{u,i} + \hat{n}_2 \rho_{v,i}) \tag{7.47}$$

where $\rho_{u,i}$ is the skew associated with $bin_{u,i}$ and $\rho_{v,i}$ is the skew associated with $bin_{v,i}$. And the final normal vector is determined by the following:

$$\vec{c}_{local,new} = \frac{\vec{c}_{local} + \sum \vec{\rho}_i}{|\vec{c}_{local} + \sum \vec{\rho}_i|} \tag{7.48}$$

As an example of the application of the multiscale procedure, assume that the map is a 2x2 map along $\theta = [0, 2\pi)$ and $\phi = [0, \pi)$, the point to be mapped lies in the local reference frame at $\theta = \frac{3\pi}{2} + 0.1$ and $\phi = \frac{\pi}{2} + 0.2$, and the map is a 2 level hierarchy with the scaling factor $\beta_i = 0.5 \cdot \beta_{i-1}$. The 2x2 map is specified as:

$$map = \begin{bmatrix} -1 & 1 \\ -1 & 1 \end{bmatrix}$$

The procedure would be applied as follows:

First level:

$$bin_{u,0} = floor\left(2\frac{\frac{3\pi}{2} + 0.1 - 0}{2\pi}\right) = 1$$

$$bin_{v,0} = floor\left(2\frac{\frac{\pi}{2} + 0.2 - 0}{\pi}\right) = 1$$

$$\theta_0 = 2\pi\left(\frac{1}{2}\right) = \pi$$

$$\phi_0 = \pi\left(\frac{1}{2}\right) = \frac{\pi}{2}$$

Second level:

$$bin_{u,1} = floor\left(4\frac{\frac{3\pi}{2} + 0.1 - \pi}{2\pi}\right) = 1$$

$$bin_{v,1} = floor\left(4\frac{\frac{\pi}{2} + 0.2 - \frac{\pi}{2}}{\pi}\right) = 0$$

$$\theta_1 = 2\pi\left(\frac{1}{2} + \frac{1}{4}\right) = \frac{3\pi}{2}$$

$$\phi_1 = \pi\left(\frac{1}{2} + \frac{0}{4}\right) = \frac{\pi}{2}$$

Note that at level 1, $\theta_1 = \frac{3\pi}{2} = \theta - R$ where the residual $R = 0.1$, and $\phi_1 = \frac{\pi}{2} =$

$\phi - R$ where the residual $R = 0.2$.

To acquire the multiscale reconstruction of the map value, take:

$$\sum (\beta_i \cdot map_i) = \beta_0 \cdot map[1,1] + \beta_1 \cdot map[1,0]$$

$$= 1 \cdot 1 + 0.5 \cdot 1 = 1.5$$

Because the subdivision can result in a self-similar grid (i.e., a fractal material), the memory required to capture an infinite number of details for a fractally self-similar material for all of the particles in a system can be as little as $O(u_{num} \cdot v_{num})$. This is possible because a reference to the map can be applied to each particle in the system and only one data structure needs to be stored.

7.5 Contributions

The research into a multi-physics and multi-resolution framework for specifying surface properties of granular materials allows a wider range of microscopic parameters to be probed when modeling a granular material. The types of information are myriad, and it has been shown that the model can capture several parameters of interest to researchers.

It has been shown that when the framework can be used to represent microscopic asperities on a smooth particle's surface. The model introduced here assumes that the asperities are conical and are much smaller than the dimensions of the smooth particle on which they are applied. This model has been shown to correctly reproduce geometric interlocking of two particles during a rotation test where the particles are given no friction.

Relating cohesive properties to surface characterizations is an important part of the research into powder production. This thesis has demonstrated that the multiphysics framework can be used to localize cohesive surface energy to specific locations on the powder grain. Numerical pull-off tests of randomly oriented particles confirm that the distribution of surface energies agrees well with that determined from experiment.

Finally, an extension of the framework has been proposed that allows for multiresolution representation of the desired surface parameter. Optimizations such as an assumption of fractal similarity between scales can be exploited to limit the memory resources required.

Chapter 8

Polyhedral Boundaries

Developing polyhedral boundary elements is necessary to efficiently analyze granular behavior in blenders and mixing devices of arbitrary geometries. For instance, v and y- shaped blenders, as well as, double helical ribbon blenders are difficult to represent efficiently using non-polyhedral elements. From a user viewpoint, most computer aided design applications export 3-D machine representations to some type of polyhedral mesh format. For usability, DEM3D supports VRML and 3DS formatted files as inputs for defining polyhedral meshes.

There have been several approaches that attempt to create simple polyhedron-polyhedron and polyhedron-primitive contact algorithms. These usually assume that the facets are much smaller than the contacting primitive. In the case of boundary elements for DEM, the facets are often much larger than the grains contacting them. Representation of boundaries as polyhedral elements has been studied by Favier [27] for use in a sphere-based DEM code. Their approach attempts to transform the contact problem from sphere-face to point-prism.

8.1 Contact Resolution

The procedure illustrated in 8.1 and 8.1 functions through a series of constraining checks. First, the buffered surface contact check is performed to determine whether the contact lies completely in a face. If so, the contact normal is set as the normal

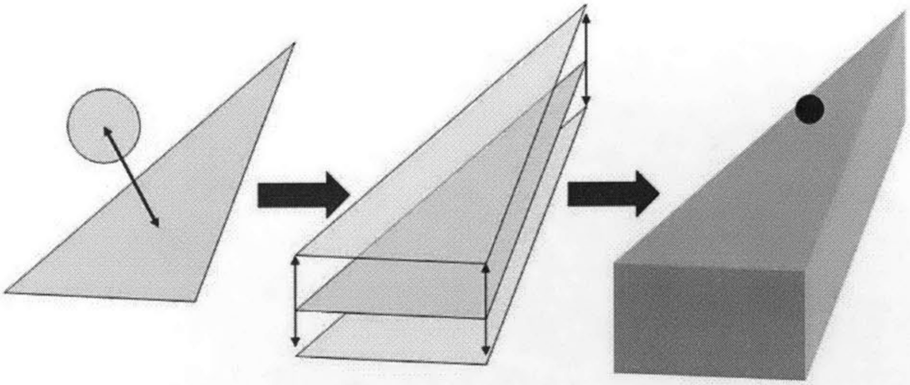


Figure 8-1: Favier pre-check approach: sphere-triangle contact (left), create buffer region based on sphere radius (center), and reduce the problem to a 3-D point to prism contact problem (right).

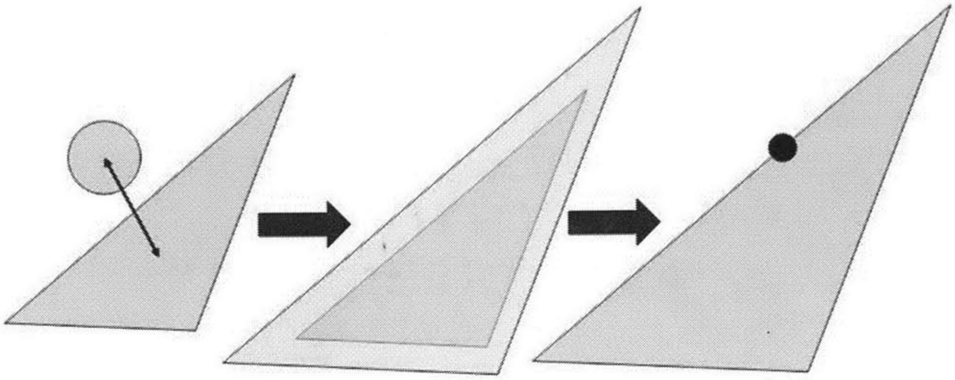


Figure 8-2: Author's pre-check approach: sphere-triangle contact (left), then create buffer region based on the intersection of the sphere with the plane containing the face (center), and finally reduce the problem to a 2-D point to triangle contact problem (right).

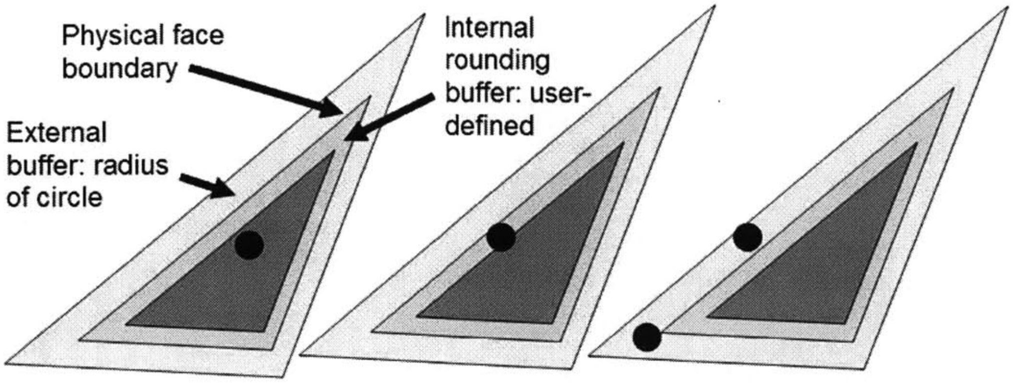

Physical face boundary
External buffer: radius of circle
Internal rounding buffer: user-defined

Figure 8-3: Contact can be divided into 3 regions: point in the face-only contact region (left), point in the computationally-rounded area (center), point in the edge- or vertex-only contact region (right).

to the face. Second, a complete surface check is performed to determine whether the contact lies in a buffered region of curvature. If so, the contact normal is set as passing orthogonal to a line segment which lies a distance of the buffer size along the angle bisector of the closest edge. Third, a check is performed to find whether contact lies at a vertex, and if so the contact normal is set as the vertex normal (typically, a weighted averaged of the normals of the faces to which the vertex belongs). Finally, an edge check is performed to determine if contact lies on an edge to the face. If so, the contact normal is set as the vector from the edge to the body's centroid. The decision tree for the algorithm is illustrated in 8.1

Energy conservation during single particle contact has been verified over multiple contact instances. Testing of the individual contact cases has shown agreement with hypothesized results

- Face contact
- Face-Edge contact
- Edge contact
- Vertex contact

From figures 8.1, 8.1, and 8.1, it can immediately be seen that the contact profiles for 3 consecutive contacts conserves momentum. Moreover, the contact profiles are

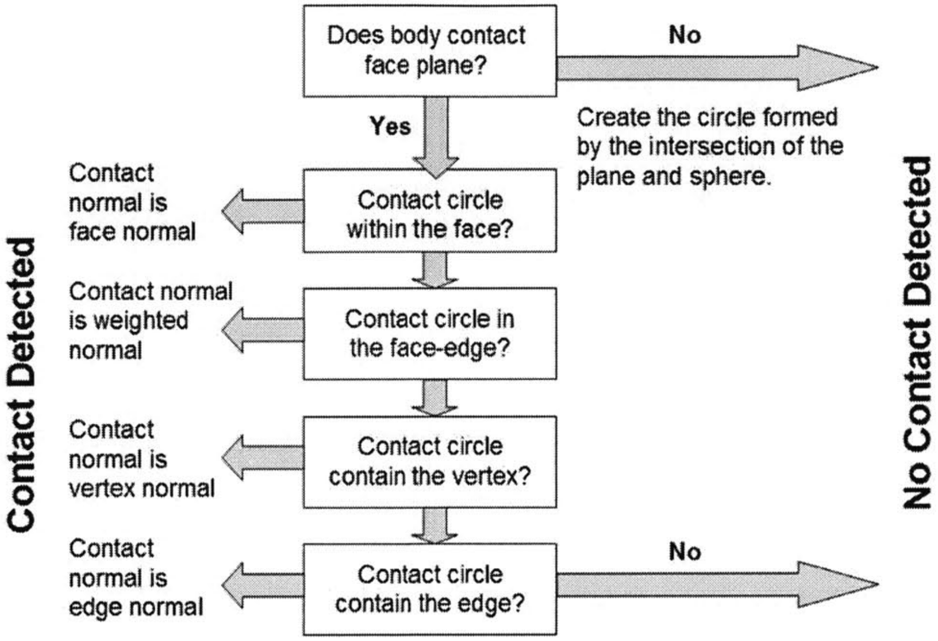

Does body contact face plane?
No
Yes
Create the circle formed by the intersection of the plane and sphere.
Contact normal is face normal
Contact circle within the face?
Contact normal is weighted normal
Contact circle in the face-edge?
Contact normal is vertex normal
Contact circle contain the vertex?
Contact normal is edge normal
Contact circle contain the edge?
No
Contact Detected
No Contact Detected

Figure 8-4: Procedure for detailed contact check between triangular polyhedron face and a sphere.

