YUNIBESITI YA BOKONE-BOPHIRIMA
NORTH-WEST UNIVERSITY
NOORDWES-UNIVERSITEIT

NUMERICAL SIMULATION OF SYSTEMS OF RIGID BODIES

Rufus Stephanus Neethling

B.Ing. (PU for CHE), M.Ing. (PU for CHE)

Thesis submitted in the School of Mechanical and Materials Engineering at the Potchefstroom Campus of the North-West University in fulfilment of the requirements for the degree of Philosophiae Doctor Ingeniare.

Supervisor: Prof. C.G. du Toit

POTCHEFSTROOM

2004

ACKNOWLEDGEMENTS

Academic/Professional

School for Mechanical and Materials Engineering, North-West University (formerly Potchefstroom University for Christian Higher Education) for this opportunity and the funding availed to me through the years.

M-Tech Industrial (CC) for financial assistance, opportunities to learn and gather experience and granting me the opportunity to complete my studies.

Prof. C.G. (Jat) du Toit for being my promoter and for proof reading this dissertation, as well as being a valuable source of information and assistance.

Alexander Polson for helping me obtain the necessary verification results by doing the required PFC3D test runs.

Personal

Our Creator for creating me, the world we live in and all that is within and around it and for granting me the talent and opportunity to pursue higher education to this level.

My parents for loving me, caring for me, raising me and helping me spiritually as well as materially during my early years of study and even now.

Anne-Marie Redelinghuys for being my best friend in the world, always believing in me and encouraging me to be the best I can be, I shall always love you and hold you dear.

Jan-Hendrik Kruger for being a great friend and advisor on some of the many problems that I had faced during the writing of this dissertation as well as for being at all interested in what I had to say about my field of study.

Robert Coetzee for being a friend and taking an interest in my field of study and the work I had been doing.

Eugene van Heerden and Martin MacRobert for being friends as well as esteemed colleagues at M-Tech with valuable advice on programming and solving the particular problems I had faced.

All those I might have forgotten or left out, the people who helped me become who I am, and do what I had done up to now.

ABSTRACT

This study investigated the various techniques available for rigid body system simulation, thus providing a thorough overview of the latest relevant literature. It became apparent that no engineering level accurate impulsive approaches for rigid body system simulation with concurrent contacts were in use or even existed up to now. A general formulation and solution technique for multiple concurrent contact problems were developed and tested on simple systems. Advantages of the formulation used are that the minimum of material properties, i.e. density, normal and tangential restitution coefficients and static and dynamic friction factors – all measurable, need to be specified and the results should be physically correct to a very high precision. Results obtained were encouraging and demonstrated that the original binary collision model available thus far could at least be extended to a heptenary simultaneous collision model.

OPSOMMING

Hierdie studie het die verskeie tegnieke beskikbaar vir star-liggaam-stelselsimulasie ondersoek en sodoende 'n deeglike oorsig oor die betrokke literatuur verskaf. Dit het duidelik geword dat geen ingenieursvlak-akkurate impulsiewe benaderings vir star-liggaam-stelselsimulasie met gelyktydige kontakte in gebruik is of selfs bestaan het tot op hede nie. 'n Algemene formulering en oplossingstegniek vir veelvuldige kontak-probleme is ontwikkel en getoets vir eenvoudige stelsels. Voordele van die formulering wat gebruik is, sluit in dat die minimum materiaaleienskappe, d.i. digtheid, normaal- en tangensiaalrestitusiekoëffisiënte en statiese en dinamiese wrywingsfaktore – alles meetbaar, gespesifiseer hoef te word en dat die resultate fisies korrek behoort te wees tot 'n hoë presisievlak. Resultate verkry is bemoedigend en het getoon dat die oorspronklike binêre model tot dusver beskikbaar, ten minste uitgebrei kan word tot 'n heptenêre gelyktydige botsingsmodel.

SAMMANFATTNING

Den här studien forskade varjehanda sätt som får användas för att simulera styva lekamsystem, och alltså förskaffade en tillbörlig översikt av tillhörande literatur. Det upptäckats att det fanns inga användade eller även existerande impulsbasserade flerkontaktmetoder som levererar resultater användbar i ingenjörsmiljön. En allmän formulering och lösningsteknik för flerkontaktproblem utvecklades och provades för enkla system. Formuleringen som utvecklades har några fördelar, som inkluderar det att man behöver bara materialegenskapen, liksom tätheten, normal- och tangensialrestitution och statiskt och dinamiskt friktionfaktorer – alla mätbara, och att resultater är fisiskt korrekta till en hög precision. Resultater som producerades är uppmuntrande och tydade på det att ursprungliga binärkollisionmodellen, som användades tills nu, kunde åtminståne utbredas till en sjulekamskontaktmodel (heptenärkollisionmodel).

CONTENTS

SUBJECT **PAGE**

NOMENCLATURE

ABBREVIATIONS AND ACCRONYMS

ANSI	- American National Standards Institute
ASCII	- American Standard Code for Information Interchange
BiCGM	- Bi-Conjugate Gradient Method sparse matrix solution algorithm
BiCGS	- Bi-Conjugate Gradient Squared sparse matrix solution algorithm
BiCGSTAB	- Bi-Conjugate Gradient Stabilised sparse matrix solution algorithm
CGM	- Conjugate Gradient Method symmetric sparse matrix solution algorithm
DAE	- Differential Algebraic Equation
DE	- Differential Equation
DEM	- Distinct Element Method
FEM	- Finite Element Method
FPGA	- Field Programmable Gate Array, a type of computer cluster
GLCP	- Gauss' Least Constraints Principle
ISO	- International Standards Organisation
LDE	- Linear Differential Equation
LCP	- Linear Complementarity Problem
MFC	- Microsoft Foundation Classes®
MLS	- Moving Least Squares method
MNCP	- Mixed Non-Linear Complementarity Problem
MPCC	- Mathematical Program with Complementarity Constraints
NCP	- Non-Linear Complementarity Problem
ODE	- Ordinary Differential Equations
OOP	- Object Oriented Programming
PBMR	- Pebble Bed Modular Reactor
PDF	- Portable Document Format, default extension for files of that type
PS	- Postscript, default extension for files of that type.
QMRES	- Quasi-Minimal Residual sparse matrix solution method
SRS	- Software Requirement Specification
SVD	- Singular Value Decomposition
RBSS	- Rigid Body System Simulation technique
TXT	- Preferred Extension for ASCII text files.

CONCEPTS AND DEFINITIONS

CONCEPT	DEFINITION/DESCRIPTION
Array	- An orderly arrangement of numbers or objects in any number of dimensions.
Binary Collision	- A collision involving only two bodies.
Cluster Collision	- A collision scenario where more than two bodies are involved concurrently.
Co-located Collisions	- Concurrent collisions sharing at least one common rigid body number in their collision pair lists.
Concurrent Collisions	- Collisions occurring at exactly the same instant in time.
Container Class	- A template class designed to be able to contain any type of data in a generic manner.
Convergence Level	- A measure of the similarity between solution values in two successive iterations.
Converging Solutions	- Values obtained by successive iterations are tending towards specific values.
Design Patterns	- Sets of standardised programming solutions to often encountered problems.
Divergence	- Iteration results not settling on any specific values but rather appearing to change by ever greater values.
Heptenary Collision	- A cluster collision involving seven rigid bodies.
Hexenary Collision	- A cluster collision involving six rigid bodies.
Iteration	- Repetitive calculation or algorithm execution that stops when a certain termination condition is met.
Lattice	- A three-dimensional array of values or objects.
Matrix	- A two-dimensional array of values or objects.
Objective Function	- A function providing a measure of the success or reliability of a solution to an optimisation problem when the solution is substituted into the expression.
Octenary Collision	- A cluster collision involving eight rigid bodies.
Optimisation Problem	- A computational problem with the aim of finding an optimal or best solution for a set of requirements.
Orthogonal Vector	- If a vector is oriented exactly perpendicular to some reference vector, it is said to be orthogonal to that reference vector and it implies that their scalar product (dot product) should be zero.

CONCEPT	DEFINITION/DESCRIPTION
Orthonormal Vector	- A vector oriented exactly perpendicular to both of two other mutually orthogonal reference vectors is said to be orthonormal to the other two and is usually obtained by calculating the cross product of those other two vectors.
Quaternary Collision	- A cluster collision involving four rigid bodies.
Quintenary Collision	- A cluster collision involving five rigid bodies.
Tensor	- A special real number matrix with three rows and three columns representing some physical properties such as angular inertia for an object.
Termination Condition	- The criterion governing the repetition of any process, represented in programming by loops.
Ternary Collision	- A cluster collision involving three rigid bodies.
Vector	- A one-dimensional array of values or objects, more specifically in geometry and physics a special three-element one-dimensional real number array representing a physical quantity with both magnitude and direction.

LIST OF SYMBOLS

STANDARD SYMBOLS

$C_{dp,i}$ - Profile drag coefficient tensor for body *i*.

$C_{dt,i}$ - Tangential drag coefficient tensor for body *i*.

$d_{f,ij}$ - Point *i* to facet *j* perpendicular projection distance.

d_{ij} - Inter-object distance between geometrical objects *i* and *j*.

dV_i - Infinitesimal volume for rigid body *i*.

$\underline{F}_i$ - Force vector acting upon body *i*.

I_i - Mass moment of inertia tensor for body *i*.

$I_{i,jk}$ - Mass moment of inertia tensor entry in row *j*, column *k* for body *i*.

m_i - Mass of body *i*.

M_i - Mass tensor for body *i*.

$\underline{n}_{f,i}$ - Surface normal vector for facet *i*.

$\underline{n}_{ij}$ - General contact normal vector between object *i* and *j* (pointing to $\underline{i}$).

$\underline{n}_{ij}^{nb}$ - Contact normal vector between object *i* and *j* (pointing to $\underline{i}$) at contact number *nb*.

$n_{grid,k}$ - Number of evenly spaced bins in Cartesian axis direction *k*.

$\underline{N}_{plane}$ - Plane normal vector for any arbitrary plane to be intersected by an edge.

$\underline{o}_{ij}^{nb}$ - Orthonormal tangential vector between object *i* and *j* at contact number *nb*.

$\underline{p}_i$ - Any arbitrary point position vector.

$\underline{p}_{ij}$ - Perpendicularly projected point from point *i* to edge *j*.

$\underline{p}_{e,ij}$ - Perpendicularly projected point from point *i* to geometric entity *j*.

$\underline{p}_{f,i,k}$ - Perpendicularly projected point from facet *i*'s vertex *k* to its opposite facet edge.

$\underline{p}_{plane}$ - Point on any arbitrary plane to be intersected by an edge.

$\underline{p}'_i$ - General point on and edge.

r_i - Radius of spherical body *i*.

r_i^{nb} - Radius of spherical body *i* for contact number *nb*.

$\underline{r}$ - Any radial position relative to a reference point in metres (*m*)

$\underline{r}_i^{nb}$ - Radial position vector from body i centre of mass ($\underline{x}_i$) to contact number nb.

r_i' - Component i of a general radius relative to a fixed point ($\underline{r}'$).

$\underline{r}'$ - General radius relative to the centre of mass for any body.

$\underline{r}'_i$ - Position radius of body i relative to the system centre of mass ($\underline{x}_{cm}$).

$\underline{t}_{ij}^{nb}$ - Tangential vector between object i and j at contact number nb.

$\underline{T}_i$ - Torque vector acting upon body i.

V_i - Total volume of rigid body i.

$\underline{x}_{cm}$ - Centre of mass position vector for total rigid body system.

$\underline{x}_i$ - Centre of mass position vector for body i.

$\underline{x}_{e,i,1}$ - Position vector for first vertex of edge i.

$\underline{x}_{e,i,2}$ - Position vector for second vertex of edge i.

$\underline{x}_{f,i,1}$ - Position vector for first vertex of facet i.

$\underline{x}_{f,i,2}$ - Position vector for second vertex of facet i.

$\underline{x}_{f,i,3}$ - Position vector for third vertex of facet i.

$\underline{x}_{v,i}$ - Position vector for vertex i.

$\underline{\dot{x}}$ - Any general linear velocity in metres per second ($m.s^{-1}$).

$\underline{\dot{x}}_{cm}$ - Centre of mass ($\underline{x}_{cm}$) average linear velocity for total rigid body system.

$\underline{\dot{x}}_i$ - Time derivative of centre of mass position vector for body i.

$\underline{\dot{x}}_{ij}$ - Velocity of object i relative to object j.

$\underline{\dot{x}}_{n,ij}$ - Normal component of velocity of object i relative to object j.

$\underline{\dot{x}}_{t,ij}$ - Tangential component of velocity of object i relative to object j.

$\underline{\dot{x}}_{total}$ - Resultant velocity due to linear ($\underline{\dot{x}}$) and angular ($\underline{\dot{\theta}}$) velocities at given radius ($\underline{r}$).

GREEK SYMBOLS

$\alpha_{e,ij}$ - Point i to-edge j perpendicular projection ratio.

$\alpha_{f,ij,k}$ - Point-to-edge projection ratio from point i's perpendicular line k of facet j.

$\Delta_{grid,k}$ - Original sorting bin division size in Cartesian axis direction k.

$\Delta_{overlap,k}$ - Sorting bin division overlap in Cartesian axis direction k.

Δt_t - The time step adjustment to be made during a critical time search iteration process.

Δt_{adj} - The time step adjustment to be made during a critical time search iteration process.

$\varepsilon_{n,ij}^{nb}$ - Restitution coefficient in normal ($\underline{n}_{ij}^{nb}$) direction at contact number nb.

$\varepsilon_{t,ij}^{nb}$ - Restitution coefficient in tangential ($\underline{t}_{ij}^{nb}$) direction at contact number nb.

$\lambda_{n,ij}^{nb}$ - Impulse magnitude in normal ($\underline{n}_{ij}^{nb}$) direction at contact number nb.

$\lambda_{t,ij}^{nb}$ - Impulse magnitude in tangential ($\underline{t}_{ij}^{nb}$) direction at contact number nb.

$\lambda_{t,ij,adj}^{nb}$ - Adjusted impulse magnitude in tangential ($\underline{t}_{ij}^{nb}$) direction at contact number nb.

ρ_{ext} - Edge extension ratio to describe any point on an edge relative to its first vertex.

$\rho_{ext,plane}$ - Edge extension ratio for the edge and plane intersection point.

ρ_i - Material density of rigid body i.

$\underline{\theta}_i$ - Angular orientation vector about centre of mass ($\underline{x}_i$) for body i.

$\underline{\dot{\theta}}$ - Any general angular velocity in radians per second ($rad.s^{-1}$).

$\underline{\dot{\theta}}_{cm}$ - Average angular velocity round centre of mass ($\underline{x}_{cm}$) for total rigid body system.

$\underline{\dot{\theta}}_i$ - Time derivative of angular orientation vector about centre of mass ($\underline{x}_i$) for body i.

LIST OF FIGURES

LIST OF TABLES

Chapter 1
Introduction

Ever since the dawn of mankind, there was the irrepressible need to feed an enquiring mind, to go beyond the apparent boundaries set by those preceding them. Had it not been for this seemingly simple fact, we as the human population could not and would not have been able to dominate the earth as is the case today. Envision a world where you have nothing, not even the clothes on your back, and everything has to be acquired and processed by utilising your own skill and physical power. The only way to really progress from this state, is to manufacture and use application specific tools, and the "designers" of such things were really taking the first steps in the direction of what we now call the discipline of engineering.

The following depiction is entirely fictional and any characters or incidents resembling actual people or events are purely and entirely co-incidental.

On some day, quite a while back, some person (it might have been a male or a female, no one knows) picked up a rock, and saw that it had a very dangerous side (having cut a left foot sole to the bone when stepping on it), and a relatively safe side (the side that would have been better to step upon). Looking around, other similar pieces of rock could be found, and it could be deduced that these rocks were a special kind of rock that tended to have sharp edges on some sides, but why, could not be figured out just then. They found that it was easier to bark trees, or defend themselves against predators when attacked using these wondrous stones. One lazy person, like I am too, got tired of going back to the place of origin too often, and started carrying hands full of them to the camp site, to have them handy when the first went dull. This, of course, did not always go smoothly, and some of the rocks fell, hit other rocks, and broke off shards: small, very much lighter and sharper shards with sharp points, once again taking a physical toll on the bearer, who now had quite a few holes and cuts to testify to a much too inquisitive mind accompanied by a clumsy and lazy body. But, if this sharp rock can peg into a human, it might work well on smaller animals if cast purposefully, somehow (somewhat like the stones used to kill hares, up until then, only sharper). The idea of a spear was born after several trials and many errors. The days of the naked (or vegetative matter-clothed – says a friend of mine) human were numbered, since larger and larger animals could be pursued and utilised for their meat, fur, leather and even their bones. Humans had entered the era of purposeful invention, rather than

relying on accidental discovery of useful tools or principles. One man saw that the men with longer arms could cast spears with more speed, whilst the taller men's spears went further, so he manufactured some alternate means of launching spears, e.g. spear slingers, of wood, or a bow and a modified smaller spear, called an arrow, that penetrated fairly large game with deadly accuracy at quite great distances. It would be quite a while before the answers to many of his questions would be given by a man named Sir Isaac Newton, but still, the principles that were at work could be utilised without understanding them ("Maths, what's that, is it edible??").

The biggest problem with not knowing, or understanding fully, the principles governing some phenomena, is that one can then only reach a serviceable solution by trial and error (or should that be, "trial and success"?), developing a "gut feel" for the better solutions as time goes by and experience accumulates. As the years marched on from the time of the first hunters, physical testing became more and more expensive, until, today, it has become nearly too costly to do at all. This, however, happened in a time when another handy tool had been developed, one that millions of people can use today and keeps on getting cheaper and cheaper: the personal computer (PC). But there is just one catch, one has to really understand the principles that govern and describe the phenomena in order to reproduce them on a computer, since a computer is a mathematical tool as far as engineers and many other scientifically oriented users are concerned. One can only describe the phenomena to be simulated and, later, applications of these to be tested, using mathematical expressions and techniques on a computer. But, once that hurdle is passed, the possibilities are almost infinite, and the testing can be done at literally a fraction of the cost of physical testing. Not to mention how much raw material is spared unnecessary usage, which is really good, since resources are currently under immense pressure and dwindling rapidly. This brings us to the field of interest to this study, as well as slightly related to the kind of design problems facing the people in the story relating the spear throwing quest.

The research topic of analytical dynamics proper had been initialised by Sir Isaac Newton himself. Terms both endearing and inspiring to many a physics or engineering student, such as "What goes up must come down", can be ascribed to this great thinker. But, moving on from the reverie, Newton, as he is most often referred to by the general scientific community, had contributed greatly to various fields of science. For everyday practical calculations, of particular interest to the dynamics researcher, the relationships between mass, displacement and time that had been determined by Newton are still applicable to this very day. Had Newton been alive today, he would most probably have taken on (and solved) very much the same types of

problems facing the dynamics researchers of today, and with the aid of computer power he might very well have been unstoppable. But, alas, that genius had passed away and we normal people have to struggle along without him. Fortunately, the man did what I sometimes fail to do, he wrote down everything. Newton was, of course, neither the first nor the only scientist to be interested in mathematics, geometry, bodies, forces and motion. The Greeks Thales, Pythagoras, Plato, Aristoteles, Euclidius, Archimedes and Ptolemeus, the Pole Nicolaus Copernicus (Mikolaj Koppernigk), the Italians Galileo Galilei and Leonardo da Vinci and the German Kepler, amongst many others, were way ahead of their times and worked on aspects of the topics related to mathematics, geometry, astronomy, mechanics and dynamics as we know them today. Galileo even had a lifelong imprisonment imposed upon him by the Roman Catholic Church for supporting the Copernican theory, thus being a heretic to their minds. Only recently, on October 31, 1992, 350 years later, Pope John Paul II admitted to mistakes made during the prosecution by the clergy of the time, but still did not overturn the conviction. Even though Galileo was thus convicted, he continued working and looking for answers, and developing new theories no matter what the resistance to them had been – something he had in common with all the other great contributors to modern scientific knowledge and theory, both before and after him. We all can learn from such brave and dedicated scientific practice: nothing should ever be considered totally impossible to understand or improve upon.

1.1 Scope and Goal of this Study

All the work done by the scientists mentioned earlier (and others), can only be useful if applied to one or more of the problems faced today, especially in the fields of engineering and science. Combining and utilising all relevant knowledge one had learnt from previous researchers, exchanging ideas with contemporary researchers and then producing a new scientific theory or providing tools for advancing insight into lesser studied principles and phenomena, can be considered as useful. *This study mainly aims to provide more detailed insight into the behaviour of and simple, yet accurate simulation capabilities for multi-body simultaneous collisions (henceforth termed cluster collisions) as encountered in engineering applications, such as ball mills, fluidised chemical beds or other dynamic multi-body systems, through evaluation or solution of physically based analytical expressions.* The latest appropriate models for rigid body motion and interactions need to be found, studied and compared with one another where and if possible. Applicable mathematical solution techniques will also have to be identified and employed, either in adapted form or as-is, with full cognisance of strengths and weaknesses of the techniques eventually employed (e.g. when severe calculation errors can be expected, when

techniques will break down, or when techniques will be highly appropriate for or ideally suited to a certain calculation).

1.2 Brief Overview of the Rest of this Thesis

In order to determine the range of rigid body motion and interaction models currently used, as well as associated solution techniques currently employed, the standard research procedure is followed. Firstly a need is identified, and in this case, that was the more detailed study and eventual simulation of cluster collisions. Knowing what the aim might be, one then has to make sure that which will be attempted was not yet done, since that would be time well wasted if it had been done. The only way to ascertain the uniqueness of any study and the accompanying theory to be developed is to do a thorough literature survey, the results of which can be found summarised in Chapter 2. Apart from ascertaining uniqueness, various handy pieces of information which might help with solving the problem at hand – should the topic prove to be unique after all – are uncovered. After a general literature survey determining the contribution and niche of the intended study, a more detailed literature "study" commences, during which the researcher has to identify and extract theory relevant to the problem intended to be solved from the reviewed and other literature and these can be found in Chapter 3.

Having the appropriate theory all identified and summarised, it is time for the development and implementation of additional theory, derived from, in extension to, or modified from existing theory. This new theory development and/or implementation is at the heart of the study, since this is where the real intellectual contribution would be seated, and this can be seen in Chapter 4. Once implementation had been done, a certain amount of fine tuning and improvements have to be done using test cases and their results as guidelines. The test results also provide the researcher with a measure of the success and accuracy achieved with the study, since it can be compared with the results of certain bench marks, alternative implementations/solutions or physical experiments. All of these test cases and results need to be neatly summarised, as is done in Chapter 5, in order to convince any scientific non-believers or sceptics that the latest solution indeed does provide whatever the researcher claimed it would. After running various tests and test cases to establish the degree of reliability, the results are discussed, conclusions are made and aberrant behaviour and inaccuracies are also stated and explained wherever possible. Finally, Chapter 6 deals with a summary of all work done, pointing out all the new contributions that had been made, and also suggesting topics for further research.

Chapter 2
Literature Survey

The literature survey for any study is crucial to ascertain the uniqueness of any research activity, but it also serves to determine the context of a researcher's work within the broader field of relevant work and provides the researcher with enough applicable theory and methods to be able to develop something useful to contribute to the field of study in question. In this chapter various literature sources are classified, discussed and rated with respect to relevance to the current research effort. From the literature, motivation for the current study can be extracted and neatly formulated.

2.1 Methodology

By far the most of the literature that had been collected is available on the internet as PDF and PS format documents, since they had been obtained by doing literature searches using keywords such as "rigid body collision" and "rigid body impact", on internet based search engines Google (at http://www.google.com) and Scirus (http://www.scirus.com). Further specialising search results by looking for "multiple contact" or "simultaneous impact", the most appropriate papers were obtained from all kinds of electronic sources. Once the sources had been verified (i.e. the internet-links had been checked, or their existence in a journal), they are ordered by date and category for chronological reading and discussion in groups of relevance, as can be seen in section 2.2 below. As the literature survey progresses, it might be necessary to add more references related to a specific field of interest for the sake of completeness, or in order to better place the whole study in context for the more inquisitive reader, or simply because it is a key reference with respect to better understanding certain concepts or phenomena.

2.2 Reviewed Literature

The reviewed literature can be divided into several sub categories, all relevant to various extents to the study undertaken. The obvious main field of interest would be physics, but without disciplines of mathematics (in particular geometry, calculus and numerical maths) and computer programming (design patterns, object oriented analysis and problem solution using C++ in particular), the research would not but very successful. These fields of interest and some of the

more relevant literature are now described and discussed in more detail in the following sub-sections.

2.2.1 Physics

The formal definition of physics according to the Concise Oxford Dictionary by Thompson (1995) reads: "The science dealing with the properties and interactions of matter and energy", from the Greek "phusika" for "natural things" (Thompson, 1995:1030). This general statement encompasses a vast array and multitude of sub-disciplines, of which most are not of direct interest or applicability to the present study. However wide the physics discipline, engineering will always be dependent on the results of physics research in its quest to design, develop and deliver products for the ever more impatient and comfort-loving consumer market. Without knowing the basic principles to apply, engineers would have no alternative but to become demi-physicists themselves. Fortunately there are many physicists and equally many research results to apply for engineering means. A good and sufficiently brief overview of the history and achievements of physics can be found on the web page named "Physics Time Line" at http://www.weburbia.demon.co.uk/pg/historia.htm and its sub-pages (last accessed January 23, 2004). If the contents of those pages are studied, it can be seen that the origin and development of physics can not truly be traced to any one of those contributors specifically, but, rather, to all of them, and most probably there had been some Babylonian, Chinese, Phoenician, Egyptian, Mayan, Scythian, Celtic, Germanic or other even extinct populations' "proto-physicists" preceding Thales of Miletus. The fact of the matter is that it does not really matter. What does matter, is what is left of what was learnt and what can be done with that. The Greeks started writing down all kinds of things, including their mathematical and astronomical hypotheses, and thus they could be considered the originators of physics in the modern format. Physics can be sub-divided into several sub-disciplines, such as the study of energy transfer, dynamics, statics, mechanics, nuclear physics, etc. The sub-disciplines of direct relevance to the present study are listed in the following sub-sections.

2.2.1.1 Mechanics

The Concise Oxford Dictionary by Thompson (1995) states that mechanics is: "the branch of applied mathematics dealing with motion and tendencies to motion" or, perhaps more applicable to the current study topic, "the science of machinery", from the Greek "mekhane" for "machine" (Thompson, 1995:846, under "mechano-"). Judging by such a definition of the discipline of mechanics, it would indeed be the most logical place to start looking for information on and

related to rigid body systems, their motions, their interactions and their simulation. First, the basics need to be understood, and these can be found in various books available on the topic of mechanics, of which Goldstein (1980) appears to have been a popular choice in the literature encountered. In this text, information can be found on the elementary principles, such as particle and particle system mechanics, constraints, D'Alembert's principle, Lagrange's equations, velocity-dependent potentials and dissipation functions. Furthermore, these basic principles are expanded or utilised in more detailed subsequent chapters, variational principles, rigid body kinematics and equations of motion and, perhaps of special interest to this study, the Lagrangian and Hamiltonian formulations for continuous systems and fields.

2.2.1.1.1 Dynamics

The Concise Oxford Dictionary by Thompson (1995) states that dynamics is: "the branch of mechanics concerned with the motion of bodies under the action of forces", indirectly from the Greek "dunamis" for power (Thompson, 1995:424, see dynamic). Glancing at the description, there has to be some form of relationship between forces and movement, and Sir Isaac Newton determined and penned down just such a relationship which is still quite applicable for the range of velocities, accelerations, masses and energies that the everyday mechanical engineer encounters, barring applications such as space travel and other (currently) less conventional modes of transportation, where the influence of Einstein's famous theory of relativity plays a larger than negligible role. More about Newton's equations of motion and impact can be found in Chapter 3. Some of the basic background theory for dynamics was obtained from Goldstein (1980) and some of the more mathematically oriented theory could be found in publications such as Woodhouse (1987), though the latter mostly appears to contain a subset of the former.

2.2.1.1.1.1 Free Body Dynamics

As the description might suggest, the discipline of free body dynamics entails the study of the motion of freely moving bodies in a three-dimensional domain under the influence of various kinds of forces. This is usually done by the application of the equations of motion and energy balance developed by Newton. Though Newton's equations are simple in principle, they are highly applicable to various engineering problems, barring the effects of friction/viscous drag one can, for instance, determine trajectories of projectiles (this would greatly have assisted the poor man with the spear-throwing problem in the previous section, who probably did not have that much time to test all his designs with the pressure of hunting and general survival being so high). The study of free motion is applicable to any system where projectile movement is of

importance and even in some systems where only a balance of forces and momentum is needed to determine, for instance, the amount of fuel necessary in a rocket.

2.2.1.1.1.2 Interactive Dynamics

As might be expected, Newton also had a stake in the principles of interaction between two free moving bodies, under the influence of forces such as weight and drag. The principles of conservation of momentum were formulated by Newton, and they still need to be satisfied for most earth-bound types of applications when occurrences such as collisions take place. A slightly humoristic definition of a two-body (also called binary) collision can be given as follows: "The result of two physical entities attempting to partially or entirely occupy one another's volumetric space at the exact same instant". Extending this definition of a collision to the more general case of multiple concurrent collisions or contacts, the following can be stated: "Two or more physical entities attempting to partially or entirely mutually occupy the volumetric space of one or all of the other entities present in the interacting cluster at the exact same time". Newton also formulated the inter-body gravitational law, which is, of course, another type of interaction between bodies and not presently under scrutiny for the masses considered in our investigations.

2.2.1.1.1.3 Rigid Body Interaction

Bodies that do not deform much (i.e. the point of impact does not migrate in any direction during collision and the physical geometries of the colliding objects do not change significantly) when undergoing a collision, can be considered rigid. Rigid bodies are, of course, a simplification of the actual physical reality, but the theory is simple enough and apparently easy to implement on a computer (more about the actual simplicity will follow later in this study), which we are ultimately aiming at. For the record it should be noted that Chatterjee and Ruina (1997) discussed and provided a proper account of the measure of rigidity and basically concluded that rigidity is well defined for smooth motions but not so for collisions. Even though this is the case, rigidity can still be assumed, since the assumption does not lead to too much inaccuracy in the types of applications the results of this study are intended to be employed. Quite a few materials used in engineering exhibit properties that are more "rigid-body"-like than they are "pliant-body"-like, such as steel, graphite, rocks/stones and glass.

Rigid-body type behaviour can be simulated using various kinds of approaches, even ones that seemingly contradict the very definition of rigidity (such as the penalty method that will be discussed in more detail in a section below), and all of these methods have their pros and cons.

The greater part of this literature survey which follows, is dedicated to identifying, describing and understanding as many of these alternative simulation approaches as possible in order to make informed decisions about the research objectives. From a brief overview of the literature and by reading highly informative and very thorough summaries of literature published in various papers, it could be determined that there are basically two main types of approaches, namely (mathematically) continuous contact models and (mathematically) discontinuous contact models (Gilardi & Sharf, 2002:1214), both of which approaches are described in detail and supplemented with references to literature in the following sections.

In order to gradually and gently introduce the whole field of rigid body system simulation, some more generally summarising and reviewing sources, such as McAllester (1996), are first discussed. In the aforementioned paper, the increasing importance to and usage of computers by the modern engineer are stated and various solution algorithms for non-linear mathematical programs (i.e. sets of equations and constraints, either linear or non-linear) are discussed, analysed and compared. Various alternative granular media modeling techniques for engineering purposes had been investigated by Herrmann and Luding (1998), including the hard sphere binary collision model, penalty methods and stochastic models, providing a valuable source of possible references. A more general approach was discussed by Keller *et al.* (1998), describing and using a combination of both continuous and impulsive contact models to simulate systems of rigid bodies and expressed the need for a simultaneous multiple impact model to be developed.

With relevance to computer graphics, Mirtich (1998) discussed penalty methods, Linear Complementarity Problem (LCP) methods and impulse-based methods and also describes a Singular Value Decomposition (SVD) method which can be used to find the solution vector with the minimum norm for a complementarity problem as defined for normal LCPs. Contact detection using a V-Clip algorithm for polygonal bodies is also discussed. LCP and impulse-based methods was also investigated by Sauer *et al.* (1998), describing their use for real-time rigid body simulations. In more granular media research for engineering applications, Hoomans (1999) provided a thorough summary of fluidised granular media simulation techniques available at the time, and amongst the techniques were binary collision impulse-based and penalty-force based simulation, respectively called hard sphere and soft sphere models. Another text providing a general overview is that of Engelsson (2000), who produced a dissertation containing a chapter on the use of the impulse-based and collision force based methods for contact resolution and in

that same year Luh (2000) did a thorough and critical synopsis of the paper on "Analytical methods for dynamic simulation of non-penetrating rigid bodies" by Baraff (1989).

A comprehensive text on rigid body system simulation related topics, also aimed at the computer graphics field of interest, were compiled by Erleben (2001), ranging from linear algebraic concepts through 3D geometry, contact detection, contact resolution methods (such as LCP formulations and impulsive impact theory) to the laws of motion with their applications and the underlying mathematics and numerical techniques necessary for solution. At roughly the same time and once more in the same genre, Milenkovic and Schmidl (2001) investigated optimisation based simulation and compared their technique with the alternatives, especially constraint based methods, such as LCPs. Their proposed optimisation based method addressed many of the problems, such as non-solutions and long calculation times for large numbers of concurrent contacts associated with LCPs. A later text by Erleben (2002) is a document describing the outlines of a modular rigid body simulation algorithm, as well as summarizing the various simulator paradigms that were known at the time, such as analytical methods, penalty methods, impulse-based methods and hybrid methods combining the aforementioned other paradigms. Still on the topic of computer graphics and rigid body modeling, Kry and Pai (2002) suggested a continuous contact simulation technique for smooth surfaced rigid bodies, unfortunately only for single contacts.

Also in the realm of computer graphics, Benedetti (2002) investigated physical response to collisions between deformable objects, describing in some detail the Coulomb friction law and Hooke's spring force law, as well as the concept of an impulse. Some attention was also given to describing the penalty methods and the constraint based approach taken by Baraff (1993) and the consequently necessary LCP formulation and its solution, as well as the impulse-based approach taken by Mirtich (1996). A handy table, providing comparative analysis of the various methods that had been scrutinized, had also been compiled (Benedetti, 2002:30). In their paper, Gilardi and Sharf (2002) provided a comprehensive literature survey of contact dynamics modelling, wherein topics ranged from a general impact definition through the various contact resolution paradigms and solution techniques to restitution definitions and friction models, but eventually the penalty method was preferred due to its ease of implementation and generality (can be used for both static and dynamic contact). The field of computer graphics and rigid body modelling was also the one in which Holmlund (2002) compiled a set of lecture slides that explain the rationale for physics-based modeling and related topics such as free body motion and

numerical integration were discussed and the penalty and impulse-based methods, chiefly the binary collision approach for the latter, were also briefly touched upon.

Computer graphics related rigid body simulation techniques were also investigated by Egan (2003), who discussed standard real time techniques, such as penalty methods, and impulse-based methods, and analytical methods, such as the original formulation (LCP-approach), Gauss' principle of least constraints, position based time stepping and energy based methods, briefly referring to several possibly useful literature sources for further work.

Progress had also been made in the field of Finite Element Modeling (FEM) regarding force calculations for interactive solid mechanics, as Jing (2003) demonstrated in a very thorough investigation of the various techniques, advances and outstanding issues in numerical modelling of rock mechanics, judging by the list of references provided and also the content of the paper. Of relevance to the present study is the fleeting mention of a penalty-like method to determine boundary conditions for a Finite Element Method (FEM) rock simulation grid. In the meantime, back in the computer graphics rigid body simulation category, Klein (2003) combined the approach of Chatterjee and Ruina (1998) for collisions, with some type of penalty method, for relatively static contact.

Another paper by Schmidl and Milenkovic (2003) provided a brief overview of the methods available and employed till early 2003, but mainly concentrated on utilising the impulse method for use in rigid body system simulations, while Stronge (2003) investigated the influence of several factors playing a role in a system containing multiple contacts and the validity of assumption that instantaneous impulse transfer occur in such systems, mentioning the work of Ivanov (1995) who investigated the validity of assumptions that simultaneous momentum transfer can be simulated using a series of binary contact resolutions and found that the calculated momentum distribution was flawed when compared with an analytical solution of general cases in both co-axial and planar three-body cluster collisions.

The subject of contact detection is a whole separate field of study, but it is nonetheless relevant and important to rigid body modelling, since it is the most time consuming part of simulation algorithms. The more recent work done by Fortin and Coorevits (2004) suggested an improved contact searching routine, able to find potential neighbours in large systems ($\pm 10^4$ bodies) using a connectivity table. In recent times, three dimensional elastoplastic frictional contact problems using Boundary Element methods had been investigated by Gun (2004) and might prove a valuable source of information for the present study, while Renouf *et al.* (2004:377-378) used a

method similar to that of Baraff (1993:9-13) for finding contact forces, employing non-linear Gauss-Seidel iteration in a parallel processor environment.

2.2.1.1.1.3.1 Continuous Contact Models

The continuous methods are so called because they do not require a halt in normal free body motion integration to first calculate interactive forces if time steps are chosen sufficiently small (typically of the order 1.8e-6 seconds per step, according to Mishra and Rajamani (1992), but highly dependent on the physical dimensions of the bodies being modelled, their velocities, and the amount of maximum overlap that is considered allowable). The most commonly known continuous contact resolution methods are penalty methods, which simply refers to the way interactions are handled or, more specifically, how inter-penetrations (be they due to collisions or static contact) are handled, namely by "penalising" a penetration with some counteractive force related to the distance of penetration or overlapping volume, material type and also the relative velocity. Penalty methods enable simulators to avoid having to distinguish between static or dynamic contact (i.e. collisions) by treating both types of contact the same way using the same types of parameters. A brief chronologically ordered discussion of contributors to this type of approach can be found in the next paragraphs.

Cundall and Strack (1979) were amongst the first researchers to implement a model having a spring and damper (dashpot) in both the normal and tangential directions of a contact (henceforth called Distinct Element Modelling or DEM) to simulate the physical material response during contact, but considered the solution a single contact at a time. Further work was done on this method by several other authors since then, especially in more recent years. Amongst others, Mishra and Rajamani (1992), (1993) used the basic DEM principles to develop 2D ball mill simulators and Hustrulid (1995) extended the DEM algorithm to make provision for parallel calculations in the 3D simulation of rock mechanics using an implicit formulation for the contact force expressions. Related to DEM, Kraus *et al.* (1997), (1998) and Kraus and Kumar (1997) proposed a modified spring-damper approach combining distance integrations and also an LCP solution to the frictional contact problem making provision for rolling and sliding with multiple concurrent impacts.