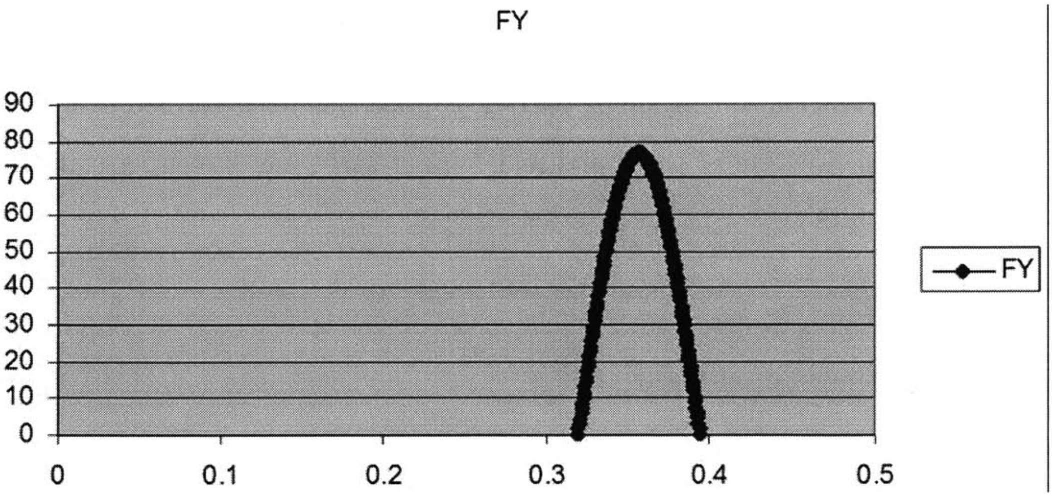


Figure 8-5: 1 of 3 consecutive contacts

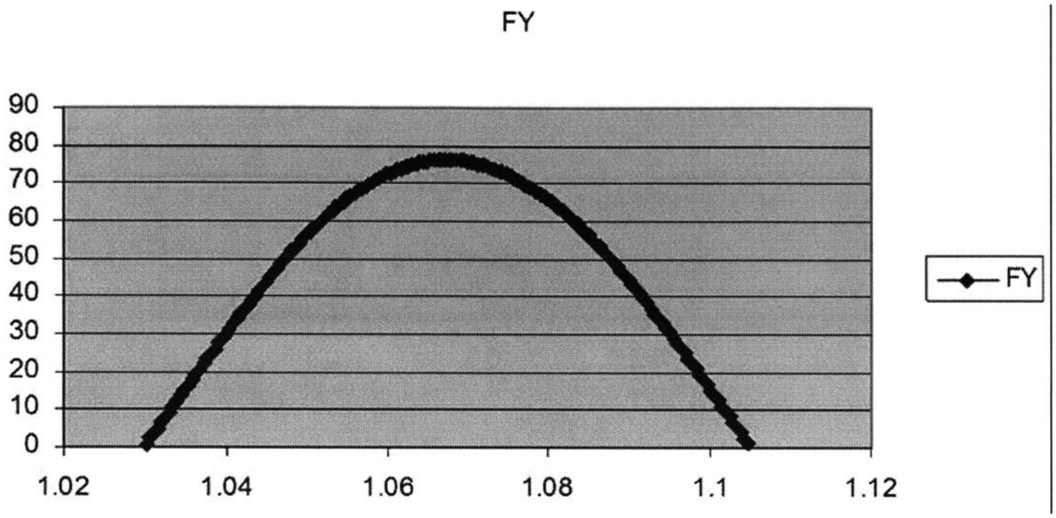


Figure 8-6: 2 of 3 consecutive contacts

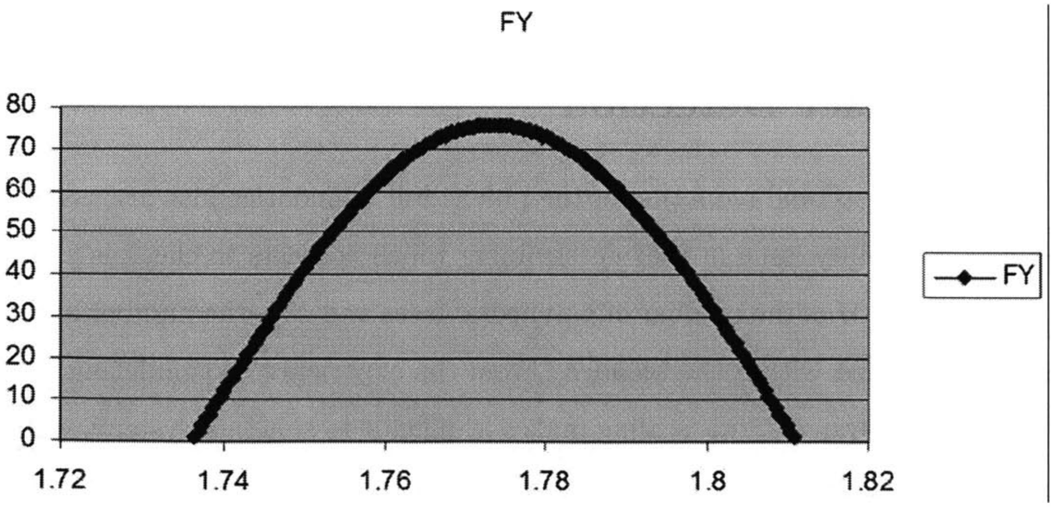


Figure 8-7: 3 of 3 consecutive contacts

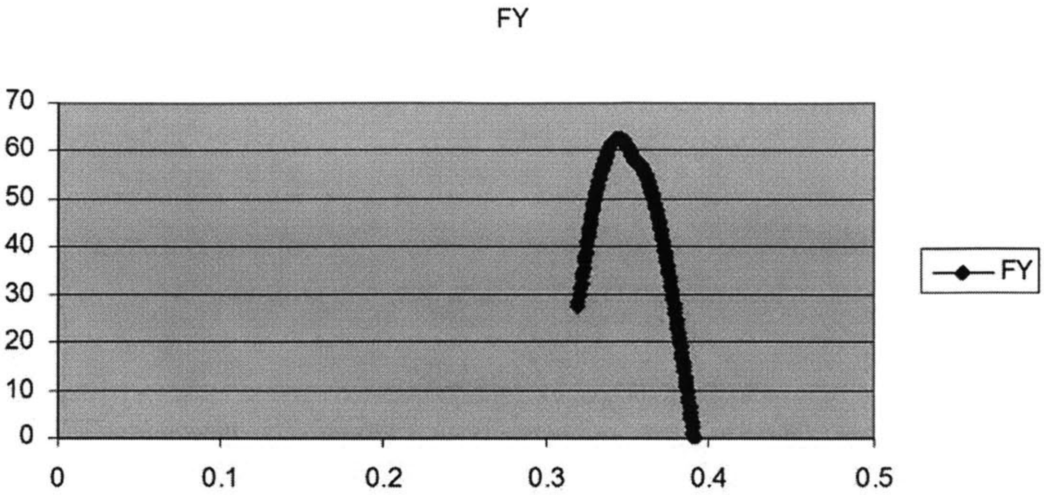


Figure 8-8: Case of contact on a face interior region with initial overlap

identical across the contacts with a max force of 78 N and a contact duration of 0.08 s.

As expected, cases 1, 2, and 3 produce identical contact force profiles. From the above discussion, it can be seen that the polyhedral contact resolution algorithm conserves momentum and energy over multiple similar contacts. Contacts on similar topology are resolved identically numerically for the case of flat surface contact.

8.2 Contact Detection

Despite the dimensional reduction of the polyhedral boundaries just discussed, there remains a scalability issue in that determining which contacts to check is a $O(MN)$ operation where M is the number of polyhedral faces and N is the number of discrete elements contained within the blender. From the experience of simulating pharmaceutical powder systems, this scaling makes it difficult to simulate systems where the number of polyhedral facets are above $O(10^2)$.

It would be preferable to have an algorithmic method of reducing the computational complexity of the contact problem. This section will introduce a method to achieve a reduced scaling to $O(M + N)$, where N is the number of discrete elements. Two common methods of reducing the complexity of intersection checking are

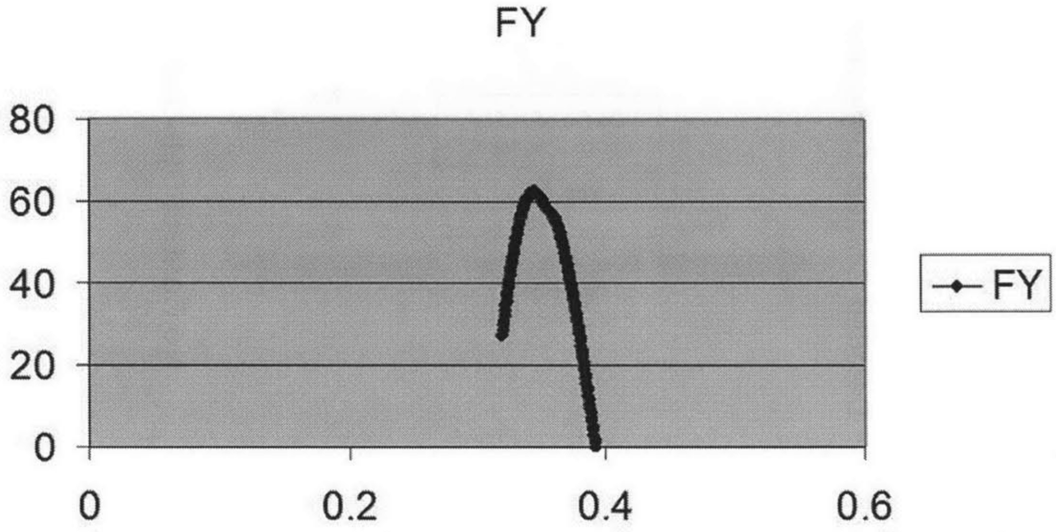


Figure 8-9: Case of contact along an edge region with initial overlap

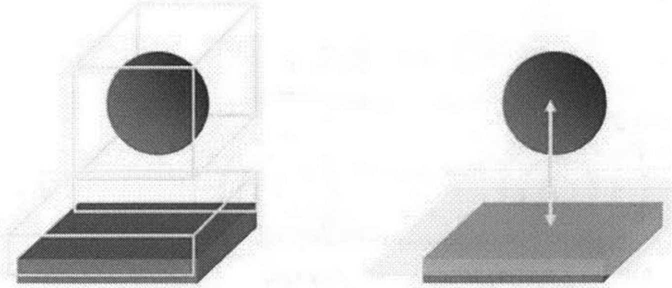


Figure 8-10: Illustration of common techniques for reducing the complexity of neighbor checks involving polyhedra.

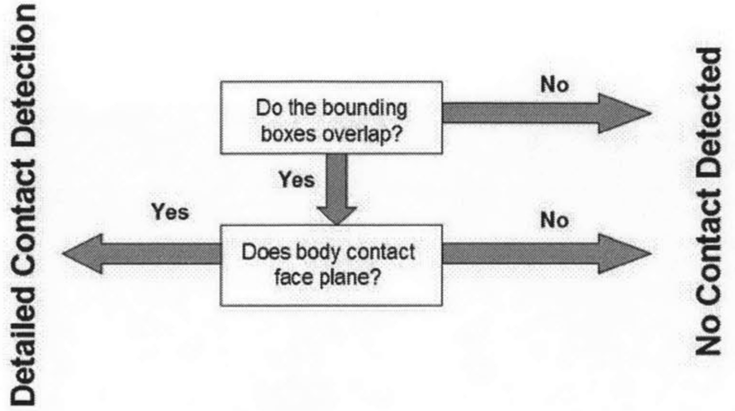

Detailed Contact Detection
Do the bounding boxes overlap?
No
Yes
Yes
Does body contact face plane?
No
No Contact Detected

Figure 8-11: Illustration of coupled technique for reducing the complexity of neighbor checks involving polyhedra.

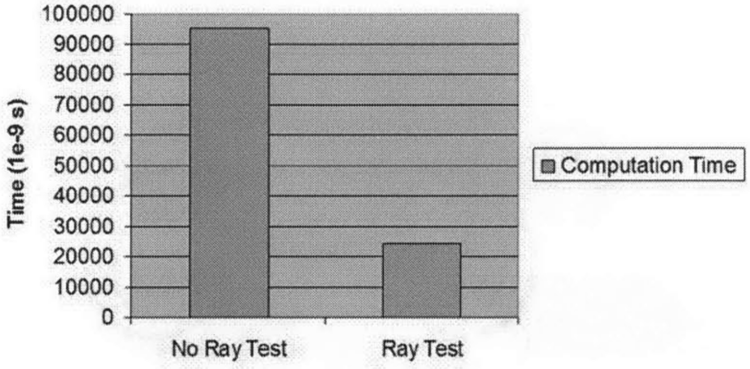


Figure 8-12: Illustration of 75% reduction in computation time by implementing a the pre-filtering strategy illustrated in Figure 8-10.

through bounding boxes and containment planes, as illustrated in Figure 8-10. By coupling these two techniques, we have the decision tree shown in Figure 8-11. The computational gains over the case of no filtering can be seen in a graph of the test problem shown in Figure 8-12.

Though these techniques help to reduce the computational intensity, the scaling remains $O(NM)$, though with a smaller constant coefficient. However, by adding in a neighbor-sorting algorithm to reduce the number of facets against which a discrete element must be checked, the desired scaling properties can be achieved.

The binning algorithm bins based on largest discrete element grain (as opposed to largest facet). Facets are binned into all bins in which they lie. This results in a greater density of voxels (and greater memory requirements); however, it reduces the possible matches between grain and boundary where detailed contact resolution can be costly. This is deemed acceptable if $N >> M$, where N is the number of discrete elements, and M is the number of faces. Bins are defined by minimum coordinate, and an object is assigned to a bin based on the centroid of its bounding box:

$$bin_i = \left(floor \left(\frac{extremum_i|_{upper} + extremum_i|_{lower}}{2} \right) - min_i \right) / bins_i \tag{8.1}$$

where $extremum_i|_{upper}$ is the upper extremum of the body along dimension i, $extremum_i|_{lower}$ is the lower extremum of the body along dimension i, min_i is the minimum coordinate for any body along dimension i, and $bins_i$ is the number of bins used to discretize the dimension i.

The algorithm uses a sparse matrix representation of the 3-D space occupied by the model. The sparse matrix representation is a regressive data structure that has been generalized to represent any N-dimensional space of bins. In common terms, the data structure is a linked list that is linked $N - 1$ times. Each list is maintained ordered such that a binary find operation can be performed on each dimension to find the requested bin on that dimension. Since a bin is defined by a tuple of 3 integers (in 3-D space), which represent bin addresses along each dimension, the failure to locate

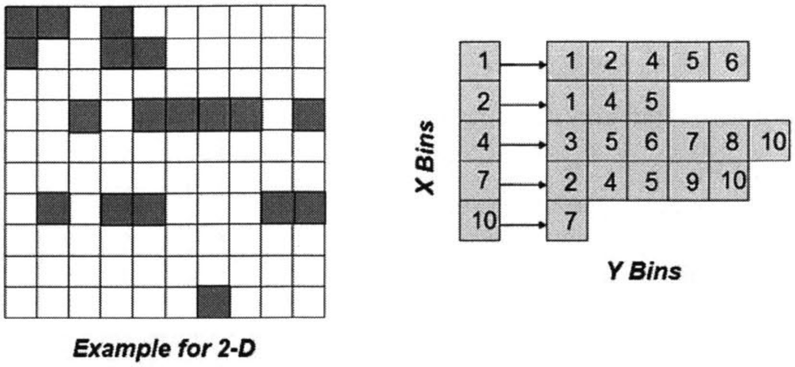


Figure 8-13: Illustration of data structure for storing the contents of bins.

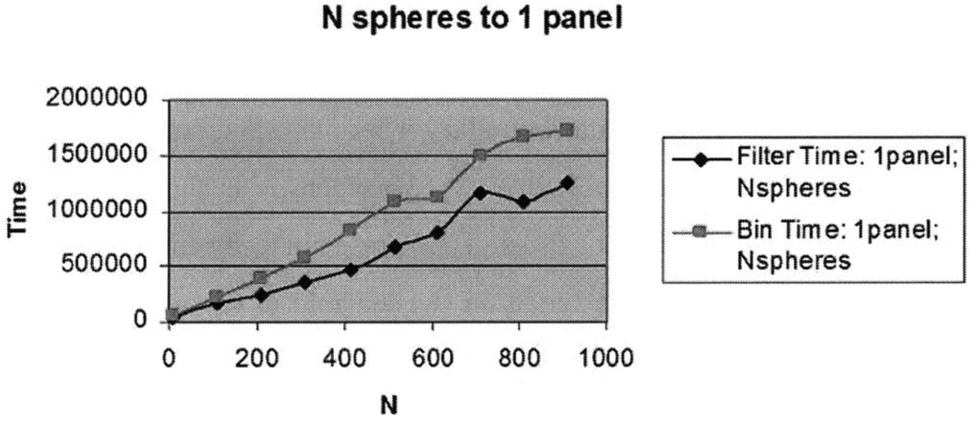


Figure 8-14: Illustration of the efficiency of the binning algorithm and simple $O(NM)$ algorithm for 1 boundary facet and N grains, where N is varied along the horizontal and computation time is represented on the vertical axis.

a portion of the bin address automatically results in an empty bin for the requested tuple and a reduction of the search time. A graphical representation of the data structure as an example in 2-D is shown in Figure 8-13.

To increase the efficiency of finding neighboring bins, index of the bin in the linked list can be queried and used to find the right and left side bin indices. If the bin indices are +/-1 of the current bin index, then the adjacent bins cannot be eliminated as candidates for neighbors.

The algorithm has been tested against two limit cases and a mean case. Figure 8-14, 8-15, and 8-16 show the difference between using filters with an $O(NM)$ algorithm

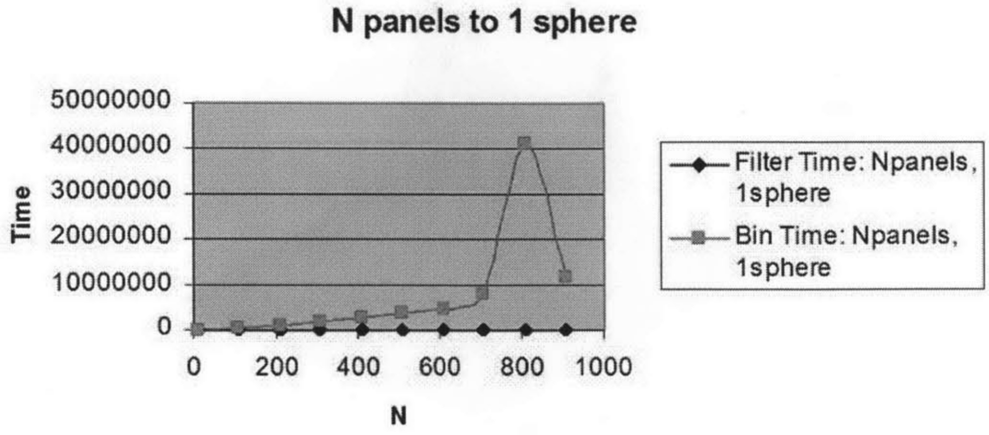


Figure 8-15: Illustration of the efficiency of the binning algorithm and simple $O(NM)$ algorithm for N boundary facets and 1 grain, where N is varied along the horizontal and computation time is represented on the vertical axis.

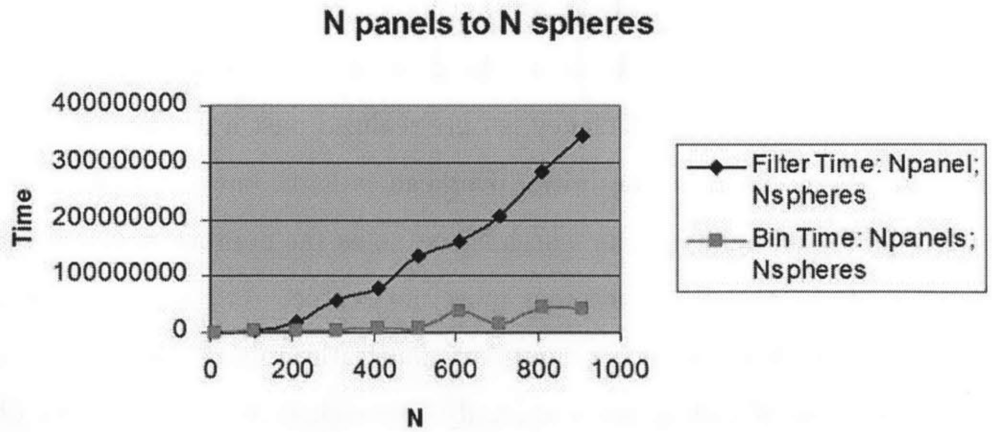


Figure 8-16: Illustration of the efficiency of the binning algorithm and simple $O(NM)$ algorithm for N boundary facets and N grains, where N is varied along the horizontal and computation time is represented on the vertical axis.

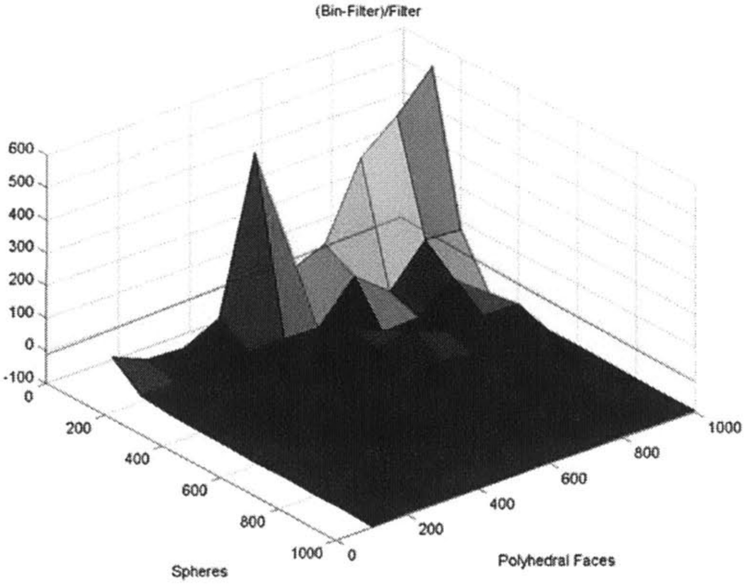


Figure 8-17: Illustration of the efficiency of the binning algorithm over a simple $O(NM)$ algorithm for y boundary facets (y-axis) and x grains (x-axis), where the height indicates (indicates time of the binning algorithm-time of the filter with $O(NM)$ algorithm)/(time of the filter with $O(NM)$ algorithm).

and using the binning algorithm just described. Figure 8-15 illustrates that it is more effective to avoid the binning algorithm as the ratio of boundary facets to grains increases. Because of the significant overhead of this approach in initially binning the facets, the computational efficiency is only realized past a certain ratio of grains to boundary facets. That is, the initial overhead is high, but the cost of lookup is low and is effective at reducing the search space, so as the number of grains increases the efficiency of the algorithm becomes more noticeable. As Figure 8-14 indicates, for instance, when there is only 1 boundary facet, the initial cost of binning and the incremental cost of lookup are marginally outweighed by the cost of an $O(NM)$ search. When the number of bodies increases at approximately the same rate as the number of boundary panels, Figure 8-16 shows that significant computational intensity can be avoided by using the binning algorithm.

Using a regular grid of data points representing numbers of polyhedral facets and numbers of grains, a contour plot can be created to find the efficiency frontier for

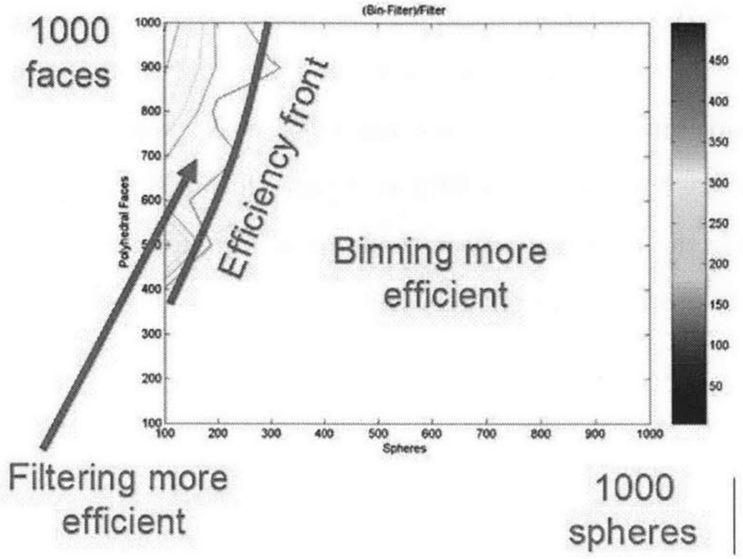


Figure 8-18: Illustration of the efficiency of the binning algorithm over a simple $O(NM)$ algorithm for y boundary facets (y-axis) and x grains (x-axis), where the contour (indicates time of the binning algorithm-time of the filter with $O(NM)$ algorithm)/(time of the filter with $O(NM)$ algorithm).

deciding whether to use the binning algorithm or whether to use a simple $O(NM)$ algorithm in combination with a prefilter (e.g., bounding box or containing plane). These are shown in Figure 8-17 and 8-18.

8.3 Contributions

Two contributions to the representation of polyhedral boundaries have been offered here.

First, this thesis provides less computationally intensive method of determining detailed contact between discrete elements and polyhedral boundary facets with a specific formulation for sphere primitives. This contact resolution algorithm offers a theoretical basis for interpolating C^1 continuity of contact with a C^0 continuous boundary and provides numerical validation.

Second, a contact detection algorithm is offered for boundary elements that reduces the computational complexity for $O(NM)$ to $O(M + N)$ through a binning algorithm, where N is the number of discrete elements and M is the number of polyhedral facets in the boundary representation. The contact detection algorithm also offers further optimizations to reduce the $O(M + N)$ scaling coefficient through a streamlined binary space division data structure that offers quick neighbor checks to bins neighboring the candidate discrete element. Scaling graphs have been given to demonstrate the properties of the algorithm.

Chapter 9

Application: Modeling Blending of Pharmaceutical Powders

Pharmaceutical powder manufacturing is a key component in the $200+ billion pharmaceutical industry. The production of pharmaceuticals is a highly profitable business; however, the profitability of most products developed is highly sensitive to time and supply. Both of these variables are impacted by the quality and productivity pharmaceutical manufacturing facilities. Because of rigid regulation set in place by the United State Food and Drug Administration (USFDA), pharmaceuticals sold in the United States or by companies chartered in the United States must pass a series of quality control regulations before they can be released to the public for sale.

The loss from batches that fail to pass quality control typically ranges from 10-50% per product based on the maturity of the product (higher yields for products that have been in manufacture longer) as well as nature of the material (certain products have proven difficult to work with) and unidentified failures along the process line (recurrent operator caused failures, etc.). These ranges are typically proprietary knowledge to the pharmaceutical companies and are offered here as anecdotal evidence gathered from informal meeting with pharmaceutical company researchers.

Coupled with this high loss rate is a newer threat to production offered by the gradual phasing in of the USFDA 2001 CGMP regulations. This new initiative aims to push the pharmaceutical manufacturing industry (among others) to be able to not

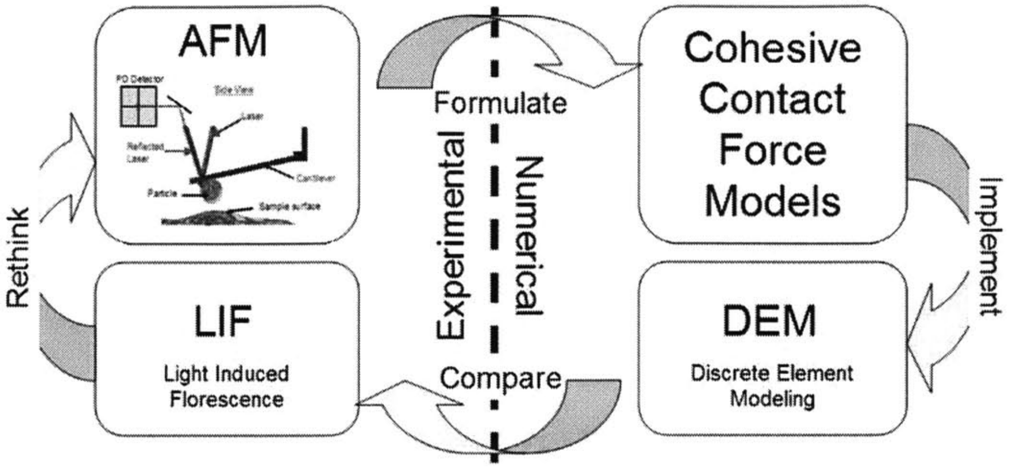


Figure 9-1: Illustration of the coupled numerical/experimental approach adopted here for refining and validating numerical models and DEM.

only detect failed batches but to also state why the batches failed and how that will be prevented in the future. This approach is commonly known as process improvement. The penalties for being unable to satisfactorily identify a process failure and offer a plan of remediation can include fines as well as a shut-down of production.

Lost revenues from disposed product and an inactive line as well as the cost of fines are imminent threats to the profitability of many pharmaceutical products. Solving process problems can offer immediate and significant profit gains for pharmaceutical manufacturers. A particular problem faced is in the homogeneous mixing of powders consisting of active pharmaceutical ingredients (API's) with inactive ingredients (excipients). Because of the dearth of basic understanding of the behavior of granular materials, tracing process failures and offering remediation plans in powder processing is often difficult. This chapter focuses on a particular process step of the powder manufacturing process where powders are blended.

A coupled numerical and experimental approach has been implemented between the author and researchers in Prof. Charles Cooney's laboratory affiliated with the Department of Chemical Engineering at MIT. The approach has been referred to in related theses by Domike [24] and Ngai [88]. This approach bridges micro-mechanical experiments with models of micro-mechanical behavior. These models are then implemented in the DEM3D framework and simulations are run. The results of the simulations can then be compared with experiment to determine the accuracy of the numerical approach in determining meso- and macro-scale material behavior. The process is illustrated in Figure 9-1.

Previous numerical studies have compared blending kinetics in a mixer from magnetic resonance imaging (MRI) [87] and positron emission particle tracking (PEPT) [112] data with discrete element models. Hopper kinetics for pharmaceuticals have also been performed, comparing experimental results against DEM results [7]. This section concentrates on the modeling of granular mixing in a "v" and "y" type bench-scale blender using the discrete element method with a new stochastic version of the classic JKR model of cohesion and polyhedral boundary conditions as discussed in the previous chapters.

9.1 Experimental Methods

Two types of experiments were conducted on the powder systems to determine the properties at the macro-scale and the micro-scale. To determine the properties of pair-wise interactions in both normal and shear directions, atomic force microscopy (AFM) was used. The AFM measurements determine a matrix of cohesive pull-off forces and inter-particle friction coefficients. In the macroscopic system, the homogeneity of the sample was measured using a light induced fluorescence (LIF) technique developed in our collaborators' laboratory [88, 58, 59, 60]. Together, the two techniques offer experimental measurements at both the micro-scale and macro-scale and provide validation data for the upscaling of inter-particle interactions to powder system behavior via discrete element modeling.

9.1.1 Atomic Force Microscopy

Atomic force microscopy (AFM) is a technique for measuring the forces that develop between surfaces at the micron scale order. For this particular research, AFM has been used to measure the friction, cohesion, and adhesion forces for pharmaceutical

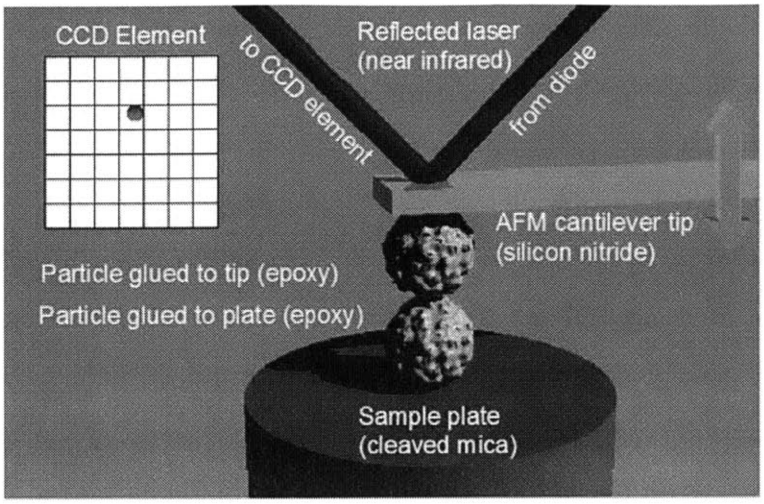


Figure 9-2: Model of the atomic force microscope experiment setup

powders. Because AFM can resolve weak forces in the nN range, the technique is ideal for the studies detailed here. These measurements are then used to calibrate force-displacement models used in the discrete element model to represent inter-particle attraction/repulsion. Similar work in characterizing powders has been performed [16, 49, 48, 54, 93] as has research in adapting AFM data for use in DEM [67].