In frictionless collision modelling related research, Neethling (1998) compared basic frictionless collision handling using DEM with an alternative analytical method based on momentum conservation and impact equations and found that static contacts were easily and reliably handled

by DEM, but dynamic contacts had undesirable artificial energy gains and spatial rounding errors associated with them, whilst the alternative method had no such problems for impulsive impacts. What could also be seen from computer simulations is that realistic spring and damper constants are very hard to obtain, and that they are sensitive to relative velocities and spatial considerations (e.g. exact positioning relative to one another and number of simultaneous contacts). Related to DEM, the simulation paradigm of Finite Element Modelling (FEM) had some further contributions in the form of Gu *et al.* (2002), who used a variant of the penalty method for Finite Element Modelling (FEM) contact resolution, and Pandolfi *et al.* (2002), who developed a time discretised variational algorithm for the determination of inter-object forces, able to detect contact terminations and slippage, apparently related in some way to the Least Constraints Principle described by Redon *et al.* (2002), while Hasegawa *et al.* (2003) improved upon the simple overlap-distance-based penalty method model by taking into account the overlapping volume and determining a reactionary force based thereupon.

Stronge (2003) identified various contact resolution methods and investigated the phenomenon of impulse propagation. It was found that true simultaneous impacts can be assumed for high relative velocities and materials with high wave propagation properties, whilst true simultaneous impacts cannot at all occur for low relative velocities and low wave propagation properties. Another further improvement on the penalty method with specific application in FEM for solid mechanics was also proposed by Chamoret *et al.* (2004) along with a contact smoothing algorithm, employing automatically adjusting penalty parameter and time step based on the measure of interpenetration, only for frictionless contacts at the time of publication. In another, more recent, contribution to the field of DEM, Schäfer *et al.* (2004) investigated the use of FPGAs to accelerate calculation routines by means of optimised parallel processing.

2.2.1.1.1.3.2 Discontinuous Contact Models

A discontinuous contact model is only concerned with the states of a system of contacting rigid bodies immediately before and after a seemingly instantaneous event, such as a rigid body collision, and one could therefore call it a macroscopic view. It is called discontinuous, since velocities within the system appear to instantaneously change at the time of impact, thus a mathematical representation of each body's velocity appears "discontinuous" with respect to time. The advantages of using such a model to calculate the post-collision state of a system are that collisions are calculated at one single time step and that this avoids rounding errors which might have occurred as in the case of penalty methods. As mentioned earlier, the correctness of

assuming perfectly instantaneous momentum transfer due to impact had been investigated by Stronge (2003) and it was found to be related to, amongst other factors, relative velocity and wave propagation speed (e.g. the higher the relative velocity and wave propagation coefficient, the more correct the assumption). For this study, instantaneous momentum transfer through the whole system will always be assumed, since materials that have high wave propagation speeds are mainly under consideration, and thus the impulsive momentum transfer model can be used. The first impulsive momentum transfer investigations were done by Newton, who studied the phenomenon of pendular motion and impulsive momentum transfer during collisions, hence the name "Newton's cradle" for a certain well known apparatus decorating many a CEO's desk. The motion of the Newton-cradle penduli can be used to demonstrate the relationship between kinetic and potential energy, as well as the concept of momentum transfer. The binary collision model, developed by Newton to explain and quantify the phenomenon of momentum transfer during impact serves the investigator of such systems as the Newton's cradle well (until work done by Ceanga and Hurmuzlu (2001) shed some more and totally different light on this simple assumption), but there are shortcomings to this collision model, as will be seen in the course of this literature survey related to discontinuous contact resolution models.

The discontinuous models for contact resolution have steadily advanced in complexity and extent of their applicability, mainly due to the advances in computing power and available memory. Subsequent usage of more complex or previously impractical solution techniques had now become a reality and there are now a number of different approaches to handling contact/collision problems in a discontinuous manner, with those encountered during this literature survey listed in the following paragraphs.

The currently most popular method for handling concurrent contacts and collisions, encountered during this survey, appears to be the Linear Complementarity Problem (LCP) approach, which is essentially a linear or quadratic programming application, i.e. finding a "best solution" for a set of equations given a set of constraints to be satisfied. After an extensive study of existing techniques available at the time, Baraff (1989) proposed a scheme making use of heuristic LCP solution techniques to analytically solve for systems of simultaneous frictionless contacts, both static and impulsive, after first discussing the disadvantages of penalty methods. The concept of vanishing contact points is also discussed and a solution to the associated problem of predicting such points is proposed. The original proposed method was then also extended by Baraff (1993), (1994) to also incorporate friction and the various difficulties associated with both 2D and 3D solutions of this type were discussed, also citing other researchers working on related problems.

Stewart and Trinkle (1995) conducted a thorough survey of literature and is an excellent source on the origins of complementarity problems in general, as well as related formulations. There was also a very informative discussion on the principles of the existence and uniqueness of solutions and their meanings in general, as well as convergence and dynamics issues. Yet another improvement on previous work was made by Baraff (1997), (1999) with each consecutive set of conference contributions and in a collaborative effort, Baraff and Witkin (1997) also looked at partitioned dynamics which appears to be a good approach to approximately solve for multiple contacts between many rigid bodies at once. Similar work had been done by Sauer and Schömer (1998) and Sauer *et al.* (1998), who used a modified LCP approach with a complicated complementarity relationship imposed for two types of initially non-linear constraints (Sauer *et al.*, 1998:3) to solve concurrent contacts, also describing the handling of rolling contact, while Song *et al.* (1999) used an LCP approach to solve frictional contact problems incorporating a model for contact compliance, which is essentially a penalty-based approach.

The impulse based research of Anitescu and Hart (2000a), (2000b) employed velocity-impulse LCP-based techniques for calculating impulse magnitudes for concurrent collisions, which avoids the lack of existence of a specific solution as might be the case with some Differential Algebraic Equation (DAE) approaches, as well as the artificial stiffness resulting from the penalty methods, with their work only focusing on totally plastic collisions. It was El Kahoui *et al.* (2001) who could prove that the algorithm developed by Baraff (1989) does terminate for LCPs derived from contact problems, but not for general LCPs, subsequently developing an algorithm to treat the unsolvable subclass of these problems. As is the case for others, Anitescu and Hart (2002) extended their original work (Anitescu & Hart, 2000a), (Anitescu & Hart, 2000b) too, to handle problems with small friction using a fixed-point iteration technique. Also using the LCP approach, Cline (2002) dealt mainly with contact force determination, also summarising a basic LCP solution technique, called the algorithm of Lemke, which is the algorithm Stewart and Trinkle (2002) also employed to solve the set inequalities arising from non-penetration and frictional constraint problems as described in Stewart and Trinkle (1995) while also pointing out the relationship between Ordinary Differential Equations (ODEs) and complementarity problems, as well as describing in some detail the phenomenon of friction.

A method related to the LCP approach, is the Non-Linear Complementarity Problem (NCP) formulation, more applicable to situations where non-linear constraints are involved, such as in work by Buck and Schömer (1998), who used LCP and NCP approaches to solve non-linear

constraints for inter-body force calculations, employing a Newton type of iteration for the NCP. A thorough summary of work done to date, especially related to time stepping techniques for solving LCPs, was provided by Tzitzouris (2001) and Trinkle *et al.* (2001) did an informative study on the solution of concurrent contact problems for engineering applications using an NCP-approach. A slightly different approach was investigated by Adreani *et al.* (2002), showing that an LCP modeling three-dimensional multi-rigid-body contact with friction can be formulated as equivalent non-linear bound constrained optimisation problems, also describing the concept of Mixed Non-linear Complementarity Problems (MNCPs) and their solution using merit functions to be minimized, which yielded promising results.

Another method, related to LCP and NCP methods (and basically employing one or both of these methods as sub-procedures), is the Mathematical Program with Complementarity Constraints (MPCC) approach, pertaining to which Anitescu (2003) investigated the possibility of using algorithms and procedures usually applied to smooth non-linear programming for the solution of general non-linear programming problems. A constraint-stabilisation scheme for linear-complementarity-based time stepping was described in Anitescu *et al.* (2003) to deal with problems encountered in multibody dynamics with joints and contacts taking friction into account, which might be more useful since it avoids the need for multiple LCP solutions in favour of only one per time step (Anitescu *et al.*, 2003:4). The MPCC approach was also implemented by Peng *et al.* (2004) for the simulation of interacting robot systems with friction and seems to be using much of the theory in Anitescu *et al.* (2003).

A more physically based approach is the impulse-based range of simulation methods (in which the present study can also be included). Some form of momentum or energy transfer is considered and used to determine post-collision states. Mirtich and Canny (1994), (1995) described a method for and investigated the behaviour and applicability of impulse integration for binary collisions, while the Ph.D. dissertation by Mirtich (1996) conglomerated and implemented this research. It was concluded that the new impulse based method proposed and used for simulations were faster and more accurate than its contemporary penalty based counterparts. In a different approach, Kawachi *et al.* (1997) handled frictional multiple contact point impulse calculations by using an extension of the LCP formulation proposed by Baraff (1989:230). A more advanced approach in impulse based simulations was followed by Chatterjee and Ruina (1998), who proposed a new approach for handling simultaneous collisions by ordering the various collisions with respect to relative normal approach velocities, but still stuck to binary impulse based collision handling, while Goldberg *et al.* (1999) successfully

implemented the impulse based theory originally developed by Mirtich (1996) for a part pose-statistics estimator and Mirtich (1999) also investigated the combination of impulse and constraint based methods for mixed simulation environments.

The impulse-based binary collision resolution was also one of the techniques used by Hoomans (1999:32-47) to model granular media and Ouyang and Li (1999) also used what is essentially a binary collision model coupled with a fluid solution algorithm to simulate gas-solid interactive behaviour in a two-dimensional domain. A similar approach was taken by Cafiero and Luding (2000), except that a mean field kinetic theory is applied to determine translational & rotational temperatures. The impulse-based calculations for binary collisions were also used by Zhou *et al.* (2002:1803-1805) to model fluidized granular media. Yet another example of impulse-based binary collision resolution, similar to that of Kawachi *et al.* (1997), was the work of Dingliana and O'Sullivan (2003:4). Until recently, only binary collisions had been considered for true impulse based methods, but Giang *et al.* (2003) proposed using what appears to be a hybrid scheme utilizing Newton's impulse equations and LCP solutions, yet again similar to the approaches by Kawachi *et al.* (1997) and Dingliana and O'Sullivan (2003). There are a few shortcomings to the impulsive approaches presented. They are usually frictionless and LCPs do not necessarily always yield physically correct (or sufficiently accurate) solutions.

Some other related methods have been found to exist. In the field of granular media simulations, for instance, Glocker (1997) investigated the formulation of interaction laws for rigid multibody systems by generalizing the principles of d'Alembert, Jourdain and Gauss in terms of variational inequalities with impacts excluded which provided mathematical proof for the existence and uniqueness of accelerations in a rigid body system containing static contacts. A type of time dependent binary collision formulation was proposed by Luding and McNamara (1998), making it possible to calculate finite forces acting over a finitely short time for a collision based on energy transfer considerations. In addition to previous work done, Mirtich (1998) summarised some techniques for contact reaction calculations such as penalty and LCP methods, including the impulse based approach and its advantages with respect to stability and speed, as well as its disadvantages with respect to lasting simultaneous contacts (and, in fact, all true simultaneous contacts in general). As mentioned earlier, the then relatively new SVD approach was also discussed and described in some detail as an alternative collision resolution technique by Mirtich (1998). The use of control vectors for affecting externally imposed user interactions were proposed and investigated by Popovic *et al.* (2000), implementing an impulsive collision resolution model utilizing the LCP method developed by Baraff (1994).

The work of Bruynickx and Khatib (2000) described, in detail, combining d'Alembert, Jourdain and Gauss' principles to take geometric motion constraints into account for a system of dynamic redundant and constrained manipulators, while Ceanga and Hurmuzlu (2001) investigated the phenomenon of the simplest of multi-impact systems in the form of a Newton cradle. A more accurate alternative method was sought to overcome the problems associated with the simple impulse-based and compliance based methods for co-linear impact of an assembly of three balls and the solution was found in a modified impulse correlation method. This assumes prior knowledge of the impact propagation direction, which is not always possible or even advisable to guess. Nonetheless, this paper contains novel work done with respect to the case of an arbitrary number of balls in simultaneous rectilinear collision and the impulse distribution throughout such a system. Following earlier work, an arbitrarily changeable simulation paradigm had been developed by Popovic (2001), able to alter parameters and requirements and implementing a control vector as described in Popovic *et al.* (2000) and a continuous integrator scheme utilising Jacobian computation.

A very informative and highly applicable paper was published by Milenkovic and Schmidl (2001) in which a form of Gauss' least constraints principle was apparently used (Redon *et al.*, 2002:2) to solve for large systems of concurrently contacting rigid bodies with Coulomb friction incorporated. According to them, systems of rigid bodies can be linked mathematically using impulse equations and the loss of kinetic energy is maximised by using an optimisation algorithm – e.g. quadratic programming – suitable for such calculations. It should be noted that this method was only employed to obtain visually plausible results and not necessarily for engineering level accurate solutions (Milenkovic & Schmidl, 2001:2). One of the problems apparent from the equations used (Milenkovic & Schmidl, 2001:6) is that the linear contribution of the tangential impulses is not mentioned, furthermore, the regular tangential impulse equations that resemble those for normal direction impulses are neglected.

Another method seemingly related to the SVD and least squares minimisation/optimisation approaches was implemented by Redon *et al.* (2002) using Gauss' least constraints principle, classified as a form of Nearest Point Problem (NPP), in motion space formulations without friction. The algorithm operating in motion space performed increasingly better in preliminary test cases than the algorithms operating in contact space (e.g. LCPs) as the total numer of contacts increased (Redon, 2002:1-2). The advantage of using Gauss' principle is that it allows a very intuitive formulation of constrained systems (Redon, 2002:2) although only frictionless assemblies were considered. Of particular interest are the actual minimisation objectives (Redon

et al., 2002:2, 3). The Gauss principle converts the linear programming (LP) optimisation problem to a least squares optimisation problem (Redon *et al.*, 2002:4).

2.2.1.1.2 Statics

Statics is essentially a special case of dynamics, where the sum of all forces and torques in a system should be zero, no movement relative to an inertial reference frame occurs and continuous contact between all the components of such a system is involved. For the present study the static contact resolutions will not be delved into too deeply, since it is not really within the main scope. The work done by researchers such as Anitescu (2002), (2003), Anitescu and Hart (2000a), (2000b), (2002), Anitescu *et al.* (2003), Baraff (1989), (1993), (1994), (1997), (1999), Baraff and Witkin (1997), Benedetti (2002), Cline (2002), Cundall & Strack (1979), Egan (2003), Erleben (2001), El Kahoui *et al.* (2000), Gilardi and Sharf (2002), Holmlund (2002), Hustrulid (1995), Ladeveze *et al.* (2002), Lee *at al.* (1994), Trinkle (2003), Trinkle *et al.* (2001) and Tzitzouris (2001) all dealt with or referred to static contact force resolution to some extent regarding the use of mainly constraint based methods (i.e. implementing LCPs or NCPs) and penalty methods, but also occasionally mentioning some other currently less used methods (in the field of contact resolutions, at least) such as the Gauss principle and quadratic programming.

2.2.2 Mathematics

Referencing the Concise Oxford dictionary yet again, one finds that mathematics is: "the abstract science of number, quantity and space studied in its own right" (Thompson, 1995:840), from the Greek "mathematikos", via "mathema" – "matos" 'science' from "manthano" 'learn' (Thompson, 1995:840, under "mathematical"). It would be slightly redundant to go into the same depth of detail as had been done for the section on physics, since the whole discipline of mathematics had many of the same players involved and one only has to look at names of many of the terms, concepts and theorems encountered in mathematics to realise this very fact (e.g. Euclidian geometry, theorem of Pythagoras, Gauss elimination). The larger discipline of mathematics includes algebra, calculus and applied mathematics, all of which are applicable to some extent in the fields of physics, chemistry, engineering and economics, amongst others. As mathematical knowledge progresses, the continuing synthesis of knowledge from the different divisions of mathematical research to deal with ever more complex uses, makes it hard to determine within which division some of the research efforts actually belong. This is by no means such a bad thing, since it is, ironically, how it must have started out, there was no special

division of mathematics in the time of Thales, they could try their hand at everything, and seeing the whole picture usually leads to much more innovative thinking in any discipline. Be that as it may, for the sake of simplicity and clarity, the following few sections do roughly divide the literature into some categories, simply to ease the process of setting what is known to text in a neat format. Here, then, follows the most basic of divisions to be made.

2.2.2.1 Algebra

The word "algebra" is derived from the Arabic word "al jabr", which essentially means "the reunion of broken parts" (Thompson, 1995:31) and is today used to describe the mathematical discipline concerned with mostly abstract concepts (but which are found to have evermore relevance and application in various and diverse aspects of everyday life) in mathematics, or, as stated in the Oxford Concise Dictionary (Thompson, 1995:31), algebra is: "the branch of mathematics that uses letters and other general symbols to represent numbers and quantities in formulae and equations". Many texts on algebra exist – (Durbin, 1992) and (Johnson *et al.*, 1993) to mention but two of them – and they all are equally well suited to referencing where basic concepts are concerned, though their emphases might differ slightly. Durbin (1992) appears to cover most of the bases and provides a general overview, which is why it was arbitrarily chosen as the main reference on algebra. Typical examples of the use of algebra throughout history can be found briefly summarised in Durbin (1992:1-9), and special attention is drawn to the relatively small section on computer related algebra (Durbin, 1992:7) in the introduction, since this is relevant to the present study. The chapter on algebraic coding (Durbin, 1992:273-288) deals in some detail with the aforementioned topic of computer related algebra. Substantial portions of mathematical theory relevant and applicable to engineering calculations, and to this study in particular, belong to the mathematical sub-discipline of linear algebra.

2.2.2.1.1 Linear Algebra

The specialising word "linear" is obtained from the Latin via "linearis" from "linea" for "line" (Thompson, 1995:792). It thus implies that this type of algebraic analysis concentrates on the properties of lines in three-dimensional space, which, indeed, it was originally mostly used for. Though the concepts of lines and related objects are simple, much can be done with the basic theory in, for example, curve fitting, analytical geometry, rigid body mechanics, graph theory and multi-variable calculations (including linear programming or optimisation problem solution) as can be seen if one pages through books like the ones by Rorres and Anton (1979), Durbin (1992), Johnson *et al.* (1993) and Oliver (1993). Some of the very first applications of linear

algebra one encounters as an engineering student is the calculation of torque using the cross product, as can be seen in Goldstein (1980:2) or Salas (1995:876), or the component of a force vector in the direction of a specified vector using scalar products (Salas, 1995:799). Vector operations and analysis straddle the mathematical disciplines of algebra and calculus, and is covered in texts from both fields, such as Rorres and Anton (1979:9-35), Johnson *et al.* (1993:117-214), Kreyszig (1993:325-564), Ellis and Gulick (1994:690-733) and Salas (1995:773-897), though their emphases are slightly different. The sections in Kreyszig (1993) and especially Erleben (2001:19-33) is particularly important since it provides an overview of the relationship and collaboration between linear algebra and vector calculus. Cheney and Kincaid (1999:605-621) also provides a brief summary of frequently used linear algebraic terminology as utilized in numerical analysis.

The algebraic approach tends to be much more theoretical and abstract (e.g. n-dimensional, though still applicable to practical applications for three-dimensional cases), whilst the calculus approach tends to be more real-world based, though these differences are vanishing rapidly as the advantages of cross-discipline approaches are increasingly realised, recognised and followed. One is also gradually introduced to matrix theory, e.g. Johnson *et al.* (1993:1-116), which mostly deals with the problem of solving simultaneous sets of linear multi-variable equations (Johnson *et al.*, 1993:1-8), (Henrici, 1982:169-179), and its applications (Rorres & Anton, 1979), (Oliver, 1993:297-327), such as least square curve fitting (Rorres & Anton, 1979:193-202), (Durbin, 1992:328-358), (Johnson *et al.*, 1993:196-212), multi-variable Newton-Raphson solutions (Stoecker, 1989:124-127), (Cheney & Kincaid, 1999:110-113), linear programming for optimisation problems (Oliver, 1993:310-319), (Rorres & Anton, 1979:219-274), (Cheney & Kincaid, 1999:579-603), stress calculations in mechanics of materials (Boresi *et al.*, 1993:30-43), finite element methods for flow and structures (Reddy, 1993:486-502), (Reddy, 1993:147-192) and finite difference (Anderson *et al.*, 1984:479-517) methods for flow. Once the theory of matrices is understood, literally a whole new world of applications unfolds to the researcher. The current developments in matrix usage seems to not as much focus on the development of new applications as it is on finding more accurate and efficient ways to determine solutions for literally massive sets of simultaneous linear equations. More about these methods (and their applications) will be said in the section on Sparse Matrix Solution Algorithms.

2.2.2.2 Calculus

According to Thompson (1995:185), calculus is: "a particular calculation or reasoning" or, more appropriate to engineering, "the infinitesimal calculuses of integration or differentiation", from

the Latin "calculus" for a "small stone used in reckoning on an abacus" (Thompson, 1995:185). Due to the inherently time-dependent nature of processes and phenomena, differential calculus is of extreme importance and relevance to most engineering problems attempting to reflect real world situations accurately. Integral calculus has an equally important and prominent role to play in engineering, for example in the determinations of volumes (Ellis & Gulick, 1994:494-507), (Salas, 1995:363-370), curve lengths (Ellis & Gulick, 1994:507-511), (Salas, 1995:612-619), surface areas (Ellis & Gulick, 1994:512-516), (Salas, 1995:347-351), work done (Ellis & Gulick, 1994:516-522), (Salas, 1995:379-384), moments and centers of gravity (Ellis & Gulick, 1994:525-531), (Salas, 1995:371-376), hydrostatic force, (Ellis & Gulick, 1994:533-537), (Salas, 1995:386-389), etc. As is the case with algebra, there exist many books on calculus, of which the ones by Ellis and Gulick (1994), Salas (1995) and Stewart (1998a), (1998b) are but four, though four very good, examples. It can be seen that the common thread running through all of these books, is the basic theory pertaining to functions and function properties, limits, differentiation, integration, deduction and expansions of [Taylor & Fourier] sequences and series, as well as their usages in everyday calculations and applications, such as those seen in Stoecker (1989:121-127), Henrici (1993:222-273, 275-330), (1993:332-396), Ellis and Gulick (1994:544-634), Salas (1995:693-769), Stewart (1998b:558-640) and Cheney and Kincaid (1999:102-113, 123-129, 374-389, 412-414).

Of particular interest to the present study are the sections dealing with Lagrange multipliers , as can be found in Ellis and Gulick (1994:865-878) or Salas (1995:1008-1013), since they are mentioned more than a few times in the literature on linear complementarity (Anitescu & Hart, 2000a:4), (Anitescu & Hart, 2002:2, 9, 12), (Anitescu, 2003:3, 4), (Anitescu *et al.*, 2003:2) and general mechanics (Goldstein, 1980:35-64), (Woodhouse, 1987:31-80). From the texts on calculus surveyed (Ellis & Gulick, 1994), (Salas, 1995), it can be seen that calculus is indeed highly applicable to and useful for various physics and engineering problems or calculations. A few examples can be given. The Divergence Theorem, e.g. in Ellis and Gulick (1994:1013-1020), relating the total surface area and volume of a region, is used in computational grid calculations to simplify volume calculations. The relationship between finite difference and finite volume approaches for flow solutions can also be demonstrated by virtue of the Divergence Theorem. The first order simple Taylor series expansion for multiple variable functions is easily used to derive the basic theory for the Newton-Raphson iterative solution for multiple variables (Stoecker, 1989:124-130), (Cheney & Kincaid, 1999:110-113).

2.2.2.2.1 Geometry

Thompson (1995:566) states that geometry is: "the branch of mathematics concerned with the properties and relations of points, lines, surfaces and solids", from the Greek "ge" for "earth" (Thompson, 1995:566, under "geo") and "-metria" from "metron" for "measure" (Thompson, 1995: under "metre" and "metry"). Though geometry qualifies as a discipline in its own right (and most likely was the origin of mathematical theory as we know it today), it is included in calculus texts, and thus most often considered to be a sub-discipline of calculus. When working with the physical properties (e.g. volume, surface area, projected area, mass, centre of mass) and spatial attributes (e.g. inter-object distances) of objects such as rigid bodies, the use of analytical geometry is encountered very early in the investigations conducted. In the present study, the fast and efficient calculation of inter-object distances between various types of simple geometrical entities (e.g. vertices, lines, triangular facets and spheres) is of primary concern and utmost importance. Other types of geometrically based information that need to be efficiently calculated are collision and contact normal vectors, and constructed orthonormal bases for each of the collisions and contact points. As already mentioned some relevant theory for vector based analytical geometry, such as line magnitudes, vector component projections, orthogonal vector constructions, vector addition, etc. can be found in calculus texts such as Ellis and Gulick (1994:691-735), Salas (1995:773-837) or Stewart (1998b:644-682), while other relevant theory or examples of usage can also be found in algebra texts such as Rorres and Anton (1979:1-8, 9-18, 121-129) and Johnson *et al.* (1993:117-203) and also in the texts by Kreyszig (1993:325-564) and Erleben (2001:34-41).

2.2.2.3 Applied Mathematics

The Oxford Concise Dictionary (Thompson, 1995:840 under "mathematics") states that applied mathematics is: "[mathematics] as applied to other disciplines such as physics, engineering, etc." Judging by the foregoing description, applied mathematics is essentially the practical application of theoretical mathematics in whatever fields they happen to be of use. It could be argued that applied mathematics is the final test of validity for otherwise purely theoretical mathematics. Applied mathematics is the reason for the current blurring of the lines of distinction between, for instance, algebra and calculus, since it is the main source and cause of cross pollination between various fields of mathematical research. Specifically in the heat transfer field of engineering, Stoecker (1989) provides several examples of the practical application of mathematics to solve engineering problems such as financial calculations (Stoecker, 1989:27-45), mathematical modelling (Stoecker, 1989:53-74), system simulation (Stoecker, 1989:111-131) and optimisation

(Stoecker, 1989:143-152). There is also a section on Lagrange multipliers and their use in calculus methods of optimisation (Stoecker, 1989:161-176). Useful sections on search methods pertaining to optimisation are also included (Stoecker, 1989:186-209), (Stoecker, 1989:454-465).

More details on geometric programming (Stoecker, 1989:240-255), linear programming (Stoecker, 1989:260-291) and calculus based methods of optimisation (Stoecker, 1989:430-451) can also be found in Stoecker (1989). Other texts used during the course of this study are Henrici (1982), Press *et al.* (1992), Kreyszig (1993) and Cheney and Kincaid (1999), and they provide more general information on how to numerically approach the applied mathematics needed in various divisions of engineering.

2.2.2.3.1 Numerical Analysis

Numerical analysis is: "the branch of mathematics that deals with the development and use of numerical methods for solving problems." (Thompson, 1995:934) and the word "numerical" is derived from Latin via "numericus" from "numerus" (Thompson, 1995:934, under "number" and "numerical"). Numerical analysis or mathematics is, thus, part of the real-world application of mathematics mentioned before. Practical problems pertaining to the evaluation or calculation of many types of mathematical expressions are addressed by various types of numerical solutions and one can see quite a few of these summarised in texts like Henrici (1982), Press *et al.* (1992), Kreyszig (1993) and Cheney and Kincaid (1999). One should always remember that there are various types of errors (Henrici, 1982:39-50), (Press *et al.*, 1992:28-31), (Kreyszig, 1993:920-921, 922-924), (Cheney & Kincaid, 1999:44-89), (Pettersson, 2003:5-7), such as instabilities (Henrici, 1982:33-38), (Kreyszig, 1993:921), (Cheney & Kincaid, 1999:402-404), conditioning aspects (Henrici, 1982:51-62), (Kreyszig, 1993:993-998), (Cheney & Kincaid, 1999:249-250) and representation-related problems (Henrici, 1982:4-30), (Press *et al.*, 1992:29-31), (Kreyszig, 1993:919-920), (Cheney & Kincaid, 1999:63-68, 74-82) associated with numerical work, since almost all numerical work now done is done on a digital computer representing all numbers in sets of bits. Failure to take into account the aforementioned affects or to somehow make provision for them when coding, could lead to very rapid procurement (due to high processor speeds nowadays) of very wrong, and thus useless, expensive or even dangerous, answers in some cases.

Many of the various mathematical concepts that exist in theoretical mathematics (i.e. algebra and calculus) had been found to have one or more application in especially engineering, as could be seen in the various chapters of Stoecker (1989), who concentrated on thermal system simulation,

design and analysis, Press *et al.* (1992) which is also a compilation of numerical methods and the whole of Kreyszig (1993), which is an extensive compendium of engineering related mathematical theory, their applications and specialised numerical techniques to solve the expressions and equations originating from such typical engineering problems. A good overview of various numerical methods can be found in a section of and Press *et al.* (1992), Kreyszig (1993:916-1082) and the greater part of Cheney and Kincaid (1999). Methods of possible particular interest to the present study are highlighted and discussed in the following sections.

2.2.2.3.1.1 Linear Multiple Variable Equation Solution Techniques

Simultaneous sets of linearly independent linear equations can always be represented by an augmented matrix (Johnson *et al.*, 1993:23) type of problem which can be solved using several alternative techniques. Several practical applications involve the solution of simultaneous linear equations at some stage as part of the algorithms. Perhaps the most powerful and applicable method known to engineers are the Newton-Raphson iteration and its related Secant iteration for solving systems of non-linear equations, which rely on sets of linear equations resulting from the Taylor expansion of multi-variable functions to be solved. The solution of linear systems is also encountered and required in Least Squares curve fitting and minimisation techniques, the solution of linear ordinary differential equations with specified initial values and the equations resulting from the discretisations of equations representing the physical properties and behaviour of fluids and materials (e.g. Finite Difference Methods, Finite Volume Methods, Finite Element Methods).

Returning to the topic of solving systems of linear equations; augmented matrix expressions can easily be directly solved by using Gaussian elimination (Henrici, 1989:169-179), (Johnson *et al.*, 1993:5-8, 10-17, 21-32), (Kreyszig, 1993:972-979), (Cheney & Kincaid, 1999:241-265) which is an analytically-based algorithm that eliminates all sub-diagonal entries. Although it is the most analytically correct method, it unfortunately requires a full matrix to store all the coefficients as the elimination progresses, thus making it only applicable for solving relatively small systems of linear equations.

For larger, usually sparse, matrices several solution methods had to be devised and do exist. The simplest of all is the Jacobi iteration (Kreyszig, 1993:990-991), (Barret *et al.*, 1994:8-9), (Cheney & Kincaid, 1999:309) which can be used to solve relatively large systems of linear equations by approximating new guess values for all variables using only the original guess values. Similar in approach to the Jacobi iteration, the Gauss-Seidel iteration (Kreyszig,

1993:986-988), (Barret *et al.*, 1994:9-10), (Cheney & Kincaid, 1999:309) simply makes use of only the latest guess values (i.e. the old guesses where not available, else, the new values) to calculate new guess values. Successive Over-Relaxation (SOR) (Barret *et al.*, 1994:10-14), (Cheney & Kincaid, 1999:309-310) is, in its turn, an acceleration of the Gauss-Seidel iteration where the contribution ratio (called relaxation factor) of the old guess values and new guess values to the new guess values can be specified. Setting the relaxation to 0.0 yields the Jacobi iteration, whilst using a relaxation of 1.0 yields the original Gauss-Seidel iteration. The advantages for any of these techniques are that they can also be used to solve sparse matrices, thus making the number of variables that can be solved for simultaneously much greater and the storage requirement can be minimised as far as possible.

Spectral property based methods for solving large sparse matrices such as the Conjugate Gradient Method (CGM) and its derivative methods such as Bi-Conjugate Gradient Method (BiCGM), Bi-Conjugate Gradient Stabilised (BiCGSTAB) and Bi-Conjugate Gradient Squared (BiCGS) had been developed to address the need for rapid solution of massive systems of linear equations (Barret *et al.*, 1995), (Bücker, 2002). Conjugate Gradient methods (as well as some other solvers) emphasise the importance and applicability of combining linear algebra and calculus concepts once more, since they are based on the construction of series of mutually orthogonal residual vectors – for example, in the Bi-Conjugate methods this is attempted for two series of residuals. Iterative solution methods for parallel implementation was also discussed by Bücker (2002), naming such alternatives as the Classic (Bücker, 2002:526-528), e.g. Gauss-Seidel, Jacobi and SOR, Projection (Bücker, 2002:528-529) and Krylov Subspace (Bücker, 2002:530-540), e.g. Power Method, Arnoldi, Lanczos, Bi-Lanczos, Ritz-Galerkin, Petrov-Galerkin, Generalised Mimimum Residual (GMRes) and Quasi-Minimal Residual (QMR), methods. In their recent paper, Alanelli and Hadjimos (2004) described the use of block Gauss elimination followed by classic iterative solution techniques, e.g. Jacobi, Gauss-Seidel and SOR, for solving linear systems.

The convergence properties for matrix solution techniques are important to take notice of (Barret *et al.* 1994:30-34), (Cheney & Kincaid, 1999:312-315), (Bücker, 2002:540-541), since this can have an influence on the general stability of algorithms where matrices are employed. Some literature dealing specifically with convergence properties had been surveyed. Amongst others, Axelsson (2003) studied the iteration number for the Conjugate Gradient method providing insight into and explanations for the behaviour of the CGM when solving large ill-conditioned

linear systems, as might be encountered in this study. Yun and Kim (2004) studied the convergence of two-stage iterative methods using incomplete factorisation.

Whether applying SOR, CGM, BiCGM, BiCGSTAB, BiCGS or other related solution methods, conditioning of the matrices and convergence rates can be improved by using preconditioning algorithms (Barret *et al.*, 1994:39), (Bücker, 2002:540). Basic preconditioning algorithms for various methods are described extensively in Barett *et al.* (1994:39-55) and Oppe *et al.* (2002:12-17) as well as very briefly in Bücker (2002:540-541). In relation to more specialised applications, examples were selected only from a few more recent papers. Preconditioning as used in the solution of the general second order elliptic problems were described by Chen *et al.* (2000) and Zhang (2002) proposed an alternative approximate inverse preconditioner suitable for parallel preconditioning of general sparse matrices solved using Krylov methods. The accuracy of preconditioned CG-type methods as applied to least squares problems specifically was investigated by Yang (2003). Analysis of separate displacement preconditioners for matrices arising from the solution of nonlinear elasticity problems was done by Axelsson and Karátson (2004). In another type of application, the solution of linear ODEs, Ghoreshi and Hosseini (2004) made use of a new preconditioning technique in conjunction with an adapted "Tau" method.

2.2.2.3.1.2 Differential Equation Solution Techniques

First order differential equations (Kreyszig, 1993:1034-1047), (Cheney & Kincaid, 1999:410-419), a type of ODE (Cheney & Kincaid, 1999:370-409), are usually used to describe phenomena that only have variable(s) implicitly related by first derivatives to some other variable(s) in the same equation. This, informally, implies that a single integration would yield the value of the independent variable, given some initial values for the problem being solved. Integrating for location from velocity with respect to time is a simple example, as is the related integration from acceleration with respect to time to obtain velocities. Second order differential equations (Kreyszig, 1993:1048-1054), (Cheney & Kincaid, 1999:421-425) are used to describe phenomena where the second derivative of some other variable is the highest level of differentiation required, and thus also only two integrations are required for such an expression, again given some initial values. A typical example of this would be the relationship between acceleration and position with respect to time derivatives.

Solving differential equations by direct integration is not always easy, and some numerical techniques need to be employed, such as those that can be found in Kreyszig (1993:1034-1082).

For second order DEs in particular, the most popular method currently in use, at least in the field of rigid body dynamics, is the Fourth Order Runge-Kutta method (Kreyszig, 1993:1040-1043), (Hibbeler, 1995:602-604), (Cheney & Kincaid, 1999:387-389, 415-419) since it is sufficiently accurate, fast and easy to implement. The Runge-Kutta-Nyström method (Kreyszig, 1993:1051-1054) for general second order DE integration should also be taken note of. Possibly useful predictor-corrector techniques are described in Cheney and Kincaid (1999:428-436).

2.2.2.3.1.3 Non-Linear Multiple Variable Equation Solution Techniques

One of the most powerful numerical solution methods used for solving large systems of non-linear equations with easily evaluated/calculated partial derivatives is the Newton-Raphson iteration (Stoecker, 1989:124-130), (Press *et al.*, 1992:362-368, 379-383), (Cheney & Kincaid, 1999:110-113) or related, e.g. Secant, methods. Important and useful aspects related to Newton-Raphson iterations are discussed in Stoecker (1989:339-359) and Press *et al.* (1992:383-393). Being powerful and popular, and it has been used in original form or slightly modified in several paradigms, including image registration/processing (Lucas & Kanade, 1981), anisotropic nonlinear magnetic media modelling (Jänicke & Kost, 1996), image reconstruction (Åström *et al.*, 1996) as an alternative to the Gauss-Newton method and general root-finding (Erleben, 2001:236). The Newton-Raphson iteration had also been employed in some form by Shin *et al.* (2002:1090) in the analysis of dynamic characteristics of a combined cycle power plant, as well as by Pettersson (2003:45-56) in a new polynomial based approach to numeric division, amongst many others. In a related field of study, a paper by Moore and Weiss (1979) described and discussed recursive prediction error methods for adaptive estimation, and it contains a section of possible interest or relevance to the Newton-Raphson iteration (Moore & Weiss, 1979:198-199) with respect to initial guess values or correction of such values.

2.2.2.3.1.4 Unconstrained and Constrained Multiple Variable Optimisation Techniques

The basic concepts and terminology used in unconstrained dynamics need some explanation, and this is done compactly and clearly in Kreyszig (1993:1084-1086). Further descriptions for terminology used in constrained optimisation can also be found in Kreyszig (1993:1088-1104) as well as in Cheney and Kincaid (1999:548-578). Of particular importance are the definitions of an objective function, control variables and stationary points since one or all of these are used quite often in some of the literature sources procured, such as Baraff (1989), Baraff (1994), Anitescu and Hart (2000a), (2000b), Erleben (2001), Milenkovic and Schmidl (2001), Popovic

(2001), Adreani *et al.* (2002), Anitescu (2002), Anitescu and Hart (2002), Redon *et al.* (2002), Anitescu (2003), Rhagunathan and Biegler (2003), Peng *et al.* (2004).

The simplex method (Stoecker, 1989:265-279), (Kreyszig, 1993:1092-1104), (Cheney & Kincaid, 1999:592-595), or variants thereof, is most often used to solve for both constrained and unconstrained objective functions. A more recent development in use of the simplex algorithm is described in Maros (2003), namely the dual simplex algorithm. A popular method for determining optima/extrema, i.e. maxima and minima (Salas, 1995:991-998, 999-1005, 1007-1013), for constrained sets of equations (equality and inequality constraints) is called linear programming (LP) (Henrici, 1982:210-220), (Stoecker, 1989:260-291), (Kreyszig, 1993:1088-1090), (Oliver, 1993:310-320), (Cheney & Kincaid, 1999:579-587) which employs the simplex numerical method or variants thereof to determine the best solution.

The concept of Lagrange multipliers (Stoecker, 1989:161-176) is an analytical approach to finding extrema (Salas, 1995:1008-1013) subject to certain side conditions or equality constraints, although it equally well applies to unconstrained problems (Stoecker, 1989:164) but it cannot be directly used to solve for systems with inequality constraints. Although direct solution using Lagrange multipliers is not possible, repeated application of equality constraints to a problem can be used to iteratively optimise inequality constrained problems (Stoecker, 1989:448-451). A fairly recent addition to optimisation techniques is geometric programming (Stoecker, 1989:240-255) capable of solving equality constrained and non-constrained problems, and even inequality constrained problems, though they are a little more complicated in nature. Wall (1991) used a Newton-Raphson-like iteration in the field of thermo-economics for the optimisation of a heat pump.