The AFM apparatus is arranged as shown in Figure 9-2. A laser beam is sent from a light-emitting diode toward the microscopically flat surface of the AFM tip. This beam is reflected off of the Silicone Nitride surface to light-sensing CCD diode. The relative position of the beam on the sensor is determined from the relative magnitude of light sensed on individual matrix elements and resolved into a single point, which is again converted into a deflection of the AFM tip. Realizing that the force-displacement relationship is non-Hookean, an effective interpolation of the force can be acquired.

In a typical cohesive/adhesive force experiment, the particle adhered to the tip is slowly brought into contact with the other particle, which is adhered to a cleaved mica base. This can be determined by a deflection of the AFM tip. The tip is then slowly raised until the force on the tip is relieved (i.e., the cohesive/adhesive pull-off force is reached).

9.1.2 Light-Induced Fluorescence

The mesoscopic experimental validation of the numerical models is performed using a bench-scale blender and a sensor for determining the powder composition of the boundary layer of pharmaceutical powder grains at the apex of the bench-scale blender. This sensor is based on light-induced fluorescence (LIF) technology developed at MIT.

Light-induced fluorescence [59, 60, 58] exploits the fluorescing property of pharmaceutical ingredients when exposed to a light source in the ultraviolet band. The illuminated material is excited into a higher energy state and releases photons of a higher wavelength when no longer illuminated, resulting in a strong signal over several wavelengths corresponding to the fluorescing of the different constituent materials. By filtering this signal, the strength of different components can be isolated. Because the material fluoresced in restricted to a finite window of material, the signal strength is directly related to the amount of a particular material in the sampled window. This fluoresced light is captured by a light sensor, which converts the signal into a measure of the homogeneity of the pharmaceutical product.

LIF is used in this study to determine the homogeneity of a mixed pharmaceutical material during each rotation of the material in a variety of bench-scale powder mixers. Fluorescent properties of lactose, microcrystalline cellulose, and caffeine have been determined and are used to determine the homogeneity. Each bench-scale blender is composed of stainless steel with a polycarbonate window to allow for LIF measurements. Two types of blenders are used: a simple cylindrical blender with no baffles and a V-mixer.

9.2 Simulation

This section details the numerical experiments performed to determine the blending kinetics of pharmaceutical blenders. Both 2-D and 3-D studies have been performed and validated for this approach.

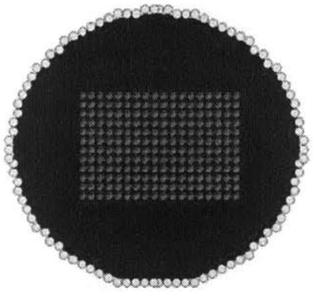


Figure 9-3: Numerical experiment setup for simulating blending experiments

9.2.1 2-D Modeling

The first part of the strategy required validation of DEM as a technique for investigating particle blending kinetics. Blending experiments using two different mixtures: (1) lactose and caffeine and (2) MCC and caffeine were conducted using the LIF technique, which was discussed in the previous section. Lactose and MCC are inactive ingredients in a pharmaceutical blend and will be referred to generically as excipients. Conversely, the caffeine particles are the active ingredient and will be referred to generically by the acronym for active pharmaceutical ingredient (API).

Using the MIMES framework described in [101], a system of 3280 lactose and MCC particles are added to a circular blender comprised of a rigid body assembly of constituent particles about the circular blender circumference, as illustrated in Figure 9-3. This rigid body assembly of boundary particles is comprised of $40\mu m$ disk-shaped particles, yielding an effective friction to the blender surface.

MCC (microcrystalline cellulose) particles are modeled as a polydisperse system of disk-shaped elements with specific density of 1.78. The radii of these particles have a mean of $75\mu m$ and follow a log normal distribution. Lactose and caffeine particles are also modeled as poly-disperse disk-shaped elements with specific densities of 1.27 and 1.23, respectively, and mean radii of $70\mu m$ and $50\mu m$ respectively, both following a log-normal size distribution.

Parameter studies of stiffness and friction helped to constrain the choices for these parameters in the 2-D simulations. From parameter sensitivity studies of similar systems [77, 57], stiffness is not explicitly considered and has been selected to maximize the time step while not allowing particles to interpenetrate more than 10% of their respective radii, as recommended by Walton [123]. Interpenetration constraints are necessary to satisfy the assumptions of local deformation implicit in the DEM formulation. This results in a constant stiffness of $2.4e4N/m$ across all particles. For friction coefficient, [24] determined that friction coefficients in excess of 0.2 did not affect the flow properties of the material for disk-shaped simulations, as is corroborated by Walton in his discussion of the limitations of circular elements in DEM [124], and a constant friction coefficient of 0.3 is applied to all materials in the simulation.

A cohesive model of particle interaction was formulated using simple point-to-point spring bond elements. The average cohesive force derived from experiments detailed in [88] was used to determine the yield strength of the bond as shown in Figure 9-4. Though AFM cohesion measurements found a log-normal distribution of cohesive pull-off forces for each pair of interacting particles and the interaction forces between excipient, API, and API-excipient are different, the pull-off force is modeled as a constant, using the average lactose-lactose pull-off force of 90nN and the average MCC-MCC pull-off force of 280nN to characterize all cohesive interactions within the lactose-caffeine and MCC-caffeine simulations, respectively. This choice is somewhat valid, considering the systems investigated are 90% excipient content. Extrapolating from the insensitivity of the models to contact stiffness, the same assumption is used in determining the cohesive force-displacement law, with the linearized stiffness set to the stiffness of the individual particles - $24,000N/m$.

Simulations were monitored using a similar procedure to that used in the actual LIF experiments. That is, a small "window" was used to filter the DEM particles at time steps corresponding to the acquisition frequency of the LIF sensor (1 acquisition per second). Using this filter and time step, the summation of the resultant caffeine particle masses was divided by the summation of the total mass of the resultant particles and compared to the corresponding LIF measurements in the actual experiment.

The parameter space was also expanded to compare simulation results with LIF experiments over a range of blender rotation rates: 10, 20, 30 RPM.

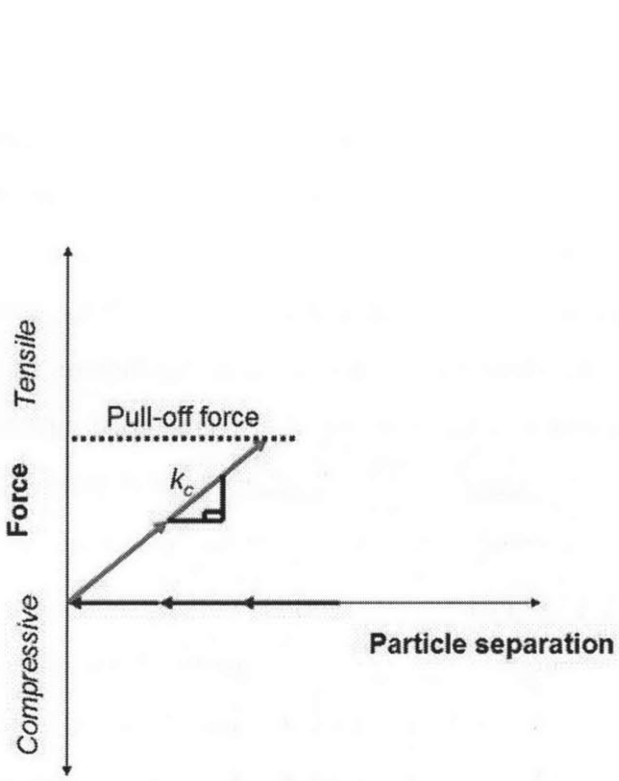


Figure 9-4: Cohesive contact law implemented for 2-D circular blender studies

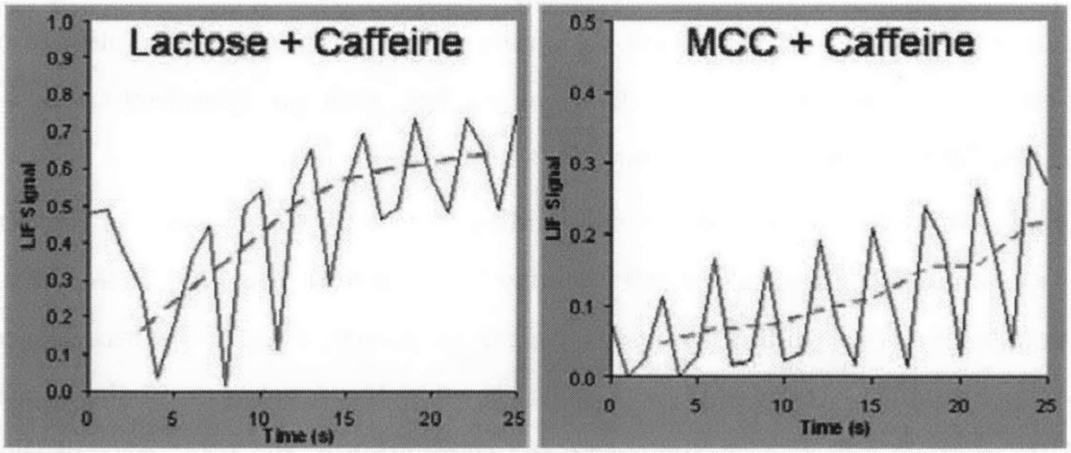


Figure 9-5: Figure from [88] showing the simulated LIF signals for lactose-caffeine (left) and MCC-caffeine (right) simulations

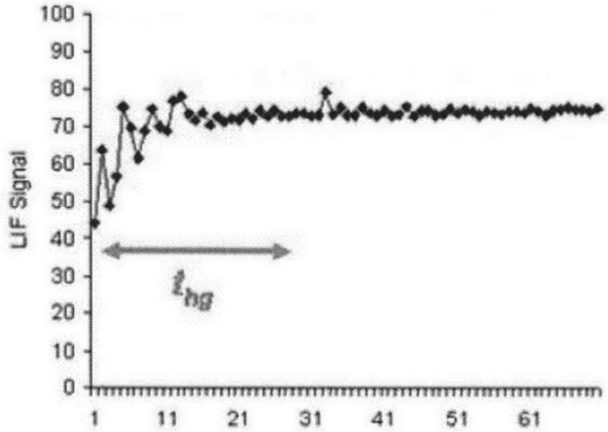


Figure 9-6: Experimental value of LIF signal (figure from [88])

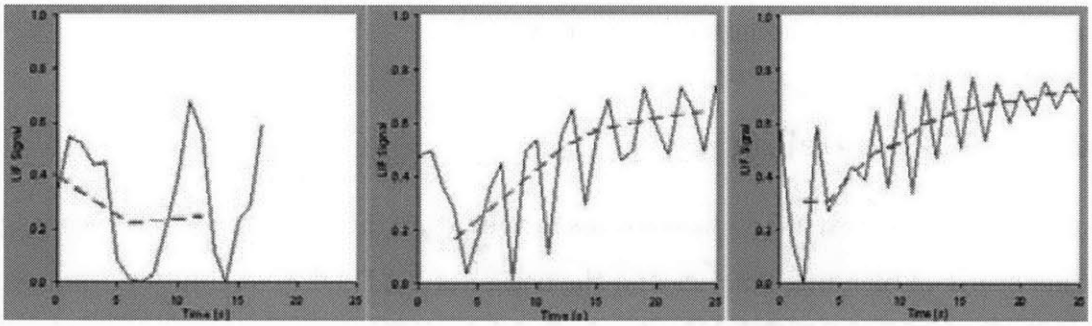


Figure 9-7: Simulated LIF signal for lactose-caffeine blending at 10, 20, 30 RPM (from [88])

The assumptions of 2-D planar motion, simple cohesion, and disk-shape were shown to prevent the method from achieving comparable quantitative results; however, there was good qualitative agreement between simulation and experiment.

As expected from experiment, the lactose-caffeine powder system required a shorter period than the MCC-caffeine system (70 s versus 79 s) to achieve a steady-state homogeneity. Also, the periodic trend for the simulation shown in Figure 9-5 is qualitatively similar to that of the experiments, as shown in Figure 9-6 with decreasing amplitude of homogeneity variation about a monotonically increasing slope as time progresses. The red dotted slope shown in Figure 9-5 represents a moving average of the homogeneity to illustrate the increasing percentage of API in the recorded window as the API diffuses into the static excipient core. This phenomenon is more pronounced in the MCC simulations, which also correlates well with the observations of the experimental system and agrees with theory, which would assume that the more cohesive MCC matrix would require more time for API particles to permeate. From visualizations of the powder flow with colors discriminating excipient from API, the avalanching action of the cylindrical blender can be discerned as the cause of the periodic signal. As the avalanching action leads to diffusion of the API into the rigid body rotation core composed mostly of excipient particles, the homogeneity signal begins to stabilize. Figure 9-7 illustrates that as rotation rate is increased, the time to achieve steady state homogeneity is decreased.

9.2.2 3-D Modeling

The success of the coarse 2-D simulations in validating the DEM approach encouraged further study of blending kinetics and the design of a 3-D DEM simulation framework to capture not only a 3-D spatial representation of particles but to also increase the fidelity of the cohesion model and to implement the highly irregular boundary conditions extant in most production blenders, especially the ubiquitous V-mixer and the double helical ribbon blender under development and testing at MIT.

Using particle properties for glass beads, a simulation has been carried out to validate the use of arbitrary boundary conditions in the DEM simulation. As shown

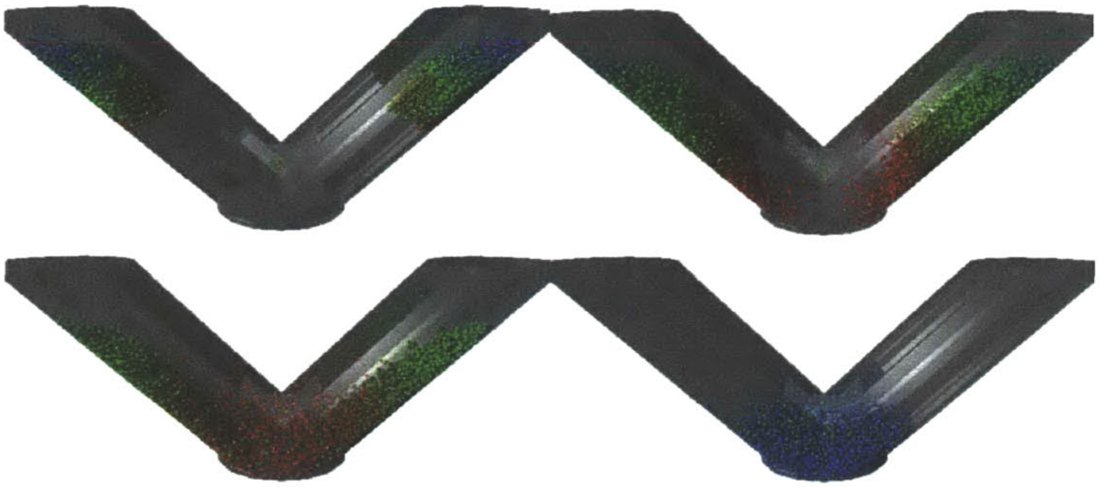


Figure 9-8: Illustration of the validation of the arbitrary boundary conditions for a V-mixer

in Figure 9-8, particles are allowed to flow down the blender walls and settle at the apex of the V-mixer. The particles are modeled as dry granular matter, and cohesion is neglected.

With a reasonable assurance of appropriate behavior of the simulation, full-scale tests have been implemented using an independently validated cohesion law. The cohesive model is based on a cohesive surface energy matrix designed to add stochastic variability to the standard JKR model of adhesion [44], as repeated here from the earlier section on cohesion:

$$\gamma(X) = \frac{2F(R_1 + R_2)}{3\pi R_1 R_2} \tag{9.1}$$

A matrix of values for the mean and standard deviation of the surface energy was extracted from experimental data. The surface energy, instead of the pull-off force, was used in order to remove the size dependence of the cohesive force. A similar procedure is used to extract the mean and standard deviation for the friction coefficient from experimental data. These matrices are used during the simulation to determine the cohesion force and Coulombic friction force. Coulombic friction is implemented with a cumulative shear formulation to accurately capture the stick-slip phenomenon.

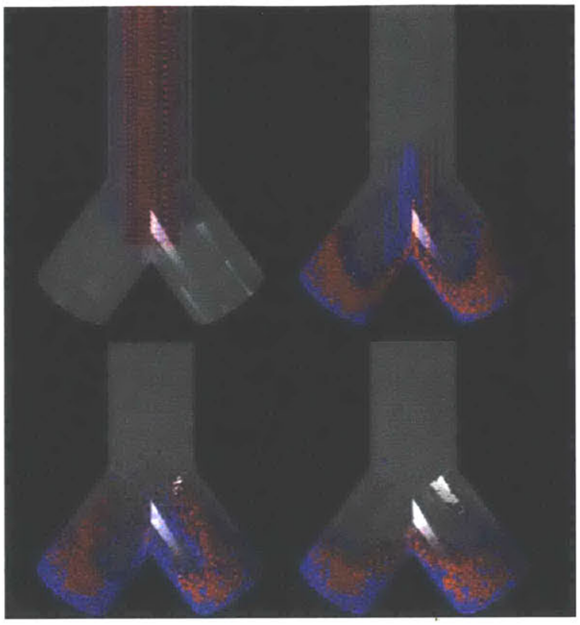


Figure 9-9: Deposition of cohesive powders into the Y-mixer colored by material with red as the API

Particles are deposited via the extruded end of a Y-mixer oriented perpendicular to the axis of rotation as shown in Figure 9-9 and 9-10. Initially, the particles are unmixed with the API in a layer in the plane of the illustration closest to the camera with the excipients along the back of this layer. As noted from experiment, the settling of the powder exhibits the characteristic curling flow as it reflects from the inner sides of the Y and dissipates energy quickly.

Using the visualization capabilities discussed in the architecture, particles are filtered by material type and only API particles rendered. As can be seen in Figure 9-11, the particles remain relatively unmixed in the static core of the system with the majority of permeation of API particles at the free surfaces of the system (surfaces unconstrained by the mixer wall). Diffusion occurs parallel to the axis of rotation, catalyzed by the interaction of the system with the blender wall during the separation phase.

The system was run for 3 complete rotations to study the evolution of the homogeneity of the system over time. The system exhibited flow characteristics observed in experiment such as sudden slump of the granular matter under gravity, as shown

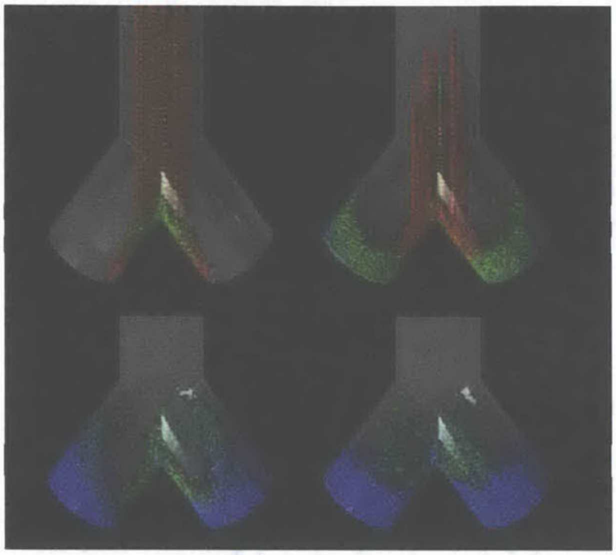


Figure 9-10: Deposition of cohesive powders into the Y-mixer colored by velocity with red indicating higher magnitude

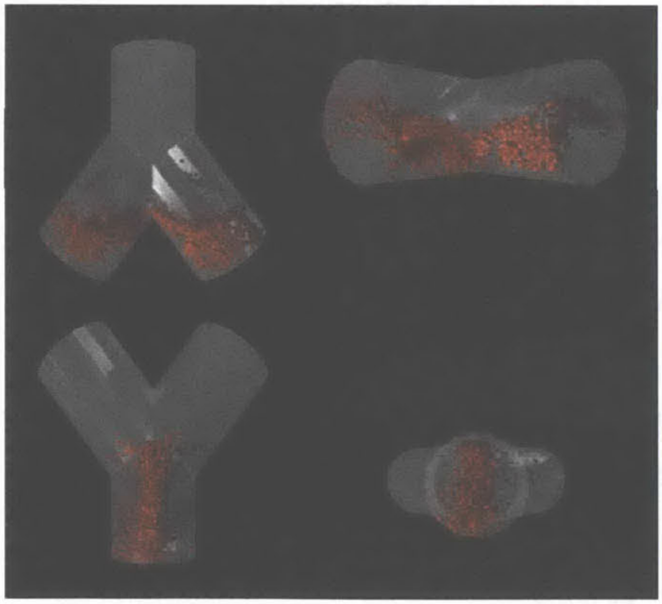


Figure 9-11: Rotation of the Y-blender with only API particles rendered

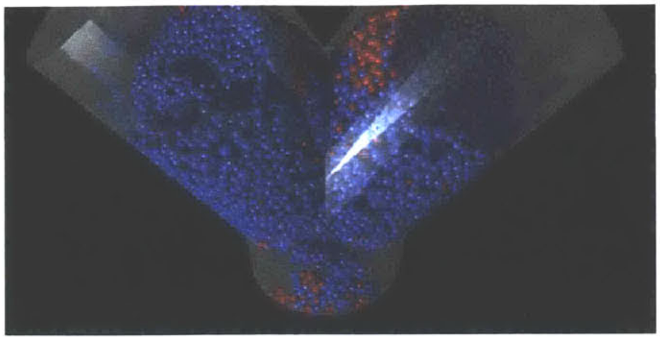


Figure 9-12: Sudden rearrangement of the cohesion matrix under gravity

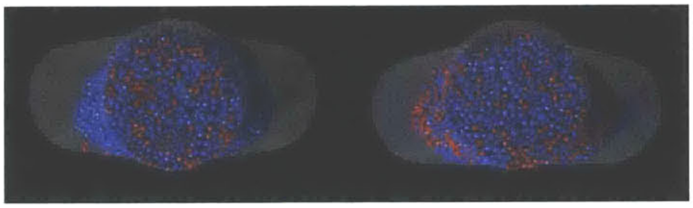


Figure 9-13: API homogeneity/content at the measurement window for the 3D Y-mixer for rotation 1 (left) and rotation 2 (right)

in Figure 9-12.

Observing the system across revolutions, the API content at the measurement position (facing the mixer axis that is perpendicular to the axis of rotation) is observed to increase in time as the free surface composition changes during the separation and aggregation phases of the mixing cycle. The average decrease in the number of same material neighbors for particles can be discerned between the two rendered images.

Diffusion of API particles away from the core results in greater mixing throughout the blend with de-aggregation occurring faster at the free surfaces. The diffusion of API particles away from the core does not act to separate API particles from cohesive clusters as shown in Figure 9-14 and Figure 9-15.

9.3 Contributions

Using the DEM framework described in the beginning of this thesis, we have been able to successfully validate the 2-D DEM formulation using a simple point-to-point

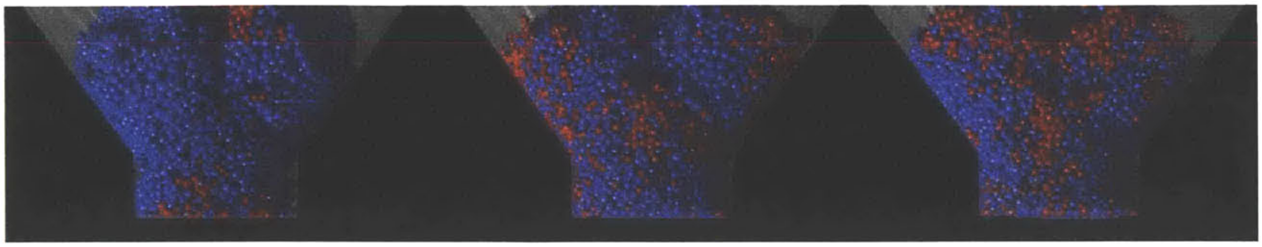


Figure 9-14: Diffusion of API particles over time from the core of the system

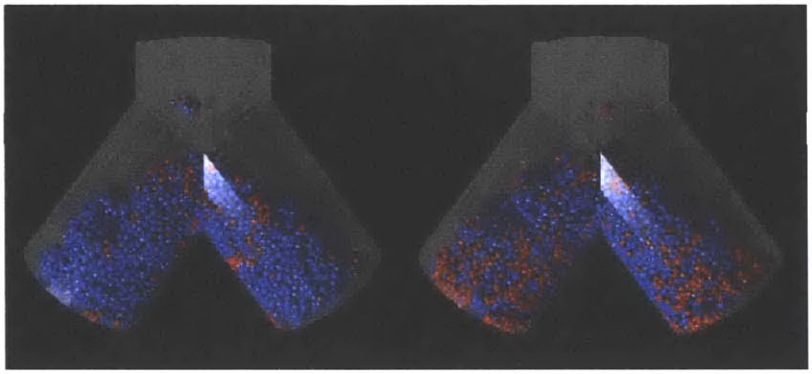


Figure 9-15: Diffusion of API particles over time from the core of the system

beam element with no shear or moment capacity. Building on the 2-D formulation a new cohesion model has been proposed to capture the stochastic nature of cohesive and frictional interactions between powder particles. This model has qualitatively captured the behavior of experimentally observed granular mixes.

Chapter 10

Conclusion

The thesis has first set out both a framework and design for a DEM simulator in Chapter 3, in combination with an abstraction framework based around SQL Server for database storage and analysis (as shown in Appendix A). Visualization is built into the application via a wrapper for POV-Ray application for visualization and Matlab for additional analysis and visualization as described in Sections 3.4 and 3.6. However, this is not a thesis primarily about aggregating commercial off-the-shelf technology, it is about using this technology in a way that elucidates the understanding of granular materials in new and novel ways with specific application to investigating the resolution of grain-scale interactions.

Specifically, this thesis has contributed to the representation of shape and surface properties and the effects of this representation on granular system behavior. The contributions lie in five main areas: computational geometry, cohesion modeling, quadrature, DEM data analysis, and understanding of the phenomenon of stress dips forming in the center of granular heaps.

1. In computational geometry, this thesis offers a new contact resolution algorithm for general triaxial ellipsoids in Section 4.3, which performs an order of magnitude faster in resolving contact than the previous algorithm, as shown in Section 4.4. This thesis develops a multi-scale representation of a particle surface in Chapter 7 with applications shown in resolving both surface-varying cohesive

interactions in Section 7.3 as well as micro-asperity geometric interlocking in Section 7.2. This thesis extends the use of arbitrary polyhedral boundary conditions in Chapter 8 through the development of a contact resolution algorithm in Section 8.1 and a contact detection algorithm in Section 8.2.

2. This thesis has developed a method for modeling cohesive contact between micron scale powders. By defining a lognormally distributed cohesion parameter across the surface of a particle, good agreement has been shown in comparison to experimental results using AFM pull-off tests, as shown in Section 7.3.

3. A new quadrature algorithm for accurately resolving rotational motion is developed in Chapter 5 and shown to exhibit superior accuracy and performance to existing methods for resolving finite rotational motion in Section 5.6.1.

4. This thesis has introduces a data analysis framework for sifting through the data produced by a DEM simulation in Appendix A, including performance statistics and contact analysis.

5. Through the analysis of DEM data in a DBMS, a correlative relationship between average stress in successive annuli centered about the heap center and the average age (residence) of the contacts in that region has been uncovered, as discussed in Chapter 6. This relationship is interpreted as an indication of stress fabric damage at the center of the heap during construction as a mechanism for the development of a local stress minimum at the heap center.

In combination, these contributions have enabled a better understanding of granular materials through the lens of discrete element modeling by providing enhanced resolution of particle geometry and improved data analysis capabilities. The effectiveness of these contributions are illustrated through the studies of formation of stress dips in granular heaps and in the numerical experiments simulating powder blending.

Chapter 11

Future Work

One of the largest sources of excitement in a PhD is to see how one's research opens doors to whole new avenues of research. For instance, both the analysis of heap formation and stress network formation in general as well as the modeling of pharmaceutical blending are relatively immature research fields and could benefit from this type of research. Some of the specific extensions are:

- Application of the multiscale, multiphysics surface mapping algorithm to numerical simulation of powders

- Coupling of micromechanical laws of particle interactions currently being researched into the DEM framework to increase the fidelity of the approach

- Parallelization of the code for extension to the solution of larger problems or faster solution of current scale problems

- Extension of the visualization capabilities to provide new illustrations of granular behavior as well as more detailed animations of real systems.

- Investigation of new ways of analyzing data from DEM simulations to extract important mechanisms for different particle behaviors.

- Further work on modeling continuous blending production of pharmaceuticals in a double helical ribbon blender

- Sensitivity analysis of the simulations to different parameters, such as particle shape, contact law, etc. and refining these findings for applications to particular types of systems. For instance, blended systems are relatively insensitive to material stiffness; however, stress wave propagation experiments are controlled by the material stiffness and contact damping models.