Another method that might gain popularity for optimisation is the Gauss principle (Glocker, 1997) which had been used specifically in the determination of impulse and force magnitudes and directions (Redon *et al.*, 2002) in simultaneously colliding and contacting systems of rigid bodies. El Kahoui *et al.* (2001) discussed improved algorithms for solution of linear complementarity problems arising from collision response and described the mathematical background thoroughly (El Kahoui *et al.* 2001:72-73), (El Kahoui *et al.* 2001:73-90). A good explanatory description of the linear complementarity problem (LCP) and solution techniques for it can be found in Erleben (2001:42-48).

Xiu *et al.* (2001) and Noor (2003) discussed the usage of tangent projection equations in the analysis of convergence behaviour of iterative solution techniques for variational inequalities.

Tangent projection methods also provide alternative solution approaches for solving various kinds of variational inequalities. Yang (2003) did some investigations on the accuracy of the preconditioned CGM for Least Squares type problems, which, incidentally, are always symmetric (Johnson *et al.*, 1993:69, 196-212) and, thus, quicker to solve and more accurate, and, since the least squares method is a type of optimisation technique, it might be of relevance to the present study in some form. An improved simplex algorithm with primal and dual pivoting, suitable for possible use in LCP approaches, had been proposed by Chen *et al.* (1993) and it might be worth considering some or all of the principles used therein for further developments for optimisation techniques. On the possibly applicable subject of Least Squares approximations, Fasshauer (2004) investigated the use of approximated Jacobians with matrix free techniques for approximate Moving Least Squares algorithms. A slightly different approach to solving variational problems were followed by Hsiao (2004a), (2004b), using Haar-wavelets, warranting further investigation of its applicability in the field of concurrent contact resolution using LCP approaches.

2.2.3 Computer Programming

Computer programming is the process of specifying instructions to be executed by a computer. The instructions to be executed is usually stored in text format on some kind of medium, such as a magnetic (hard disk, floppy disk) or an optical (CD, DVD) storage device and always in the computer's native language called ASSEMBLER or ASSEMBLY. The ASSEMBLER instructions have to be generated somehow, and this is done by some form of translator or compiler, which basically interprets one of the various available human-like instruction sets (called higher level computer languages) and translates them into native ASSEMBLER instructions, which can be interpreted and executed by the processor. Computer programming is swiftly becoming evermore accessible to a larger subset of the modern population as general- and computer literacy progresses, due to the efforts of various software development companies to simplify and reduce the process of programming to such an extent that one scarcely needs to type a single line of code, one can simply point and click on a "concept", and the rest is done "automagically" like an Australian friend of mine (thanks, Martin MacRobert!) often remarks.

The greater availability and user friendliness of programming tools still does not necessarily guarantee that there will suddenly be an abundance of sufficiently skilled programmers available for development of complex and specialised code required by the engineering disciplines in particular, which is not necessarily such a bad thing, since it does strengthen the market position

of those engineers who do decide to take that route. Computer programming is becoming a very important, and in some cases indispensable, part of modern engineering analyses. The reason for this is the increasing reliance of engineering design processes on computational methods to simulate and investigate evermore complex problems and phenomena at least at some stage of the development cycle, usually in the preliminary/conceptual design phase, since physical prototyping could be both time consuming and expensive, as well as impact on the already limited natural resources used in the manufacture of such articles. The complicated solution techniques necessary for such complicated analyses also require evermore efficient programming techniques or increasingly powerful processors. The latter option is, of course, usually the more expensive one, and should be avoided as far as possible until the option of more efficient programming had been pushed to the limits. This practically leaves the engineer with only one best option: code better, more efficiently, more maintainable and more re-usable (Heller, 1995).

2.2.3.1 Programming Languages

The most basic computer language and thus also the lowest level of computer languages, is a language that the computer's processor would always understand, called the ASSEMBLY or ASSEMBLER language. Many new generation programmers not formally trained in computer science, never learn how to utilise ASSEMBLER, since really complicated operations have to be performed using the minimum set of instructions available to control hardware, manipulate memory and perform arithmetic operations. Apart from making it difficult (and thus, time consuming) to program, using a very basic set of instructions makes coding extremely error-prone. Fortunately for us, most of the very basic hardware handling is done by a pre-defined set of instructions, dubbed the Basic Input-Output System (BIOS) – formally defined as the hardware-abstraction-layer, and it is provided as standard software with every computer's motherboard. Most basic (and treacherous) memory handling and hardware operation is also taken care of by an Operating System (OS) which one needs to procure and install on the hard disk (though there are versions that run off storage media such as floppy drives and CDs). This basic set of enhanced ASSEMBLER instructions provide the most important and basic functionalities necessary for more complicated sets of instructions to be executed, but that is, unfortunately, where it ends, and this still leaves the programmer with quite a lot of additional work to be done for performing the simplest of tasks, such as calculating the values of simple trigonometric expressions (e.g. sin(), cos(), tan(), etc.) and performing repetitions of sub-instructions or specifying execution flow paths. This is where the value and use of higher level computer programming languages become clear (Stroustrup, 2000:9). Many higher level

computer languages, such as ADA, BASIC, COBOL, FORTRAN, Pascal, LOGO, C, Objective C, C++, Java and C# had been developed throughout the years since the initial use of punch cards in the 1960s, all of which have their own specific strengths and weaknesses. Personal experience and extensive use (and a slight amount of indoctrination throughout the past 9 years) had proven C++ to be a trusty companion to the engineer aiming at developing efficient and fast, yet highly maintainable applications with mainly numerical goals in mind. C++ furthermore lends itself extremely well to the object oriented (OO) design paradigm, since it supports all the constructs and concepts necessary for object oriented programming (OOP), see Martin and Odell (1992) for detailed discussion, but before a final decision can be made about the choice of programming language, some alternatives should at least be considered. Very recent additions to OOP languages are the Java language, developed by Sun Microsystems™, and C# (pronounced C-sharp, as in the musical term), developed by Microsoft™. Both Java and C# are aimed at the rapid development of Graphical User Interface (GUI) driven applications with minimum additional user coding for as wide an operating system range as possible. Java has the upper hand in this endeavour, for the moment, and it has proven to be really handy with respect to making it possible to run visually driven or visually oriented programs on a very wide range of operating systems with the use of the Java Virtual Machine (JVM), which is basically an in-time translator to the ASSEMBLER instructions native to various machines, but also taking into account OS-specific behaviour, such as the form and functionality of windows, mouse pointer behaviour, error and warning message reflection, GUI control behaviour, etc. Both Java and C# appear to be visually-enabled extensions to or supersets of C++, and, although they might have been re-designed from basic principles and apparently dangerous or useless concepts were either totally discarded (e.g. pointers), or their functionality were restricted to the absolute minimum, they still have the look and feel that basic C as well as C++ had lent to the programming community. With respect to restricting or removing language features, Stroustrup (2000:9) comments that "...restricting language features with the intent of eliminating programmer errors is at best dangerous." and "Good design and the absence of errors cannot be guaranteed merely by the presence or absence of specific language features."

What also makes C++ attractive is that, lack of GUI functionality aside, it is every bit as platform-independent as Java, and it is, in fact, used as the development language of choice on several UNIX flavours, as well as for all Linux flavours. The GNU's Not UNIX (GNU) project also contributes to platform independence by providing a (free) American National Standards Institute (ANSI) and International Standards Organisation (ISO) compliant GNU C++ compiler

for almost every imaginable OS which is regularly revised and updated. Only a few drawbacks exist for the standard ANSI/ISO C++ user: it does not currently support threading (letting different parts of a program run on separate processor paths), it does not have default generic visual components for GUI transportability, it does not fully support user defined type information (i.e. what member variables exist, the name of the type itself, etc.) extraction for all user defined types and it does not by default have observer (Gamma *et al.*, 1995:293-303) or any other design pattern concepts (Gamma *et al.*, 1995) built in, as is the case with Java. Drawbacks notwithstanding, C++ remains the programming language of choice for engineering applications requiring rapid calculations ("number crunching") or processing of vast amounts of data whilst still utilising an OOP approach and gaining from the resulting maintainability. In the following section the various sources of C++ knowledge that had been utilised throughout this study can be found.

2.2.3.1.1 The C++ Programming Language

A brief historical overview of the development of C++ is given by Stroustrup (2000:10-11), who is the original designer and implementer of C++, and this section provides the reader with some reasons for the incorporation of and the original inspiration for the extensions added to the C programming language. Firstly, it should be stressed that C++ had been designed to be useful for various levels of expertise, including the apprentice programmer, just starting out, right through to the most advanced levels of programming imaginable. Stroustrup (2000:7) states that "you will be using C++ – often for building real systems – before understanding every language feature and technique. By supporting several programming paradigms ... C++ supports productive programming at several levels of expertise." and furthermore "C++ is organised so that you can learn its concepts in roughly linear order and gain practical benefits along the way." What this basically implies is that you do not need to first become OOP literate (as might be the case with Java and C#) to start writing basic programs in C++ and learning about basic C++ functionality and concepts, i.e. C++ supports incremental learning. A basic text on C++, such as Cohoon and Davidson (1999), would thus be sufficient to introduce oneself to the C++ programming experience and, in fact, it would be wise not to tackle the Stroustrup (2000) text right away, since it is tough reading for a novice. The text by Stroustrup (2000) is, however, very useful once the more intricate peculiarities of C++ need to be investigated or understood, as well as explained, in some cases, and is an essential part of the C++ programmer's repertoire of references.

Once basic concepts of C++ had been mastered, one can move on to the OOP paradigm, as described, for instance, in Martin and Odell (1992) or Nielsen (1995), and which is often referred to by Stroustrup (2000:37-39, 301-302, 692, 726). Having been introduced to OOP, one can start to look at ways to improve the execution time performance of compiled code by utilising techniques such as described in Heller (1995). Large software projects usually start to bloat terribly, code-wise, and the optimisation of file inclusion and linker dependencies for the software project, such as described in Lakos (1996) becomes essential to minimise compilation times and thus maintain or improve the productivity of a programmer-team (a project that compiles for a long time, wastes time during debugging when several re-compilations might be required, and it also impacts on normal production builds). Armed with an array of applicable theory, one then has to look at compendia, such as Oliver (1993), Lippman and Lajoie (1998) and Friedman *et al.* (1999) containing useful algorithms, solutions and, sometimes, also implementation details for several oft-encountered programming problems.

2.2.3.2 Program Development and Design

The development and design of software is an engineering discipline in its own right. A good overview of the software development and design process can be found in Stroustrup (2000:691-764), where some aspects of the design and implementation processes are highlighted. Further considerations to take into account for large scale systems can also be found in Lakos (1996). A new trend that is emerging is the use of design patterns (Gamma *et al.*, 1995) to address recurring problems uniformly. Design patterns can be more easily implemented or applied when object oriented programming techniques, such as described in Martin & Odell (1992), Nielsen (1995) and Stroustrup (2000), are applied. The C++ programming language specifically lends itself to very efficient and easy use of design patterns by enabling the programmer to implement concepts using objects, in the form of classes, for re-usability, as well as further generalisation of design patterns, and thus even better re-usability, through the feature of template classes. One should always view design patterns, general OOP techniques and all the other factors named as one big collaborative and interactive collection of techniques to simplify and guide the design process as a whole. OOP works, and provides many answers, but it is not the alpha and the omega of programming. Neither are design patterns. Where applicable they can, and should, be used to the advantage of the programmer, but one should always strive to not compromise on execution and compilation times as far as possible, which is one of the main drawbacks of using a highly and purely object oriented language such as Java.

2.2.3.2.1 Algorithms

An algorithm is a set of instructions to be completed to obtain a required or desired output or outputs given some input or inputs. Apart from the design paradigms and approaches to be selected from or combined and eventually implemented, there are some basic algorithms that can (and usually should) be implemented and re-used. These include basic linear algebra routines (Rorres & Anton, 1979), (Henrici, 1989), (Press *et al.*, 1992), (Kreyszig, 1993), (Oliver, 1993), (Barret *et al.*, 1995), (Bücker, 2002), (Oppe *et al.* 2002) preconditioning routines (Press *et al.*, 1992), (Barret *et al.*, 1995), (Oppe *et al.* 2002) initial value problem solution routines (Cheney & Kincaid, 1999), optimisation routines (Stoecker, 1989), (Wall, 1991), (Press *et al.*, 1992), (Cheney & Kincaid, 1999), search routines (Lippman & Lajoie, 1998), non-linear simultaneous equation solution routines (Stoecker, 1989), (Wall, 1991), (Press *et al.*, 1992), (Cheney & Kincaid, 1999). These basic routines are usually employed at some stage or another within more complex routines or algorithms, which are obtained from literature such as those in the papers and texts surveyed and discussed in the earlier sub-section on physics. Some custom algorithms might also be developed to deal with specific problems or challenges encountered during the course of this study.

2.3 Summary of Literature Survey – Current States of Affairs

This section re-capitulates and summarises the current state of the art in the various fields of interest relevant to the present study as indicated by and deduced from the literature reviewed. Only the most recent and relevant literature sources are considered, since they already encompass most of the earlier work done, and where that is not the case, the earlier work is discussed on its own. Relevant suggestions for further work obtained from the literature considered are also provided as part of the formal motivation for this study.

2.3.1 Mechanics

Mechanics is a very comprehensive discipline, and the advent of the digital computer caused a dramatic increase in the advances in research seen in its various divisions, e.g. fluid mechanics, structural mechanics, rigid body mechanics, etc., especially in more recent years. Despite the greater efficiency with respect to cost and time, as well as increasingly easy user interfaces for most mechanics modelling tools, they are not that popular yet, since the general opinion regarding reliability of non-physical design paradigms is still not very positive. This will change in time as more verification and validation results are obtained and published.

In rigid body mechanics, the topic of free moving rigid body motion had not advanced much in recent years, apart from improvements to the motion integrators employed (Faure, 1997), (Keller *et al.*, 1998:21-23), (Erleben, 2001:49-56), (Holmlund, 2002:17-18), (Celledoni & Owren, 2003:425-428), (Klein, 2003:4), of which the Fourth Order Runge-Kutta integrator (Keller *et al.*, 1998:21-22), (Erleben, 2001:51-52) appears to be the most popular. In contrast to the aforementioned free body dynamics, the topic of interactive rigid body dynamics – especially involving multiple concurrent contacts – appears to be continuously growing and is being investigated by several researchers across the globe, such as Anitescu (2003), Anitescu and Hart (2002), Baraff (1999), Benedetti (2002), Chatterjee and Ruina (1998), Cline (2002), Egan (2003), El Kahoui *et al.* (2001), Gilardi and Sharf (2002), Glocker (1997), Herrmann and Luding (1998), Holmlund (2002), Hoomans (1999), Kawachi *et al.* (1997), Kraus and Kumar (1997), Kraus *et al.* (1997), Kraus *et al.* (1998), Lee *et al.* (1994), Luh (2000), Mirtich and Canny (1995), Mirtich (2000), Pang and Trinkle (1996), Popovic (2001), Raghunathan and Biegler (2003), Redon *et al.* (2002), Sauer and Schömer (1998), Schmidl and Milenkovic (2003), Song *et al.* (1999), Stewart and Trinkle (2002), Stronge (2003) and Tzitzouris (2001). A more detailed summary of the current state of interactive dynamics research can be found in the following section.

2.3.1.1 Contact- and Collision Resolution Methods

Several researchers (Keller *et al.*, 1998:15), (Anitescu *et al.*, 2003:1) pointed out the *need for development of* a *multiple concurrent contact resolution algorithm and* the *advantages of physically based modelling* (Baraff, 1999:A1), (Holmlund, 2002:2-3, 6). Gilardi and Sharf (2002) provided a very thorough summary and overview of all the most recent research and the state of collision and contact modelling in general at that time and had been a direct and source of most references with respect to rigid body collision and contact modelling for this study. As mentioned earlier, they stated that there are basically two main divisions in the rigid body contact simulation arena, namely the continuous contact (force-based) models and discontinuous/discrete (impulse-momentum) models (Gillardi & Sharf, 2002:1214).

According to Gilardi and Sharf (2002:1235-1236), discrete models pose several problems, such as the determination of the appropriate coefficient of restitution and collision models employing these coefficients of restitution. The *discrete formulation* also *appeared to be difficult to extend to multi-point impact scenarios in general* when *applying* the *available solution methods to* the *formulations currently known and* that the use of *Coulomb's friction law may lead to inconsistencies or multiple solutions* (Gilardi & Sharf, 2002:1236). In contrast to the discrete

contact model, the continuous contact models were considered to be more versatile, since one can model both static and dynamic contact and it is easily extendible to multi-contact scenarios (Gilardi & Sharf, 2002:1236). The *possible problems* that might be *encountered in highly dynamic systems of very rigid bodies with high relative velocities, such as* the *difficulty of determining the correct spring and damper coefficients at various approach velocities and also possibly numerically unstable differential equations* (Mirtich, 1998:11), (Holmlund, 2002:38), as well as *excessive (i.e. physically non-realistic) overlaps*, seem to have been *neglected.* Much research and investigation had been conducted in the discrete formulation field for both static and dynamic contacts, and the *main approaches for multiple-contact calculations* tend to be *centred around optimisation of non-linear inequalities* using various techniques, the latest of which are the Linear Complementarity Problem (LCP), Non-linear Complementarity Problem (NCP), Mathematical Program with Complementarity Constraints (Anitescu, 2003) and Gauss' Principle of Least Constraints. The discrete contact methods encountered in the literature all appear to approach the solution of multiple contact problems in a somewhat theoretical manner with regards to the mathematics employed (e.g. references to abstract matrix and vector theory abound, sometimes confusingly so).

A discrete contact approach that generated interest regarding the present study was proposed by Ruspini and Khatib (1997:3-6), extending the impulse-based approach developed by Mirtich (1996) and then casting it in an LCP format to utilise solution techniques for such problems, as, for instance, described by Erleben (2001:42-48), Anitescu and Hart (2002) or Trinkle (2003). *This study* will be *unique* since it will be less based on optimisation type strategies – of which LCP, NCP, MPCC and Gauss' Least Constraints all are examples – and *focus* more *on the basic principles of physics* as far as possible, utilising only the absolutely essential mathematical techniques where applicable. ***The reason for attempting to develop a more physically based approach is that optimisation strategies for systems with non-linear constraints are not necessarily guaranteed to have unique, and thus physically accurate, solutions for the frictional contact problem*** – see Kraus *et al.* (1998:1-2), Mirtich, (1998:11) and Milenkovic and Schmidl (2001:2).

The *proposed physically-based approach*, and *in particular* the use of the *impulse-based contact formulation for multiple concurrent contacts*, was *inspired by* a combination of the *work done by* Ivanov (1995), Kawachi *et al.* (1997), Baraff (1999), Mirtich (1996), (1998), Ruspini and Khatib (1997), Kraus *et al.* (1998), Milenkovic and Schmidl (2001), Tzitzouris (2001), Holmlund (2002), Redon *et al.* (2002), Giang *et al.* (2003), Noor (2003), Schmidl and Milenkovic (2003),

Trinkle (2003) and Hsiao (2004a), (2004b), as well as preliminary work done by Neethling (1998), amongst others. It is *hoped* that, by selectively combining some or all of the approaches and supporting solution techniques encountered in the literature, a *more physically accurate yet simpler algorithm* for *handling multiple concurrent contacts for a system of rigid bodies can be developed.*

2.3.2 Mathematics

The general current state of the mathematical art cannot possibly be represented in any measure of completeness within this study, since only parts relevant to rigid body simulation were considered, but the current situation of mathematical disciplines for these relevant fields of interest can be given. These states of the art for the various mathematical techniques hitherto employed are briefly discussed in the following sections.

2.3.2.1 Linear Algebra

Basic linear algebraic theory had not really advanced that much in recent years, if one is to judge by the uniform content of texts containing sections on linear algebra, e.g. there is not much difference between the contents of Rorres and Anton (1979:9-35), Johnson *et al.* (1993:117-214) or Kreyszig (1993:325-564). On the flipside, the usage or application of basic linear algebraic theory is advancing at a tremendous rate. Papers from diverse fields of interest – e.g. computer graphics (Baraff, 1999), (Erleben, 2001), (Anitescu, 2003), engineering (Trinkle, 2003:3-10), mechanics (Burov, 2003:641-646), pure and applied/numerical mathematics (Chen *et al.*, 1993), (Xiu *et al.*, 2001), (Noor, 2003), (Alanelli & Hadjimos, 2004), etc. – attest to the use of linear algebra as a tool for simplifying, organising and even solving expressions (with the aid of vectors and matrices) in problems that might otherwise appear extremely complex and confusing to comprehend.

2.3.2.2 Calculus

As in the case of linear algebra, the theory for basic calculus had remained largely unchanged in recent times. Also similar to the situation for linear algebra, the application of calculus principles, theory and techniques have increasing relevance in various other fields of interest. As an example, a recently proposed, more efficient, accurate and reliable optimisation strategy was devised using basic calculus and function theory, combined with linear algebra where needed (Xiu *et al.*, 2001), (Celledoni & Owren, 2003), (Noor, 2003). Basic calculus is also being used as basis for the derivation of more efficient or accurate numerical algorithms applied in other contexts, such as error analysis, function approximations, differentiation and integration,

amongst others. The most useful basic theory appears to be the multiple variable Taylor series expansion used, for instance, in the various Runge-Kutta integration schemes and the Newton-Raphson/Secant root finding algorithms. Calculus can still be a rich source of ideas, concepts and theory to be mined in the quest for more efficient, elegant or accurate methods to be employed in various disciplines.

2.3.3 Numerical Analysis

Numerical analysis had gained in popularity, applicability and general relevance for various disciplines in recent times due to the greater affordability, general availability and relatively powerful processing capabilities of the digital computer. A "symbiotic", constructive interaction exists between the development of theory for numerical analysis and the development of algorithms from such theory. This happens since advances in numerical analysis theory leads to improved numerical algorithms when implemented and in its turn numerical analysis theory improves due to the greater insights into behaviour of numerical techniques gained from studying the results obtained from preceding implementations.

2.3.3.1 Numerical and Iterative Solution Methods for Linear Systems

In recent years, the focus and emphasis in numerical analysis pertaining linear algebra had been on the development of more computationally and memory efficient algorithms to deal with the solution of increasingly larger systems of linear equations (Press *et al.*, 1992:71-89), (Barret *et al.*, 1994), (Bücker, 2002), (Oppe *et al.*, 2002), (Alanelli & Hadjimos, 2004), as well as the analysis (Axelsson, 2003), (Yang, 2003), (Axelsson & Karátson, 2004), (Yun & Kim, 2004) and improvement (Zhang, 2002) of the convergence properties of these algorithms. Understanding the convergence properties and other aspects of linear system solution techniques is important, since the solution of non-linear systems of equations and inequalities depend either directly or indirectly upon them.

2.3.3.2 Multi-Variable Solution Techniques for Non-Linear Systems

In the relevant literature surveyed, multi-variable solution techniques all seem to be increasingly based on some form of the Newton-Raphson algorithm derived from the first-order Taylor expansion for multiple variable functions (Stoecker, 1989:124-130), (Press *et al.*, 1992:362-368, 379-383), (Cheney & Kincaid, 1999:110-113). Though the Newton-Raphson algorithm is very powerful with excellent convergence properties for good guess values, it diverges as rapidly when bad initial values are guessed, but there seem to be some techniques available to overcome this problem, such as the Globally Convergent Method (Press *et al.*, 1992:362-368, 383-393). If

a system is convergent, the rate of convergence can be accelerated employing a technique proposed by Homeier (2004), i.e. a modified Newton-Raphson technique using a "half step" for improved convergence, with quadratic convergence properties.

2.3.3.3 Multi-Variable Optimisation Techniques

Optimisation techniques are related to or dependent on multi-variable solution techniques in several ways, since the minimisation or maximisation of certain expressions or enforcement of some constraints are usually combined with non-linear system solution techniques, such as Newton-Raphson or Secant, which, in turn, is reliant on solution techniques for large linear systems. There are several examples of how the various mathematical disciplines are used in collaboration or combination to achieve better solutions to optimisation and related problems (Bruynickx & Khatib, 2000), (El Kaoui *et al.*, 2001), (Adreani *et al.*, 2002), (Horowitz & Afonso, 2002), (Redon *et al.*, 2002), (Maros, 2003), (Noor, 2003), (Hsiao, 2004b).

2.3.4 Computer Programming

Computer programming had become an indispensable part of numerical analysis, since it is the only way that the algorithms devised by researchers can be converted into functioning and usable tools for computer simulation of phenomena too complex to be studied otherwise. Only the state of the art for selected aspects of computer programming relevant to rigid body simulation and the associated numerical algorithms are discussed in the following sections.

2.3.4.1 Programming Languages

Programming languages had evolved into highly specialised tools for organised and high-level computer instruction specification. The focus is more on rapid application development, and the latest programming languages available, such as Visual C++ .NET™, C# .NET™, Java, VB .NET™, Objective C, etc., support quick and easy Graphical User Interface (GUI)-development as well as the latest design paradigms, e.g. Object Oriented (OO) and the use of Design Patterns. Even though each of the programming languages have their own particular sets of advantages, the standard ANSI/ISO compliant C++ programming language (and Visual C++ .NET in particular) remains the vehicle of choice for coding in the engineering field, since it allows for hybrid coding, e.g. procedural and object-oriented mixed, as well as meta-programming (using expression templates) and the use of templates in general.

2.3.4.2 Program/System Design

The design of software had become an important aspect in almost every field of interest known today, since the use of software had extended into almost every niche imaginable. Object oriented (OO) design is the current topic, paradigm and approach of choice amongst programmers. Object oriented programming techniques pose several advantages, especially with respect to maintainability and modularity, but one should remember that there are some penalties with respect to execution speed and memory usage, so moderation is the key. Along with the object oriented paradigm, comes the usage of design patterns to address certain recurring problems encountered by programmers, since most of these patterns rely on the knowledge about and usage of object oriented approaches. Programming languages that support object oriented programming, such as C++, Java, etc., is of cardinal importance today and it will become even more so in future.

2.4 CONCLUSION

This literature survey unequivocally identified several shortcomings or opportunities for further work in various fields of interest. Since the main focus had originally been on rigid body simulation, it was decided to *pursue the development of an improved numerical algorithm for multiple concurrent contacts (cluster collisions) in rigid body systems based on impulse and momentum balance equations in three dimensions.* This is due to the fact that a simpler contact resolution model can be useful in general study, understanding and teaching of system momentum principles at undergraduate level (physics). Handling multiple concurrent contacts if and when they do occur in 3-D engineering applications, especially where coupled with flow-grids such as in fluidised beds (Hoomans, 1999) is also a possible application. The gaming and general computer simulation community might also benefit from a simple general multi-contact capable solution algoritm.

Several methods of addressing the non-linear equations encountered and inequality constraints involved in contact problems had also been identified and will be evaluated for merit in the course of this study. After finding or developing the best algorithm or algorithms for handling the problem obtained from the approach taken, the C++ implementation should be able to handle cluster collisions of at least more than two simultaneously colliding spheres and can serve as a basis for further investigation into multiple concurrent collision phenomena. This then concludes the literature survey and the synthesis of existing theory and possible development of new theory can now commence in the following chapter (Chapter 3).

Chapter 3
Theoretical Background

In this chapter, the theory used in the current study is discussed, analysed, developed and motivated where necessary. Theory necessary for free body motion, Newtonian rigid body interaction, analytical geometry, collision searches, collision resolution, matrix solution techniques, multi-variable solution techniques and multi-variable optimisation techniques are all discussed here, and improvements or modifications are suggested where considered necessary.

3.1 Basic Theory of Mechanics

Only a small subdivision of the full discipline of mechanics is applicable to the present study, since it concentrates on the motion and interaction of rigid bodies – spheres to be precise. In this section the little theory is summarised and discussed. Conventions adopted with respect to mathematical expressions in this study need to be highlighted first. In all the equations shown, vector quantities are always underlined (e.g. $\underline{x}_i$), matrices/tensors are capitalised non-underlined letters (e.g. M_i), scalars are normal non-underlined letters (e.g. $\lambda_{n,ij}^{nb}$) and first derivatives of any variables with respect to time are indicated with a dot (e.g. $\dot{\underline{x}}_i$). Also to be noted is that superscripts do not refer to exponents, but are used to keep expressions as compact as possible, unless otherwise stated.

3.1.1 Rigid Body Mechanics

As might be deduced from definitions given earlier, the term "rigid body mechanics" is self-explanatory. The motion, interactions and associated behaviour of rigid bodies are of interest in mechanics, and basic theory for these phenomena are described in the following sections.

3.1.1.1 Free Body Motion

Free body motion is the motion of a body not interacting with another in any way other than through mass related gravitational influences. Any linear displacement of the centre of mass of a body in a three dimensional domain is termed translation, whilst any angular displacement of a body round its own centre of mass is termed rotation. Since the earth is by far the most proximate body involved in earthbound applications, it is safe to assume that the influence of other stellar bodies can be neglected and only the gravitation due to earth needs to be considered.

Furthermore, the small charges present on most rigid bodies can only generate negligible forces and torques due to electromagnetic influence compared to the gravitational force. Following these assumptions, the only external forces and torques experienced by all bodies in simulations for this study are due to gravity and drag, thus leading to the modified Newton's equation for linear motion as seen in equation (3.1) below.

$$\frac{\partial}{\partial t}[M_i \frac{\partial}{\partial t}(\underline{x}_i)] = \underline{F}_i - C_{dp,i} \frac{\partial}{\partial t}(\underline{x}_i) \tag{3.1}$$

In equation (3.1) the $\underline{F}_i$ and $\underline{x}_i$ vectors respectively represent the external force and linear position and the $C_{dp,i}$ tensor is the profile drag coefficient of rigid body *i*. The mass tensor, M_i, in equation (3.1) is simply the diagonal matrix as shown in equation (3.2) below, representing the linear inertia of any rigid body.

$$M_i = \begin{bmatrix} m_i & 0 & 0 \\ 0 & m_i & 0 \\ 0 & 0 & m_i \end{bmatrix} \tag{3.2}$$

Similar to the linear case, a Newton's equation for angular motion, as seen in equation (3.3) below, can be formulated.

$$\frac{\partial}{\partial t}[I_i \frac{\partial}{\partial t}(\underline{\theta}_i)] = \underline{T}_i - C_{dt,i} \frac{\partial}{\partial t}(\underline{\theta}_i) \tag{3.3}$$

In the case of the angular momentum equation, the vectors $\underline{T}_i$ and $\underline{\theta}_i$ respectively represent the external torque angular orientation, the tensor $C_{dt,i}$ represents the tangential viscous drag of rigid body *i*. The angular inertia tensor of any rigid body is represented by the matrix, I_i, the components of which are shown in equation (3.4) below, with $I_{i,j,k}$ the tensor component in row *j*, column *k*, ρ_i the density, $\underline{r}'$ the general radius relative to the centre and dV_i the infinitessimal volume of rigid body *i*.

$$I_{i,j,k} = \int_i \rho_i (\underline{r}' \bullet \underline{r}' - r'_j r'_k) dV_i \tag{3.4}$$

A further simplifying assumption had been made by neglecting the effects of viscous drag for the time being, resulting in yet simpler equations. The equation for linear motion can then be written as in equation (3.5) below, which is very familiar to most students having studied basic physics at some stage.

$$\frac{\partial}{\partial t}[M_i \frac{\partial}{\partial t}(\underline{x}_i)] = \underline{F}_i \tag{3.5}$$

Similarly, the equation for angular motion reduces to equation (3.6) below.

$$\frac{\partial}{\partial t}[I_i \frac{\partial}{\partial t}(\underline{\theta}_i)] = \underline{T}_i \tag{3.6}$$

That concludes this section on the basic theory relevant to free body motion as applicable to the present study.

3.1.1.2 Rigid Body System Properties

Related to the theory within the foregoing section, some theory for the properties of systems of rigid bodies could be of importance as well and this section briefly discusses them, starting off with the expression for the total mass of the whole rigid body system (m_{total}), as shown in equation (3.7) below, where each individual rigid body has the mass m_i.

$$m_{total} = \sum_{i=1}^{bodies} m_i \tag{3.7}$$

The centre of mass ($\underline{x}_{cm}$) for such an assembly of rigid bodies can then be described as in equation (3.8) below.

$$\underline{x}_{cm} = m_{total}^{-1} \sum_{i=1}^{bodies} m_i \underline{x}_i \tag{3.8}$$

And then the average velocity of a rigid body system ($\underline{\dot{x}}_{cm}$) can be written as in equation (3.9) below.

$$\underline{\dot{x}}_{cm} = m_{total}^{-1} \sum_{i=1}^{bodies} m_i \underline{\dot{x}}_i \tag{3.9}$$

The total angular inertia of a rigid body system (I_{total}) can be computed using the expression in equation (3.10) below, where $\underline{r}'_i$ is the position radius vector of rigid body i relative to the system centre of mass ($\underline{x}_{cm}$).

$$\begin{aligned} I_{total} &= \sum_{i=1}^{bodies} [I_i + m_i \underline{r}'_i \underline{r}'^T_i] \\ &= \begin{bmatrix} I_{i,xx} + m_i r'_{i,x} r'_{i,x} & I_{i,xy} + m_i r'_{i,x} r'_{i,y} & I_{i,xz} + m_i r'_{i,x} r'_{i,z} \\ I_{i,yx} + m_i r'_{i,y} r'_{i,x} & I_{i,yy} + m_i r'_{i,y} r'_{i,y} & I_{i,yz} + m_i r'_{i,y} r'_{i,z} \\ I_{i,zx} + m_i r'_{i,z} r'_{i,x} & I_{i,zy} + m_i r'_{i,z} r'_{i,y} & I_{i,zz} + m_i r'_{i,z} r'_{i,z} \end{bmatrix} \end{aligned} \tag{3.10}$$

The average angular velocity ($\underline{\dot{\theta}}_{cm}$) for such a system can then be calculated using equation (3.11).

$$\underline{\dot{\theta}}_{cm} = I_{total}^{-1} \sum_{i=1}^{bodies} [I_i \underline{\dot{\theta}}_i + \underline{r'}_i \times m_i \underline{\dot{x}}_i] \tag{3.11}$$

The position vector for body i relative to the system centre of mass, $\underline{r'}_i$, seen in equations (3.9) and (3.10) can be calculated using equation (3.12) below.

$$\underline{r'}_i = \underline{x}_i - \underline{x}_{cm} \tag{3.12}$$

Using the general expression for resultant velocity due to angular and linear components relative to any point ($\underline{p}$) in a Cartesian coordinate system, as shown in equation (3.13), an expression for the resultant velocity at any position ($\underline{r'}_k$) in terms of the system centre of mass, as shown in equation (3.14), can be derived.

$$\underline{\dot{x}}_{total} = \underline{\dot{x}} - (\underline{r} - \underline{p}) \times \underline{\dot{\theta}} \tag{3.13}$$

$$\Rightarrow \underline{\dot{x}}_k = \underline{\dot{x}}_{cm} - \underline{r'}_k \times \underline{\dot{\theta}}_{cm} \tag{3.14}$$

Some of the foregoing equations and expressions might be employed to improve the initial guess values for the contact tangent vectors ($\underline{t}_{ij}^{nb}$) to be determined by some or other iterative procedure.

3.1.1.3 Rigid Body Interaction

Rigid body interaction refers to how rigid bodies in continuous or momentary contact behave as an assembly in general. From the literature surveyed it could be deduced that one can employ either a continuous or a discrete contact model (Gilardi & Sharf, 2002:1214), both with their unique advantages and disadvantages. For this study, a form of discrete contact model had been selected since it does yield physically more realistic results for nearly perfectly rigid bodies when inter-body forces or impulses are very large (thus avoiding the problem of excessive inter-penetration). The investigation will be restricted to momentary contacts (i.e. collisions) for this study and hence only collision modelling theory will be discussed. Furthermore, since this study is mainly aimed at providing accurate basic analytical expressions for more detailed study of multiple concurrent contacts by extending the simplest existing collision and momentum conservation laws. The various models of contact and accompanying theory are discussed in the following sections.

3.1.1.3.1 Continuous Contact

In the continuous contact model, contacts are considered to occur within a finite length of time, as opposed to being instantaneous singular events. That being the approach, the physical response of the actual materials comprising the bodies in contact is approximated using virtual springs and dampers at the points of contact in both the normal and tangential directions. This method is usually referred to the soft body/penalty method or Distinct Element Method (DEM) and is well suited to problems with both static and dynamic contacts. More recent work (Hustrulid, 1995), (Kraus *et al.*, 1998) also made it possible to consider all contacts simultaneously, imparting greater realism and accuracy to simulations. The spring-and-damper models tend to be numerically very unstable (Benedetti, 2002:22) for high-speed simulations and are also restrictive on time step size.

The various parameters needed to approximate collision restitution behaviour via penalty methods correctly have the added disadvantage of not being very easy to determine using experimental results since it is dependent on shape and relative approach velocities of the colliding objects. The most problematic aspect of the penalty approach is excessive overlapping/inter-penetration of bodies, especially with near perfect rigidity, when very high compressive forces occur, due to the very nature of the contact force calculation procedure (i.e. overlap/inter-penetration based penalty forces). Due to the highly rigid nature of the type of bodies (i.e. steel, graphite or nylon spheres) intended for consideration, it was decided to rather pursue the alternative hard body or instantaneous approach for better accuracy.

3.1.1.3.2 Instantaneous Contact

Discrete contact models assume that all present (sometimes concurrent) contacts are instantaneous in nature and can thus usually be resolved in one single computational step. Several methods exist for handling such assumptions, and all are especially suitable for ideal rigid body simulations, where overlaps or inter-penetrations cannot be allowed. The section on discontinuous contact in the literature survey chapter presents several popular methods available and currently employed and a creative blend of these will most likely yield a better solution technique. More about these techniques will follow in a later chapter, but first, the basic theory for collision modelling needs to be discussed in this section.

The basic theory relevant to discrete contact modelling encompasses expressions for describing interactive momentum conservation and transfer. For each rigid body in a system, the sum of all linear momentum before and after a collision should be equal, which can be expressed

mathematically as in equation (3.15) below, where a $^{+}$-superscript refers to instantaneous post-collision states, a $^{-}$-superscript refers to instantaneous pre-collision states, the vectors $\underline{n}_{ij}^{nb}$ and $\underline{t}_{ij}^{nb}$ respectively refer to the normal and tangential unit vectors and $\lambda_{n,ij}^{nb}$ and $\lambda_{t,ij}^{nb}$ respectively refer to the normal and tangential impulse magnitudes at at contact number *nb*.

$$M_i \underline{\dot{x}}_i^{+} - \sum_{nb} (\lambda_{n,ij}^{nb} \underline{n}_{ij}^{nb} + \lambda_{t,ij}^{nb} \underline{t}_{ij}^{nb}) = M_i \underline{\dot{x}}_i^{-} \tag{3.15}$$

Before discussion of the relevant theory can continue, it would be wise to first study the illustration of the contact-related unit vectors and their conventional orientations, regarding the subscripts *i* and *j*, as seen depicted in Figure 3.1 for two contacting spheres below.

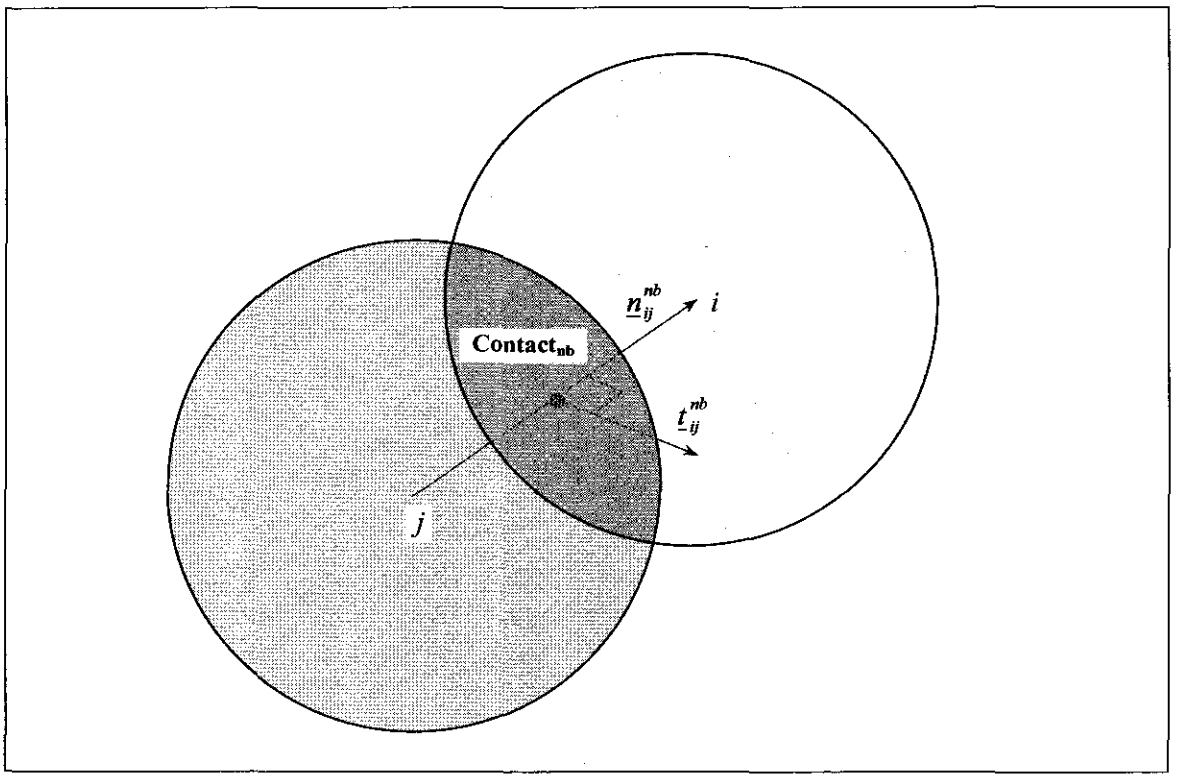


Figure 3.1: The contact normal and tangential unit vectors between spheres *i* and *j*

In equation (3.15) above, the tangential vectors ($\underline{t}_{ij}^{nb}$) all have to meet the requirement for orthogonality with respect to their associated contact normal vectors ($\underline{n}_{ij}^{nb}$) for each existing contact, which can be expressed mathematically as in equation (3.16) below, where the ^{nb}-superscript refers to the contact under consideration.

$$\underline{n}_{ij}^{nb} \bullet \underline{t}_{ij}^{nb} = 0 \tag{3.16}$$

Furthermore, the tangential vector has to be of unit magnitude, which can be specified mathematically by the expression shown in equation (3.17) below, with the ^{nb}-superscript referring to the contact being considered.

$$\underline{t}_{ij}^{nb} \bullet \underline{t}_{ij}^{nb} = 1 \tag{3.17}$$

For angular momentum, a relationship similar to the one for linear momentum holds, which can be described mathematically as in equation (3.18) below, where $\underline{r}_i^{nb}$ refers to the radial position vector of contact *nb* relative to the centre of mass for rigid body *i*.

$$I_i \underline{\dot{\theta}}_i^{+} - \sum_{nb} (\underline{r}_i^{nb} \times \lambda_{n,ij}^{nb} \underline{n}_{ij}^{nb} + \underline{r}_i^{nb} \times \lambda_{t,ij}^{nb} \underline{t}_{ij}^{nb}) = I_i \underline{\dot{\theta}}_i^{-} \tag{3.18}$$

When spheres are the types of rigid bodies under consideration, some simplifications can be made. Firstly, the position vector for all contacts in any direction relative to the sphere centre of mass is parallel to the contact normal, thus this contact position vector can be described as a scalar multiple (equal to the sphere radius) of the unit normal, and secondly, the cross product of this position vector with the contact normal would then always be zero. These simplifications then result in the modified angular momentum expression as shown in equation (3.19) below, where r_i^{nb} is the radius of sphere *i*.

$$I_i \underline{\dot{\theta}}_i^{+} + \sum_{nb} r_i^{nb} \lambda_{t,ij}^{nb} (\underline{n}_{ij}^{nb} \times \underline{t}_{ij}^{nb}) = I_i \underline{\dot{\theta}}_i^{-} \tag{3.19}$$

Momentum transfer during collisions in the normal and tangential directions can be accounted for using one of various equations relating pre- and post collision properties, such as velocity, momentum or energy, between contacting bodies. The simplest collision relationships that can be used would be Newton's equations for restitution in the normal and tangential directions, relating relative velocities in the normal and tangential directions, which is why they were selected for this study. Equation (3.20) below, implicitly employing the expression in equation (3.13), describes the normal restitution relationship in mathematical terms, where the ^{nb}-superscript once more refers to the contact under consideration.

$$\begin{aligned} &[(\underline{\dot{x}}_i^{nb,+} + r_i^{nb} \underline{n}_{ij}^{nb} \times \underline{\dot{\theta}}_i^{nb,+}) - (\underline{\dot{x}}_j^{nb,+} - r_j^{nb} \underline{n}_{ij}^{nb} \times \underline{\dot{\theta}}_j^{nb,+})] \bullet \underline{n}_{ij}^{nb} \\ &= -\varepsilon_{n,ij}^{nb} [(\underline{\dot{x}}_i^{nb,-} + r_i^{nb} \underline{n}_{ij}^{nb} \times \underline{\dot{\theta}}_i^{nb,-}) - (\underline{\dot{x}}_j^{nb,-} - r_j^{nb} \underline{n}_{ij}^{nb} \times \underline{\dot{\theta}}_j^{nb,-})] \bullet \underline{n}_{ij}^{nb} \end{aligned} \tag{3.20}$$

Equation (3.20) can be simplified further, since the tangential velocity has no normal component for a sphere, resulting in equation (3.21).

$$(\underline{\dot{x}}_i^{nb,+} - \underline{\dot{x}}_j^{nb,+}) \bullet \underline{n}_{ij}^{nb} = -\varepsilon_{n,ij}^{nb} (\underline{\dot{x}}_i^{nb,-} - \underline{\dot{x}}_j^{nb,-}) \bullet \underline{n}_{ij}^{nb} \tag{3.21}$$

Similar to equation (3.20), the tangential restitution in the direction of the tangential unit vector can be expressed mathematically as in equation (3.22) below, the ^{nb}-superscript yet again referring to the contact under consideration.

$$\begin{aligned} &[(\underline{\dot{x}}_i^{nb,+} + r_i^{nb}\, \underline{n}_{ij}^{nb} \times \underline{\dot{\theta}}_i^{nb,+}) - (\underline{\dot{x}}_j^{nb,+} - r_j^{nb}\, \underline{n}_{ij}^{nb} \times \underline{\dot{\theta}}_j^{nb,+})] \bullet \underline{t}_{ij}^{nb} \\ &= -\varepsilon_{t,ij}^{nb} [(\underline{\dot{x}}_i^{nb,-} + r_i^{nb}\, \underline{n}_{ij}^{nb} \times \underline{\dot{\theta}}_i^{nb,-}) - (\underline{\dot{x}}_j^{nb,-} - r_j^{nb}\, \underline{n}_{ij}^{nb} \times \underline{\dot{\theta}}_j^{nb,-})] \bullet \underline{t}_{ij}^{nb} \end{aligned} \tag{3.22}$$

The relationship specified in equation (3.22) will not be enough to ensure that the tangential impulse is of maximum possible magnitude (i.e. in the direction of the sliding velocity if slip were to occur), but the additional specification of zero tangential restitution relationship for the orthonormal tangential direction, such as mathematically expressed in equation (3.23), will remedy this problem, with an ^{nb}-superscript again referring to the contact under consideration.

$$[(\underline{\dot{x}}_i^{nb,+} + r_i^{nb}\, \underline{n}_{ij}^{nb} \times \underline{\dot{\theta}}_i^{nb,+}) - (\underline{\dot{x}}_j^{nb,+} - r_j^{nb}\, \underline{n}_{ij}^{nb} \times \underline{\dot{\theta}}_j^{nb,+})] \bullet (\underline{n}_{ij}^{nb} \times \underline{t}_{ij}^{nb}) = 0 \tag{3.23}$$

The phenomenon of friction, or rather slippage, makes tangential restitution less user friendly since it should allow for slippage if the tangential impulse exceeds the maximum tangential-to-normal impulse ratio for the contact under consideration, which would be the static friction factor according to the standard Coulomb friction law. The aforementioned requirement can be expressed in mathematical terms as shown in equation (3.24) below, with the ^{nb}-superscript referring to the contact under consideration as before.

$$\lambda_{t,ij,adj}^{nb} = \mu_{ij,dyn} \lambda_{n,ij}^{nb} \cup \lambda_{t,ij}^{nb} > \mu_{ij,stat} \lambda_{n,ij}^{nb} \tag{3.24}$$

Typically the relationship between normal and tangential impulse magnitude as described in equation (3.24) should replace the regular relative velocity relationship as shown in equation (3.22) only when equation (3.22) failed to result in a tangential impulse of magnitude bounded by the static friction coefficient. The approach to friction taken in this study avoids the use of an approximated friction pyramid and allows the use of the actual isotropic friction cone, which is an approach unique to this study not found in any of the references consulted.

Another problematic condition applies to normal impulses, which can never be negative for freely contacting bodies and this also has to be taken into account. Mathematically the non-negative normal impulse condition can be described as in equation (3.25) below.

$$\lambda^{nb}_{n.ij,adj} = 0 \cup \lambda^{nb}_{n,ij} < 0 \tag{3.25}$$

A combination of all the foregoing equations can be used to describe implicit relationships between the various rigid bodies in a system with concurrent contacts occurring, which then has to be solved by appropriate techniques taking into account the necessary constraints, such as described by equations (3.24) and (3.25). The solution techniques for such types of problems will be discussed in a later chapter.

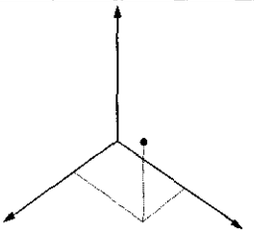
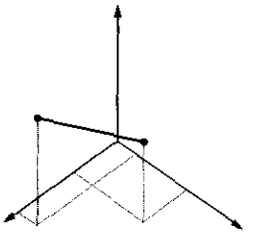
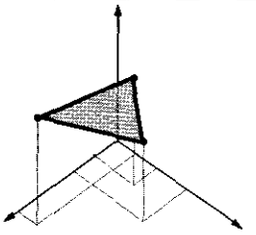
3.2 BASIC GEOMETRY

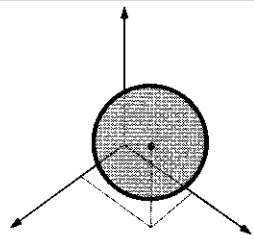
Basic geometrical theory is a key element in contact determinations and calculations. It has to be used in contact detection and also for the calculation of actual contact point parameters such as exact location and surface normal orientation. The following sections briefly summarise the analytical geometry theory necessary for the relevant calculations.

3.2.1 Analytical Geometry

Analytical geometry refers to the branch of mathematics utilising algebraic and calculus concepts to represent various geometrical concepts, such as curves, intersections, volumes and surfaces. The basic geometrical entities used in this study are summarised in Table 3.1 below.

Table 3.1: Basic geometric entities

Entity/ Concept	Depiction	Description	Properties
Vertex		A point in 3-D space described by a triad of real numbers representing Cartesian co-ordinates.	- 3-D Vector - Magnitude - Direction
Edge		A straight line in 3-D space connecting and bounded by two vertices.	- 3-D Vector - Length - Direction - Offset Vertex - Bounded
Facet		A perfectly flat 2-D surface oriented in 3-D space, bounded by and sharing three vertices with three edges.	- Triangular - Surface Area - Orientation - Bounded

Entity/ Concept	Depiction	Description	Properties
Sphere		A perfectly smooth-surfaced volumetric entity with its entire surface perfectly equidistant from its volumetric centre.	- Surface Area - Volume - Orientation - Centre of Volume

In this study, the spheres are the only moving objects considered, but their interactions with or relationships to other geometrical entities such as vertices, edges and facets, need to be described mathematically. Before one can tackle the problem of determining inter-object distances, possible closest neighbours to be checked for interaction need to be determined by some elimination or sorting technique. The following sections summarise the sorting technique employed, the mathematical expressions for inter-object distances and related quantities and also other aspects of contact related geometry.

3.2.2 Sorting and Locating Objects in a 3D Domain

Doing an exhaustive search for proximate objects in a three-dimensional domain containing a large number of such objects would take search times increasing exponentially with the increment in number of rigid bodies to be cross-checked. Narrowing down the number of possible object pairs to check for is a crucial part of contact determination in rigid body systems and fortunately there are some ways to handle such problems. The most efficient way of determining possible proximity is to use bounding boxes into which adjacent objects can be sorted using geometrical sorting techniques. Geometrical sorting can be done mathematically by evaluating coordinates and sorting them according to some criteria, about which more will be said in the implementation chapter, along with storage schemes and other relevant design information. Once objects had been sorted into bins, i.e. the various bounding boxes into which objects are sorted, the search for contacting objects can be accomplished much faster, since only spatially closest neighbours are added to any specific bin and will thus be checked for interference against other objects in the same bin.

3.2.3 Inter-Object Distances

Inter object distances are easy to calculate when one has a little vector algebra and calculus theory in stock. In this study, the objects are very simple and only interactions between spheres, spheres and vertices, spheres and edges and spheres and facets need to be considered. All inter-object distance calculations reduce to mathematically describing the magnitude of the shortest

possible straight distance between the geometrical objects considered. For spheres and vertices, the inter-object distance is the difference between the sphere's radius and the magnitude of the vector joining the vertex and the centre of the sphere, which can be expressed mathematically as in equation (3.26) below.

$$d_{ij} = \left\| \underline{x}_i - \underline{x}_{v,j} \right\| - r_i \tag{3.26}$$

The inter-object distance between two spheres is defined in a similar fashion, as can be seen in equation (3.27) below.

$$d_{ij} = \left\| \underline{x}_i - \underline{x}_j \right\| - r_i - r_j \tag{3.27}$$

Before continuing, it is helpful to know that the perpendicular projection of any random point in three-dimensional space ($\underline{p}_i$) onto an edge can be expressed mathematically as in equation (3.28) below, which is similar and related to, but not based on, the formula in Mirtich (1998:9).

$$\underline{p}_{e,ij} = \frac{(\underline{x}_{e,j,2} - \underline{x}_{e,j,1}) \bullet (\underline{p}_i - \underline{x}_{e,j,1})}{(\underline{x}_{e,j,2} - \underline{x}_{e,j,1}) \bullet (\underline{x}_{e,j,2} - \underline{x}_{e,j,1})} (\underline{x}_{e,j,2} - \underline{x}_{e,j,1}) + \underline{x}_{e,j,1} \tag{3.28}$$

In equation (3.28) above, an important scalar ($\alpha_{e,ij}$) termed the point-to-edge projection ratio can be isolated and is defined by the expression given in equation (3.29) below.

$$\alpha_{e,ij} = \frac{(\underline{x}_{e,j,2} - \underline{x}_{e,j,1}) \bullet (\underline{p}_i - \underline{x}_{e,j,1})}{(\underline{x}_{e,j,2} - \underline{x}_{e,j,1}) \bullet (\underline{x}_{e,j,2} - \underline{x}_{e,j,1})} \tag{3.29}$$

The point-to-edge projection ratio can be used as a criterion during contact checking, as will be seen later, and should be taken special note of. Knowing the expression for a perpendicular projection, it can be shown that the inter-object distance between a sphere and an edge is given by the expression shown in equation (3.30) below.

$$d_{ij} = \left\| \underline{x}_i - \frac{(\underline{x}_{e,j,2} - \underline{x}_{e,j,1}) \bullet (\underline{x}_i - \underline{x}_{e,j,1})}{(\underline{x}_{e,j,2} - \underline{x}_{e,j,1}) \bullet (\underline{x}_{e,j,2} - \underline{x}_{e,j,1})} (\underline{x}_{e,j,2} - \underline{x}_{e,j,1}) - \underline{x}_{e,j,1} \right\| - r_i \tag{3.30}$$

The sphere-facet inter-object distance formula can be shown to be represented by the expression in equation (3.31) below.

$$d_{ij} = \left| (\underline{x}_i - \underline{x}_{f,j,1}) \bullet \underline{n}_{f,j} \right| - r_i \tag{3.31}$$

Another important scalar quantity named the point-to-facet projection distance can be isolated in equation (3.31) above and is shown in equation (3.32).

$$d_{f,ij} = (\underline{x}_i - \underline{x}_{f,j,1}) \bullet \underline{n}_{f,j} \tag{3.32}$$

The point-to-facet projection distance can be used during contact checking to determine whether a point to be projected lies above or below a triangular facet with respect to its "positive surface side", but more about this later.

3.2.4 Contact Related Theory

The previous section summarised the theory related to determining inter-contact distances which can be used to check for direct contact, in which case the inter-object distances would be zero. One has to take into account some additional concepts when determining points of contact and their associated contact normal vectors, such as whether a contact is at all physically possible, and this will also be described in this section.

The definitions for the basic geometric objects as listed in Table 3.1 are ideally suited for their intended purpose, i.e. describing surfaces and spherical rigid bodies unambiguously. First considering facets, the true facet region does not include the corner vertices or bounding edges, only the bounded triangular region, thus delegating the handling of collisions to edge entities where they occur right between two facets. In their turn, the line described by an edge only spans the range between two vertices, never including them, thus delegating the handling of collisions at the junction of two or more edges to vertex entities. Theoretically this approach leads to a set of unique sphere-and-entity-contacts that do not need to be checked for duplication after being determined.

If the distance between a sphere and a vertex as shown in equation (3.26) is zero, they are automatically in contact, and the unit normal for this contact is then defined by the expression as seen in equation (3.33) below.

$$\underline{n}_{ij} = \frac{\underline{x}_i - \underline{x}_{v,j}}{r_i} \tag{3.33}$$

For inter-sphere contacts, a very similar expression, as seen in equation (3.34), exists for the contact normal whenever the inter-object distance as shown in equation (3.27) is zero.

$$\underline{n}_{ij} = \frac{\underline{x}_i - \underline{x}_j}{r_i + r_j} \tag{3.34}$$

The unit normal vector for contact between a sphere and an edge can be expressed as in equation (3.35) below if two conditions are met. The first condition is that the point-to-edge projection ratio, as shown in equation (3.29), should have a value of between 0 and 1.0 and the second condition is that the inter-object distance as shown in equation (3.30), must be zero.

$$\underline{n}_{ij} = \frac{\underline{x}_i - \dfrac{(\underline{x}_{e,j,2} - \underline{x}_{e,j,1}) \bullet (\underline{x}_i - \underline{x}_{e,j,1})}{(\underline{x}_{e,j,2} - \underline{x}_{e,j,1}) \bullet (\underline{x}_{e,j,2} - \underline{x}_{e,j,1})} (\underline{x}_{e,j,2} - \underline{x}_{e,j,1}) - \underline{x}_{e,j,1}}{r_i} \tag{3.35}$$

Contact between a facet and a sphere exists when four conditions are met. The first three conditions have to be met to ensure that the sphere lies directly above the facet region (i.e. the perpendicularly projected point would lie within the region bound by the three edges) and they are a special application of the point-to-edge-projection ratio (see Figure 3.2 below for clarity).

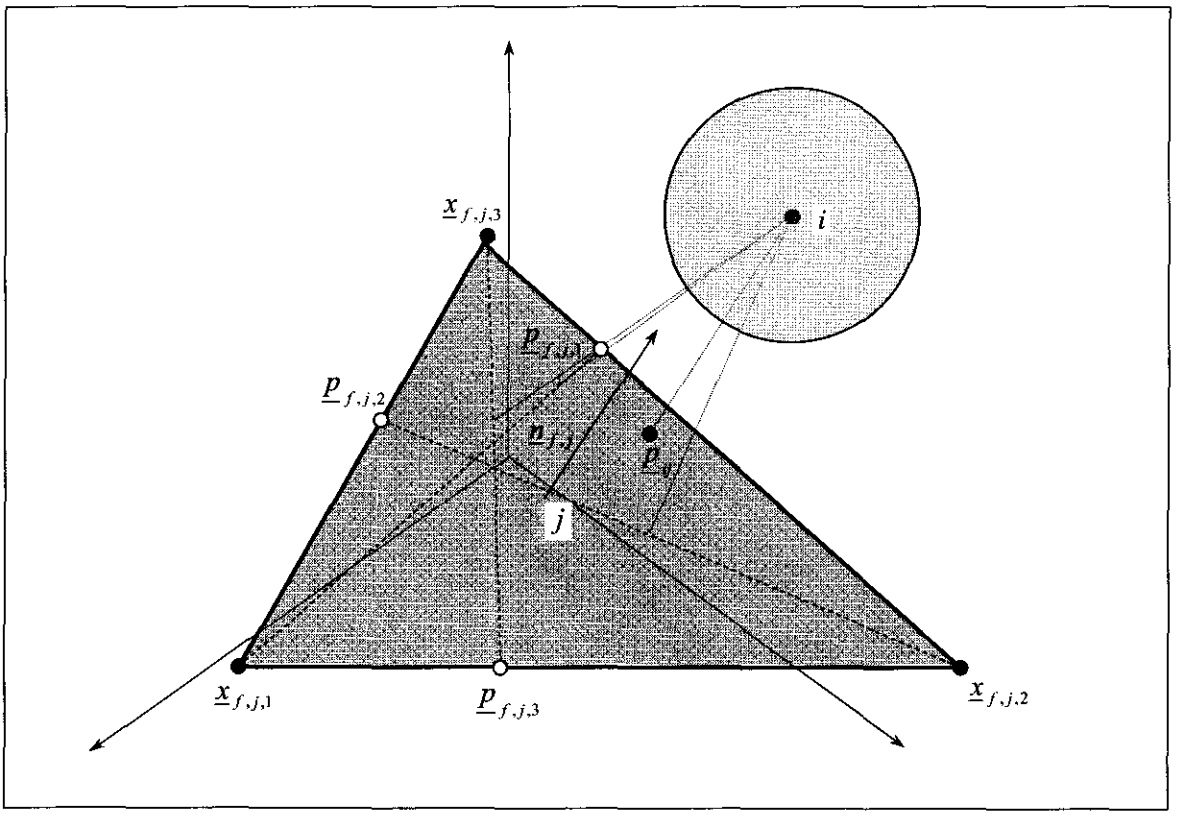


Figure 3.2: The various parameters involved in distance calculations for sphere *i* and facet *j*

The point-to-edge projection ratio of the sphere centre to the projected perpendicular from each corner of a facet to the opposite side of the facet, as given in mathematical form for each line by equation (3.36), should all have values between 0 and 1.0 (see figure below for clarity).

$$\alpha_{f,ij,k} = \frac{(\underline{p}_{f,j,k} - \underline{x}_{f,j,k}) \bullet (\underline{x}_i - \underline{x}_{f,j,k})}{(\underline{p}_{f,j,k} - \underline{x}_{f,j,k}) \bullet (\underline{p}_{f,j,k} - \underline{x}_{f,j,k})} \tag{3.36}$$

The perpendicularly projected points in equation (3.36) can be obtained by doing a normal projection using the point-to-edge projection expression as shown in equation (3.28), where the points to be projected for a facet, $\underline{p}_i$, will be the corner vertices of the facet, $\underline{x}_{f,i,k}$, and the edges opposite to each vertex will be the edges projected to, e.g. for $\underline{x}_{f,j,1}$, the opposite edge's first and second vertices would respectively be $\underline{x}_{f,i,2}$ and $\underline{x}_{f,i,3}$, for $\underline{x}_{f,j,2}$ they would be $\underline{x}_{f,i,3}$ and $\underline{x}_{f,i,1}$, and for $\underline{x}_{f,i,3}$ they would be $\underline{x}_{f,i,1}$ and $\underline{x}_{f,i,2}$. The last condition, as for all other types of contact, is that the inter-object distance, as described in equation (3.31), be zero. The unit normal at a contact between a facet and a sphere is always the unit normal of the facet surface. This concludes the section on contact related theory and also on analytical geometry as a whole.

3.3 Basic Software Development Principles

Most of the theory related to programming principles belongs in the implementation chapter and that is where they are listed, but some principles should be considered as basic theory and those are listed in this section.

3.3.1 General Software Development Cycle

The software development cycle is a guideline for approaching the process of software development in a systematic and consistent manner, though definitely always iterative (Stroustrup, 2000:696). Following such a development cycle might seem tedious, but it leads to easier and better quality control and can also be enforced to ensure product quality and improved reliability. In addition to the quality benefits, there are also the advantages of better project scope definition, quicker development times and better and easier maintenance due to better and quicker error tracing at requirements level. Ultimately this leads to saving money in commercial environments. The most basic phases of the development cycle, as applicable to scientific and engineering code development, loosely based on the software development process described by Nielsen (1995:13), are summarised in the following sub-sections.

3.3.1.1 Software Requirements Analysis

The first step necessary for the successful completion of a software development project would be the identification or compilation of a software requirement specification (SRS) during the software requirement analysis phase. An SRS need not necessarily be a large individual document; a section listing all the intended or required capabilities a computer program (application) should have, would be good enough.

3.3.1.2 Software Design

A software design phase follows the compilation of an SRS. In this phase, the actual implementation strategy necessary for fulfilling the requirements specified in the SRS is described and motivated. The sub-steps to be taken during the design phase are listed in the following sections. It should be duly noted that the actual design might not exactly follow the design steps in sequential fashion, but rather a highly iterative parallel execution of the various sub-steps might result due to unforeseen difficulties or amendments to the original requirement specification.

3.3.1.2.1 Functional Specification

Firstly, a functional specification that describes what exactly is expected from the software or software components to be developed has to be deduced from the software requirement specification. A functional specification differs from an SRS in that, in addition to the primary requirements as stated in the SRS, it also has to include secondary requirements to be met in support of those primary requirements. Secondary requirements usually deal with concepts such as data input, storage, retrieval, transfer, display and internal data representation structures as well as interactions and collaborations between various data representation structures. Having finished a functional specification, the proper architectural design can be done.

3.3.1.2.2 Conceptual Architectural Design

From the functional specification, a conceptual architectural design can be done by translating functional requirements into conceptual code, depicting relationships, collaborations, data groupings and operations to be executed diagrammatically. The object oriented programming (OOP) approach as well as the related design patterns can be used with great success in architectural design to deal with highly complex problems by delegating tasks, identifying common data storage expressions, etc.

3.3.1.3 Coding

From the conceptual architectural design, a detailed architectural implementation can be done by translating design concepts into actual computer programming language expressions, constructs and instructions. After the implementation had been done, the code can be compiled and is ready for testing.

3.3.1.4 Testing

In order to ensure that the implementation done actually works like it should, a test plan has to be compiled and followed. Such a test plan usually only verifies whether what was expected in the SRS was actually delivered and it can thus be compiled from the SRS even before the design phase is started. In most applications, verifying whether requirements as stated in the SRS had been met is, unfortunately, not sufficient to ensure reliability. The results obtained from the implementation of the theory used in the code always have to be validated by comparing to the results generated by some alternative implementation employing the same or a different approach, preferably not originating from the same programmer.

3.3.1.5 Release

After sufficient testing yielding no errors when run through the full test plan, the software can be released for general use and then usually only requires maintenance and support.

3.3.1.6 Maintenance and Support

Once a software package or component had been made available and is in use, one cannot ever expect perfectly problem free operation. Having a contingency plan for handling errors is thus part of the development process. Test versions are usually first made available to testers who then run through the test plans and report errors, but they may also opt to use the software in everyday situations, which sometimes uncovers less obvious errors in the code. Test plan entries to check for these everyday errors encountered then need to be added. All errors need to be promptly handled and updates of the software should be made available. In addition to fixing problems, software product developers should provide end users with assistance in using their product in whatever capacity it was designed to function. Apart from the normal assistance, expert advice might occasionally also be required for more specialised applications by some users and having an in-depth knowledge of both the code and its various applications, and limitations, is thus always essential.

3.3.1.7 Decommissioning

From the time software is commissioned into service its relevance already starts diminishing, since no user's requirements ever remain exactly constant. After some time which could range from a few months to several years, the software simply becomes too outdated and will need to be decommissioned, with all support for that version terminating. By such time, though, most software developers have a newer, more relevant version (based on customer complaints, requests and general refactoring) ready to take the place of the defunct product.

3.3.2 Object Oriented Programming

Object Oriented Programming (OOP) techniques reach high levels of complexity, but that need not be the case for the theory relevant to this study, since simplicity is of greater importance and thus only the very basic concepts will be discussed here. First and foremost, object oriented programming can be seen as a "divide and conquer"-strategy for computer software development and programming problem solution. The basic components and aspects of OOP are listed and described in the following sections, all of which were found in Nielsen (1995:40-53) and shown here either slightly modified or as it originally appears in the text.

3.3.2.1 Classes and Objects

The basic unit of object oriented programming is the object, as might be evident from the name for this programming paradigm. An object is a self-contained data type containing data and methods to operate on that data in various ways, usually representing a real life concept or object, but not necessarily restricted to that. Objects are defined using the class construct in C++ which is extremely well suited to do this.

3.3.2.2 Encapsulation and Information Hiding

Due to the fact that data can be grouped into objects and access to them can be limited to the usage of discrete methods operating on that data, data is encapsulated inside an object. Due to the encapsulation, data is also hidden from the user.

3.3.2.3 Data Abstraction

Objects provide for a means of data abstraction by being self contained entities. This means that an object can be used to represent a more complex concept (data type) without burdening the user with confusing or intimidating implementation details such as exact internal data representation and methods implementing algorithms to operate on or using the contained data.

3.3.2.4 Responsibilities

The methods operating on and within an object are termed the responsibilities of that object, though a more accurate description might be the functionality of the object.

3.3.2.5 Collaborations and Message Passing

The service provided by one object for usage by others is termed collaboration. This can be either in the form of an operation or data storage capability. Message passing is a concept closely related to collaboration, since it describes the manner in which software modules communicate. Messages can be sent as variables by value in function parameters and received by return value or by referenced function parameters (such as references or pointers).

3.3.2.6 Inheritance

Inheritance refers to the re-use of the attributes of an existing object (superclass) in more complex or specialised objects (subclasses). Using the attributes of more than one superclass is termed multiple inheritance and can be useful for rapid subclassing whenever appropriate superclasses already exist and can be combined. One should keep in mind that there is a downside to multiple inheritance, especially where architecture and delegation of responsibilities was not done properly, since there might be clashes between certain identically named data members and methods contained within the various superclasses. It is evident that inheritance is the way to accomplish re-use of object code and thus minimising source file sizes and redundant object coding.

3.3.2.7 Polymorphism

"Polymorphism refers to the characteristic that an object can exist as an instance of different classes at run-time." (Nielsen, 1995:48). This is a useful characteristic for the writing of generic code which relies on objects behaving differently in various situations whilst still appearing to have the same interface.

3.3.2.8 Binding

In programming, binding refers to the association of method calls or names with the actual implemented code to be executed upon invocation of that method. Binding can be done at compilation time (static binding) or be postponed to execution time (late or dynamic binding). Both types of binding have their particular uses in programming, and, as one might expect, dynamic binding is mostly used in event-driven scenarios, while static binding is preferred for

standard code excerpt replacements, such as quickly running a loop to copy data or some similar operation.

3.3.2.9 Modularity

Modularity refers to the degree of segregation and delegation of responsibilities to be handled within a software project. Due to its inherent encapsulating and abstracting nature, OOP automatically simplifies and enforces the process of modularisation during the design phase. Greater modularity reduces the apparent complexity of large systems and also improves maintainability.

3.3.2.10 Genericity

Objects can be designed to be able to contain different parameterised data types specified at the time of object instantiation, thus usually being containers. The template class construct is used for this purpose in C++ and further contributes to the extent of code reuse.

3.3.3 Generic Programming

Generic programming is a term referring to creating code that represents concepts, such as containers or algorithms in parameterised format. As mentioned earlier, generic programming greatly contributes to code-re-use and general maintainability, since a wrongly implemented algorithm or container need be fixed at only one location, and the problem will be addressed for instances of usage, to name but one of the implied advantages.

3.3.4 Design Patterns

According to Gamma *et al.* (1995:3) design patterns are "descriptions of communicating objects and classes that are customised to solve a general design problem in a particular context." From the definition can be deduced that, in a computer programming environment, design patterns could definitely rely on and can be used in object oriented programming. Design patterns are ideal to help maximise code reuse as well as simplify, organise and identify data abstractions, object collaborations and general architectural layout.

3.3.5 Code Optimisation

Code optimisation can be done with respect to architecture, execution speed, memory usage, user friendliness and maintainability. Optimisation is a simple concept that is difficult to implement, since it basically entails making code more desirable or sensible for use.

3.3.5.1 Architectural Optimisation

Architectural optimisation can usually be done even though a highly efficient design was implemented initially. Architectural improvements usually reduce to re-factoring of code, concentrating on no particular construct, but rather taking out common code in functions and objects and placing it in separate independent functions or objects. Dependencies can also be reworked and improved as can be seen in the whole text by Lakos (1996) as part of large scale C++ program design.

3.3.5.2 Speed Optimisation

Speed optimisation entails the minimisation of all time consuming operations, such as mathematical operations, class indirections, copy operations, memory allocations and de-allocations, etc. Stroustrup (2000:675) neatly summarises the aspects necessary to address when considering performance improvements.

3.3.5.3 Memory Usage Optimisation

Apart from speed considerations, memory usage is also a problem, though not as much as it used to be a few decades ago. Memory usage usually increases with most techniques employed to decrease execution times, since use is often made of temporary variables. There are some instances where using less memory can increase execution times too, though, for instance when a lot of data needs to be transferred, since it takes longer to copy more data. Minimising the amount of memory used at any stage in a program's execution remains a good objective, no matter what the ultimate reason for it would be, since no-one appreciates a memory-hogging application.

3.4 Conclusion of Theoretical Background

The theory listed and briefly discussed in this chapter is only the simplest, but nonetheless important basics for this study. Using the basic theory, new techniques can be developed or devised and finally implemented in a self-contained software component.

Chapter 4 Algorithms and Implementation

In this chapter, the extended or new theory and the algorithms to solve such problems are presented, along with strategies for implementation thereof in computer code. The implementation of theory is the first step to verifying whether new theory is actually feasible and correct, along with the validation and verification process to be done after (or more aptly, during) implementation and therefore plays a pivotal role in the research process.

4.1 Algorithms to be Used

In this section, the numerical approaches to the solution of problems posed or implied by the theory in the theoretical background are listed and described briefly. Several algorithms need to be implemented in this study in order to accomplish the goal of doing multiple concurrent contact simulation in a rigid body system. As can be seen in Figure 4.1 below, one has to set up an initial configuration of the rigid bodies and then move the rigid bodies, detect contacts and resolve contacts for one single simulation time step.

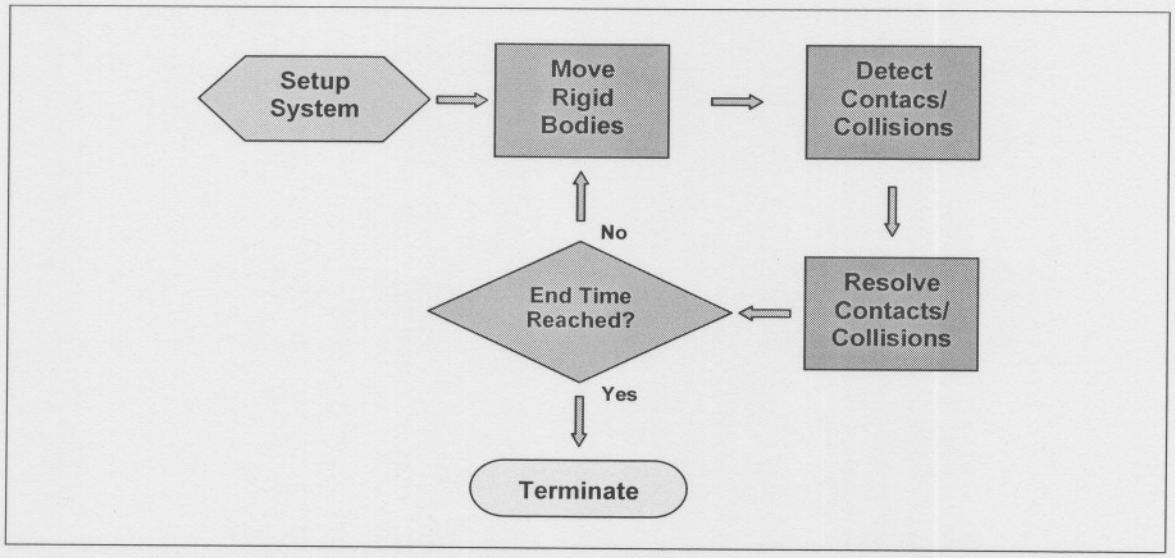


Figure 4.1: The basic rigid body simulation steps and simulation loop.

A little more will be said about the setup of rigid body systems later, but for now only the core algorithms needed for basic simulation steps directly relevant to collisions will be discussed.

4.1.1 Free Body Motion

Free body motion is most often described using a simplified version of Newton's equations of motion as seen in equations (3.5) and (3.6).

4.1.1.1 Problems

The equations of motion to be used are second order ordinary differential equations and thus cannot be solved using simple manipulation and substitution techniques. Differential equations need to be integrated with respect to the independent variable, in this case, the time, i.e. t in the relevant equations. One also needs to take into account the spatial restrictions or constraints that apply to a rigid body system, since no inter-penetration is allowed.

4.1.1.2 Solutions/Algorithms

The relevant algorithms or solutions for the types of differential equations encountered, as well as some strategies for avoiding inter-penetration are summarised in the following sections.

4.1.1.2.1 Second Order Differential Equation Integration

From the literature, the most popular technique for solving/integrating second order differential equations appears to be the Fourth-Order accurate Runge-Kutta technique (Kreyszig, 1993:1040-1043), (Hibbeler, 1995:602-604), (Cheney & Kincaid, 1999:387-389, 415-419), (Erleben, 2001:51-52), which is based on a type of Taylor series expansion. Looking in any of the literature sources mentioned, the general Runge-Kutta integrator steps, such as those extracted from Hibbeler (1995:602) and shown slightly modified in Figure 4.2, can be found or deduced.