- Research into the validity of downscaling the bench-scale systems to smaller systems of particles and answering the question of how few particles are required to capture the essential behavior in blended powder systems

- Analysis of other blended aggregate systems, such as concrete, where the blending kinetics are different due to drastically different force scales and particle distributions compared with pharmaceutical powder systems

- Finishing the validation of the smoothed particle hydrodynamics (SPH) component of the DEM framework to allow for computational fluid dynamics calculations to provide coupled fluid-solid flow

- Implementation of a Lattice-Boltzmann cell-based integration algorithm for coupling fluid-flow and solid-fluid interaction into the model

An issue with the use of light induced fluorescence for studying blended systems is that resolution is restricted to a few microns into the window of a bench-scale blender, so only the wall layer of the powder is measured for homogeneity. Using DEM, it would be a valuable study to show whether the window layer is a characteristic representation of the whole system in terms of homogeneity or whether wall interactions reduce the usefulness of this measurement.

As demonstrated by this thesis and [88], the possibilities for using DEM in understanding blending kinetics and extending this understanding to the design and testing of blender designs before the production of the blender itself could be of enormous value to the pharmaceutical manufacturing industry.Eventually in pharmaceutical blender modeling, the goal is to be able not only to predict the behavior inside an existing blender and determine the parameters that affect the effective mixing of different types of powders but to also provide a design tool for effectively testing and validating new blender designs before they go into production. The continuous blender mentioned above is an example of this; validation and analysis of the blending kinetics of this blender design numerically could provide a degree of pre-production assurance that the blender will provide sufficient homogeneity.

Upscaling is another issue in discrete element modeling. In performing numerical simulations to reproduce the behavior of powders in a bench-scale blender, it is not currently possible to simulate every grain. Instead, it is assumed that the blending kinetics of a scaled blender will approximate those of the bench-scale blender, as, similarly, the kinetics in a bench-scale blender are assumed to approximate those in the production size blender. The validity of this assumption is currently untested, and it is necessary to identify the proper number of particle necessary to capture the characteristics of interest in the production scale system.

An existing and unanswered question is how do we know when we have achieved enough resolution (i.e., enough parameters are being modeled) to accurately capture a particular system's behavior? What weight should be given to each parameter? Extension of the work on granular heaps to a parameter study of the effects of different variables on the behavior of the system would possibly shed new light on the important variables in emergent behavior in these systems as well as possibly offering insight into a more general method for determining a priori the important parameters to consider. As an avenue for exploring this field, extension of the studies on granular heaps could be extended to provide a deeper micro-mechanistic view on the stress state of these systems as the evolve under different conditions.

Finally, as touched on in the section on data, storage and analysis of data can yield surprising and interesting results. DEM simulations, unlike experiments, have the capacity to give us a perfect knowledge of the states of each grain at a microscopic temporal resolution. This is currently not being exploited in many studies with focus instead on simulating ever greater numbers of particles. Data analysis considering both temporal and spatial components concurrently can be executed on a number of problems and should be investigated in greater detail.

Though much research has been performed to elucidate the mechanisms involved in granular behavior in fluids, it is still an open question as to how particle geometry affects the flow of fluid through granular media. Coupling the Lattice-Boltzmann algorithm for resolving fluid flow and solid-fluid interaction could allow us to investigate the role of particle geometry and other parameters on the behavior of grains in fluids, as, for instance, in the problem of modeling sand production in the near well-bore region of oil wells.

Appendix A

Data Management

Efficient and fast computation of a simulation is only half of the effort involved in a successful numerical modeling approach. Often overlooked is the importance of analyzing the data resultant from an analysis. For particle methods, this is an important consideration. As simulations involve more particles over more timesteps, the amount of data generated from the typical simulation increases accordingly. For illustration, a 10^6 particle system simulated over 10^6 timesteps produces $1.2e13$ bits, or 1.5 terabytes, of location data if assuming 3-D spatial coordinates stored using 4-bit floating point precision. Storing and analyzing this magnitude of data is non-trivial. It is also often necessary to query a variety of parameters concurrently, making it difficult to logically divide data into separate data repositories (e.g., by particle or by time slice).

This chapter provides a method for piping data from a DEM simulation into a commercial off-the-shelf (COTS) database system coupled with a framework for filtering, rolling up, and visualizing phenomena of interest.

A.1 Data Archiving

One of the key principles of development adopted for DEM3D, as stated in the previous chapter, is to provide a framework that is agile to change. One way of providing less sensitivity to change is through well-defined functionality both vertically (in coherent class hierarchy) and horizontally (in design of interfaces). In the data storage

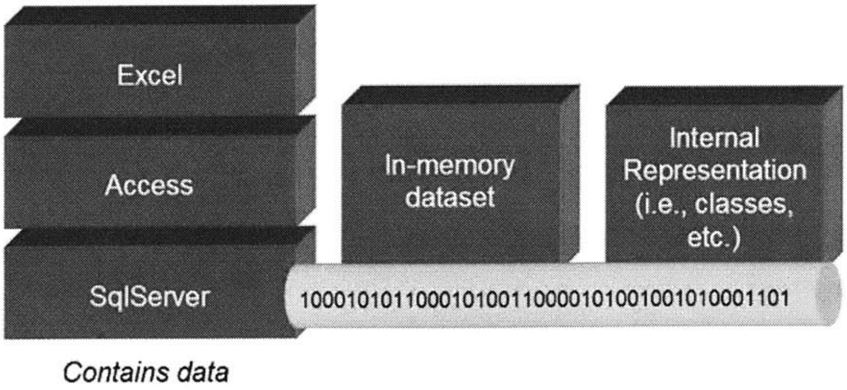


Figure A-1: Process of piping data from the simulation to the backend data source.

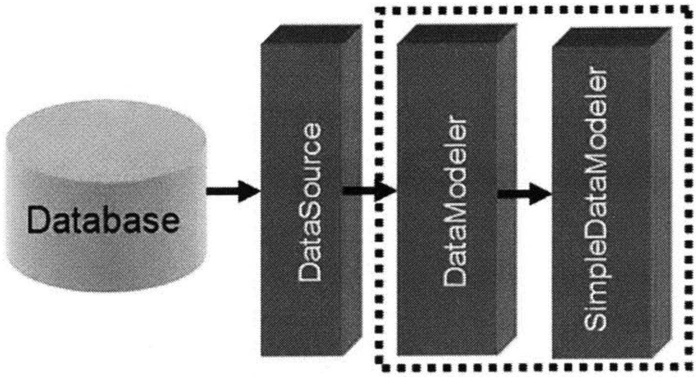


Figure A-2: Illustration of abstracting the semantics of data in a general database.

component, this was retained with a general class-hierarchy to adapt to changing database schemas throughout product design. Because of the flexibility of the approach, it has been extended to allow runtime specification of the data schema through the *DataSource* class hierarchy.

Figure A-1 illustrates the desired process for bringing data from the simulation application and pushing it into the data source of choice. Figure A-2 shows the class hierarchy for adapting data without data or process semantics (stored in a database) into a data object which has not only well-defined data in the context of the simulation but also optimized processes for storing and accessing the data.

The *DataSource* class serves to adapt the data held in a database, spreadsheet, or other structured format to adhere to an in-memory dataset schema as well as providing interfaces for reflecting changes back to the original data source. The

DataSource class also provides methods to cache and batch-update the original source file depending on the user's needs. The *DataSource* class also has inheriting classes for each of the data source types that must be interfaced (e.g., SQL Server, Excel, XML, etc.), which may override the some of the core methods of the *DataSource* class. The attractive part of this is that the base class provides the necessary abstraction to prevent calling classes from handling the details of individual data source types.

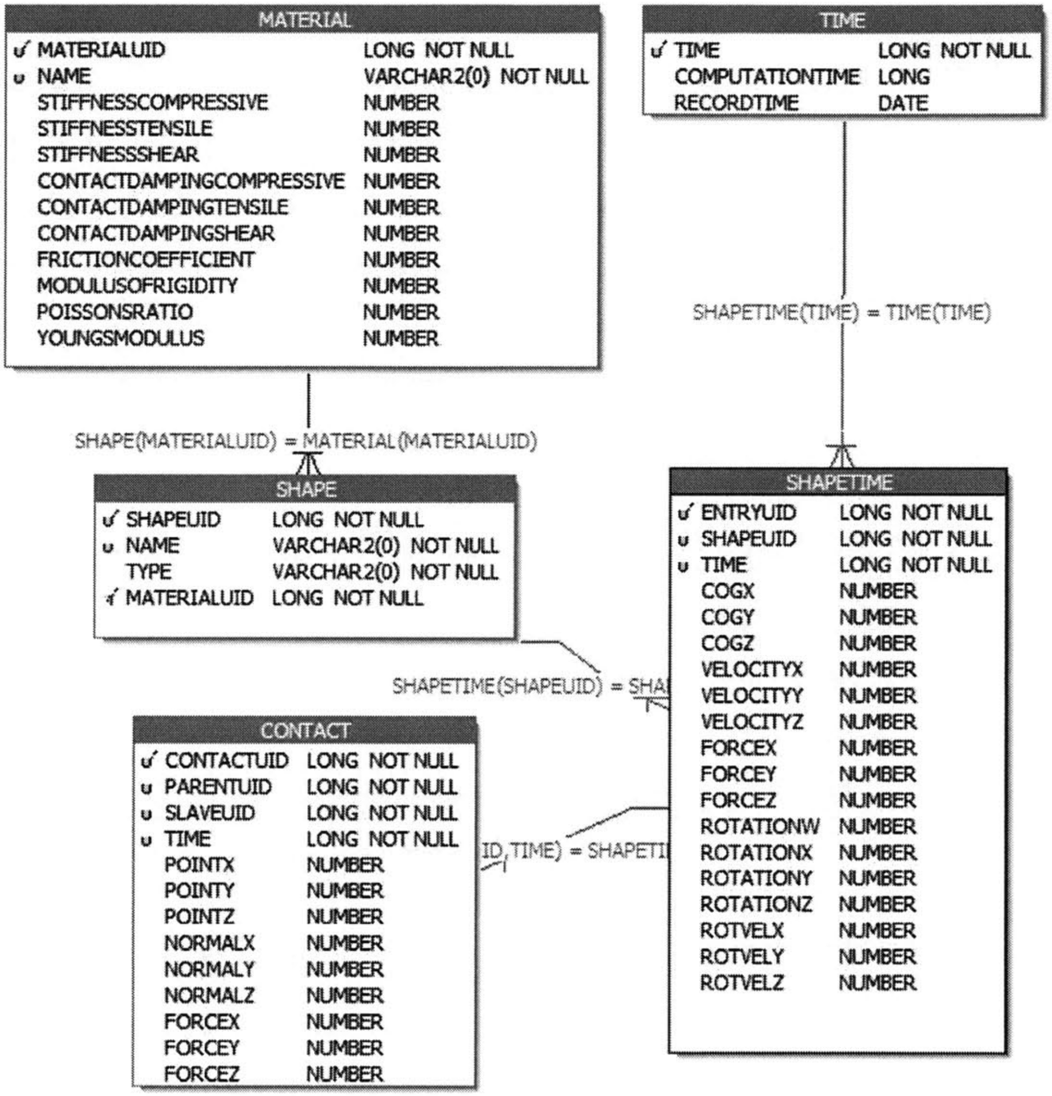
Caching and batch-updating are both a subset of the business logic of the data flow. From tests with sample DEM systems, caching could reduce the time for output by approximately 65%. Referential integrity is guaranteed by the DEM simulator itself, so there is no need to check relationships during the updating of the database; however, it would be advantageous to have relationships defined when querying the data. These behaviors are defined in the *DataModeler* class, which holds an instance of the *DataSource* class (which may be an inheriting class instance).

Finally, the abstraction of data must be constrained to only represent a particular data schema and to give rules about which data table and data columns should be created in the in-memory data schema. This is handled by inheriting classes of the *DataModeler* class such as the *SimpleDataModeler* class, as illustrated in Figure A-2.

The result of this architecture is a decoupling of the functional steps in storing data. A simple illustration of this is that if one were to change the way a shape is represented in the database or how timesteps are captured, the only change in the code for data storage would be in the *SimpleDataModeler* class. Methods also allow some simple changes to the underlying data schema to be handled through an XSD input file.

In the *SimpleDataModeler* class, data archiving in discrete element modeling is modeled as a strictly non-transactional data flow. That is, data is fed into data storage with no feedback to the simulation. Data is only withdrawn to perform analyses on the results of the simulation and to create visualizations of the output. This yields a model that is well-conditioned for fast updating given that an appropriate design is used for the representation of data.

In this work, the class structure of the DEM3D simulation application is amenable



	MATERIAL	
✓	MATERIALUID	LONG NOT NULL
u	NAME	VARCHAR2(0) NOT NULL
	STIFFNESSCOMPRESSIVE	NUMBER
	STIFFNESSTENSILE	NUMBER
	STIFFNESSSHEAR	NUMBER
	CONTACTDAMPINGCOMPRESSIVE	NUMBER
	CONTACTDAMPINGTENSILE	NUMBER
	CONTACTDAMPINGSHEAR	NUMBER
	FRICTIONCOEFFICIENT	NUMBER
	MODULUSOFRIGIDITY	NUMBER
	POISSONSRATIO	NUMBER
	YOUNGSMODULUS	NUMBER

	TIME	
✓	TIME	LONG NOT NULL
	COMPUTATIONTIME	LONG
	RECORDTIME	DATE

	SHAPE	
✓	SHAPEUID	LONG NOT NULL
u	NAME	VARCHAR2(0) NOT NULL
	TYPE	VARCHAR2(0) NOT NULL
✓	MATERIALUID	LONG NOT NULL

	SHAPETIME	
✓	ENTRYUID	LONG NOT NULL
u	SHAPEUID	LONG NOT NULL
u	TIME	LONG NOT NULL
	COGX	NUMBER
	COGY	NUMBER
	COGZ	NUMBER
	VELOCITYX	NUMBER
	VELOCITYY	NUMBER
	VELOCITYZ	NUMBER
	FORCEX	NUMBER
	FORCEY	NUMBER
	FORCEZ	NUMBER
	ROTATIONW	NUMBER
	ROTATIONX	NUMBER
	ROTATIONY	NUMBER
	ROTATIONZ	NUMBER
	ROTVELX	NUMBER
	ROTVELY	NUMBER
	ROTVELZ	NUMBER

	CONTACT	
✓	CONTACTUID	LONG NOT NULL
u	PARENTUID	LONG NOT NULL
u	SLAVEUID	LONG NOT NULL
u	TIME	LONG NOT NULL
	POINTX	NUMBER
	POINTY	NUMBER
	POINTZ	NUMBER
	NORMALX	NUMBER
	NORMALY	NUMBER
	NORMALZ	NUMBER
	FORCEX	NUMBER
	FORCEY	NUMBER
	FORCEZ	NUMBER

SHAPETIME(TIME) = TIME(TIME)
SHAPE(MATERIALUID) = MATERIAL(MATERIALUID)
SHAPETIME(SHAPEUID) = SHA
ID,TIME) = SHAPETI

Figure A-3: Data model for archiving data in DEM3D (sans independent simulation settings table).

to the relational data model that serves as the foundation of most modern enterprise database systems. The class design of bodies, boundaries, contacts, and simulation settings in DEM3D have intuitive analogues in a relational model. The final model is illustrated in Figure A-3.

As described in the introduction to the chapter, a large amount of data can result from trying to capture even a small subset of the data. It is often desirable to prefilter the information being archived to avoid large amounts of data being sent to the database. Prefiltering can take on the form of a reduced number of parameters for each particle, or it may be to sample timesteps at a lower rate or select only certain particles for tracking. With the 10^6 particles over 10^6 timesteps, we may only need to capture behavior of the simulation at a certain resolution (e.g., every 10^3 timesteps), over a limited interval (e.g., 10^3 timesteps after a certain stress state is detected), or over limited particles (e.g., a set of 10^3 tracer particles). All of these cases would result in a reduction of the stored information to 1.5 gigabytes, an easily addressed data size on a modern personal computer. It would be advantageous to build this pre-filtering into the system.

Pre-filtering of timesteps is handled in the simulation application itself as a parameter in the simulation settings. Individual particles can be pre-filtered based on filter objects, which determine, during the simulation time-stepping, which particle data to send to the *DataModeler* instance. In the *DataModeler* instance, the parameters are filtered based on the business logic of the class. For the *SimpleDataModeler* class, filtering can be specified by the user at runtime either through a programmatic interface. This programmatic interface is structured in a wizard format, making it easily adapted to a form-based specification of parameters.

A.2 Data Mining

Extracting data requires a series of steps to transform the raw data into a flat data structure. A flat data structure is advantageous in several ways:

1. It can be easily delimited by separator characters and imported into popular

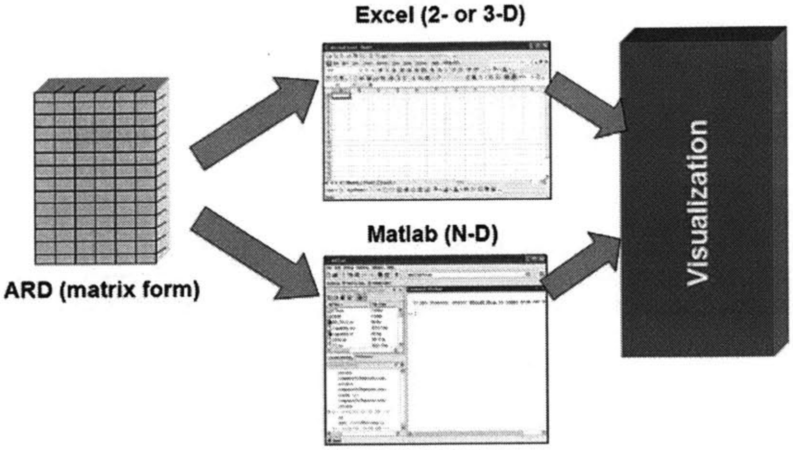


Figure A-4: Consumption of flat data table data into a visualization engine.

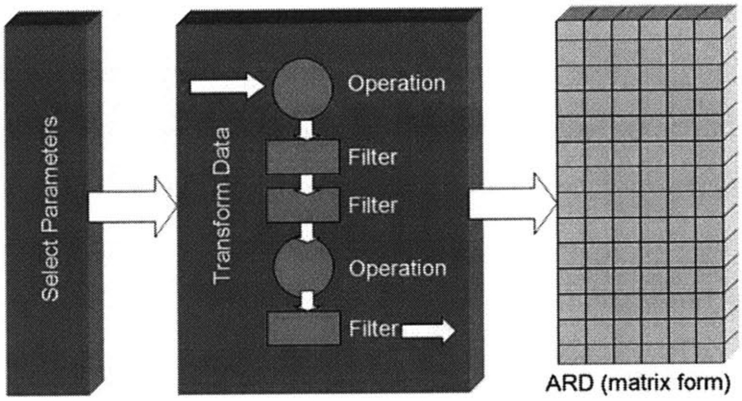


Figure A-5: Process for rolling up queried data into a flat data table via operations and filters.

statistical analysis programs, such as Minitab, SAS, and SPSS, as well as popular spreadsheets, such as Excel, for further processing.

2. Plotting of temporally-based states is naturally enabled by a flat data structure where time is captured along the row dimension and state, such as location, velocity, or force, along the column dimension.

3. Database queries produce flat data table results.

Bibliography

[1] Current good manufacturing practice for finished pharmaceuticals, April 1996.

[2] B.J. Alder and T.E. Wainwright. Studies in molecular dynamics. i. general method. *Journal of Chemistry and Physics*, 31:459–466, 1959.

[3] Kevin Amaratunga. A wavelet-based approach for compressing kernel data in large-scale simulations of 3d integral problems. *IEEE Computing in Science and Engineering*, 2(4):34–45, 2000.

[4] Kevin Amaratunga and Julio E. Castrillon-Candas. Surface wavelets: A multiresolution signal processing tool for 3d computational modeling. *International Journal for Numerical Methods in Engineering*, 52(3):239–271, 2001.

[5] A.H. Barr. Superquadrics and angle-preserving transformations. *IEEE Computer Graphics and Animation*, 1:11–23, January 1981.

[6] Guido Bartels, Tamas Unger, Dirk Kadau, D.E. Wolf, and Janos Kertesz. The effect of contact torques on porosity of cohesive powders. *Granular Matter*, 7(2):139–143, 2005.

[7] J. Baxter, H. Abou-Chakra, U. Tzn, and B. Mills Lamptey. A dem simulation and experimental strategy for solving fine powder flow problems. *Transactions of the Institute of Chemical Engineers*, 78(A):1019–1025, 2000.

[8] Barry W. Boehm. *Software Engineering Economics.* Prentice Hall, Englewood Cliffs, NJ, 1981.

[9] Clint Boulton. Ibm's blue gene supercomputer is for sale, November 2004.

[10] David Boutt, Benjamin Cook, Brian McPherson, and John Williams. Direct simulation of fluid-solid mechanics in porous media using the discrete element and lattice-boltzmann methods. *Journal of Geophysical Research*, submitted, 2004.

[11] John Bridgewater. Understanding how to process particulate matter. Lectures in Chemical Engineering, Massachusetts Institute of Technology, Cambridge, MA, April 2002.

[12] Volkhard Buchholtz and Thorsten Poeschel. Numerical investigation of the evolution of sandpiles. *Physica A*, 202(390), 1994.

[13] M.E. Cates, J.P. Wittmer, J-P Bouchard, and P. Claudin. Development of stresses in cohesionless poured sand. *Philosophical Transactions of the Royal Society of London*, 356(1747):2535–2560, 1998.

[14] Nora Y.K. Chew and Hak-Kim Chan. The role of particle properties in pharmaceutical powder inhalation formulations. *Journal of Aerosol Medicine*, 15(3):325–330, 2002.

[15] Jaehyuk Choi, Arshad Kudrolli, R. Ruben Rosales, and M.Z. Bazant. Diffusion and mixing in gravity-driven dense granular flows. *Physical Review Letters*, 92, 2004.

[16] Jamie A. S. Cleaver and J.W.G. Tyrrell. The influence of relative humidity on particle adhesion - a review of previous work and the anomalous behaviour of soda-lime glass. *KONA*, 22, 2004.

[17] B. K. Cook, David R. Noble, Dale S. Preece, and J. R. Williams. Direct simulation of particle-laden fluids. Technical Report SAND2000-1178C, Sandia National Laboratories, 2000.

[18] Benjamin Cook, David Noble, and John Williams. A direct simulation method for particle-fluid systems. *Engineering Computations*, 21(2-4):151–168, 2003.

[19] Benjamin K. Cook. *A Numerical Framework for the Direct Simulation of Solid Fluid Systems.* PhD thesis, Massachusetts Institute of Technology, Cambridge, Massachusetts, 2001.

[20] Peter A. Cundall. Computer simulations of dense sphere assemblies. In *Proceedings of the U.S./Japan Seminar on the Micromechanics of Granular Materials*, Sendai-Zao, Japan, October 1987.

[21] Peter A. Cundall, H. Konietzky, and David O. Potyondy. Pfc ein neues werkzeug fuer numerische modellierung. *Bautechnik*, 73(8):492–498, 1996.

[22] Peter A. Cundall and O.D.L. Strack. A discrete numerical model for granular assemblies. *Geotechnique*, 29(1):47–65, 1979.

[23] D. d' Humires, P. Lallemand, and U Frisch. Lattice gas models for 3d hydrodynamics. *Europhysics Letters*, 2:291–297, 1986.

[24] Reuben Domike. *Pharmaceutical Powders in Experiment and Simulation : Towards a Fundamental Understanding.* Phd, Department of Chemical Engineering, Massachusetts Institute of Technology, 2004.

[25] S.J. Eichhorn and R.J. Horn. The young's modulus of microcrystalline cellulose. *Cellulose*, 8:197–207, 2001.

[26] Laurene V. Fausett. *Applied Numerical Analysis Using Matlab®*, volume 1. Prentice Hall, Upper Saddle River, NJ, 1st edition, 1999.

[27] J.F. Favier. Integrating particle dynamics and boundary kinematics in discrete element analysis. In *15th ASCE Engineering Mechanics Conference*, number Number, Columbia University, New York, NY, 2002. ASCE.

[28] J.F. Favier, M.H. Abbaspour-Fard, and H. Kremmer. Modeling nonspherical particles using multisphere discrete elements. *Journal of Engineering Mechanics*, 127(10):971–977, 2001.

[29] J.F. Favier, M.H. Abbaspour-Fard, H. Kremmer, and A.O. Raji. Shape representation of axi-symmetric, non-spherical particles in discrete element simulation using multi-element model particles. *Engineering Computations*, 16(4):467–480, 1999.

[30] U. Frisch, B. Hasslacher, and Y. Pomeau. Lattice-gas automata for the navier-stokes equation. *Physical Review Letters*, 56:1505–1508, 1986.

[31] E.J. Garboczi. Three-dimensional mathematical analysis of particle shape using x-ray tomography and spherical harmonics: application to aggregates used in concrete. *Cement and Concrete Research*, 32:1621–1638, 2002.

[32] Russell Gold, Thomas Hammell, and Tom Snyder. *Test Driven Development: A J2EE Example*. APress, 2005.

[33] Robert Grady. *Practical Software Metrics for Project Management and Process Improvement*. Prentice Hall, Englewood Clifs, NJ, 1992.

[34] Gary Grest, James Landry, Leo Silbert, and Steven Plimpton. Rheology of granular flow, 2001.

[35] Gary Grest, James Landry, Leo Silbert, and Steven Plimpton. Rheology of granular flow. Lecture at the Kavli Institute for Theoretical Physics, Santa Barbera, CA, June 2001.

[36] J. Hardy, Y. Pomeau, and O. de Pazzis. Time evolution of two-dimensional model system i: invariant states and time correlation functions. *Journal of Mathematical Physics*, 14:1746–1759, 1973.

[37] H. Hertz. *Journal of Mathematics*, 92, 1881.

[38] R.C. Hibbeler. *Engineering mechanics: dynamics.* MacMillan, London, England, 1974.

[39] Chia-Lu Ho. Calculating the mean and variance of power sums with two lognormal components. *IEEE Transactions on Vehicular Technology*, 44(4):756–762, 1995.

[40] Mark A. Hopkins. Discrete element modeling based on mathematical morphology. In *Discrete Element Methods: Numerical Modeling of Discontinua*, Santa Fe, NM, USA, September 2002. ASCE, American Society of Civil Engineers.

[41] F.H. Hummel and E.J. Finnan. The distribution of pressure on surfaces supporting a mass of granular material. *Proceedings of the Institute of Civil Engineering*, 212:369–392, 1920.

[42] Richard P. Jensen, Tuncer B. Edil, Peter J. Bosscher, Michael E. Plesha, and Nabil Ben Kahla. Effect of particle shape on interface behavior of dem-simulated granular materials. *International Journal of Geomechanics*, 1(1):1–19, 2002.

[43] R.P. Jensen, Peter J. Bosscher, Michael E. Plesha, and Tuncer B. Edil. Dem simulation of granular media-structure interface: Effects of surface roughness and particle shape. *International Journal for Numerical and Analytical Methods in Geomechanics*, 23(6):531–547, 1999.

[44] K.L. Johnson, K. Kendall, and A.D. Roberts. Surface energy and the contact of elastic solids. *Proceedings of the Royal Society of London*, 324:301–313, 1971.

[45] Scott M. Johnson. Investigation of efficient geometric shape algorithms for numerical simulation of discrete particle systems. Master's thesis, Massachusetts Institute of Technology, Cambridge, Massachusetts, May 2003.

[46] Scott M. Johnson, John R. Williams, and Benjamin K. Cook. Contact resolution algorithm for an ellipsoid approximation for discrete element modeling. *Engineering Computations*, 21(2,3,4):215–234, January 2004.

[47] Capers Jones. Programming languages table, March 1996.

[48] Robert Jones, Hubert M. Pollack, Jamie A. S. Cleaver, and Christopher S. Hodges. Adhesion forces between glass and silicon surfaces in air studied by afm: effects of relative humidity, particle size, roughness, and surface treatment. *Langmuir*, 18:8045–8055, 2002.

[49] Robert Jones, Hubert M. Pollack, D. Geldart, and A. Verlinden. Inter-particle forces in cohesive powders studied by afm: effects of relative humidity, particle size, and wall adhesion. *Powder Technology*, 132:196–210, 2003.

[50] A.C. Kadak and M.Z. Bazant. Pebble flow experiments for pebble-bed reactors. In *Second International Topical Meeting on High Temperature Reactor Technology*, Beijing, China, 2004.

[51] Dirk Kadau, Guido Bartels, L. Brendel, and D.E. Wolf. Contact dynamics simulations of compacting cohesive granular systems. *Computational Physics Communications*, 147(1-2):190–193, 2002.

[52] C. Kane, J.E. Marsden, M. Ortiz, and M. West. Variational integrators and the newmark algorithm for conservative and dissipative mechanical systems. *International Journal for Numerical Methods in Engineering*, 49:1295–1325, 2000.

[53] Odagi Katsuaki, Toshitugu Tanaka, and Kenji Yamane. Rough-surface model and its applications to dem simulation of compression test of particles. In B. K. Cook and R.P. Jensen, editors, *Discrete Element Methods: Numerical Modeling of Discontinua*, number Number, pages 299–304. ASCE, 2002.

[54] Kevin Kendall. Adhesion: molecules and mechanics. *Science*, 263:1720–1725, 1994.

[55] Matthew R. Kuhn. A torus primative for particle shapes with the discrete element method. In *Discrete Element Methods: Numerical Modeling of Discontinua*, Santa Fe, NM, USA, September 2002. ASCE, American Society of Civil Engineers.

[56] Matthew R. Kuhn. Smooth convex three-dimensional particle for the discrete-element method. *Journal of Engineering Mechanics*, 129(5):539–547, 2003.

[57] H.P. Kuo, P.C. Knight, D.J. Parker, Y. Tsuji, M.J. Adams, and J.P.K. Seville. The influence of dem simulation parameters on the particle behaviour in a v-mixer. *Chemical Engineering Science*, 57(17):3621–3638, 2002.