- *Calculate the four fourth-order integration constants:*

$$k_1 = hF(t_i, x_i, \dot{x}_i)$$
$$k_2 = hF(t_i + 0.5h, x_i + 0.5h\dot{x}_i, \dot{x}_i + 0.5k_1)$$
$$k_3 = hF(t_i + 0.5h, x_i + 0.5h(\dot{x}_i + 0.5k_1), \dot{x}_i + 0.5k_2)$$
$$k_4 = hF(t_i + h, x_i + h(\dot{x}_i + 0.5k_2), \dot{x}_i + k_3)$$

  where F is the second order derivative function ($\ddot{x}$) defined in terms of t, x and $\dot{x}$, with x any time-dependent variable, $\dot{x}$ its first derivative with respect to time and t the actual time.

- *Calculate new approximated first integral by evaluating:*

$$\dot{x}_{i+1} = \dot{x}_{i+1} + (k_1 + 2(k_2 + k_3) + k_4)/6$$

- *Calculate new approximated second integral by evaluating:*

$$x_{i+1} = x_i + h(\dot{x}_i + (k_1 + k_2 + k_3)/6)$$

Figure 4.2: The fourth-order Runge-Kutta integration steps

An actual integration time step (Δt_{step}) can be divided into several sub-divisions of size h for greater accuracy, but with sufficiently small time steps, h can be set equal to the full time step (Δt_{step}) and the terms with the subscript $_{i+1}$ will then refer to the next approximated time step values. In the case of the simple equations of motion, such as (3.5) and (3.6), the time step becomes less critical, due to their simple nature, but care should always be taken not to make time steps too large in any case. Integration of each linear and angular velocity vector component can be done separately by using the fourth order Runge-Kutta method as outlined in Figure 4.2.

4.1.1.2.2 Avoidance of Inter-Penetration

In order to avoid inter-penetration, the spatial distribution results yielded by the integration routine, need to be checked for overlaps anywhere after each attempted integration time step. The description of the overlap detection will be delegated to the following section, since it is basically the same as the contact detection routine described there and only the main algorithm steps, as seen in Figure 4.3 will be described here.

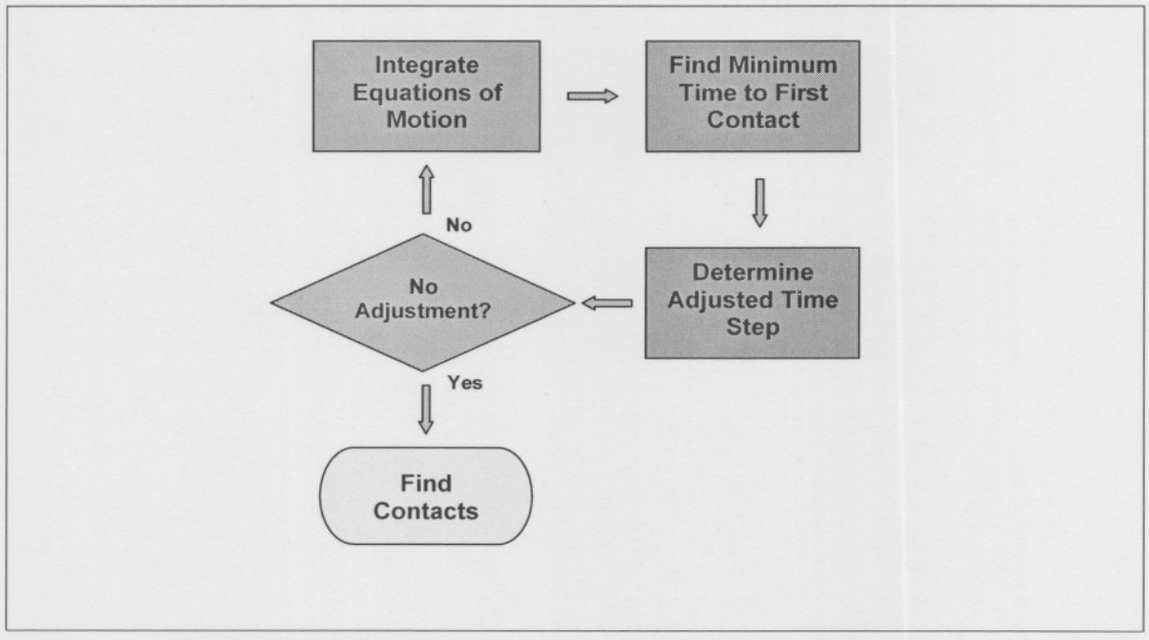


Figure 4.3: The basic rigid body motion loop.

The most difficult part of the free motion simulation with inter-penetration avoidance is the determination of the critical time step to be taken from the original state so as to avoid inter-penetration. This is an iterative process and shown in Figure 4.4 integrated with the rest of the free motion routine.

- *Integrate equations of motion for each rigid body using fourth order Runge-Kutta.*
- *If an overlap exists anywhere, start or continue iterative integration.*
    - *Estimate the time step adjustment needed for a legal state by evaluating:*

      $$\Delta t_{adj} = \left\lfloor d_{ij} / \left\| \dot{\underline{x}}_{n,ij} \right\| \right\rfloor_{\min}$$

      where Δt_{adj} *is the time step adjustment and* $\left\lfloor d_{ij} / \left\| \dot{\underline{x}}_{n,ij} \right\| \right\rfloor_{\min}$ *is the minimum overlap-to-relative velocity magnitude-ratio (i.e. estimated time to first collision) found at a particular overlap between body i and j.*
    - *Adjust the previous integration time step by adding* Δt_{adj} *to it.*
    - *Return to integration with new estimated time step as parameter.*
- *Else, store new positions and exit.*

Figure 4.4: The basic rigid body motion algorithm with critical time estimation.

The expression for the velocity of object *i* relative to object *j* can be written mathematically as in equation (4.1) below, once again implicitly based on the expression in equation (3.13), where it is specialised for the sphere-sphere interaction case, but can easily be further specialised for the sphere-fixed object interaction case by zeroing the *j* components.

$$\dot{\underline{x}}_{ij} = (\dot{\underline{x}}_i + r_i \, \underline{n}_{ij} \times \dot{\underline{\theta}}_i) - (\dot{\underline{x}}_j - r_j \, \underline{n}_{ij} \times \dot{\underline{\theta}}_j) \tag{4.1}$$

Knowing what the general expression for relative velocity between two objects is, one can derive an expression for its normal component by taking its scalar product with the unit normal between the interacting objects, given mathematically as in equation (4.2) below, once again simplified since the tangential velocity for a sphere has no normal component.

$$\dot{\underline{x}}_{n,ij} = [(\dot{\underline{x}}_i - \dot{\underline{x}}_j) \bullet \underline{n}_{ij}] \underline{n}_{ij} \tag{4.2}$$

Having found the critical time step, the contacts can finally be detected and stored and then the contact/collision resolution can commence.

4.1.2 Overlap and Contact/Collision Detection

Once all objects had been moved using the motion integrator, they need to be sorted into their respective bins for grouping prospective contact candidates and then the occurrence of such contacts or overlaps can be determined by evaluating the expressions for inter-object distances in equations (3.26), (3.27), (3.30) and (3.31). Due to the repetitive nature of collision detections,

the evaluation routines need to be very efficient, especially since they are repetitively used within the free body motion algorithm too.

4.1.2.1 Problems

The problems faced by the contact detection sub-routine can basically be divided into a macro-scale search problem to determine physical proximity of objects and then more refined detailed checking for geometrical object pairs actually touching or overlapping. Overlap needs to be checked continuously, whilst contact detection only needs to be done once. When contact detection is completed, contact parameters need to be determined and stored.

4.1.2.2 Solutions/Algorithms

The relevant algorithms or solutions for the general proximity sorting as well as some strategies for determining overlaps and contacts efficiently are discussed in this section. In addition to this, the handling of the various parameters involved in contact/collisions, such as contact normal vectors, combined normal and tangential restitution factors and static friction and dynamic friction factors.

4.1.2.2.1 Determining Closest Neighbours Using Bin Sorting

The best way to narrow the number of object pairs to check for contact, is to sort them into bins based on their geometrical locations in the 3D domain. This list of static objects should be retained, since sorting does consume time, and the re-sorting should thus be kept to a minimum. Geometric bin sorting is a fairly simple process based on lower and upper x, y and z Cartesian coordinates for the sorting bins to determine the x, y or z Cartesian "index" of the bin lattice the objects are sorted into. Depending on what types of objects are being sorted, their "critical coordinates" are "located" using a simple division of each ordinate by the bin grid interval in that Cartesian direction (see Figure 4.5 for an illustration of how the sub-division of a one-dimensional domain works and Figure 4.6 for a summary of the calculations to be done in 3D).

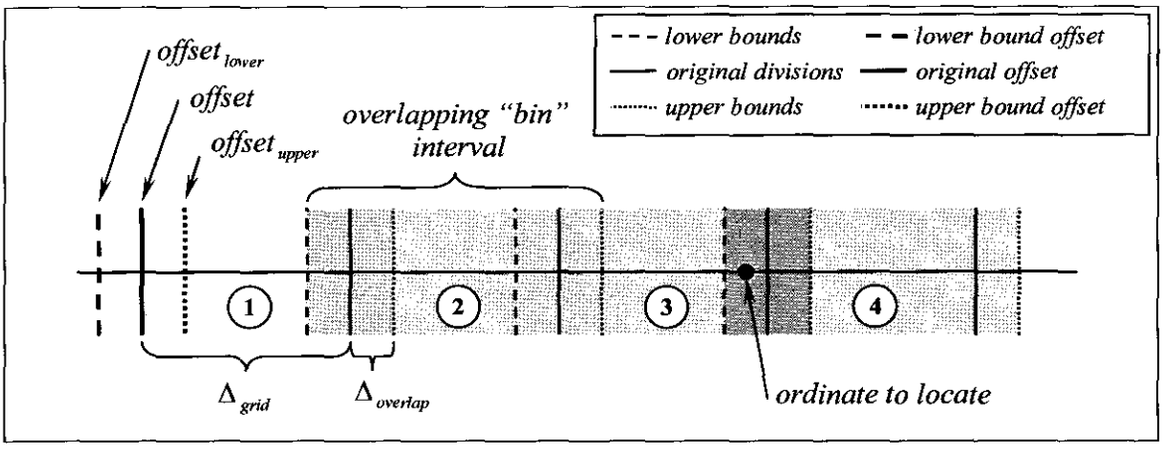


Figure 4.5: Depiction of the overlapping "bin" layout in 1D.

In Figure 4.5 the solid vertical lines indicate the original grid into which the domain is divided and the dotted lines indicate lower and upper bounds for all extended bins, which are deliberately chosen to overlap with adjoining bins in order to not miss any possible contacts due to objects lying in separate bins. The lower limit of the original domain division is termed the *offset* and the *upper* and *lower offsets* are then set a distance of $\Delta_{overlap}$ respectively to the negative and positive side thereof. It should be noted that the *upper offset* should actually only start at the right of the original division between the first and second bins, but for computational simplicity and consistency (see the section on calculation of the upper and lower indices in Figure 4.6) the upper offset is chosen as indicated.

The domain division depicted in Figure 4.5 can be extended to the 3D case without much alteration, since the three Cartesian coordinate directions are independent of one another with respect to the grid and the general procedure for determining position indices, as used to locate spheres and vertices in a 3D domain can be seen in Figure 4.6 with the subscript k referring to x, y and z Cartesian axis directions.

The principle of geometric point, vertex or sphere locating is perhaps best illustrated by a simple example involving the domain illustrated in Figure 4.5 and calculations indicated in Figure 4.6. First, consider the one-dimensional domain in Figure 4.5; by visual inspection it can be seen that the ordinate to be located lies within both bin 3 and 4. The actual principle of this visual inspection, however, needs to be translated into some mathematical expression if it is to be useful in computer coding. The expressions in the section on object location index determination seen in Figure 4.6 would yield the results $position_{upper} = 3$ and $position_{lower} = 4$ for the ordinate indicated on Figure 4.5.

- *Divide the 3D domain into an evenly spaced lattice of overlapping "bins".*
- *Calculate and store the lower grid offsets by subtracting the overlaps from the original grid offsets, or mathematically:*

$$offset_{lower,k} = offset_k - \Delta_{overlap,k}$$

- *Calculate and store the upper grid offsets by adding the overlaps to the original grid offsets, or mathematically:*

$$offset_{upper,k} = offset_k + \Delta_{overlap,k}$$

- *Determine each object's location indices by evaluating its "critical coordinate indices" with the following expressions:*

$$position_{lower,i,k} = roundup((ordinate_{i,k} - offset_{lower,k})/\Delta_{grid,k})$$

subject to

$$1 \le position_{lower,i,k} \le n_{grid,k}$$

and

$$position_{upper,i,k} = roundup((ordinate_{i,k} - offset_{upper,k})/\Delta_{grid,k})$$

subject to

$$1 \le position_{upper,i,k} \le n_{grid,k}$$

where $n_{grid,k}$ *is the number of bins in the* k^{th} *Cartesian axis direction.*

- *Add the sorted object's number to appropriate lists in the "bins" in lattice positions given by the following set of combinations:*

```
for upper and lower x indices
    for upper and lower y indices
        for upper and lower z indices
            Add object i to appropriate list in bin [x-index][y-index][z-index]
```

Figure 4.6: The geometrical sorting routine.

Dealing with more complex objects is more difficult since they might span several bins in a three dimensional domain and simple solutions had been devised for determining those bins for both edges and facets. Firstly, determining edge location in the bin grid requires interpolating for intersection positions where an edge crosses the original bin division planes between its endpoints (see Figure 4.7 for a 2-D example of edge locating considerations). The general interpolation routine for edge j, with endpoints $\underline{x}_{e,j,1}$ and $\underline{x}_{e,j,2}$, can be described mathematically as in equation (4.3) below, where ρ_{ext} is a scalar representing the edge extension ratio which can be used to describe any point ($\underline{p'}_i$) along the line vector described by the edge.

$$\underline{p'}_i = \rho_{ext}(\underline{x}_{e,j,2} - \underline{x}_{e,j,1}) + \underline{x}_{e,j,1} \tag{4.3}$$

If the general plane ordinate in the k^{th} Cartesian axis direction is defined as $p'_{i,k}$, it can be shown that the intersection point for the i^{th} plane in the k^{th} axis direction and the edge in equation (4.3) would be given by the expression in equation (4.4) below.

$$\underline{p}'_{i,k} = \frac{p'_{i,k} - x_{e,j,1,k}}{x_{e,j,2,k} - x_{e,j,1,k}} (\underline{x}_{e,j,2} - \underline{x}_{e,j,1}) + \underline{x}_{e,j,1} \tag{4.4}$$

On closer inspection of equation (4.4) it can be seen that only two coordinates need to be interpolated for, since the third (k^{th} direction) coordinate is specified by the plane where the intersection is to be interpolated.

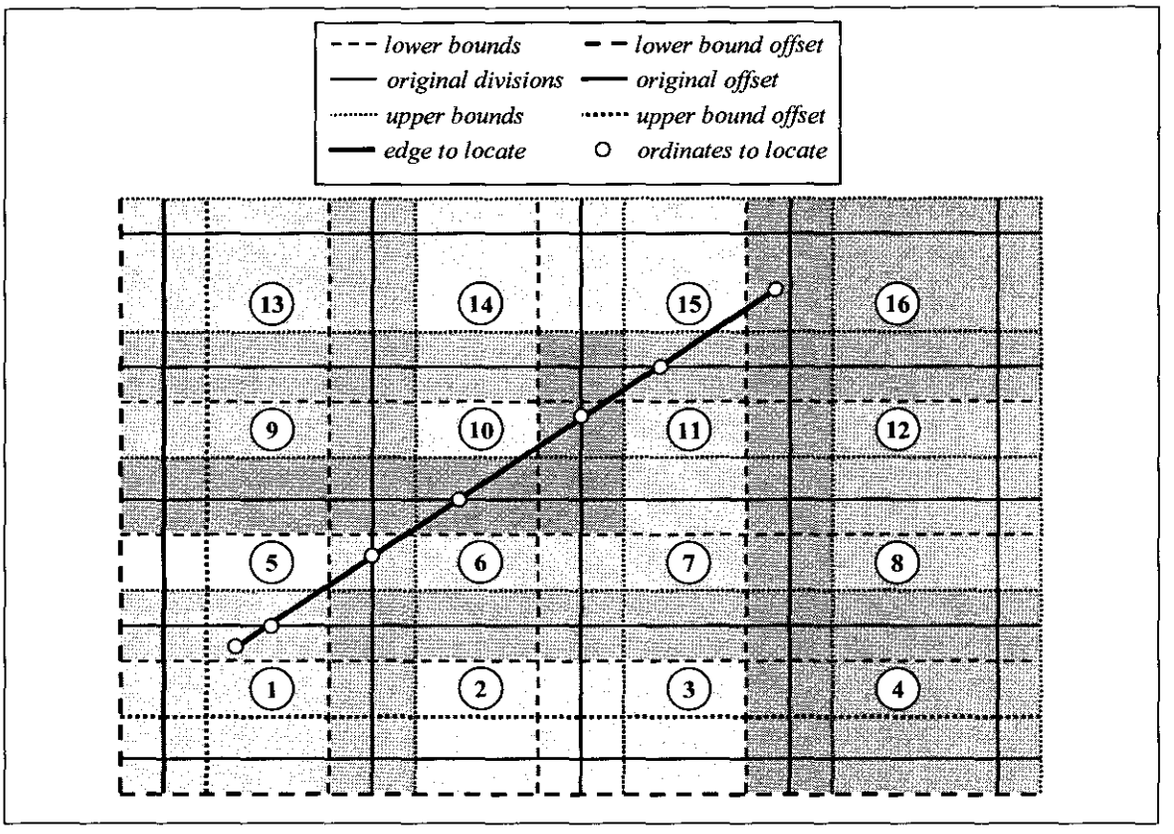


Figure 4.7: Depiction of edge location in a 2D domain.

The endpoints and interpolated coordinates (i.e. critical coordinates) on the edge can subsequently be used to locate the edge subsections using the regular procedure as outlined in Figure 4.6 and the unique bin entries can be added to the bins' edge lists.

A physical example might once again be employed here to demonstrate the principle of locating a line in a 2-D domain. Consider the line in Figure 4.7; sections of the line cut across bins 1, 5, 6, 10, 11 and 15. Evaluating the critical coordinates on the line using the locating procedure outlined in Figure 4.6, the index pairs that are found would be 1 & 5, 1 & 5, 5 & 6, 6 & 10, 10 &

11 and 11 & 15. Removing the copy bin indices, the list of indices obtained is 1, 5, 6, 10, 11 and 15. Strictly speaking, taking the extended size of the bins into account, bin 14 would also have to be included into the list, but since the overlaps are actually only necessary to ensure that partial spheres are not missed in contact detections, this method is acceptable. This method works equally well in a 3D domain, since only plane intersection points are located and sorted into bins.

Before continuing to the next type of object to locate (facets), a simple expression for use in arbitrary edge-plane intersection determinations is given in equation (4.5), in case one needs to rotate or transform the bin grid for more efficient or appropriate domain subdivision at some stage and intersections still need to be determined very quickly.

$$\underline{p}'_i = \frac{(\underline{x}_{e,j,1} - \underline{p}_{plane}) \bullet \underline{N}_{plane}}{(\underline{x}_{e,j,2} - \underline{x}_{e,j,1}) \bullet \underline{N}_{plane}} (\underline{x}_{e,j,2} - \underline{x}_{e,j,1}) + \underline{x}_{e,j,1} \tag{4.5}$$

In equation (4.5) $\underline{x}_{e,j,1}$ and $\underline{x}_{e,j,2}$ are once again the endpoints of edge j of which the intersection with any arbitrary plane with normal $\underline{N}_{plane}$ and any point on the plane $\underline{p}_{plane}$ needs to be determined. Inspecting equation (4.5), yet another handy ratio as shown in equation (4.6) below can be isolated, this time called the edge-to-plane intersection extension ratio, since it refers to the factor with which an edge's original length needs to be multiplied to end on the exact point on the plane where the intersection occurs when starting from $\underline{x}_{e,j,1}$ along edge j's unit vector direction, being the special plane-intersection case equivalent of the general line extension ratio (ρ_{ext}) in equation (4.3).

$$\rho_{ext,plane} = \frac{(\underline{x}_{e,j,1} - \underline{p}_{plane}) \bullet \underline{N}_{plane}}{(\underline{x}_{e,j,2} - \underline{x}_{e,j,1}) \bullet \underline{N}_{plane}} \tag{4.6}$$

As with other ratios, the edge extension ratio indicates that a plane cuts an edge whenever the ratio has a value of between or equal to 0.0 and 1.0, otherwise the edge and plane do not physically intersect. What makes the expression for the edge extension ratio numerically appealing is that the plane normal ($\underline{N}_{plane}$) need not be of unit magnitude, thus avoiding costly and unnecessary square root determinations. It can be shown that the general case in equation (4.5) reduces to the specialised case in equation (4.4) whenever the normal vectors point along Cartesian axes and the points on the planes lie directly on those axes.

The last geometric entity to be located in this study, namely facets, can now be discussed. Facets are more complex than edges to locate, since they represent a 2-D sub-region in 3-D space and

can therefore be part of several bins spread throughout a 3D domain. The way facet locating is handled in this study, is to do a critical line-representation of the plane as seen in Figure 4.8 and then locating the plane intersection points for all critical lines, adding the unique facet numbers to the various bins' facet lists as they are encountered.

A degree of redundancy is to be expected (e.g. bin 11 in Figure 4.8 is "hit" by the locator no less than 9 times), but this is acceptable for the time being, since not many such extremely large facets spanning several bins at a time are normally used in actual simulations.

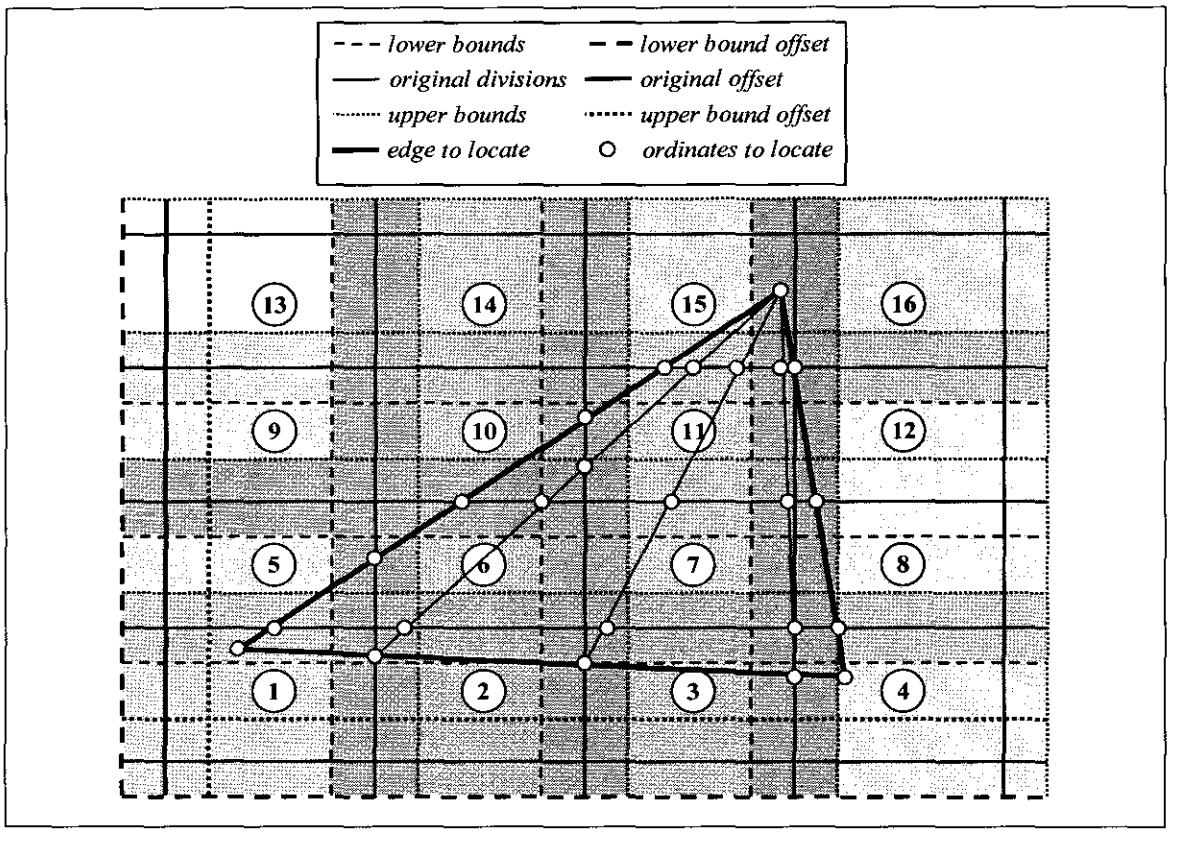


Figure 4.8: Depiction of facet location in a 2D domain.

Another physical example would be redundant and will thus be refrained from, but it can be seen in Figure 4.8 that the appropriate bins will be located from the various intersection points by the algorithm outlined. This concludes the geometric location and sorting of entities. Once all the objects present in the domain had been sorted into bins using the geometric locator routines, the search for overlaps can commence.

4.1.2.2.2 Searching for Overlaps

Searching for geometric entity overlaps is much simplified and highly localised due to the sorting into bins that precedes the detailed search. The lists of spheres and other objects present

in each bin can be checked against one another for overlaps by evaluating their inter-object distances as described in equations (3.26), (3.27), (3.30) and (3.31). An overlap is constituted by an inter-object distance less than zero and it is an illegal state in rigid body system simulation. The overlap distance and the relative approach velocity together provide an approximation for the time since impact. Thus, if one encounters an overlap during an integration time step, the original time step is shortened by the maximum estimated overlap time and integration is restarted from the previous time step. To find an exact time of impact, one then usually has to iterate between a forward and a backward time step adjustment (Δt_{adj}, as defined in Figure 4.4) of the previous time step till the adjustment is close enough to zero within a prescribed tolerance. Once a legal state with only contacts but no overlaps had been found, contacts can be located and stored with the necessary parameters.

4.1.2.2.3 Searching for Contacts

After determination of the critical time step using estimations based on overlaps and relative normal velocities (see Figure 4.4), and integrating the total system to that instant in time, the lists of spheres and other objects present in each bin (already sorted during the overlap-routine) can be checked against one another for contacts by evaluating their inter-object distances as described in equations (3.26), (3.27), (3.30) and (3.31). Upon encountering a contact, where an inter-object distance is within tolerance distance from zero, contact parameters need to be determined and stored in the contact list.

4.1.2.2.4 Determining Contact Parameters

Various contact parameters need to be calculated and stored for use during the contact resolution phase. Firstly, the object index number and type needs to be stored for both contacting objects at each contact in order to facilitate the correct setup of the equations during the resolution phase. The aforementioned approach assumes that all contacts are always bilateral (i.e. only two objects are involved at each specific contact point), and it would be a reasonable assumption based on physical and spatial considerations. Secondly, one needs to determine the contact normal ($\underline{n}_{ij}$), obtained by the evaluation of the appropriate one of three different types of contact normal expressions in equations (3.33), (3.34) and (3.35) or, for facets, the unit normal of the facet needs to be stored. The full relative velocity vector and its normal and tangential components all need to be calculated and stored for each contact, along with the normal and tangential restitution factors and the effective static and dynamic friction factors. Interactive restitution factors determined from experimental results can be used, hashing those of various materials against one

another in a table. The friction factors can, likewise, be determined from hash tables correlating the static and dynamic friction factors for various material combinations. In order to simplify initial model evaluation, the restitution and friction factors will be initially be set to some arbitrary constant values. Once all the parameters had been determined, the collision resolution phase can commence.

4.1.3 Contact/Collision Resolution

From the literature survey conducted, it is evident that multiple concurrent impact analysis is by no means yet a fully understood or accurately modelled phenomenon and that a multitude of alternative approaches to model them exists. All of the approaches pose some difficulties, advantages and disadvantages and this makes each well suited for specific applications, and less suitable for others. It could be seen that the various methods are usually not reconcilable, since they are often based on different principles, e.g. some elect to model problems at material mechanics level by using spring and damper configurations (continuous contact or soft-body approaches) while others prefer to consider the macroscopic or instantaneous dynamics of colliding systems (discrete or discontinuous contact or hard body approaches). For this study, a form of discrete contact model was chosen due to the mathematical simplicity and elegance of its fundamental equations, and physical accuracy during high speed or large force interactions, where continuous contact models pose several difficulties with respect to parameter determination (e.g. restitution coefficients do not translate well into a spring-and-damper model at all) and numerical instabilities (e.g. stiff numerical equations restricting time steps and the effectively non-concurrent impact resulting from delays due to spring-and-damper approximations for material responses).

4.1.3.1 Problems

The most difficult problem encountered in this simulation process would be solving the inequality constrained non-linear equalities arising from the set of equations as described in equations (3.15) through (3.17) and (3.19) through (3.25). Solution of inequality constrained non-linear equalities usually involves application of non-linear equality solution techniques, linear equality solution techniques and general variational calculus. It also requires knowledge of the mathematical behaviour and properties of all the aforementioned solution techniques. The manner in which these problems are addressed is described in the following section.

4.1.3.2 Solutions/Algorithms

Various problems had been identified in the preceding section, all of which have to be addressed to some extent if a workable total solution is to be produced. The best strategy would be to divide and conquer the problem, delegating whatever tasks can be delegated to more suitable algorithms or solution techniques where possible. The general problem had been described in the previous chapter, using the set of equations (3.15) through (3.17) and (3.19) through (3.25) for a whole system of simultaneously contacting rigid bodies. In general, the statement as summarised in Figure 4.9 below can be made regarding the set of equations to be solved for collision resolution.

Find the set of post-collision linear and angular velocities, normal and tangential impulses and tangential unit vectors given by:

$$\underline{S} = \left[\underline{\dot{x}}^{+}, \underline{\dot{\theta}}^{+}, \underline{\lambda}_n, \underline{\lambda}_t, \underline{t}\right]^T$$

where: $\underline{\dot{x}}^{+} = \left[\dot{x}^{+}_{1,x}, \dot{x}^{+}_{1,y}, \dot{x}^{+}_{1,z}, \ldots, \dot{x}^{+}_{bodies,x}, \dot{x}^{+}_{bodies,y}, \dot{x}^{+}_{bodies,x}\right]^T$ *is the global linear velocity vector,*

$\underline{\dot{\theta}}^{+} = \left[\dot{\theta}^{+}_{1,x}, \dot{\theta}^{+}_{1,y}, \dot{\theta}^{+}_{1,z}, \ldots, \dot{\theta}^{+}_{bodies,x}, \dot{\theta}^{+}_{bodies,y}, \dot{\theta}^{+}_{bodies,z}\right]^T$ *is the global angular velocity vector,*

$\underline{\lambda}_n = \left[\lambda_{n,1}, \ldots, \lambda_{n,contacts}\right]^T$ *is the global normal impulse magnitude vector,*

$\underline{\lambda}_t = \left[\lambda_{t,1}, \ldots, \lambda_{t,contacts}\right]^T$ *is the global tangential impulse magnitude vector*

and $\underline{t} = \left[t_{1,x}, t_{1,y}, t_{1,z}, \ldots, t_{contacts,x}, t_{contacts,y}, t_{contacts,z}\right]^T$ *is the global tangential vector array.*

Such that the following relationships are satisfied for all bodies i and contacts nb:

$$M_i \underline{\dot{x}}_i^{+} - \sum_{nb} (\lambda_{n,ij}^{nb} \underline{n}_{ij}^{nb} + \lambda_{t,ij}^{nb} \underline{t}_{ij}^{nb}) = M_i \underline{\dot{x}}_i^{-}$$

$$I_i \underline{\dot{\theta}}_i^{+} + \sum_{nb} r_i^{nb} \lambda_{t,ij}^{nb} (\underline{n}_{ij}^{nb} \times \underline{t}_{ij}^{nb}) = I_i \underline{\dot{\theta}}_i^{-}$$

$$(\underline{\dot{x}}_i^{nb,+} - \underline{\dot{x}}_j^{nb,+}) \bullet \underline{n}_{ij}^{nb} = -\varepsilon_{n,ij}^{nb} \underline{\dot{x}}_{ij}^{nb,-} \bullet \underline{n}_{ij}^{nb}$$

$$[(\underline{\dot{x}}_i^{nb,+} + r_i^{nb} \underline{n}_{ij}^{nb} \times \underline{\dot{\theta}}_i^{nb,+}) - (\underline{\dot{x}}_j^{nb,+} - r_j^{nb} \underline{n}_{ij}^{nb} \times \underline{\dot{\theta}}_j^{nb,+})] \bullet \underline{t}_{ij}^{nb} = -\varepsilon_{t,ij}^{nb} \underline{\dot{x}}_{ij}^{nb,-} \bullet \underline{t}_{ij}^{nb}$$

$$[(\underline{\dot{x}}_i^{nb,+} + r_i^{nb} \underline{n}_{ij}^{nb} \times \underline{\dot{\theta}}_i^{nb,+}) - (\underline{\dot{x}}_j^{nb,+} - r_j^{nb} \underline{n}_{ij}^{nb} \times \underline{\dot{\theta}}_j^{nb,+})] \bullet (\underline{n}_{ij}^{nb} \times \underline{t}_{ij}^{nb}) = 0$$

$$\underline{n}_{ij}^{nb} \bullet \underline{t}_{ij}^{nb} = 0$$

$$\underline{t}_{ij}^{nb} \bullet \underline{t}_{ij}^{nb} = 1$$

$\lambda_{n,ij}^{nb} \geq 0$ *(contact normal impulse constraints)*

$\lambda_{t,ij}^{nb} = \mu_{ij,dyn} \lambda_{n,ij}^{nb} \cup \lambda_{t,ij}^{nb} > \mu_{ij,stat} \lambda_{n,ij}^{nb}$ *(contact tangential impulse constraints)*

Figure 4.9: The mathematical problem statement for collision resolution.

From the mathematical problem statement in Figure 4.9 above, it can be seen that the simultaneous solution of the set of equations is severely complicated by the additional inequality constraints on the set of solutions with respect to the normal and tangential impulse magnitudes ($\lambda_{n,ij}^{nb}$ and $\lambda_{t,ij}^{nb}$). A further complication of the tangential impulse constraint is that the friction phenomenon is discontinuous, since it allows for tangential force and impulse magnitude to rise to a higher level during non-slipping contact before transitioning to a lower value at slipping contact and, since the tangential impulse direction is also unknown, this becomes an implicit non-linear inequality, but is not a real problem in the formulation used, since only the tangential impulse magnitude is actually considered.

What is generally needed for this problem is some kind of multi-variable solution technique that would yield physically correct and accurate solutions to the set of momentum balance and transfer equations as summarised in Figure 4.9 subject to the constraints shown. Various techniques exist for handling non-linear inequalities, of which this problem is a typical example, and these all have their unique advantages and disadvantages. In order to solve non-linear inequalities, nonlinear equality and linear inequality solution techniques usually need to be incorporated and understood, which in turn all require the usage of linear equality solution techniques, better known as matrix solution techniques, and in particular, hopefully sparse matrix storage and solution techniques. All the aforementioned types of multi-variable solution techniques are discussed in the following sections in the order they had been mentioned.

4.1.3.2.1 Non-Linear Inequality Solution

The pseudo-mathematical nonlinear inequality formulation shown in Figure 4.9 is not necessarily totally unique, since Kraus and Kumar (1997:2), Kraus *et al.* (1998:1-2) and Redon *et al.* (2002:3-4) used Lagrange equations applicable to a general system of rigid bodies for their collision simulation theory, and these are similar to the equations of impulsive interaction derived in this study using the basic Newtonian principles of momentum transfer and impulsive velocity changes, as are formulations given by Kawachi *et al.* (1997:184) and Giang *et al.* (2003:4) to some extent. What does make this study unique is the manner in which the sliding velocity direction is implicitly specified using equation (3.23), avoiding the explicit use of the assumption that energy loss should be maximised. A further unique attribute, as pointed out earlier, is the simplicity of representing the isotropic friction cone avoiding the pyramidal approximation thereof as encountered in other texts. Also unique is the approach taken for the solution of this problem, since it is more physically based than any of the alternative methods to date, at least compared to the methods described in the literature reviewed.

As remarked earlier, several candidates for linear inequality solution techniques can be isolated from the literature surveyed, but it was decided to take a more intuitive, physically based approach in stead, since the optimisation-based techniques are not yet suitable for engineering level applications (Kraus *et al.*, 1998:1-2), (Milenkovic & Schmidl, 2001:2). This does not preclude the possible use of some parts of the techniques employed, such as that of Milenkovic and Schmidl (2001) who employed a Quadratic Programming (QP) technique to solve for unknown impulses and the associated changed linear and angular momentum, later identified by Redon *et al.* (2002:2) as being an instance of Gauss' Least Constraints Principle – GLCP – (Glocker, 1997), (Redon *et al.*, 2002:2-3) at work. The GLCP method is a promising candidate for solving the type of problem to be dealt with in this study, since the number of variables to be optimised for is considerably less than those for the conventional LCP or NCP approaches, but no conclusive evidence had, as yet, been obtained or documented regarding the accuracy of results obtained from GLCP techniques (Redon *et al.*, 2002:6) and thus some of its principles can possibly only be used in a type of intuitive predictor tool to determine where contacts are likely to break or slip, leaving the more detailed calculation to a more accurate solver, since, once points of breaking and sliding contact had been determined, equalities can be instated and the problem can be solved as a nonlinear equality.

For the actual non-linear inequality solution, the technique as described in Figure 4.10, based on intuitive analysis of the physical situation, and incidentally bearing a slight resemblance to the Singular Value Decomposition (SVD) technique proposed by Mirtich (1998:12-14) regarding the handling of negative normal impulses and excessive tangential impulses, emerges. The main motivation and rationale for the particular algorithm developed, is a statement in Stoecker (1989:448-451) regarding the technique for solving inequality constrained problems by using a series of equality constraints. Though the concept of successive equality constraint application might appear to be simple, choosing which equality constraints to apply at what stage is quite tricky. In the case of this study it was assumed that the largest negative normal impulse or impulses during any particular iteration should be the first to be rendered inactive until all negative impulses were eliminated, after which the contact or contacts with the largest tangential-to-normal ratio exceeding the static friction factor should be removed.

```
If a new set of equations is used, guess new initial values for all unknown variables
Check for convergence by evaluating function value sum - terminate loop if below tolerance
If not converged :
  Do one Newton - Raphson iteration
Find contacts with maximum negative normal impulses
If negative normal impulses were found, remove the relevant contacts.
If no negative normal impulses existed :
  Find contacts with maximum tangent/normal impulse ratios exceeding static friction factor
  If exceeding tangent impulses were found :
    Alter relevant contact equations to couple normal and tangent impulse magnitudes
Check for convergence - terminate main loop if below tolerance
```

Figure 4.10: The proposed non-linear collision resolution algorithm.

Though seemingly simple, the algorithm outlined in Figure 4.10 entails a bit of theoretical gymnastics, since generically identifying the exact contacts to remove from the list and the contacts where the tangential impulse equation should be altered is no menial task. As an initial solution/approach, it will be assumed that only the maximum negative impulses are the offending ones and by sequentially removing all those, the system will be nudged towards compliance with respect to normal impulse magnitudes. The tangential impulse equations will also need to be adjusted to account for slippage where necessary as described in the algorithm in Figure 4.10. As can be seen from the algorithm, it is assumed that negative normal impulses should all be eliminated before any tangential slippage checking is done. In order to obtain the temporary solutions each time, a non-linear equality solution algorithm needs to be employed, and one is described in the following section.