[58] C. Lai and Charles Cooney. Application of a fluorescence sensor for miniscale on-line monitoring of powder mixing kinetics. *Journal of Pharmaceutical Sciences*, 93(1):60–70, 2004.

[59] C. Lai, D. Holt, J. Leung, Charles Cooney, GK Raju, and P. Hansen. Real time and noninvasive monitoring of dry powder blend homogeneity. *AIChE Journal*, 47(11):2618, 2001.

[60] C. Lai, A. Zahari, B. Miller, W. Katstra, M. Cima, and Charles Cooney. Nondestructive and on-line monitoring of tablets using light-induced fluorescence technology. *AAPS PharmSciTech*, 5(1), 2003.

[61] A. Lew, J.E. Marsden, M. Ortiz, and M. West. Variational time integrators. *International Journal for Numerical Methods in Engineering*, 60:153–212, 2004.

[62] Q. Li, D. Wills, R. Phillips, W.J. Viant, J.G. Griffiths, and J. Ward. Implicit fitting using radial basis functions with ellipsoidal constraint. *Computer Graphics*, 23(1):55–69, 2004.

[63] Guoping Lian, Colin Thornton, and Michael J. Adams. A theoretical study of the liquid bridge forces between two rigid spherical bodies. *Journal of colloid and interface science*, 161:138–147, June 1993.

[64] Kurt Liffman, Myhuong Nguyen, and Paul W. Cleary. Stress in sandpiles. In *Second International Conference on CFD in the Minerals and Process Industries*, pages 83–88, Melbourne, Australia, 1999.

[65] X. Lin and Tat-Tang Ng. Contact detection algorithms for three-dimensional ellipsoids in discrete element modeling. *International Journal for Numerical and Analytical Methods in Geomechanics*, 19:653–659, 1995.

[66] X. Lin and Tat-Tang Ng. A three dimensional discrete element model using arrays of ellipsoids. *Geotechnique*, 47(2):319–329, 1997.

[67] Ning Lu, J. Lechman, G.G.W. Mustoe, K.T. Miller, and K. Eccleston. Comparison of dem simulations and physical experiments for direct measurement of strongly attractive particle-particle interactions. In *Discrete Element Methods: Numerical Modeling of Discontinua*, Santa Fe, NM, USA, September 2002. ASCE, American Society of Civil Engineers.

[68] H.G. Mattutis and S. Luding. The effect of particle shape and friction on the stresses in heaps of granular media. In D.E. Wolf and P. Grassberger, editors, *Friction, Arching, and Contact Dynamics*. World Scientific, Singapore, 1997.

[69] H.G. Mattutis, S. Luding, and H.J. Herrmann. Discrete element simulations of dense packings and heaps made of spherical and non-spherical particles. *Powder Technology*, 109:278–292, 2000.

[70] Steve McConnell. *Rapid Development*, volume 1. Microsoft Press, Redmond, Washington, 1st edition, 1996.

[71] A. Mendez-Villas, M.J. Nuevo, and M.L. Gonzalez-Martin. On the use of rms roughness for surface roughness characterization. *Material Science Forum*, 408:245–250, 2002.

[72] Raymond D. Mindlin. Compliance of elastic bodies in contact. *Journal of Applied Mechanics*, 71:259–268, 1949.

[73] Raymond D. Mindlin and H. Deresiewicz. Elastic spheres in contact under varying oblique forces. *Journal of Applied Mechanics*, 20:269–286, 1953.

[74] A.A. Mirghasemi, L. Rothenburg, and E.L. Matyas. Influence of particle shape on engineering properties of assemblies of two-dimensional polygon-shaped particles. *Geotechnique*, 52(3):209–217, 2002.

[75] James K. Mitchell. *Fundamentals of Soil Behavior*, volume 1. John Wiley and Sons, New York, NY, 2nd edition, 1993.

[76] Toshiaki Miyajima, Ken-Ichi Yamamoto, and Masunori Sugimoto. Effect of particle properties during tapping. *Advanced Particle Technology*, 12(1):117–134, 2001.

[77] Sung Joon Moon, I.G. Kevrekidis, and S. Sundaresar. Particle simulation of vibrated gas-fluidized beds of cohesive fine powders. *submitted to I&EC Research (special edition October 2006)*, 2006.

[78] J.P. Morris, L.A. Glenn, F.E. Heuze, and S.C. Blair. Simulations of underground structures subjected to dynamic loading using the distinct element method. In *Discrete Element Methods: Numerical Modeling of Discontinua*, Santa Fe, NM, USA, September 2002. ASCE, American Society of Civil Engineers.

[79] Ante Munjiza. *The Combined Finite-Discrete Element Method.* John Wiley and Sons, Chichester, West Sussex, England, first edition, 2004.

[80] Ante Munjiza and K.R.F. Andrews. Nbs contact detection algorithm for bodies of similar size. *International Journal for Numerical Methods in Engineering*, 43(1):35–55, 1998.

[81] Ante Munjiza, J.P. Latham, and N.W.N. John. 3d dynamics of discrete element systems comprising irregular discrete elements - integration solution for finite rotations in 3d. *International Journal for Numerical Methods in Engineering*, 56:35–55, 2003.

[82] Graham G.W. Mustoe. Cluster2d discrete element computer code - users' manual and 2-d discrete element workshop notes. egweb.mines.edu/dem, 2001.

[83] NDCHealth. Pharmatrends: 2002 year in review, 2002.

[84] John Von Neumann. The general and logical theory of automata. In A.H. Taub, editor, *The Collected Works of John Von Neumann*, chapter 5. Pergamon Press, New York, NY, 1963.

[85] Tang-Tat Ng. Fabric evolution of ellipsoidal arrays with different particle shapes. *Journal of Engineering Mechanics*, 127(10):994–999, 2001.

[86] Tang-Tat Ng. Behavior of ellipsoids of two sizes. *Journal of Geotechnical and Geoenvironmental Engineering*, 130(10):1077–1083, 2004.

[87] Tang-Tat Ng and Changming Wang. Comparison of a 3-d dem simulation with mri data. *International Journal for Numerical and Analytical Methods in Geomechanics*, 25:497–507, 2001.

[88] Samuel Ngai. *Multi-Scale Analysis and Simulation of Powder Blending in Pharmaceutical Manufacturing.* Phd, Department of Chemical Engineering, Massachusetts Institute of Technology, 2005.

[89] C. Nouguier-Lehon, B. Cambou, and E. Vincens. Influence of particle shape and angularity on the behaviour of granular materials: A numerical analysis. *International Journal for Numerical and Analytical Methods in Geomechanics*, 27:1207–1226, 2003.

[90] Ruaidhri M. O'Connor and J. Friedrich. Microscale flow modeling in geologic materials. *Physics and Chemisty of the Earth (A)*, 24:611–616, 1999.

[91] Catherine O'Sullivan, Jonathan D. Bray, and Michael F. Riemer. Influence of particle shape and surface friction variability on response of rod-shaped particulate media. *Journal of Engineering Mechanics*, 128(11):1182–1192, 2002.

[92] H. Ouadfel and L. Rothenburg. An algorithm for detecting inter-ellipsoid contacts. *Computers and Geotechnics*, 24:245–263, May 1999.

[93] Chern-Perng Peng. *Analysis of Particle Properties and Inter-particle Bonds in a Pharmaceutical Process.* Bs, Department of Chemical Engineering, University of Queensland, 2002.

[94] Eric D. Perkins. *Discrete Element Computation : Algorithms and Architecture.* PhD thesis, Massachusetts Institute of Technology, Cambridge, Massachusetts, 2001.

[95] Eric D. Perkins and John R. Williams. Generalized spatial binning of bodies of different sizes. In *Discrete Element Methods: Numerical Modeling of Discontinua*, Santa Fe, NM, USA, September 2002. ASCE, American Society of Civil Engineers.

[96] Alfred M. Piestasanta. A strategy for software process improvement. In *Ninth Annual Pacific Northwest Software Quality Conference*, Portland, Oregon, 1991.

[97] Pekka Pirinen. Statistical power sum analysis for nonidentically distributed correlated lognormal signals. In *2003 Finnish Signal Processing Symposium*, Finland, 2003.

[98] Steven Plimpton. Fast parallel algorithms for short-range molecular dynamics. *Journal of Computational Physics*, 117:1–19, 1995.

[99] T. Poeschel. Molecular dynamics of arbitrarily shaped granular particles. *Journal of Physics I France*, 5:1431–1455, 1995.

[100] Dale Preece, Steven Burchell, and D.S. Scovira. Coupled explosive gas flow and rock motion modeling with comparison to bench blast field data. In *Proceedings of the 4th International Symposium on Rock Fragmentation By Blasting (FRAGBLAST)*, Vienna, Austria, 1993.

[101] Nabha V. Rege. *Computational Modeling of Granular Materials.* PhD thesis, Massachusetts Institute of Technology, Cambridge, Massachusetts, 1996.

[102] R.S. Sayles and T.R. Thomas. Surface topography as a nonstationary random process. *Nature*, 271:424–431, 1978.

[103] Peter Schroeder and Wim Sweldens. Spherical wavelets: efficiently representing functions on the sphere. *Computer Graphics*, 29:161–172, 1995.

[104] Peter Schroeder and Wim Sweldens. Spherical wavelets: efficiently representing functions on the sphere. *Computer Graphics*, 29:161–172, 1995.

[105] S.C. Schwartz and Y.S. Yeh. On the distribution function and moments of power sums with log-normal components. *The Bell System Technical Journal*, 61(7):1441–1462, 1982.

[106] R. Shivarama and E.P. Fahrenthold. Hamiltons equations with euler parameters for rigid body dynamics modeling. *ASME Journal of Dynamics Systems, Measurement, and Control*, 126:124–130, 2004.

[107] Ken Shoemake. Animating rotation using quaternions. In *SIGGRAPH85*, volume 245-254, pages 245–254, San Francisco, CA, US, 1985. ACM. This paper is one of the first to resurrect Hamilton's concept of quaternions from 1843.

[108] J.C. Simo and K.K. Wong. Unconditionally stable algorithms for rigid body dynamics that exactly preserve energy and momentum. *International Journal for Numerical Methods in Engineering*, 31:19–52, 1991.

[109] S.B. Slimane. Bounds on the distribution of a sum of independent lognormal random variables. *IEEE Transactions on Communications*, 49(6):975–978, 2001.

[110] J. Smid and J. Novosad. Pressure distribution under heaped bulk solids. *Proceedings of the 1981 Powtech Conference*, 63(3):1–12, 1981.

[111] Milos Sramek. High precision non-binary voxelization of geometric objects. In *Proceedings of the 17th Spring Conference on Computer Graphics (SCCG01)*, Slovakia, 2001.

[112] R.L. Stewart, John Bridgewater, Y.C. Zhou, and A.B. Yu. Simulated and measured flow of granules in a bladed mixer-a detailed comparison. *Chemical Engineering Science*, 56:5457–5471, 2001.

[113] Ning Sun and John Y. Walz. A model for calculating electrostatic interactions between colloidal particles of arbitrary surface topology. *Journal of colloid and interface science*, 234:90–105, 2001.

[114] L.W. Swenson and J.D. Wurtz. *DISCO-II: dynamic interaction simulation of clustered ordinance.* Lockheed Missile and Space Company, Inc., Sunnyvale, CA, 1985.

[115] L.M. Taylor and Dale S. Preece. Simulation of blasting induced rock motion using spherical element models. *Engineering Computations*, 9(2), 1992.

[116] Colin Thornton, Guoping Lian, and M.J. Adams. Modeling liquid bridges between particles in dem simulations of particle systems. In *Proceedings of the Second International Conference on Discrete Element Methods (DEM)*, Cambridge, MA, USA, 1993.

[117] Colin Thornton, Y. Sheng, C.J. Lawrence, and B.J. Briscoe. Numerical studies of uniaxial powder compaction process by 3d dem. *Engineering Computations*, 21(2/3/4):304–317, 2003.

[118] John M. Ting and B.T. Corkum. Discrete element models in geotechnical engineering. In *Proceedings of 3rd International Conference on Computing in Civil Engineering*, Vancouver, Canada, August 1988.

[119] Unknown. Qsm function point programming languages table version 3.0, September 2005.

[120] Loic Vanel, Daniel Howell, D. Clark, R.P. Behringer, and Eric Clement. Memories in sand: Experimental tests of construction history on stress distributions under sandpiles. *Physical Review E*, 60(5), 1999.

[121] Loc Vu-Quoc and X. Zhang. An accurate and efficient tangential force displacement model for elastic frictional contact in particle flow simulations. *Mechanics of Materials*, 31:235–269, 1999.

[122] Loc Vu-Quoc, X. Zhang, and O.R. Walton. A 3-d discrete element method for dry granular flows of ellipsoidal particles. *Computational Methods in Applied Mechanical Engineering*, 187:483–528, 2000.

[123] O.R. Walton. Numerical simulation of inclined chute flows of monodisperse, inelastic, frictional spheres. *Mechanics of Materials*, 16:239–247, 1993.

[124] O.R. Walton. Effects of interparticle friction and particle shape on dynamic angles of repose via particle dynamics simulation. In *Workshop on Mechanics and Statistical Physics of Particulate Media*, La Jolla, California, USA, 1994.

[125] O.R. Walton. Force models for particle dynamics simulations of granular materials. In *NATO ASI Symposium*, Mobile Particulate Systems: Proceedings of the NATO Advanced Study Institute, Corsege, Corsica, 1994.

[126] O.R. Walton. Potential discrete element simulation applications ranging from airborne fines to pellet beds. *SAE 2004 Transactions J. Aerospace*, 2004.

[127] Chung-Yue Wang, Chi-Fong Wang, and Jopan Sheng. A packing generation scheme for the granular assemblies with 3d ellipsoidal particles. *International Journal for Numerical and Analytical Methods in Geomechanics*, 23:815–828, 1999.

[128] John Williams, G. Hocking, and Graham G.W. Mustoe. The theoretical basis of the discrete element method. In *NUMETA*, volume 897-906, pages 897–906, Swansea, UK, 1985.

[129] John R. Williams and Kevin Amaratunga. Wavelet representation of geometry for analysis. In *Proceedings of the Second International Conference on Discrete Element Methods (DEM)*, Cambridge, MA, USA, March 1993. AFOSR, IESL Publications.

[130] John R. Williams and Ruaidhri M. O'Connor. A linear complexity intersection algorithm for discrete element simulation of arbitrary geometries. *International Journal for Numerical and Analytical Methods in Geomechanics*, 12(2):185–201, February 1995.

[131] John R. Williams and Nabha V. Rege. The development of circulation cells in granular materials undergoing compression. *Powder Technology*, 90:187–194, 1997.

[132] Xianjie Yi. *Numerical and Analytical Modeling of Sanding Onset Prediction.* Doctor of philosophy, Texas A and M University, 2003.

[133] Y.C. Zhou, B.H. Xu, Ai-Bing Yu, and P. Zulli. An experimental and numerical study of the angle of repose of coarse spheres. *Powder Technology*, 125:45–54, 2001.