4.1.3.2.2 Non-Linear Equality Solution

The most popular and powerful technique for solving non-linear equalities is the Newton-Raphson iteration for multiple variables, e.g. as described in Cheney and Kincaid (1999:110-112). As mentioned earlier, the multi-variable Newton-Raphson iteration is based on the first order Taylor polynomial expansion for multiple variables. When expressed in vector notation (Cheney & Kincaid, 1999:110-112), each iteration's adjustor vector ($\underline{\Delta}^k$) can be implicitly found by solving the mathematical expression in equation (4.7).

$$J(\underline{x}^k)\underline{\Delta}^k = -\underline{F}^k(\underline{x}^k) \tag{4.7}$$

In equation (4.7) each function evaluated has to be of the form $F_i(\underline{x}) = 0$ when the solution vector $\underline{x}$ contains all the roots of the equations, and to avoid confusion henceforth, all function braces and the iteration superscript (^{k}) will be omitted but implied. The Jacobian matrix for any iteration is obtained by evaluating the mathematical expression as given in equation (4.8) below,with J_{ij} the Jacobian matrix (J) entry at row i, column j.

$$J_{ij} = \frac{\partial F_i}{\partial x_j} \tag{4.8}$$

In index-notation, equation (4.7) can be rewritten mathematically as seen in equation (4.9) below.

$$\frac{\partial F_i}{\partial x_j}\Delta_j = -F_i \tag{4.9}$$

Fortunately for the current study, the coefficients of the Jacobian matrix for any iteration can be calculated directly using analytical expressions obtained from analytical differentiation of the governing equations. First, all the relevant equations need to be written in the form $F_i(\underline{x}) = 0$ as required, resulting in equations (4.10) to (4.18) below.

$$M_i \underline{\dot{x}}_i^{+} - \sum_{nb}(\lambda_{n,ij}^{nb}\underline{n}_{ij}^{nb} + \lambda_{t,ij}^{nb}\underline{t}_{ij}^{nb}) - M_i \underline{\dot{x}}_i^{-} = 0 \tag{4.10}$$

$$I_i \underline{\dot{\theta}}_i^{+} + \sum_{nb} r_i^{nb}\lambda_{t,ij}^{nb}(\underline{n}_{ij}^{nb} \times \underline{t}_{ij}^{nb}) - I_i \underline{\dot{\theta}}_i^{-} = 0 \tag{4.11}$$

$$(\underline{\dot{x}}_i^{nb,+} - \underline{\dot{x}}_j^{nb,+}) \bullet \underline{n}_{ij}^{nb} + \varepsilon_{n,ij}^{nb}\underline{\dot{x}}_{ij}^{nb,-} \bullet \underline{n}_{ij}^{nb} = 0 \tag{4.12}$$

$$[(\underline{\dot{x}}_i^{nb,+} + r_i^{nb}\underline{n}_{ij}^{nb} \times \underline{\dot{\theta}}_i^{nb,+}) - (\underline{\dot{x}}_j^{nb,+} - r_j^{nb}\underline{n}_{ij}^{nb} \times \underline{\dot{\theta}}_j^{nb,+})] \bullet \underline{t}_{ij}^{nb} + \varepsilon_{t,ij}^{nb}\underline{\dot{x}}_{ij}^{nb,-} \bullet \underline{t}_{ij}^{nb} = 0 \tag{4.13}$$

$$[(\underline{\dot{x}}_i^{nb,+} + r_i^{nb}\underline{n}_{ij}^{nb} \times \underline{\dot{\theta}}_i^{nb,+}) - (\underline{\dot{x}}_j^{nb,+} - r_j^{nb}\underline{n}_{ij}^{nb} \times \underline{\dot{\theta}}_j^{nb,+})] \bullet (\underline{n}_{ij}^{nb} \times \underline{t}_{ij}^{nb}) = 0 \tag{4.14}$$

$$\underline{n}_{ij}^{nb} \bullet \underline{t}_{ij}^{nb} = 0 \tag{4.15}$$

$$\underline{t}_{ij}^{nb} \bullet \underline{t}_{ij}^{nb} - 1 = 0 \tag{4.16}$$

For the contacts where normal impulse magnitudes are in breach of constraints, equation (4.12) should either be replaced by equation (4.17) below, or the contact should be totally removed.

$$\lambda_{n,ij}^{nb} = 0 \cup \lambda_{n,ij}^{nb} < 0 \tag{4.17}$$

And for the contacts where the tangential impulse exceeds the static friction limit as governed by the normal impulse, equation (4.13) should be replaced by equation (4.18) below.

$$\lambda_{t,ij}^{nb} - \mu_{ij,dyn}\lambda_{n,ij}^{nb} = 0 \cup \lambda_{t,ij}^{nb} > \mu_{ij,stat}\lambda_{n,ij}^{nb} \tag{4.18}$$

The Newton-Raphson iteration equations to be simultaneously solved for all the adjustors can then be written as in equations (4.19) to (4.26) below following manipulation and simplification and substituting $\underline{o}_{ij}^{nb}$ for $\underline{n}_{ij}^{nb} \times \underline{t}_{ij}^{nb}$ where possible, always using only the latest known previous iteration (^{k}) values of the variables being solved for wherever they occur in expressions.

$$\begin{aligned} & M_i \underline{\Delta\dot{x}}_i^{+} - \sum_{nb}(\underline{n}_{ij}^{nb}\Delta\lambda_{n,ij}^{nb} + \underline{t}_{ij}^{nb}\Delta\lambda_{t,ij}^{nb} + \lambda_{t,ij}^{nb}\underline{\Delta t}_{ij}^{nb}) \\ & = M_i(\underline{\dot{x}}_i^{-} - \underline{\dot{x}}_i^{+}) + \sum_{nb}(\lambda_{n,ij}^{nb}\underline{n}_{ij}^{nb} + \lambda_{t,ij}^{nb}\underline{t}_{ij}^{nb}) \end{aligned} \tag{4.19}$$

$$I_i \underline{\Delta\dot{\theta}}_i^{+} + \sum_{nb}[r_i^{nb}\underline{o}_{ij}^{nb}\Delta\lambda_{t,ij}^{nb} + r_i^{nb}\lambda_{t,ij}^{nb}(\underline{n}_{ij}^{nb} \times \underline{\Delta t}_{ij}^{nb})] = I_i(\underline{\dot{\theta}}_i^{-} - \underline{\dot{\theta}}_i^{+}) - \sum_{nb} r_i^{nb}\lambda_{t,ij}^{nb}\underline{o}_{ij}^{nb} \tag{4.20}$$

$$(\underline{\Delta\dot{x}}_i^{nb,+} - \underline{\Delta\dot{x}}_j^{nb,+}) \bullet \underline{n}_{ij}^{nb} = (\underline{\dot{x}}_j^{nb,+} - \underline{\dot{x}}_i^{nb,+}) \bullet \underline{n}_{ij}^{nb} - \varepsilon_{n,ij}^{nb}\underline{\dot{x}}_{ij}^{nb,-} \bullet \underline{n}_{ij}^{nb} \tag{4.21}$$

$$\begin{aligned} & [(\underline{\Delta\dot{x}}_i^{nb,+} - \underline{\Delta\dot{x}}_j^{nb,+}) \bullet \underline{t}_{ij}^{nb} - r_i^{nb}\underline{o}_{ij}^{nb} \bullet \underline{\Delta\dot{\theta}}_i^{nb,+} - r_j^{nb}\underline{o}_{ij}^{nb} \bullet \underline{\Delta\dot{\theta}}_j^{nb,+} + \underline{\dot{x}}_{ij}^{nb,+} \bullet \underline{\Delta t}_{ij}^{nb} \\ & = -\underline{\dot{x}}_{ij}^{nb,+} \bullet \underline{t}_{ij}^{nb} - \varepsilon_{t,ij}^{nb}\underline{\dot{x}}_{ij}^{nb,-} \bullet \underline{t}_{ij}^{nb} \end{aligned} \tag{4.22}$$

$$\begin{aligned} & [(\underline{\Delta\dot{x}}_i^{nb,+} - \underline{\Delta\dot{x}}_j^{nb,+}) \bullet \underline{o}_{ij}^{nb} + r_i^{nb}\underline{t}_{ij}^{nb} \bullet \underline{\Delta\dot{\theta}}_i^{nb,+} + r_j^{nb}\underline{t}_{ij}^{nb} \bullet \underline{\Delta\dot{\theta}}_j^{nb,+} + (\underline{\dot{x}}_{ij}^{nb,+} \times \underline{n}_{ij}^{nb}) \bullet \underline{\Delta t}_{ij}^{nb} \\ & = -\underline{\dot{x}}_{ij}^{nb,+} \bullet \underline{o}_{ij}^{nb} \end{aligned} \tag{4.23}$$

$$\underline{n}_{ij}^{nb} \bullet \underline{\Delta t}_{ij}^{nb} = -\underline{n}_{ij}^{nb} \bullet \underline{t}_{ij}^{nb} \tag{4.24}$$

$$2\underline{t}_{ij}^{nb} \bullet \underline{\Delta t}_{ij}^{nb} = 1 - \underline{t}_{ij}^{nb} \bullet \underline{t}_{ij}^{nb} \tag{4.25}$$

$$\Delta\lambda_{t,ij}^{nb} - \mu_{ij,dyn}\Delta\lambda_{n,ij}^{nb} = \mu_{ij,dyn}\lambda_{n,ij}^{nb} - \lambda_{t,ij}^{nb} \cup \lambda_{t,ij}^{nb} > \mu_{ij,stat}\lambda_{n,ij}^{nb} \tag{4.26}$$

In the foregoing equations all the symbols containing Δ represent the adjustor components to be solved for simultaneously using a linear equality solver technique, after which the adjustor components are added to the values of the previous iteration, demonstrated in vector format in equation (4.27) below, with the superscript (^{k}) once more referring to iteration number (Cheney & Kincaid, 1999:112).

$$\underline{x}^{k+1} = \underline{x}^{k} + \underline{\Delta}^{k} \tag{4.27}$$

After new solution values had been obtained, the iteration can be restarted after function evaluations had been done to ascertain whether the current solution is not sufficiently converged, in which case the iterations can terminate. The choice of initial values for the components of $\underline{x}^k (= \underline{x}^0)$ is of great importance and this will be discussed in a later section. The general Newton-Raphson algorithm implemented as part of the algorithm for this study can be seen in Figure 4.11 below.

1. *Evaluate each of the n functions in the system using the latest available solution vector, entering the values into the right-hand vector, $\underline{F}$, whilst also accumulating their absolute values to obtain a residual, r^k.*

$$r^k = \sum_{i=1}^{n} \left| F_i^k (\underline{x}^k) \right|$$

2. *If $r^k \geq r_{tolerance}$, continue iterations by going to the next step.*

   Else, terminate iteration.

3. *Set up Jacobian matrix, J, shown here in index format for the entry at row i, column j.*

$$J_{ij}^k = \frac{\partial F_i}{\partial x_j} (\underline{x}^k)$$

4. *Solve for adjustors (Δ_j^k) from the matrix-vector equation, shown in index-format here.*

$$J_{ij}^k \Delta_j^k = -F_i^k$$

5. *Adjust all old solution values, and return to step 1, using the new solution values.*

$$x_i^{k+1} = x_i^k + \Delta_i^k$$

Figure 4.11: The general Newton-Raphson multi-variable solution algorithm steps.

From the algorithm outlined in Figure 4.11 the necessity for some form of linear equality solver is clearly demonstrated. The best way to handle a linear equality solution is to cast the linear equations into an augmented matrix format and solve using some matrix solution technique – see Appendix A, section A.2 for an example of how this algorithm is deployed in actual matrix-vector format for the non-linear solution approach described in Figure 4.10. The matrix solution techniques available and used in this study are described in the following section.

4.1.3.2.3 Linear Equality Solution

The most basic and oft-used theory encountered in engineering pertains to the solution of sets of linear equations with implicitly related variables, which is usually accomplished using various

matrix solution techniques. Some methods are direct, and others are iterative – see Laurie (1983:204-224) and Barett *et al.* (1994:5-38), each with its particular pros and cons. A phenomenon often encountered in engineering is that of the sparse matrix, as is evidently the case with this study, since no rows contain all the linear components to be solved for – see equations (4.19) to (4.26) for clarity. In order to solve sparse matrices one needs to represent them first in some or other kind of data structure, various matrix operations need to be supported and subsequently one has to choose and employ an appropriate sparse matrix solution technique. These aspects are discussed more extensively below.

4.1.3.2.3.1 Sparse Matrix Setup and Representation

In order to exploit sparse matrix properties, the entries in a matrix should, literally, be few and far between. Fortunately, the type of problem considered in this study yields such linear systems, with matrix fill (the ratio of non-zero entries to total entries) ranging from about 75 % for small problems to less than 5 % for massive problems. Apart from the computational advantages, i.e. there are fewer coefficients to actually multiply, there is also the advantage of being able to make use of matrix entry sparseness to minimise memory usage. For more reading on sparse matrix issues related to storage, Barret *et al.* (1994:63-68) is recommended, where it can be seen that several sparse matrix entry storage schemes are available, but for this study the compressed row format (Barret *et al.*, 1994:64-65) was chosen as it can represent the coefficients resulting from the relevant equations in an easily understandable manner whilst retaining efficiency during setup and calculation phases. In order to simplify, abstract and automate the process of matrix entry input, several tools need to be developed, the most important of which is the matrix entry list object, which reduces the input from the user to merely entering row and column indices and coefficient value for each matrix coefficient. Since an entry list would greatly abstract the process of matrix coefficient entry, it would make it possible to implement other types of storage schemes, should they prove to be more beneficial, without affecting the end use/user coding at all.

Once storage scheme and strategies had been decided, the various matrix solution algorithms can be implemented and these as well as the various supporting operations are discussed briefly in the following sections.

4.1.3.2.3.2 Sparse Matrix Solution

Sparse matrix solution techniques are becoming evermore important as the number of applications for matrix theory increases, along with the sizes of the linear systems encountered in various genres, engineering being only one of many. Laurie (1983:204-224) and Barret *et al.* (1994) provide a good introduction to and summary of the various aspects related to sparse matrix solution techniques. Other sources of information on these techniques and related theory, such as Press *et al.* (1992:96-98, 833), Bücker (2002), Oppe (2002), Zhang (2002), Alanelli and Hadjimos (2004), Axelsson (2003), Yang (2003), Axelsson and Karátson (2004), Yun & Kim (2004) are available too, dealing with specific issues such as parallelisation (Bücker, 2002), preconditioning (Zhang, 2002) and convergence behaviour analysis (Axelsson, 2003), (Yang, 2003), (Axelsson & Karátson, 2004), (Yun & Kim, 2004). The various necessary supporting operations and the main solution algorithms are briefly discussed or summarised in the following sub sections and the actual pseudo-code will be shown in the implementation section if deemed necessary. The actual implementations of these supporting routines or operations all need to be highly efficient with respect to memory usage and execution speed and reliable with respect to the numerical results obtained.

- Matrix Pre-Conditioning

Sparse matrices can and do become very ill conditioned in certain instances, usually when the sum of the off-diagonal entries are considerably larger than the corresponding diagonal entries. The direct solution methods, such as Gauss-elimination, can cope with the problem of ill conditioning to a degree by pivoting, but most sparse matrix solution techniques do not have the luxury of being able to exchange rows, due to difficulties related to storage issues. In order to address such problems, matrix preconditioning usually needs to be done and it helps with both improving the chances of finding a solution to a sparse linear system and the speed of convergence during iterations. The incomplete Choleski preconditioner (Laurie, 1983:219) had been chosen as the main type for this study, since it is fairly quick and reasonably simple to implement, but a sparse approximate inverse preconditioner proposed and investigated by Zhang (2002) and sparse approximate preconditioner by González *et al.* (2004) might also be advantageous to have available for use with particularly ill conditioned systems.. For more information on preconditioning in general, the texts by Laurie (1983:216-219), Barret *et al.* (1994:39-56) and González *et al.* (2004:1) provide a good overview of most available preconditioning techniques and the rationale for them.

- Upper and Lower Triangle Solution

It will become apparent in the actually implemented algorithms that one can approximate the solution of the preconditioned matrix by doing a pseudo-LU-decomposition which is in actual fact still part of the preconditioning and entails solving for approximate solutions by back-substitution for the preconditioned super-diagonal, diagonal and sub-diagonal matrices in order – see Laurie (1983:216-219) and Barret *et al.* (1994:43-44). This technique seems to be loosely based on or related to the Choleski factorisation as described in Laurie (1983:206-207). The same can be done for the transpose of a matrix by doing a pseudo-LU-decomposition for the transposed upper and lower triangular and diagonal matrices.

- Vector-Scalar Product

- Vector-Vector Scalar Product

- Matrix-Vector Multiplication

- Iterative Sparse Matrix Solution

The ultimate goal of a set of linear equalities is to solve for various linearly inter-related variables simultaneously, and in many engineering applications such systems of equations tend to be very sparse, especially as the number of inter-related unknowns increase. This phenomenon has advantages with respect to storage space required, but it presents several problems at the same time. Most sparse matrix solution techniques have to be developed to avoid alteration of the coefficients present in the matrix, since this usually leads to complications such as fill-in and row-swapping when, for instance, Gauss elimination with pivoting is employed. All of the methods employed in this study were obtained from Barret *et al.* (1994:5-38) and more information on them can be obtained from there. Methods that were earmarked for implementation in this study were the Bi-Conjugate Gradient (BiCGM), Bi-Conjugate Gradient Stabilised (BiCGSTAB) and Conjugate Gradient Squared (CGSQUARE) algorithms. One should always keep in mind that these methods might not work, and one might need to resort to direct solution techniques after all, thus restricting one to fairly small problems due to storage considerations, yet allowing the simple inversion of matrices otherwise very hard to accomplish.

4.2 IMPLEMENTATION

Implementing all the necessary data structures and the routines to operate on the data therein is a crucial activity in any software project, including this one, especially with the object oriented approach being employed. The usual software development process, as typically described in section 3.3, was followed to arrive at the architecture and the various aspects thereof as summarised in the following sections. The modified C++ coding standards as described in the document obtained from M-Tech Industrial (Van Heerden, 2000) were adhered to as much as possible, including the use of Hungarian prefix qualifiers for variable names.

4.2.1 Software Architecture

Resulting from the user requirements specified, which in turn had been obtained or deduced from the study objectives, the software architecture as seen in Figure 4.12 below had been devised. The diagram in Figure 4.12 is only a partial overview of the most important components present in the code developed and in turn only some of these will be discussed in detail in the following two sections.

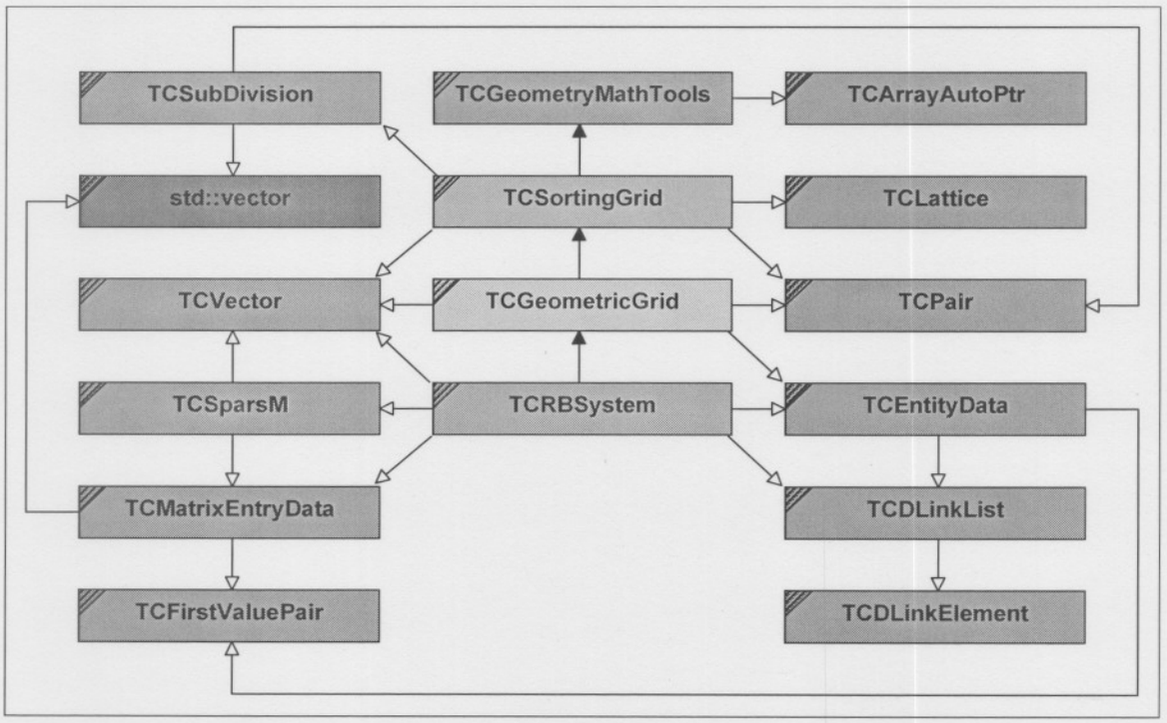


Figure 4.12: Pseudo-UML hierarchy and collaboration diagram.

The higher level objects, such as TCSortingGrid, TCGeometricGrid and TCRBSystem are all template classes at the heart of the simulation code developed and together they address the various requirements to be met for this project. The lower level storage objects such as TCPair,

TCFirstValuePair, TCArrayAutoPtr, TCVector, TCLattice, TCSubDivision, TCEntityData, TCMatrixEntryData, TCDLinkList, TCDLinkElement all mainly fulfil container object roles and they are therefore all created as template classes, developed during the course of this study. The *vector* container available in the C++ Standard Template Library (STL) is employed where memory and execution speed efficiency is of great importance and is only included in the hierarchical diagram of Figure 4.12 for the sake of completeness, since it had not been self-developed in the course of this study.

Two other objects developed for this study are the TCGeometryMathTools and TCSparsM template classes. The former provides basic 3-D geometric calculation functionality, such as point-to-line projections, line-to-line projections, point-to-plane projection, line-plane intersections, etc., and can be included as a base class by any C++ class, as long as the TCArrayAutoPtr class is also available, since pre-allocated "scratch-pad" arrays of that type are used to avoid unnecessary time consuming memory allocation and de-allocation during repeated function calls. The TCSparsM class provides the sparse matrix storage functionality and such objects can be used by various sparse matrix solution routines that are provided as globally available functions. The architectural overview in Figure 4.12 does not reflect the actual mode of usage for all the collaborating objects, since usage actually resorts under two categories, namely data storage and data operation. The various data structures, corresponding to data storage use, and the implemented supporting routines, corresponding to data operation use, are discussed in the following sections.

4.2.2 Data Structures

As mentioned before, data storage is accomplished by employing data structures representing the data to be used in whatever capacity they are needed in the code. Various data storage objects had been developed in the course of this study to meet the needs of the algorithms and routines that have been implemented.

4.2.2.1 General Array Storage

In various instances there is the need for using a raw pointer associated with dynamically allocated array memory. Since it is easy to forget to delete a dynamically allocated array, and to do it correctly to boot, it is more advantageous to make use of an auto-pointer, such as the TCArrayAutoPtr template class. It is employed in the same manner as a normal pointer, but guarantees dynamically allocated memory to be freed up whenever it goes out of scope. It also prevents unassigned pointers from having a non-null value, which can cause all sorts of memory

access related problems, since a constructor takes care of that. A related TCObjectPtr auto-pointer template is also available for non-array dynamic pointer usage. The concept of auto-pointers in C++ is similar to that of the Java pointer type, with its garbage collection. The C++ STL has a built-in auto-pointer template class, named *auto_ptr*, too, but some of its features make it less suitable for usage in the context of this study and thus necessitated the development of a similar template object. In addition to the aforementioned, the concept of an array-type of auto-pointer is not at all native to the STL.

4.2.2.2 Specialised Array Storage

The most common and efficient way of storing data, is in contiguous arrays. Since arrays are useful data structures, it is always wise to implement them as template classes, so that they can contain any data types. Sometimes, mathematical functionality is required, and this should also be provided. It was decided to group purely container-type arrays in a namespace called *Container*, whilst the versions supporting various vector-matrix mathematical operations were implemented without namespace.

4.2.2.2.1 Vector Storage

There is a *Container* and a mathematical version available for this one-dimensional array template named TCVector. Essentially it is a template encapsulation of a dynamically allocated array pointer (very similar to the TCArrayAutoPtr) supporting resizing, copying, and raw pointer extraction amongst other functionalities. The general data members for a TCVector template can be seen in Figure 4.13 below.

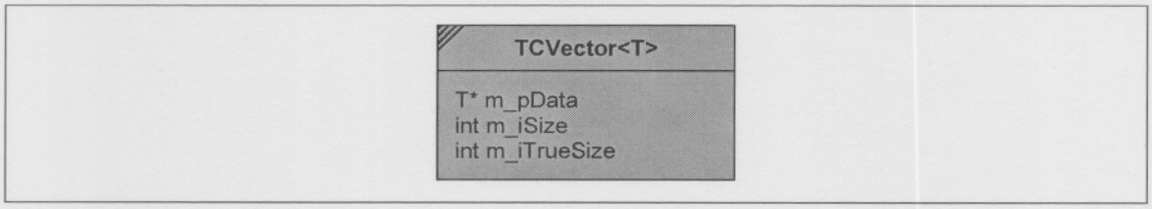


Figure 4.13: TCVector data members.

4.2.2.2.2 Sparse Vector Storage

Sparse vector storage is done by simply using a full TCVector with zero entries where appropriate.

4.2.2.2.3 Matrix Storage

There is a *Container* and a mathematical version available for this two-dimensional array template named TCMatrix. Essentially it is a template encapsulation of a dynamically allocated

pointer array pointer supporting resizing, copying, and raw pointer extraction amongst other functionalities. The mathematical version also supports Pivoting Gauss elimination matrix solution, as well as inverse, transpose and determinant calculation. The general data members for the TCMatrix template can be seen in Figure 4.14 below.

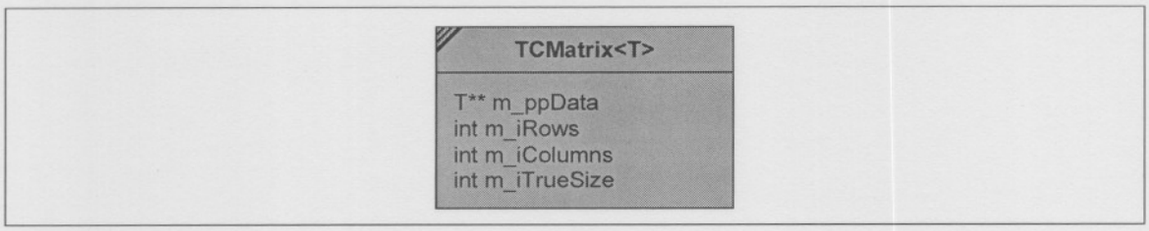


Figure 4.14: TCMatrix data members.

4.2.2.2.4 Sparse Matrix Storage

Sparse matrix storage reduces to storing the diagonal entries, the super-diagonal coefficients and their corresponding indices and the sub-diagonal coefficients and their corresponding indices, as well as some necessary offset indices. This data structure can be utilised whether the storage is compressed row or compressed column format, but for this study the former option was selected, since it is easier to enter data row-by-row for the particular set of linear equations.

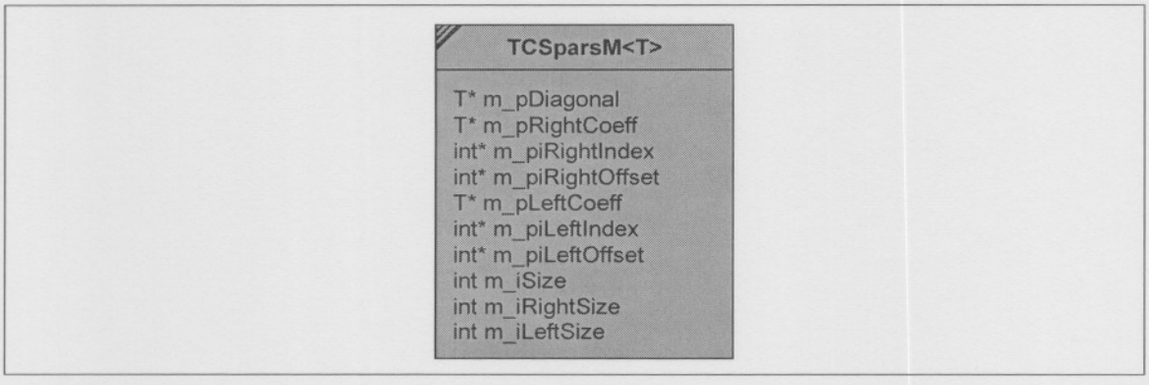


Figure 4.15: TCSparsM data members.

4.2.2.2.5 Lattice Storage

There is a *Container* and a mathematical version available for this three-dimensional array template named TCLattice. Essentially it is a template encapsulation of a dynamically allocated double pointer array pointer supporting resizing, copying, and raw pointer extraction amongst other functionalities. The general data members for the TCLattice template can be seen in Figure 4.16.

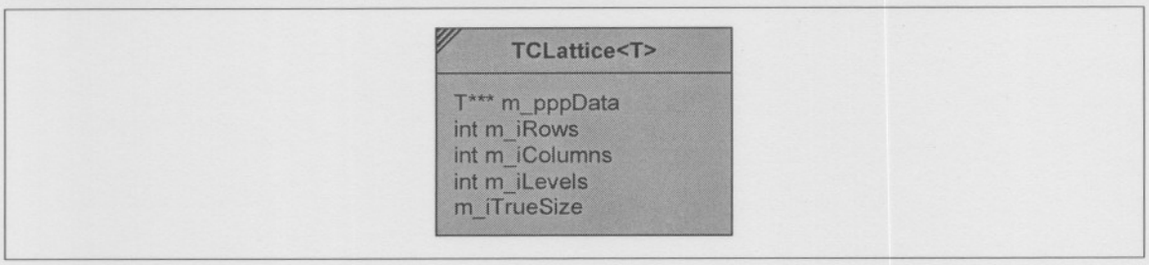


Figure 4.16: TCLattice data members.

4.2.2.3 Sorting Results Storage

The sorting algorithm needs some form of three dimensional representation or storage scheme for its results, and such a representation is what the TCSortingGrid, of which some data members are shown in Figure 4.17, provides. It is basically a three-dimensional lattice of "bins" of type TCSubDivision, each of which containing two dynamic arrays of object index-and-type-ID pairs as shown in the general data member summary shown in Figure 4.18. The lists in each bin can be used to check for inter-object contacts of physically proximate objects.

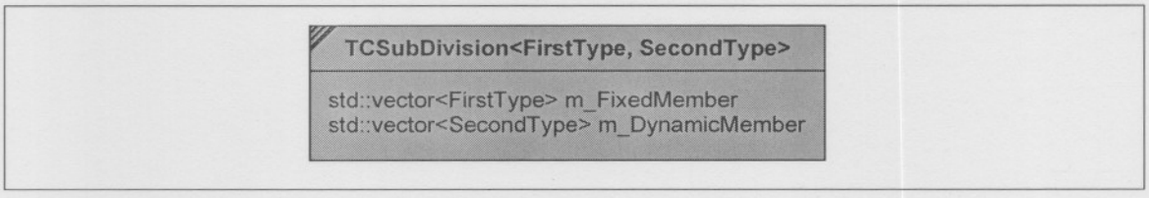


Figure 4.17: TCSubDivision data members.

Apart from the "bin" lattice, TCSortingGrid also contains additional information on the Cartesian sorting grid domain not shown in Figure 4.18 for the sake of avoiding clutter, e.g. the original and order of magnitude adjusted minima, maxima and bin division lower and upper offsets in the x, y and z Cartesian directions, the dimensions of the bins, overlap for each bin and some other information and "scratch pad" variables used during object location.

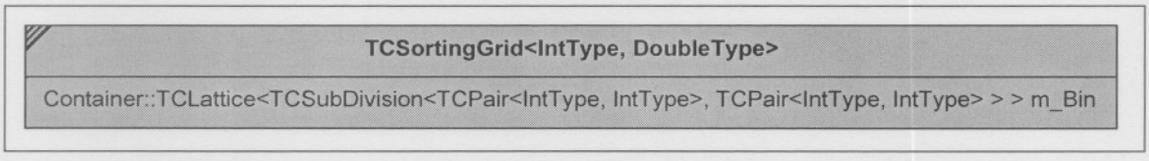


Figure 4.18: TCSortingGrid data members.

In order to do the necessary geometrical calculations, the TCGeometryMathTools utility decorator class is included as a base for TCSortingGrid and more will be said about this decorator template in the algorithm section.

4.2.2.4 Generic Data Storage

All the physical objects that need to be represented in this study have members that are of Boolean, Integer, Real, Integer Pair List or Integer List type. Representing the data for a collection of objects of the same type as rows in the appropriate types of matrices can greatly speed up access and calculation times, since object overheads are kept to a minimum. It also simplifies and generalises the function arguments whenever parameters need to be passed to generic functions, such as those used for copying or resizing. Seen as a whole, this approach is akin to the use of database tables, except that each of the "tables", i.e. matrices, in this case contains only variables of a single type, but their columns are still analogue to fields with names.

An example using the Vertex object type might serve to illustrate the concept. Vertices have various properties of importance to the present study, such as the flag indicating that the vertex had changed (Boolean), the material type index (Integer), the Cartesian x, y and z coordinates (Real) and the lists of edge, facet and contact neighbours (Integer Pair List). If, for example, provision is made for 10 vertices in the data structures, there should be a matrix with 10 rows and 1 column of Boolean entries, a matrix with 10 rows and 1 column of Integer entries, a matrix with 10 rows and 3 columns of Real entries and a matrix with 10 rows and 3 columns Integer Pair List entries.

The template container TCEntityData, with general data members shown in Figure 4.19 below, was developed to handle the memory allocation and access for such representations in a generic manner. The number of columns for each of the Boolean, Integer, Real, Integer Pair List and Integer List matrices is set to accommodate the number of variables of each type present for a particular type of entity, such as a material, vertex, edge, facet or rigid body, and the number of rows for each matrix is set to make provision for the maximum allowed number of entities expected to be used for that particular type.

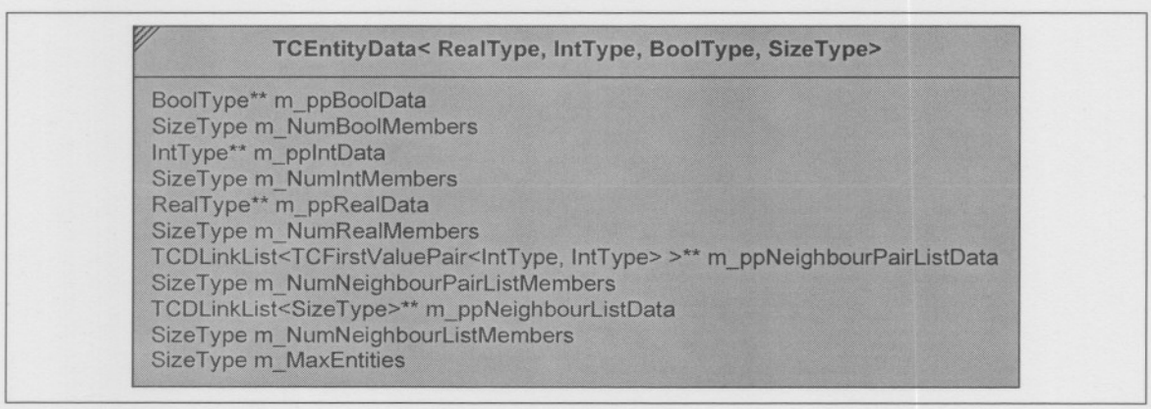


Figure 4.19: TCEntityData data members.

To extract data from a particular matrix, one simply needs to use the appropriate accessor, e.g. Int() for obtaining the full integer matrix, Int(iEntityNumber) to extract the integer data row for a particular entity or Int(iEntityNumber, iMember) for a specific integer of a particular entity.

4.2.2.5 Grid Data Storage

Since a geometrical grid consists of a collection of inter-connected and inter-related/dependent vertices, edges and facets, it is viewed as the basic unit/object encapsulating and maintaining all those objects it is comprised of. No unnecessary data needs to be passed between the various components of a grid, since the master-object, i.e. the geometrical grid, maintains and manages that data and facilitates or coordinates the interactions between its constituting components. For this study, the TCGeometricGrid template is derived from the TCSortingGrid base class in order to avail to it all the base class' sorting as well as the geometrical calculation capabilities and the data relevant to its various grid components are stored in the general entity storage format as provided by the TCEntityData container template. The most important data members are depicted in Figure 4.20 and then discussed in the paragraph below and the subsequent sub-sections.

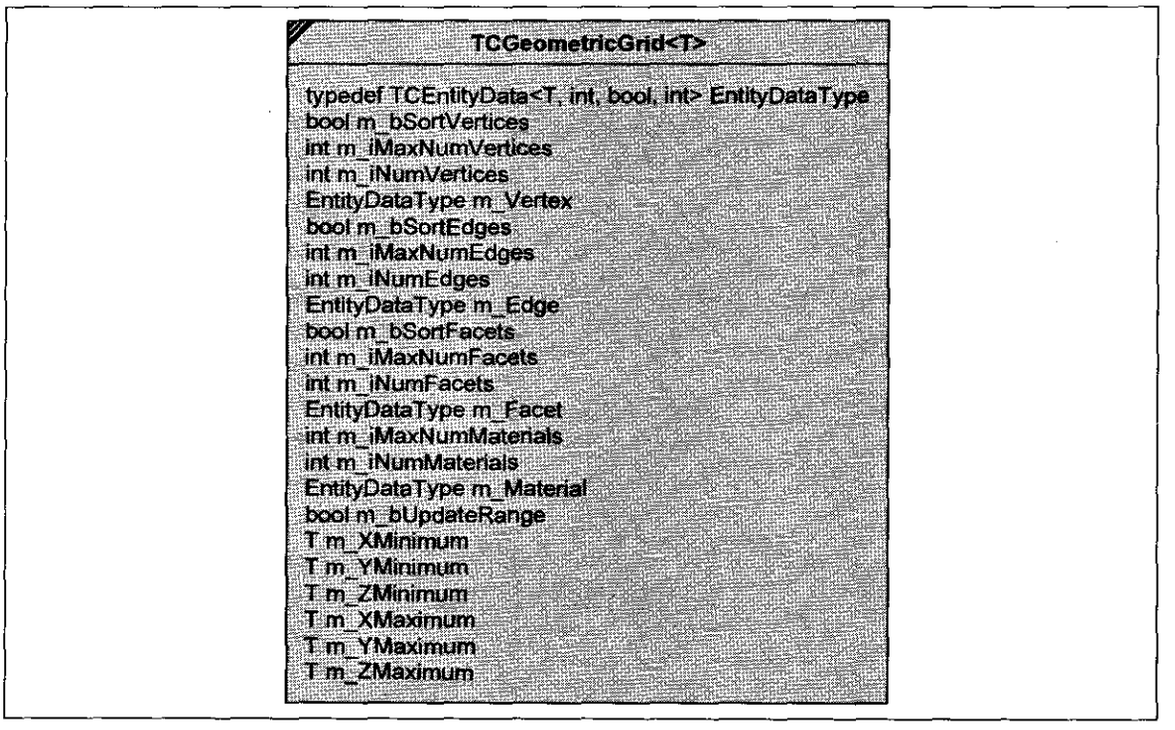


Figure 4.20: TCGeometricGrid data members.

The Boolean flags m_bSortVertices, m_bSortEdges, m_bSortFacets, i.e. Sorting Flags, are used to indicate whether any re-sorting of the associated entities/objects into bins needs to be done, and are set to true whenever anything changes for any object or objects of a particular type. The

flags will also need to be set to true for all hierarchically higher objects, e.g. whenever a vertex' data changes, all edges and facets connected to it also need to be updated, but if, for instance, a facet changes, only the facets need to be re-sorted. The flag m_bUpdateRange is set to true whenever a new vertex coordinate lies outside the present range as indicated by m_XMinimum, m_XMaximum, m_YMinimum, m_YMaximum, m_ZMinimum and m_ZMaximum, causing the sorting grid range to be adjusted and also setting all the Sorting Flags to true. All flags are set to false after the required operation is completed, e.g. after the vertices were sorted, the m_bSortVertices flag is switched off and when the range had been updated for the sorting grid, the m_bUpdateRange flag is set to false.

4.2.2.5.1 Material Storage

Using the TCEntityData template, the material data is generically stored and the enumerations to assist in data extraction based on the various member type matrix column indices are defined as shown in the code excerpt in Figure 4.21 below.

```
enum EMaterialBool
{
   embMaterialIsSolid = 1,
   // Always Add and Remove Enumerations Above This Comment
   embMax
};

enum EMaterialInt
{
   // Always Add and Remove Enumerations Above This Comment
   emiMax = 1
};

enum EMaterial
{
   emMaterialDensity = 1,
   emMaterialConductivity = 2,
   emMaterialSpecificHeat = 3,
   emMaterialViscosity = 4,
   emMaterialNormalRestitutionFactor = 5,
   emMaterialTangentialRestitutionFactor = 6,
   emMaterialStaticFrictionFactor = 7,
   emMaterialDynamicFrictionFactor = 8,
   // Always Add and Remove Enumerations Above This Comment
   emMax
};

enum EMaterialNeighbourPairList
{
   emnMaterialRigidBodyTypeNeighbours = 1,
   // Always Add and Remove Enumerations Above This Comment
   emnpMax
};

enum EMaterialNeighbourList
{
   // Always Add and Remove Enumerations Above This Comment
   emnMax = 1
};
.
.
.
m_Material.Set(m_iMaxNumMaterials,  embMax-1,  emiMax-1,  emMax-1,  emnpMax-1,  emnMax-1);
```

Figure 4.21: Material data member and helper enumerator definitions.

Some presently unused variables, relevant to heat transfer solutions, are provided for by way of the enumerations emMaterialConductivity and emMaterialSepcificHeat and due to the genericity of the code there would be no impact at all on existing code if such enumerations were to be removed, provided they were not used in any code within the project. It would be equally easy to add additional variables, such at those required for magnetic flux calculations, once again impacting only on the code where the variables are actually used, not even where parameters are passed, since this is most often done by pointer.

4.2.2.5.2 Vertex Storage

Using the TCEntityData template, the vertex data is generically stored and the enumerations to assist in data extraction based on the various member type matrix column indices are defined as shown in the code excerpt in Figure 4.22 below.

```
enum EVertexBool
{
   evbVertexChanged = 1,
   // Always Add and Remove Enumerations Above This Comment
   evbMax
};

enum EVertexInt
{
   eviVertexMaterial = 1,
   eviVertexHasNeighbours,
   // Always Add and Remove Enumerations Above This Comment
   eviMax
};

enum EVertex
{
   evVertexX = 1,
   evVertexY,
   evVertexZ,
   // Always Add and Remove Enumerations Above This Comment
   evMax
};

enum EVertexNeigbbourPairList
{
   evnpVertexEdgeNeighbours = 1,
   evnpVertexFacetNeighbours,
   evnpVertexCellNeighbours,
   evnpVertexContactNeighbours,
   // Always Add and Remove Enumerations Above This Comment
   evnpMax
};

enum EVertexNeigbbourList
{
   evnVertexContactNeighbours = 1,
   // Always Add and Remove Enumerations Above This Comment
   evnMax
};
.
.
.
m_Vertex.Set(m_iMaxNumVertices, evbMax-1, eviMax-1, evMax-1, evnpMax-1, evnMax-1);
```

Figure 4.22: Vertex data member and helper enumerator definitions.

4.2.2.5.3 Edge Storage

Using the TCEntityData template, the edge data is generically stored and the enumerations to assist in data extraction based on the various member type matrix column indices are defined as shown in the code excerpt in Figure 4.23 below.

```
enum EEdgeBool
{
   eebEdgeChanged = 1,
   // Always Add and Remove Enumerations Above This Comment
   eebMax
};

enum EEdgeInt
{
   eeiEdgeVertex1 = 1,
   eeiEdgeVertex2,
   eeiEdgeMaterial,
   eeiEdgeHasNeighbours,
   // Always Add and Remove Enumerations Above This Comment
   eeiMax
};

enum EEdge
{
   eeEdgeX = 1,
   eeEdgeY,
   eeEdgeZ,
   eeEdgeLength,
   eeEdgeUnitX,
   eeEdgeUnitY,
   eeEdgeUnitZ,
   // Always Add and Remove Enumerations Above This Comment
   eeMax
};

enum EEdgeNeigbbourPairList
{
   eenpEdgeFacetNeighbours = 1,
   eenpEdgeCellNeighbours,
   // Always Add and Remove Enumerations Above This Comment
   eenpMax
};

enum EEdgeNeigbbourList
{
   eenEdgeContactNeighbours = 1,
   // Always Add and Remove Enumerations Above This Comment
   eenMax
};
.
.
.
m_Edge.Set(m_iMaxNumEdges, eebMax-1, eeiMax-1, eeMax-1, eenpMax-1, eenMax-1);
```

Figure 4.23: Edge data member and helper enumerator definitions.

4.2.2.5.4 Facet Storage

Using the TCEntityData template, the facet data is generically stored and the enumerations to assist in data extraction based on column indices are defined as shown in the code excerpt in Figure 4.24.

```
enum EFacetBool
{
   efbFacetChanged = 1,
   // Always Add and Remove Enumerations Above This Comment
   efbMax
};

enum EFacetInt
{
   efiFacetVertex1 = 1,
   efiFacetVertex2,
   efiFacetVertex3,
   efiFacetMaterial,
   efiFacetEdge1,
   efiFacetEdge2,
   efiFacetEdge3,
   efiFacetNeighbour1,
   efiFacetNeighbour2,
   // Always Add and Remove Enumerations Above This Comment
   efiMax
};

enum EFacet
{
   efFacetX = 1,
   efFacetY,
   efFacetZ,
   efFacetNormalX,
   efFacetNormalY,
   efFacetNormalZ,
   efFacetArea,
   efFacetPerpendicularPointSide1X,
   efFacetPerpendicularPointSide1Y,
   efFacetPerpendicularPointSide1Z,
   efFacetPerpendicularPointSide2X,
   efFacetPerpendicularPointSide2Y,
   efFacetPerpendicularPointSide2Z,
   efFacetPerpendicularPointSide3X,
   efFacetPerpendicularPointSide3Y,
   efFacetPerpendicularPointSide3Z,
   // Always Add and Remove Enumerations Above This Comment
   efMax
};

enum EFacetNeigbbourPairList
{
   efnpFacetCellNeighbours = 1,
   efnpFacetContactNeighbours,
   // Always Add and Remove Enumerations Above This Comment
   efnpMax
};

enum EFacetNeigbbourList
{
   efnFacetContactNeighbours = 1,
   // Always Add and Remove Enumerations Above This Comment
   efnMax
};
.
.
.
m_Facet.Set(m_iMaxNumFacets, efbMax-1, efiMax-1, efMax-1, efnpMax-1, efnMax-1);
```

Figure 4.24: Facet data member and helper enumerator definitions.

4.2.2.6 Rigid Body System Related Storage

As with all the TCGeometricGrid components, such as vertices, edges and facets, the rigid body type, rigid body and contact related variables are stored using the generic TCEntityData container template in the TCRBSystem template derived from TCGeometricGrid to avail to it all grid-related and sorting capabilities and properties, as well as the geometric calculation methods. A summary of the data members in the TCRBSystem template can be seen in Figure 4.25 below.

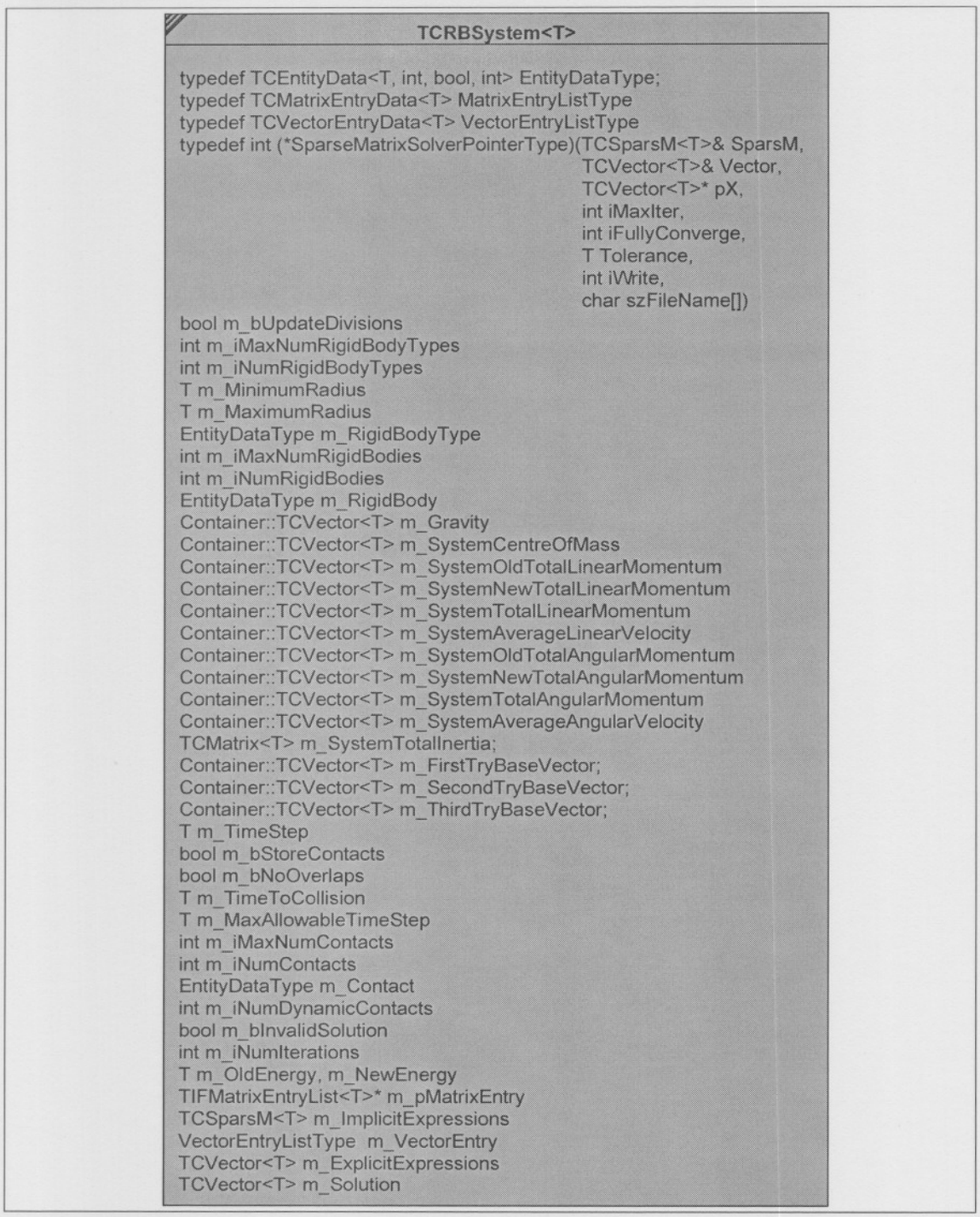


Figure 4.25: TCRBSystem data members.

The m_Material, m_Vertex, m_Edge, m_Facet, m_RigidBodyType, m_RigidBody and m_Contact members of TCRBSystem contain the data necessary for simulations to be done.

4.2.2.6.1 Rigid Body Type Parameter Storage

Using the TCEntityData template, the rigid body type, i.e. sphere, data is generically stored and the enumerations to assist in data extraction based on column indices are defined as shown in the code excerpt in Figure 4.26.

```
enum ERigidBodyTypeBool
{
   erbtbRigidBodyTypeChanged = 1,
   // Allways Add and Remove Enumerations Above This Comment
   erbtbMax
};

enum ERigidBodyTypeInt
{
   erbtiRigidBodyTypeMaterial = 1,
   erbtiRigidBodyTypeGeometricPrimitive,
   // Allways Add and Remove Enumerations Above This Comment
   erbtiMax
};

enum ERigidBodyType
{
   erbtRigidBodyTypeRadius = 1,
   erbtRigidBodyTypeVolume,
   erbtRigidBodyTypeInertia,
   erbtRigidBodyTypeMass,
   erbtRigidBodyTypeWeightX,
   erbtRigidBodyTypeWeightY,
   erbtRigidBodyTypeWeightZ,
   // Allways Add and Remove Enumerations Above This Comment
   erbtMax
};

enum ERigidBodyTypeNeigbbourPairList
{
   // Allways Add and Remove Enumerations Above This Comment
   erbtnpMax = 1
};

enum ERigidBodyTypeNeigbbourList
{
   // Allways Add and Remove Enumerations Above This Comment
   erbtnMax = 1
};
.
.
.
m_RigidBodyType.Set(m_iMaxNumRigidBodyTypes,
                    erbtbMax-1,
                    erbtiMax-1,
                    erbtMax-1,
                    erbtnpMax-1,
                    erbtnMax-1);
```

Figure 4.26: Rigid body type data member and helper enumerator definitions.

4.2.2.6.2 Rigid Body Parameter Storage

Using the TCEntityData template, the rigid body data is generically stored and the enumerations to assist in data extraction based on column indices are defined as shown in the code excerpt in Figure 4.27, Figure 4.28 and Figure 4.29. A large amount of Real type data needs to be stored for a rigid body, and therefore the code excerpt had to be shown in three segments. The first segment, Figure 4.27, contains the Boolean and Integer data related enumerations.

```
enum ERigidBodyBool
{
    erbbRigidBodyChanged = 1,
    // Allways Add and Remove Enumerations Above This Comment
    erbbMax
};

enum ERigidBodyInt
{
    erbiRigidBodyType = 1,
    // Allways Add and Remove Enumerations Above This Comment
    erbiMax
};
```

Figure 4.27: Rigid body/sphere bool & int data member and helper enumerator definitions.

The following segment, Figure 4.28, shows the real data related enumerations for rigid bodies and some of the data is only used in specific solution routines but might be redundant in others.

```
enum ERigidBody
{
    erbRigidBodyPositionX = 1,
    erbRigidBodyPositionY,
    erbRigidBodyPositionZ,
    erbRigidBodyVelocityX,
    erbRigidBodyVelocityY,
    erbRigidBodyVelocityZ,
    erbRigidBodyThetaX,
    erbRigidBodyThetaY,
    erbRigidBodyThetaZ,
    erbRigidBodyOmegaX,
    erbRigidBodyOmegaY,
    erbRigidBodyOmegaZ,
    erbRigidBodyMinX,
    erbRigidBodyMinY,
    erbRigidBodyMinZ,
    erbRigidBodyMaxX,
    erbRigidBodyMaxY,
    erbRigidBodyMaxZ,
    erbOldRigidBodyPositionX,
    erbOldRigidBodyPositionY,
    erbOldRigidBodyPositionZ,
    erbOldRigidBodyVelocityX,
    erbOldRigidBodyVelocityY,
    erbOldRigidBodyVelocityZ,
    erbOldRigidBodyThetaX,
    erbOldRigidBodyThetaY,
    erbOldRigidBodyThetaZ,
    erbOldRigidBodyOmegaX,
    erbOldRigidBodyOmegaY,
    erbOldRigidBodyOmegaZ,
    // The Following Variables will be Updated During Contact Resolution
    erbRigidBodyForceX,
    erbRigidBodyForceY,
    erbRigidBodyForceZ,
    erbRigidBodyLinearAccelerationX,
    erbRigidBodyLinearAccelerationY,
    erbRigidBodyLinearAccelerationZ,
    erbRigidBodyTorqueX,
    erbRigidBodyTorqueY,
    erbRigidBodyTorqueZ,
    erbRigidBodyAngularAccelerationX,
    erbRigidBodyAngularAccelerationY,
    erbRigidBodyAngularAccelerationZ,
    // The Following Variables are Used During Newton-Raphson Iterations
    erbRigidBodyIterationVelocityX,
    erbRigidBodyIterationVelocityY,
    erbRigidBodyIterationVelocityZ,
    erbRigidBodyIterationOmegaX,
    erbRigidBodyIterationOmegaY,
    erbRigidBodyIterationOmegaZ,
    // Allways Add and Remove Enumerations Above This Comment
    erbMax
};
```

Figure 4.28: Rigid body/sphere real data member and helper enumerator definitions.

The last segment, Figure 4.29, depicts the neighbour list related enumerations, as well as the initialisation call for the rigid body data structure. Some of the neighbour lists might not yet be in full use, but they might be handy whenever algorithm refinements are done.

```
enum ERigidBodyNeigbbourPairList
{
   // Allways Add and Remove Enumerations Above This Comment
   erbnpMax = 1
};

enum ERigidBodyNeigbbourList
{
   erbnRigidBodyVertexContactNeighbours = 1,
   erbnRigidBodyEdgeContactNeighbours,
   erbnRigidBodyFacetContactNeighbours,
   erbnRigidBodyCellContactNeighbours,
   erbnRigidBodyRigidBodyContactNeighbours,
   // Allways Add and Remove Enumerations Above This Comment
   erbnMax
};
.
.
.
m_RigidBody.Set(m_iMaxNumRigidBodies,
                erbbMax-1,
                erbiMax-1,
                erbMax-1,
                erbnpMax-1,
                erbnMax-1);
```

Figure 4.29: Rigid body/sphere neighbour data member and helper enumerator definitions.

4.2.2.6.3 Contact Parameter Storage

Using the TCEntityData template, the contact data is generically stored and the enumerations to assist in data extraction based on column indices are defined as shown in the code excerpt in Figure 4.30 and Figure 4.31, shown in segments again for lack of space.

```
enum EContactDataBool
{
   ecdbContactBinary = 1,
   ecdbContactActive,
   ecdbContactSlipping,
   ecdbContactNoTangential,
   // Allways Add and Remove Enumerations Above This Comment
   ecdbMax
};

enum EContactType
{
   ectDynamic = 1,
   ectStatic,
   // Allways Add and Remove Enumerations Above This Comment
   ectMax
};

enum EContactDataInt
{
   ecdiContactType = 1, // Either Dynamic or Static
   ecdiContactObjectNumber1,
   ecdiContactObjectType1,
   ecdiContactObjectNumber2,
   ecdiContactObjectType2,
   // Allways Add and Remove Enumerations Above This Comment
   ecdiMax
};
```

Figure 4.30: Contact bool, type & int data member and helper enumerator definitions.

```
enum EContactData
{
   ecdContactNormalX = 1,
   ecdContactNormalY,
   ecdContactNormalZ,
   ecdContactNormalOppositeX,
   ecdContactNormalOppositeY,
   ecdContactNormalOppositeZ,
   ecdContactFirstTangentialX,
   ecdContactFirstTangentialY,
   ecdContactFirstTangentialZ,
   ecdContactFirstTangentialOppositeX,
   ecdContactFirstTangentialOppositeY,
   ecdContactFirstTangentialOppositeZ,
   ecdContactSecondTangentialX,
   ecdContactSecondTangentialY,
   ecdContactSecondTangentialZ,
   ecdContactSecondTangentialOppositeX,
   ecdContactSecondTangentialOppositeY,
   ecdContactSecondTangentialOppositeZ,
   ecdContactPositionX,
   ecdContactPositionY,
   ecdContactPositionZ,
   ecdContactRelativeVelocityX,
   ecdContactRelativeVelocityY,
   ecdContactRelativeVelocityZ,
   ecdContactNormalRelativeVelocity,
   ecdContactFirstTangentialRelativeVelocity,
   ecdContactSecondTangentialRelativeVelocity,
   ecdContactNormalRestitutionFactor,
   ecdContactTangentialRestitutionFactor,
   ecdContactDynamicFrictionFactor,
   ecdContactStaticFrictionFactor,
   // The Following Variables are Used During Newton-Raphson Iterations
   ecdContactIterationNormalImpulse,
   ecdContactIterationFirstTangentialImpulse,
   ecdContactIterationSecondTangentialImpulse,
   ecdContactIterationTangentialImpulse,
   ecdContactIterationTangentialX,
   ecdContactIterationTangentialY,
   ecdContactIterationTangentialZ,
   // Allways Add and Remove Enumerations Above This Comment
   ecdMax
};
.
.
.
m_Contact.Set(m_iMaxNumContacts, ecdbMax-1, ecdiMax-1, ecdMax-1, 0, 0);
```

Figure 4.31: Contact real type data member and helper enumerator definitions.

As is the case with the rigid body data, some variables are used only in specific solution routines, whilst being redundant in others.

4.2.3 Algorithms

Once the data structures had been defined and implemented as shown in section 4.2.2, the task of implementing the various algorithms to be used with and operating on the data has to be performed. Firstly one needs to decide how and where the various methods and functions should (and thus will) best be grouped, and then these algorithms and operators need to be physically coded using the theory documented earlier. This section strives to give a simple overview of and motivation for the collaboration and delegation that had been chosen.

4.2.3.1 Geometrical Calculations

The geometrical calculation routines are of pivotal importance, particularly for the collision time calculator and contact detection routines. Since encapsulated methods are always more easily included as decorators without the hassle of having to declare them as friends wherever they are to be used, it was decided to group all the geometrical calculation routines as static methods in the TCGeometryMathTools decorator template, along with some static "scratch pad" variables and arrays to be used within and by calculation routines. At the moment this decision might possibly preclude the use of multithreading or maybe even parallel processing, but that is fine for the time being, since the code execution performance on a single processor machine, as a single threaded application is acceptable. The prototypes of the various static methods available in TCGeometryMathTools are shown in Figure 4.32 below.

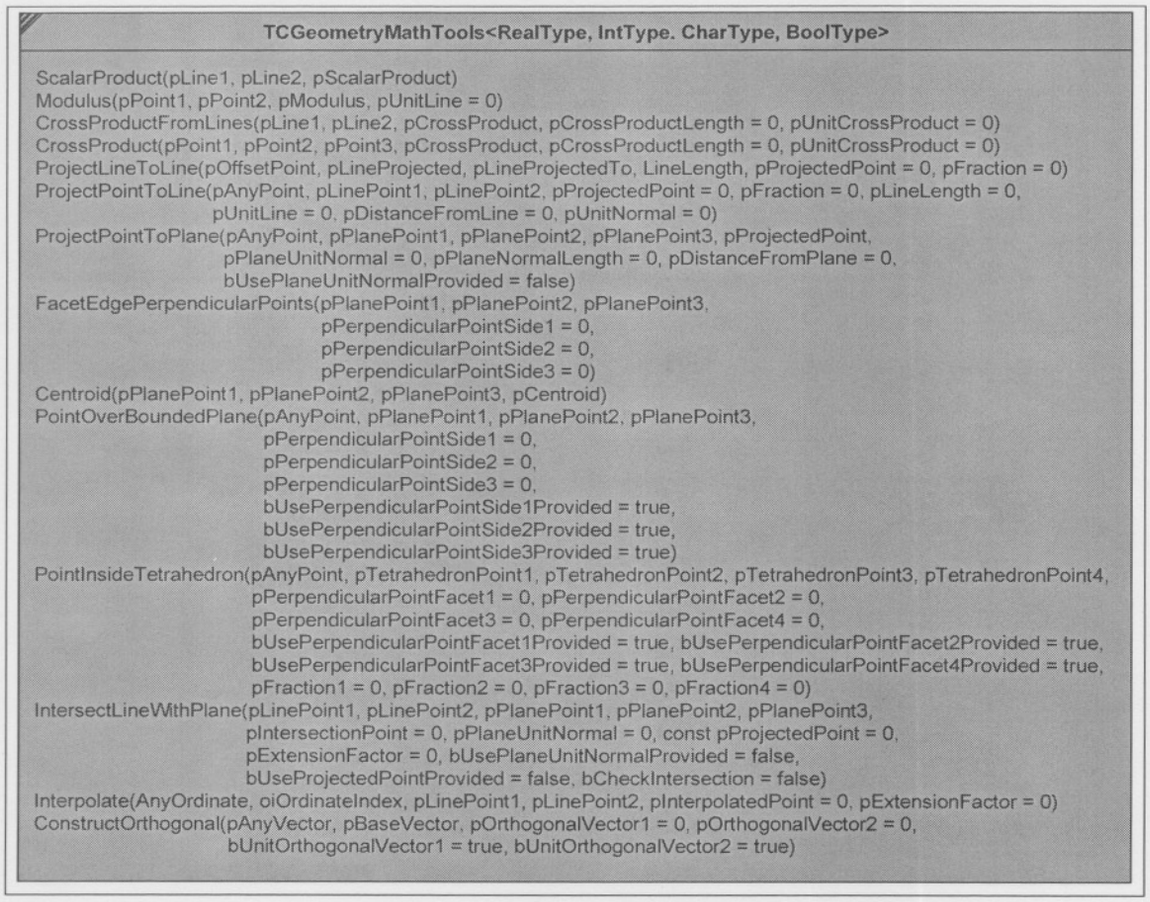


Figure 4.32: TCGeometryTools methods.

The methods shown in Figure 4.32 form the basis for all geometry-related calculations done in this study and the TCGeometryMathTools static template can be included as a decorator for any class to avail the static methods to them. All methods return a Boolean value indicating success (true) or failure (false) and contain C++ code renditions of some of the mathematical expressions

or calculations documented or referred to in sections 3.2 and 4.1.2. The pointer arguments (with prefix *p*...) all refer to three-element zero-based arrays (triads) representing three-dimensional Cartesian coordinates, i.e. a point in a 3-D domain, and it illustrates the versatility of opting for the storage scheme described in 4.2.2.4. Since the passing of any triad is done by pointer, one simply passes the address of the first element in a triad if it is stored in contiguous memory, e.g. the various triads present in the real type data storage for the Rigid Body and Contact objects – see Figure 4.28 and Figure 4.31 for practical examples.

4.2.3.2 Sorting Routines

The sorting routines are relevant to and impact on the data stored within the TCSortingGrid template and therefore they had been grouped as members thereof. The methods listed in Figure 4.33 below indirectly utilise the object locating algorithms implemented as described in section 4.1.2.2.1.

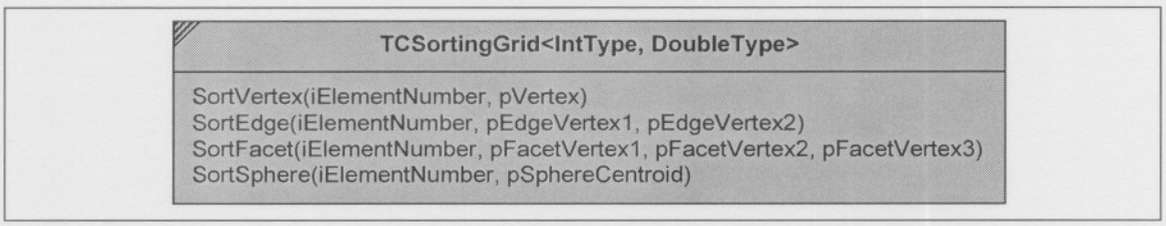


Figure 4.33: TCSortingGrid methods.

The prefix *p*... for the arguments in the method prototypes once more refers to triads being passed in pointer format. Since the TSRBSystem template is derived from the TCGeometricGrid template, which is in turn derived from the TCSortingGrid template, they have all the sorting utilities shown in Figure 4.33 at their disposal and they are indeed called within the TCRBSystem prior to any contact searching in order to narrow down the number of objects to be checked for interaction after each integration time step.

4.2.3.3 Grid Setup

The data for all the geometric entities and materials necessary for grid representation were grouped in the TCGeometricGrid object, but along with the data there is the need for some methods for the controlled adding, removing and mutation of entities as well as data storage resizing, and these are listed in Figure 4.34. As mentioned earlier, the data for vertices, edges and facets are hierarchically related and changes in one hierarchical level triggers updates of all involved entities in higher hierarchical level, as well as their re-sorting into bins. For efficiency's sake, updates should only be triggered at critical times, such as right before performing collision detection, but at present the immediate update route was chosen for the sake

of simplicity and also because the hierarchical updates are not done that often, since moving grids had not yet been implemented for this study.

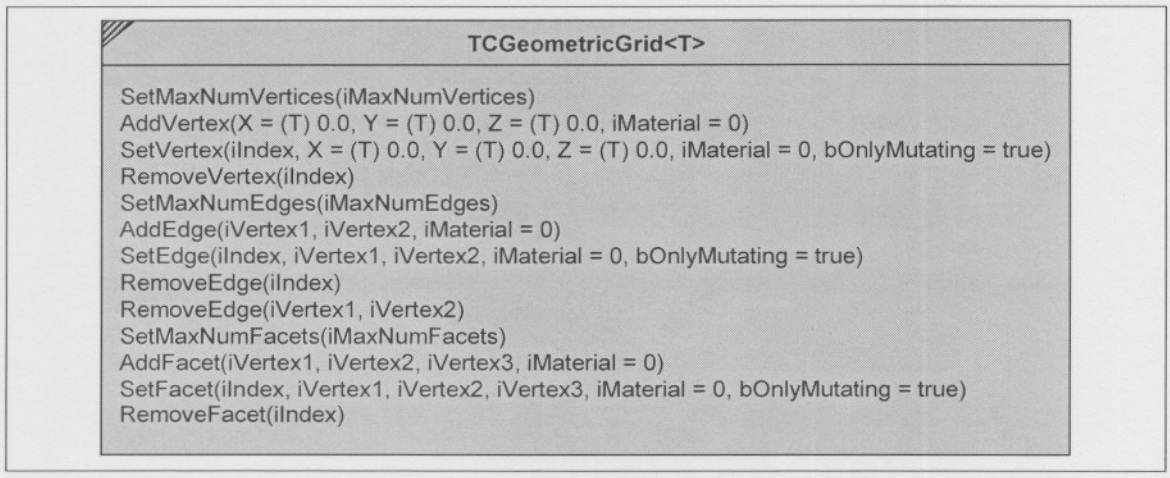


Figure 4.34: TCGeometricGrid methods.

4.2.3.4 Rigid Body Motion Routines

The rigid body simulation related data had been grouped in the TCRBSystem template, and all relevant methods operating on and using that data also needs to be included there. The methods available to the user are shown in Figure 4.35 and they exclude the core algorithms implemented for rigid body motion integration and collision resolution, which had been made protected members of the template, to avoid their inappropriate usage, since they do affect the contents of the template data members.

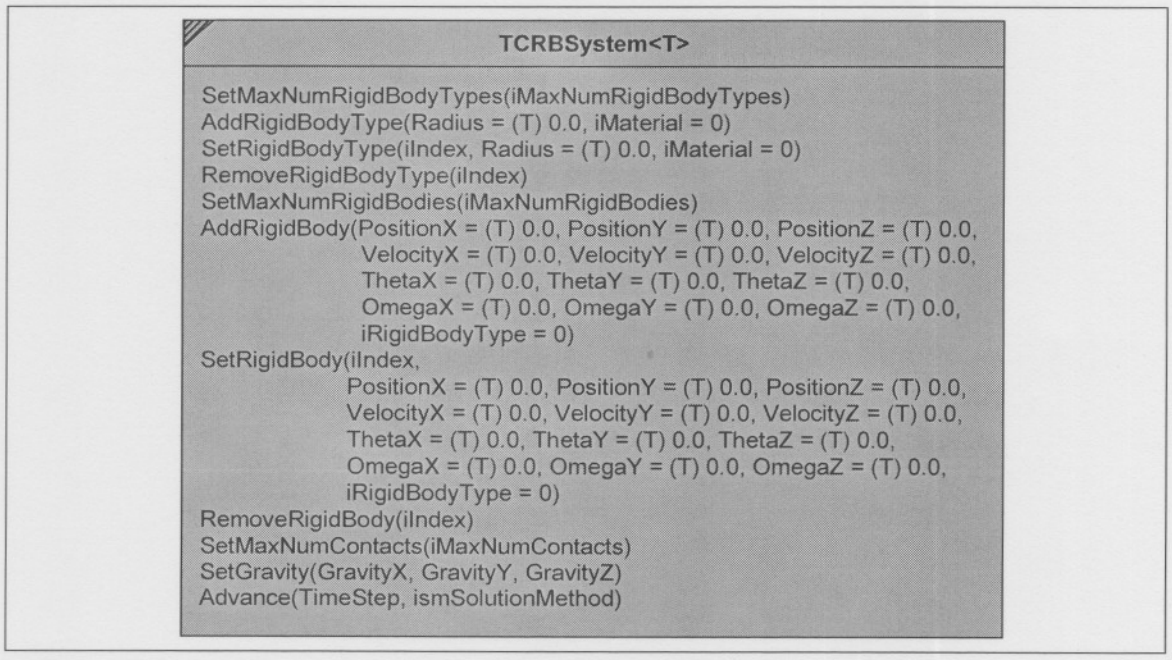


Figure 4.35: TCRBSystem methods.

The differential equations for linear and angular rigid body motion are integrated using the Runge-Kutta technique outlined in section 4.1.1.2.1, implemented in a generic protected method named IntegrateRungeKutta() and called implicitly in the course of executing the Advance(TimeStep, ismSolutionMethod) method.

4.2.3.5 Contact Searching and Critical Time Calculation Routines

The contact searching and time calculation methods are also contained within the TCRBSystem template and defined as protected to prevent undue usage. The contact searching algorithm has a dual function, since it was developed to either look for contacts or determine estimated time of impact for critical time calculation, functioning in conjunction with the motion integrator and rigid body bin sorting routines. As soon as the critical time to a collision had been established, the motion integration can be arrested temporarily and contact search can commence, after which a contact resolution has to be done to determine impulsive changes in velocities for all rigid bodies in concurrent contact.

4.2.3.6 Sparse Vector Setup

In order to simplify the setup of sparse vectors, a TCVectorEntryList template had been developed. This template object stores all the non-zero entries and their corresponding row indices and it can be used to initialise a mathematical TCVector or Container::TCVector template.

4.2.3.7 Sparse Matrix Setup

Sparse matrix setup can be very tricky and tedious and thus it was decided to develop a sparse matrix entry helper template object named TCMatrixEntryData, which can be used to initialise a TCSparsM template object. No knowledge about the actual storage scheme is required by the user of the helper template, since it reduces matrix setup to merely specifying the row and column indices and coefficient value for each non-zero matrix entry. At present only the compressed row storage scheme is implemented and employed for the actual sparse matrix data, and the supporting helper object is customised specially to ease the setup process for such a storage approach.

4.2.3.8 Matrix Solver Routines

The TCSparsM matrix storage template was developed to store sparse matrix related data and also be usable in sparse matrix solver routines or related operations, nearly all of which are implemented as free standing friend functions, not members or a particular template object but

able to operate on or utilise data within them. The solver routines implemented to date include the Bi-Conjugate Gradient Method (Bi-CGM), Bi-Conjugate Gradient Stabilised (Bi-CGSTAB) and Conjugate Gradient Squared (CGS) as they are described in Barret *et al.* (1994:21-23, 25-28). Some other sparse matrix solution techniques might be implemented if the need arises during testing, but for now the abovementioned will suffice.

4.2.3.9 Contact Resolution Routines

The contact resolution algorithm is the actual aim and intended outcome of this study and is therefore the most crucial implementation for the whole project. It also serves as an evaluation and verification tool for the theory used or synthesised in the course of this study. It was decided that the TCRBSystem template would serve the purpose of being the host for the contact resolution methods to be evaluated. The protected method ResolveContacts() is indirectly responsible for and contains the implementation of the setup and solution of the non-linear sets of equations as described in section 4.1.3. The ResolveContacts() method thus determines the post-collision state by iteratively solving the appropriate non-linear analytical expressions and these results can be used as the re-starting point for further motion integration within the Advance() method.

4.3 Conclusion of Algorithms and Implementation

In this chapter, the theory directly relevant to and used for implementation in actual C++ computer code had been discussed, following which, the actual implementation was discussed with respect to architecture and general philosophy. The following chapter summarises all the test cases run in order to verify or validate the theory implemented for this study, and is iteratively written in conjunction with this implementation chapter wherever it may have become necessary to do so.

Chapter 5
Test Cases and Results

In order to determine and ascertain whether the theory derived for and implemented in this study works and is of value, various test cases need to be run and passed, at least compared to alternative implementations where available. Test cases done for this study are described and discussed in the various sub-sections of this chapter and some suggestions for further work can be found in the consequent chapter (Chapter 6).

5.1 Test Cases

The test cases used to demonstrate that the theory actually works had been arbitrarily selected from various possible candidates and due to this fact, only very small portions of the solution domain could be evaluated, but test cases were chosen to demonstrate the developed algorithm's capability to handle the various difficulties it might be likely to encounter. In work done by Ivanov (1995:888-898), analytical expressions were derived for a few three-body clusters in simultaneous collision (collinear and planar divergent configuration), and these are the only available useful benchmarks to compare with the numerical results of the implementation for the current study. Unfortunately, the expressions developed are for the frictionless case, so they are not of great help in quantitative benchmarking. The test cases are all solved using two alternative algorithms for determining the post-collision velocities and contact impulses for the concurrently colliding systems of spheres, one is a linear model, assuming that the tangential vector, and thus the direction of sliding whenever that occurs, in the equations in Figure 4.9 stays fixed during collisions, whilst the other allows for the tangential to re-orient itself and thus it is a non-linear set of equations, as represented by equations (4.10) to (4.18). The latter, non-linear solution technique is the actual algorithm developed in the course of this study and the results obtained can be compared with those obtained by running the former linear model, or some other solution technique, such as the DEM employed by the PFC3D (Anonymous, 2002) application in use at the PBMR, results of which were kindly generated and provided by Polson (2004). Since only the velocity changes due to impact is of importance for this study, the integration time step is set to 0.0 seconds for most test cases, and thus only immediate contact is considered, requiring the sphere-system set-ups to be geometrically precise for all test cases. Material properties were set close to those of mild steel, i.e. 7850 $kg.m^{-3}$ density, 0.9 normal and tangential restitution

coefficients and static and dynamic friction factors of 0.15, and the spheres are usually all of equal size (50 mm diameter), except where otherwise specified. For the DEM verification, the parameters were set to 1.0e9 N/m for the normal stiffness, 2.86e8 N.s/m for the tangential stiffness and 0.122325548609685 N.s/m for the global damping coefficient. In addition to the verification of the collision responses, secondary aspects such as sphere-location and correct collision detection and geometrical parameter calculations are also implicitly verified, since any wrong number and position of or surface-normal orientation for contacts would be evident in the result files, and thus also easily detected by inspection.

5.1.1 Two-Body Collisions

Two-body (binary) collision test problems are the most basic and should all be passed without any error, since they verify whether the simplest physics laws are implemented correctly and yield physically correct results, verifiable by hand calculations if the need arises. This section lists all tests and their results, as well as some discussions on the results.

5.1.1.1 Setup and Execution

Six binary collision tests were performed using a general setup as shown in Figure 5.1 and varying the indicated angle, θ, between 0° and 45° in four discrete steps (45°, 30°, 15°, 0°) in the XOY plane and then also another two tests with θ equal to 45° in each case, but on the YOZ and ZOX planes respectively.

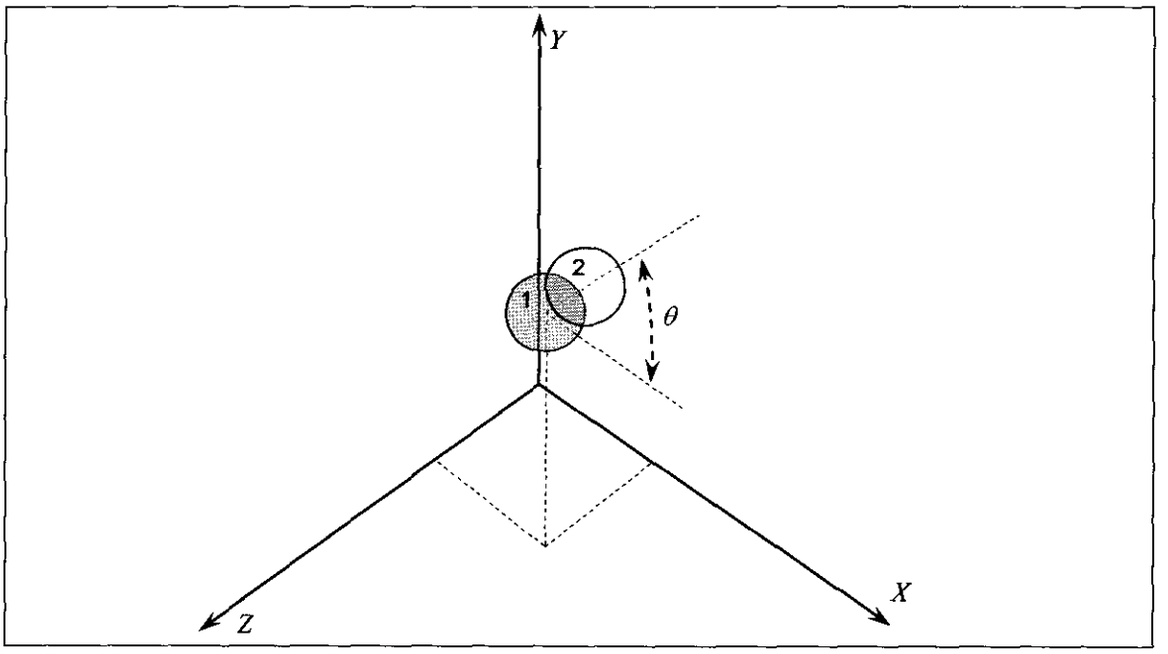


Figure 5.1: General Binary Collision Setup.

Sphere 1 always has a velocity vector of (1, 1, 1) m/s in the Cartesian coordinate system, whilst sphere 2 always has a velocity vector of (-1, -1, -1) m/s in the Cartesian coordinate system. The available linear and non-linear frictional solution methods were then used to solve for the post-collision linear and angular velocities, normal impulse and tangential impulse magnitudes. The DEM results were obtained from PBMR (Polson, 2004).

5.1.1.2 Benchmarks

Though an analytical benchmark is theoretically easy to obtain for a simple binary collision, it does involve the use of some three-dimensional mathematics, mostly geometry, thus making it difficult to write out generally, even using matrix-vector representation. Fortunately, for the binary collision case the linear frictional solution method is an exact implementation of the analytical method that would be followed to solve for the post-collision linear and angular velocities of both spheres and the normal and tangential impulse magnitudes at the contact. Comparing results for the newly developed non-linear solution algorithm with those for the linear model provides a good measure of the reliability of the former. Test cases 5 and 6 should yield exactly the same results as test case 1, only with the Cartesian vector components cyclically rotated forward on Cartesian axes once for test case 5, i.e. X becomes Y, Y becomes Z and Z becomes X, and twice for test case 6, i.e. X becomes Z, Y becomes X and Z becomes Y. In addition to the aforementioned benchmarks, the normal and tangential impulses obtained for binary test cases 1 to 4 can be verified using formulae in Hoomans (1999). After substituting the symbols used in Hoomans (1999:35-37) with the ones used in this study, the normal impulse magnitude is given by the expression in equation (5.1) whilst the tangential impulse magnitude can be calculated using the friction-factor dependant expressions in equation (5.2).

$$\lambda_{n,ij}^{nb} = -(1+\varepsilon_{n,ij}^{nb})(\underline{\dot{x}}_{ij}^{nb,-} \bullet \underline{n}_{ij}^{nb}) \Big/ (1/m_i + 1/m_j) \tag{5.1}$$

$$\lambda_{t,ij}^{nb} = \begin{cases} -2(1+\varepsilon_{t,ij}^{nb}) \left| \underline{\dot{x}}_{ij}^{nb,-} \bullet \underline{t}_{ij}^{nb} \right| \Big/ [7(1/m_i + 1/m_j)] \cup \lambda_{t,ij}^{nb} \le \mu_{stat}^{nb} \lambda_{n,ij}^{nb} \\ \mu_{dyn}^{nb} \lambda_{n,ij}^{nb} \cup \lambda_{t,ij}^{nb} > \mu_{stat}^{nb} \lambda_{n,ij}^{nb} \end{cases} \tag{5.2}$$

Substituting the appropriate values for binary collision test case 4, which was chosen as an example for the convenience of determining the normal and tangential unit vectors easily, into the expressions, the analytical intermediate steps as shown in equations (5.3) and (5.4), are obtained.

$$\begin{aligned} \lambda_n^1 &= -(1+0.9) \times (-2.0)/(1/0.513781 + 1/0.513781) \\ &= 0.976184467256141 \end{aligned} \tag{5.3}$$

$$\lambda_t^1 = \begin{cases} -2(1+0.9)\times(-2.828427125)/[7(1/0.513781+1/0.513781)] \\ = 0.394438089427958 \\ \text{or} \\ 0.15\times 0.976184467256141 \\ = 0.146427670088421 \cup \lambda_t^1 > 0.15\times 0.976184467256141 \end{cases} \quad (5.4)$$

It can be determined by inspection of the numerical values that tangential slip should occur and thus values of approximately 0.976184 kg.m/s and 0.146428 kg.m/s for the normal and tangential impulses respectively emerge as the analytical benchmark results. The impulse values for test cases 1 to 3 can also be determined using the same formulae and in the following section, in Table 5.1, a comparative listing of the results thus obtained can be found. Finally, there should never be an increase in kinetic energy (while a decrease might occur due to slippage and incomplete restitution), and momentum should remain constant during any cluster collision.

5.1.1.3 Results

The results generated from the binary collision tests can all be seen in Appendix B, section B.1. The impulse results for the first four test cases are also listed below in Table 5.1, with the analytical results obtained from the analytical expressions in equations (5.1) and (5.2) and the linear and non-linear results from the result files as shown in Appendix B, sections B.1.1 to B.1.4.

Table 5.1: Various binary collision test impulse results

Case	Impulse	Analytical	Linear	Non-Linear
Binary 1	**Normal**	1.3805333130	1.3805333130	1.3805333130
	Tangential	0.2070799969	0.2070799969	0.2070799969
Binary 2	**Normal**	1.3334927811	1.3334927811	1.3334927811
	Tangential	0.2000239172	0.2000239172	0.2000239172
Binary 3	**Normal**	1.1955769198	1.1955769198	1.1955769198
	Tangential	0.1793365380	0.1793365380	0.1793365380
Binary 4	**Normal**	0.9761844673	0.9761844673	0.9761844673
	Tangential	0.1464276701	0.1464276701	0.1464276701

5.1.1.4 Discussion

The data obtained from the binary collision test runs indicate that the linear and non-linear collision resolution methods yielded exactly the same results for all unknown variables in a binary collision – barring the occasional difference in sign for values that should be exactly zero.

Both linear and non-linear impulsive approaches conserved the system total linear and angular momentum and thus these values are also matching for the two methods. Test cases 1, 5 and 6 did, indeed, yield the same results with only the Cartesian coordinates rotated as indicated earlier, indicating that the equations were all similarly and correctly implemented and solved for all three Cartesian coordinate directions. In addition to matching values for the two methods employed – see Appendix B, section B.1, the analytical benchmark values obtained for test cases 1 to 4 – see section 5.1.1.3 above – can also be observed to match up to their tenth decimal digits with the numerical results obtained. This can be considered as sufficiently accurate to qualify both the linear and non-linear approaches as equivalent to the analytical method for the calculation of impulse magnitudes in the binary collision case. All the foregoing results, in turn, imply that, at least for binary collisions, both the linear and non-linear resolution algorithms are reliable and accurate enough for engineering purposes.

For the sake of being thorough, soft sphere (i.e. DEM) results were also obtained from PBMR (Polson, 2004), where software named PFC3D (Anonymous, 2002) is used. Results showed that the intuitively expected translational and rotational directions were correct, though the linear and angular velocities differ by around 10 % and 5 % respectively from those obtained using the analytical expressions, but this can be explained by virtue of the remarks made by Hoomans (1999:55-58), especially where the rotational velocity is concerned (Hoomans, 1999:58), since the normal and tangential damping coefficients were set to the same value. According to Hoomans (1999:54–56), the normal damping should have been set to 0.017230834 N.s/m for the selected normal spring stiffness of 1.0e9 N/m if a model similar to that of Hoomans (1999:48-56) had been used for the contact force calculations, with the corresponding tangential spring stiffness being 2.86e8 N/m and the tangential damping then being 0.004923096 N.s/m. Having no option to set the tangential damping to a value different from the normal damping in PFC3D appears to be the main reason for the occasional large discrepancies in the post-collisional rotation velocities.

5.1.2 Three-Body Concurrent Collisions

Three-body (ternary) collision test problems are still quite basic, though they can have at least two different main permutations – i.e. tests similar to the set-ups in the binary collision test, with a linear stacking of spheres, or planar collisions with one projectile sphere hitting two others – all of which should be passed without any error. Of particular interest and importance to the present study, is the situation where relative tangential velocities are not all coplanar at the commencement of the collision, since the direction of relative velocity is most likely to change

during collision. The first tests for correct handling of contact breaking and tangential slippage appear here in the ternary planar setup, and these phenomena need to be correctly detected and taken into account by the algorithm employed. This section lists all ternary tests and their results, as well as some discussions on the results.

5.1.2.1 Setup and Execution

The linear stacking ternary collision tests were performed using a general setup as shown in Figure 5.2 and varying the indicated angle, θ, between 45° and 0° in discrete steps (i.e. 45°, 30°, 15°, 0°) in the XOY plane for the first four tests. Sphere 1 had a velocity vector of (1, 1, 1) m/s in the Cartesian coordinate system, whilst sphere 3 had a velocity vector of (-1, -1, -1) m/s in the Cartesian coordinate system. The fifth and sixth tests are a repeat of test case 1 but the coordinates are respectively cyclically rotated forward once and twice (i.e. X->Y, Y->Z and Z->X for case 5 and X->Z, Y-X and Z->Y for case 6). The the initial velocities remain the same as for the first four test cases.

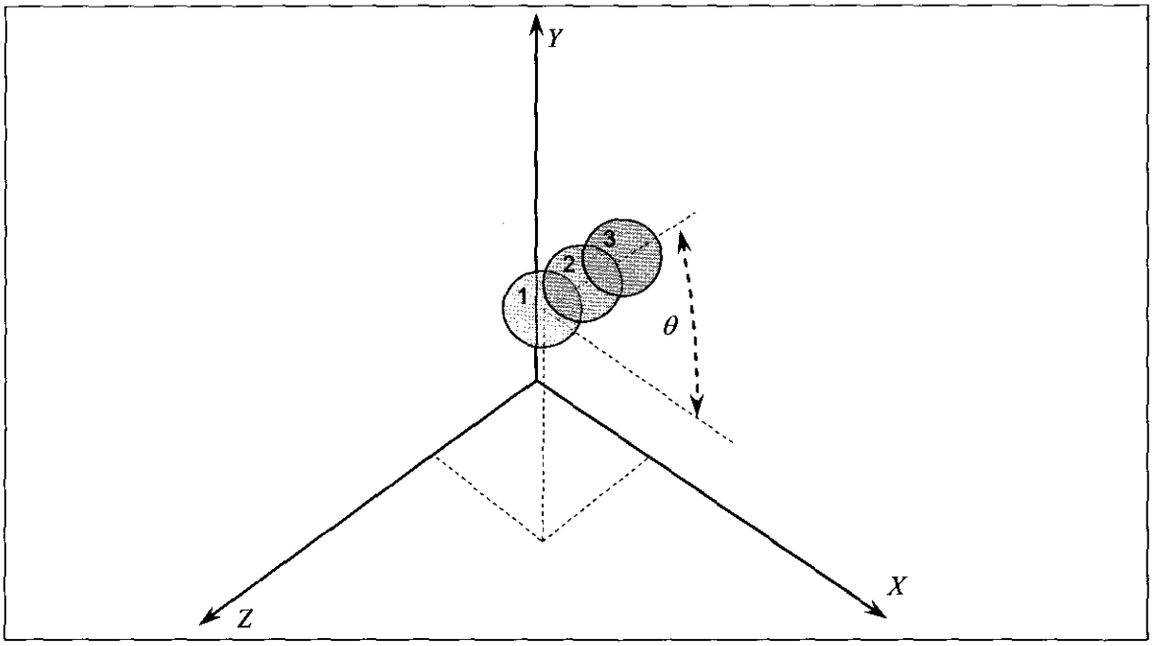


Figure 5.2: General Linearly Stacked Ternary Collision Setup.

To examine and demonstrate the effect of non-coplanar pre-collision tangential relative velocities on the eventual tangential impulse directions, test case 7 is used, with the setup identical to that for test case 4, but the velocities of spheres 1 and 3 are set to (1, 1, 1) m/s and (-1, 1, -1) m/s respectively, which resulted in the pre-collision tangential relative velocities at the two contacts to be pointing at right angles relative to one another. Test case 8 is used to verify whether momentum balance is correctly applied, with test case 4 once again used as basis, but

the initial tangential velocities at the contacts on either side of sphere 2 are pointing in the same direction, i.e. spheres 1 and 3 have velocity vectors of (1, 1, 1) m/s and (-1, 1, 1) m/s respectively. The planar ternary collision tests were performed using a general setup as shown in Figure 5.3, which is essentially a symmetrical collision test, with the angle from the symmetry line, θ, set to 30° to guarantee a three-contact system. The projectile (Sphere 1) is given one of two velocities, i.e. (1, 0, 0) m/s for test case 9 and (1, 0, 10) m/s for test case 10, whilst the other two target spheres are stationary prior to the collision.

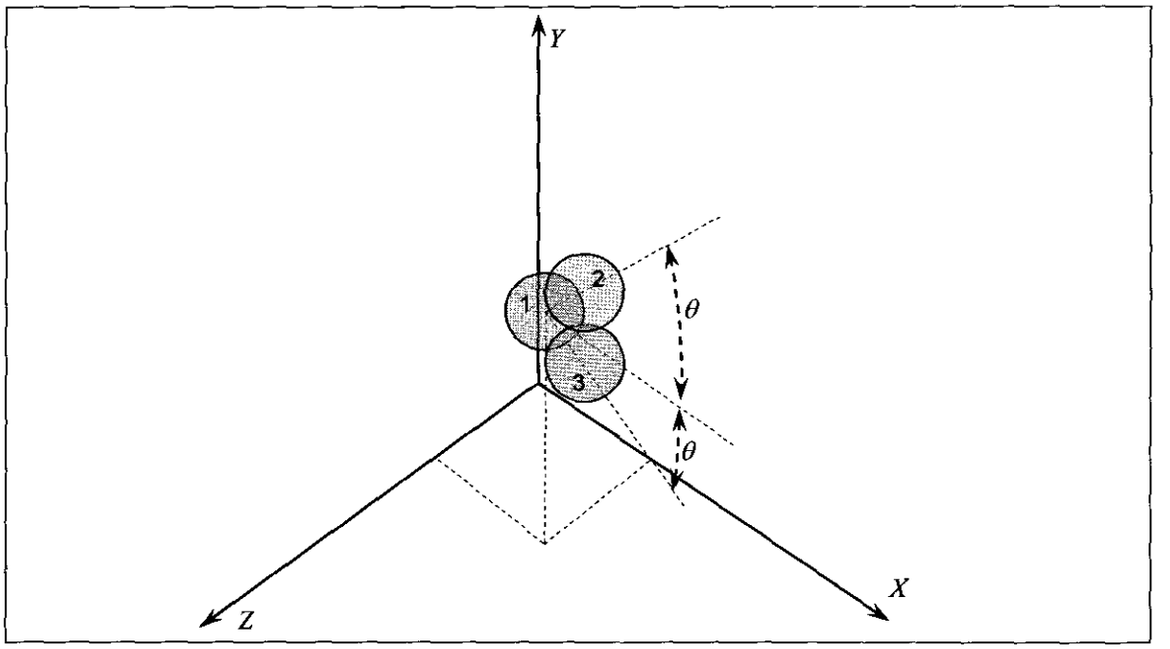


Figure 5.3: General Planar Ternary Collision Setup.

The available linear and non-linear frictional solution methods were then used to solve for the post-collision linear and angular velocities for each sphere and normal and tangential impulse magnitudes and tangential unit vector coordinates for each contact for each test case. As with the binary cases, the DEM alternative solution technique was also used to obtain results (Polson, 2004) for verification purposes.

5.1.2.2 Benchmarks

The ternary benchmark tests should all yield results either symmetrical or skew-symmetric round the central sphere (sphere 2 in Figure 5.2 and Sphere 1 in Figure 5.3). A brief summary of qualitative and some quantitative phenomena to be observed are now given.

For the linear stacking tests – as generally described by Figure 5.2, the results should exhibit the following phenomena:

- Translations
    - Should be of equal magnitude but opposite axial directions for spheres 1 and 3 in tests 1 through 7, while there should be no translation for sphere 2.
    - Should be exactly equal in case 8 for spheres 1 and 3, and sphere 2 should also translate in the same direction.
- Rotations
    - Should be in the same direction for all spheres in tests 1 through 7, with spheres 1 and 3 having exactly the same angular velocities and the central sphere (2) rotating at double the velocity of the two outer spheres.
    - Should be of equal magnitude but opposite direction in test case 8 for spheres 1 and 3 and zero for the central sphere (2).
- Contacts
    - Should have only two active contacts, between sphere 1 and 2 and also 2 and 3.
    - Normal impulses should be equal in magnitude and always compressive.
    - Tangential impulses should be parallel in direction and equal in magnitude for all linear stacking tests.

For the planar tests – as generally described by Figure 5.3, the results should exhibit the following phenomena:

- Translations
    - Should be of equal magnitude but with mirrored directions round the symmetry line parallel to the X-axis for spheres 2 and 3 in tests 9 and 10
    - Should be opposite in X-direction for sphere 1 compared to that of the other two spheres.
    - Should be in the Z direction for all spheres in case 10, sphere 1 having the highest and the other two having equal velocity components.
- Rotations
    - Should be of zero magnitude for sphere 1 round all axes in test case 9.
    - Should be equal but of opposite direction round the Z-axis for spheres 2 and 3 in tests 9 and 10.
    - Should be in the same direction round the Y-axis for all spheres in test case 10, sphere 1 having twice the Y-rotational velocity component of the other two.
    - Should be of equal magnitude but opposite direction round the X-axis for spheres 2 and 3 and zero for sphere 1 round the X-axes in case 10.

- Contacts
    - Should initially number three in total.
    - One contact, between spheres 2 and 3, should be inactive.
    - Should have only two active contacts, between sphere 1 and 2 and 1 and 3, the contact between sphere 2 and 3 should be inactive.
    - Normal impulses should be equal in magnitude and always compressive.
    - Tangential impulses should be equal in magnitude for all planar tests.
    - Tangential impulse directions should be symmetrical round the axis of symmetry, parallel to the X-axis for all planar tests.

In addition to all the aforementioned results, there should be no energy gains (though there might and should be energy losses, due to slippage and incomplete restitution) and momentum should always be conserved. In coplanar cases, the linear approach should yield the same results as the non-linear one, and in non-coplanar cases, they should not differ greatly.

5.1.2.3 Results

The various results for the ternary collision test cases can be seen in Appendix B, section B.2, listing both linear and non-linear approach results, as well as the independent DEM results (Polson, 2004).

5.1.2.4 Discussion

Most of the ternary test runs indicated that the linear and non-linear collision resolution methods yield very close to the same results where the tangential impulses are coplanar – barring the occasional difference in sign for exact zero values. The non-coplanar tangential results obtained, i.e. test cases 7 and 10, did differ mostly in their second decimal digits, indicating that the methods yield results that are at least in the same neighbourhood, including the energy balances. The fact that the values are within the same vicinity indicate that the correct roots were found by the semi-Newton-Raphson iteration for the non-linear equations and that the tangential unit vectors were well estimated, since the linear solution method is totally dependent thereupon. Upon closer inspection of the numerical results obtained for all ternary test cases, it can be seen that the resulting linear and angular velocities and acting impulses appear to be oriented in intuitively correct directions and are of acceptable magnitude. Furthermore, since the various governing equations for momentum balance, velocity changes and tangential impulse orientations are satisfied, it can be accepted that the results are physically correct too, at least

where ternary collisions are concerned. Unfortunately no alternative exact analytical results are available for verification, since this study is the first of its kind.

The linear translations obtained using DEM differed from the non-linear rigid body solution results by around 10 % for most of the tests, though there were some really large discrepancies in a few components in test cases 2, 3, 7, 9 and 10. It is also apparent that most of these tests where the large discrepancies were found had momentum conservation errors for the DEM results. Rotations mostly differed by as much as 28 %, though in test case 2 it was 44 % and in test case 10 it was a staggering 512 % for the Z-components of rotation. The directions of rotation always matched those indicated by the rigid body approach, confirming that the intuitively predicted directions of rotation are correct. As before, the large discrepancy can be explained by virtue of the remarks made by Hoomans (1999:55-58) as stated in the previous secion (5.1.1.4). One really encouraging result obtained by especially the planar ternary collisions, i.e. ternary test cases 9 and 10, is that the general impulse-based/instantaneous formulation developed as a result of this study now provides a hard-sphere/rigid body approach alternative to the soft-sphere model for the multi-contact problem as stated by Hoomans (1999:60-62). Moreover, it does appear to yield more physically realistic results compared to those obtained using DEM in such ternary collisions as occur in test cases 9 and 10, especially when the kinetic energy and the linear momentum balances are considered. Momentum-balances, in particular, should never change during any collision, whether they are multi-body or binary in nature.

5.1.3 Four-Body Concurrent Collisions

Four-body (quaternary) collision test problems are again relatively basic and should all be passed without any error. The first true implicit reactionary contact interactions, where originally passive objects in a system become involved in the cluster collision due to the instantaneous redistribution of momentum, can be found in the quaternery linear stacking as well as the symmetrical planar tests. The first three-contact cluster scenarios arise in the both the symmetrical planar test case and the full three-dimensional quaternary contact test case. Also, the three-dimensional quaternary test cases yield the first scenarios where there are supposed to be more than one negative normal impulse, and thus where contacts should be deactivated during and by calculation procedures. In addition, all slipping contacts need to be identified and handled accordingly by the contact resolution algorithm employed. This section lists all tests and their results, as well as some benchmarks for and discussions of the results.

5.1.3.1 Setup and Execution

The linear stacking quaternary collision tests, tests 1 through 4, were performed using a general setup as shown in Figure 5.4 and then varying the indicated angle, θ, between 45° and 0° in discrete steps (i.e. 45°, 30°, 15°, 0°) in the XOY plane. Sphere 1 had a velocity vector of (1, 1, 1) m/s in the Cartesian coordinate system, whilst sphere 4 had a velocity vector of (-1, -1, -1) m/s.

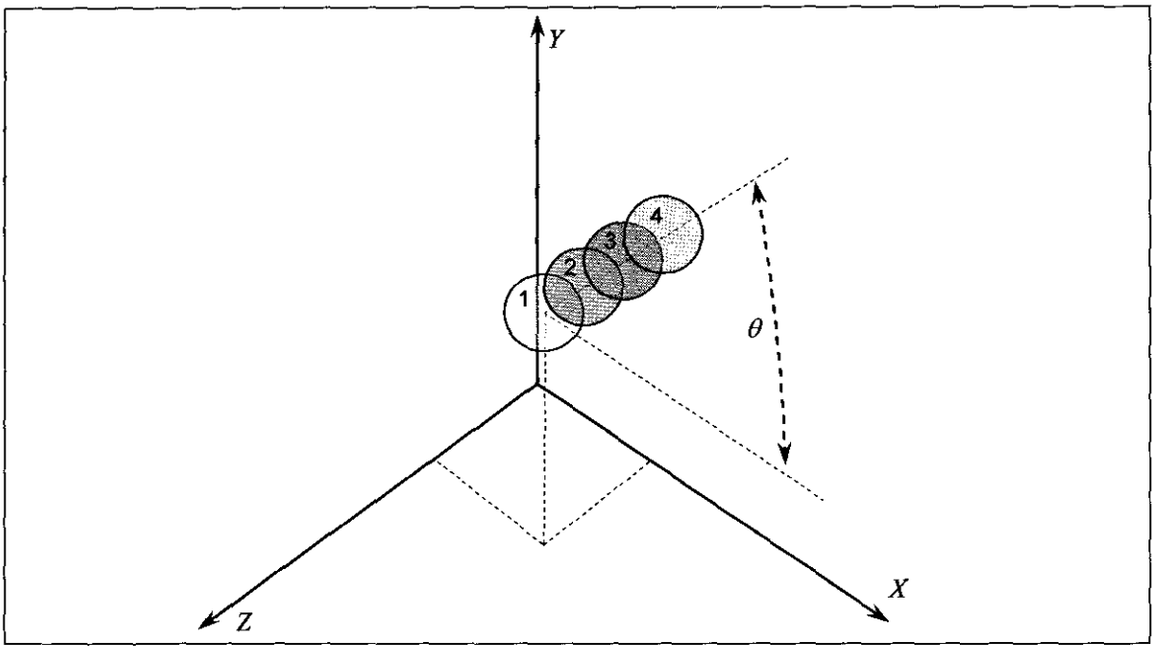


Figure 5.4: General Linearly Stacked Quaternary Collision Setup.

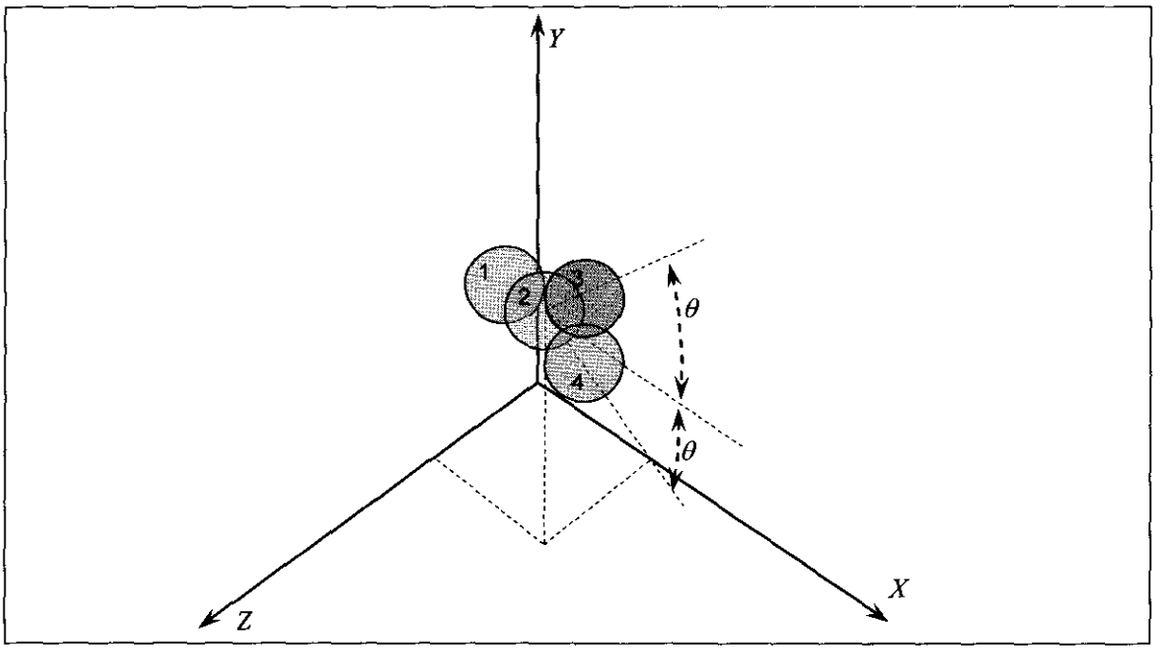


Figure 5.5: General Planar Quaternary Collision Setup.

The planar quaternary collision tests, cases 5 and 6, were performed using a general setup as shown in Figure 5.5, which is essentially a symmetrical collision test, with the angle from the symmetry line, θ, set to 30° to guarantee a four-contact system. The projectile (Sphere 1) in test case 5 was given a pre-collision velocity of (1, 0, 0) m/s and in test case 6 the projectile (Sphere 1) was given a velocity of (1, 0, 10) m/s, whilst all other spheres in the system were initially stationary. The full three dimensional quaternary tests, cases 7 through 10, were set up as indicated in Figure 5.6 with a triangular assembly of spheres being hit perpendicularly to their plane in the centre by a fourth projectile (Sphere 1). Sphere 1 had a linear velocity (0, 0, -1) m/s in case 7, (0.1, 0, -1) m/s in case 8, (-0.1*Cos(60°), -0.1*Sin(60°), -1) m/s in case 9 and (-0.1*Cos(60°), 0.1*Sin(60°), -1) m/s in case 10.

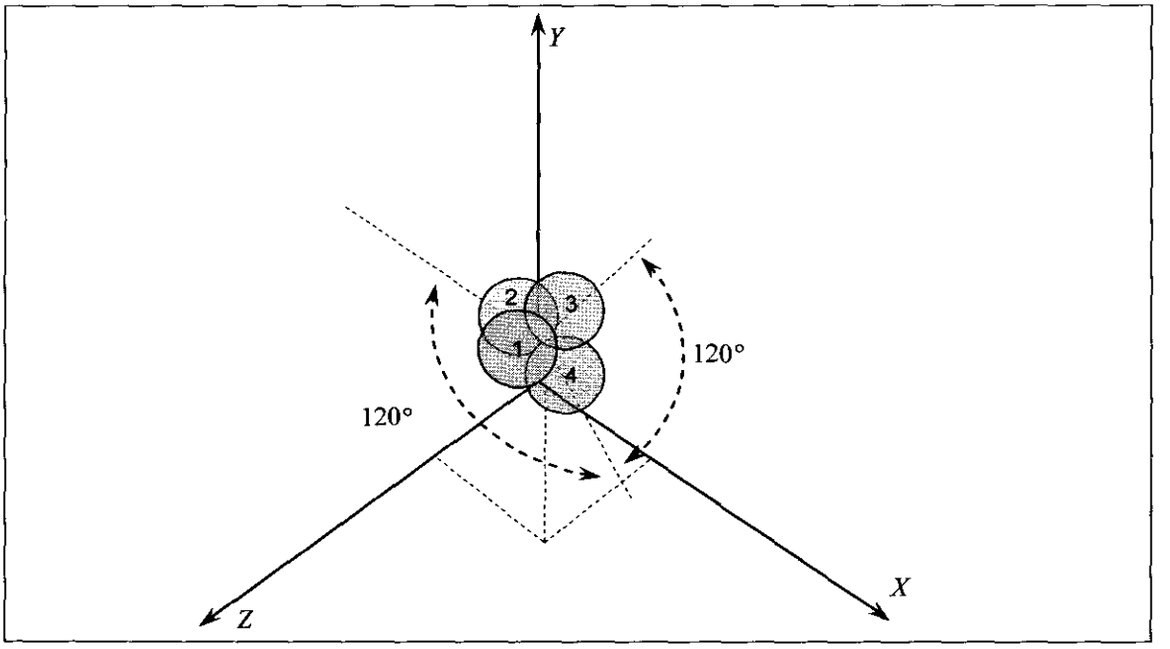


Figure 5.6: General Quaternary Collision Setup.

As for earlier test cases, the available linear and non-linear frictional solution methods were used to solve for the post-collision linear and angular velocities of each sphere and normal and tangential impulse magnitudes and tangential unit vector coordinates at each contact for each test case. An alternative set of DEM results (Polson, 2004) were also generated using PFC3D.

5.1.3.2 Benchmarks

The quaternary benchmark tests should all yield results either symmetrical or skew-symmetric round some axis or symmetry sphere. A brief summary of qualitative and some quantitative phenomena to be observed are now given.

For the linear stacking tests – as generally described by Figure 5.4, the results should exhibit the following phenomena:

- Translations
    - Should be of equal magnitude but in opposite axial directions for spheres 1 and 4 and also spheres 2 and 3 in test cases 1 through 4.
    - Should be such that relative velocities are directed away from one another between any two adjacent spheres in the system.
- Rotations
    - Should be in the same direction for all spheres in tests 1 through 4, with spheres 1 and 4 and also spheres 2 and 3 having exactly the same angular velocities.
    - Should have twice the magnitude of the inner two spheres (2 and 3) for the outer two spheres (1 and 4) in test cases 1 through 4.
- Contacts
    - Should initially number four in total.
    - Should have one inactive contact, between spheres 3 and 4.
    - Should have only three active contacts, between spheres 1 and 2, spheres 2 and 3 and also spheres 2 and 4.
    - Normal impulses should be equal in magnitude and always compressive.
    - Tangential impulses should be parallel in direction and equal in magnitude between spheres 1 and 2 and spheres 3 and 4.

For the planar tests – as generally described by Figure 5.5, the results should exhibit the following phenomena:

- Translations
    - Should be of equal magnitude but with mirrored directions round the symmetry line parallel to the X-axis for spheres 3 and 4 in tests 5 and 6.
    - Should be opposite in X-direction for sphere 1 compared to that of the other three spheres for both test cases 5 and 6.
    - Should be in the Z direction for all spheres in case 6, with sphere 1 having the highest, sphere 2 having a little lower and spheres 3 and 4 having the lowest and equal Z-velocity components.
- Rotations
    - Should initially number four in total.
    - Should be of zero magnitude for spheres 1 and 2 round all axes in test case 5.

- Should be equal but of opposite direction round the Z-axis for spheres 3 and 4 in tests 5 and 6.
- Should be in the same direction round the Y-axis for all spheres in test case 6, with sphere 2 having the greatest Y-component value.
- Should be of equal magnitude but opposite direction round the X-axis for spheres 3 and 4 and zero for spheres 1 and 2 round the X-axes in case 6.

- Contacts
  - Should have only three active contacts, between spheres 1 and 2, spheres 2 and 3 and spheres 2 and 4, the contact between sphere 3 and 4 should be inactive.
  - Normal impulses should always be compressive.
  - Tangential impulse directions and magnitudes should be symmetrical round the axis of symmetry, parallel to the X-axis for all planar tests.

For the full three-dimensional quaternary tests – as generally described by Figure 5.6, the results should exhibit the following phenomena:

- Translations
  - Should be of equal magnitude but with radially mirrored directions round the symmetry line through sphere 1 and parallel to the Z-axis for spheres 2, 3 and 4 in test case 7.
  - Should be opposite in Z-direction for sphere 1 compared to that of the other three spheres in test case 7 and only have a Z-component.
  - Should be symmetrical round the XOY-plane component of the impact velocity of sphere 1 for spheres 2, 3 and 4 in test cases 8, 9 and 10.
- Rotations
  - Should be of zero magnitude for sphere 1 and should not have any Z-components for any spheres in test case 7.
  - Should be of equal magnitude and radialy mirrored in three directions round the symmetry line for spheres 2, 3 and 4 in test case 7.
  - Should be symmetrical round the XOY-plane component of the impact velocity of sphere 1 for spheres 2, 3 and 4 in test cases 8, 9 and 10.
- Contacts
  - Should initially number six in total.
  - Should have three inactive contacts, between spheres 2 and 3, spheres 2 and 4 and spheres 3 and 4 in test case 7.

- Should have only three active contacts in test case 7, between spheres 1 and 2, spheres 2 and 3 and spheres 2 and 4, the contact between sphere 3 and 4 should be inactive.
- Normal impulses should always be compressive.
- Tangential impulse directions and magnitudes should be radially symmetric round the axis of symmetry.

In addition to all the aforementioned results, there should again be no energy gains (though there might and should be energy losses, due to slippage and incomplete restitution) and momentum should always be conserved. In coplanar cases, the linear approach should yield the same results as the non-linear one, and in non-coplanar cases, they should not differ greatly.

5.1.3.3 Results

The various results for the quaternary collision test cases can be seen in Appendix B, section B.3, listing both linear and non-linear approach results, as well as the independent DEM results (Polson, 2004).

5.1.3.4 Discussion

Data obtained from the greater part of the test runs indicate that the linear and non-linear collision resolution methods again yielded very close to the same results where the tangential impulses are coplanar or symmetric – barring the occasional difference in sign for exact zero values. The non-coplanar tangential results obtained, i.e. Test Cases 6, 8, 9 and 10, did differ mostly in their second decimal digits, indicating that the methods yield results that are at least in the same neighbourhood, including the energy balances. The fact that the values are similar, indicate that the correct roots were found for the non-linear equations. Upon closer inspection of the numerical results obtained for all quaternary test cases, it can be seen that the resulting linear and angular velocities and acting impulses appear to be oriented in intuitively correct directions and are of acceptable magnitude as was the case for the ternary collision tests. Furthermore, since the various governing equations for momentum balance, velocity changes and tangential impulse orientations are satisfied, it can be accepted that the results are physically correct too, at least where quaternary collisions are concerned. Unfortunately, as with the ternary tests, no alternative exact analytical results are as yet available for verification.

The DEM-results (Polson, 2004) for the quaternary test cases yielded results that could be expected from a partially non-concurrent collision modelling where more than one implicit "level" of collision is involved (in the case of test cases 1 through 4, it was the contact between

spheres 2 and 3 and in the case of test cases 5 and 6, it was the contacts between spheres 2 and 3 and spheres 2 and 4). The non-concurrence of collisions resulted in quite different momentum distribution patterns, if one were to look at and compare the results obtained by different means for the aforementioned test cases. The resulting discrepancies run the gamut from 0 % to 420 % depending on the specific setup solved, effectively indicating that DEM-results are not really relevant in most of the quaternary collision cases, since collisions were not handled entirely concurrent. Test case 7 revealed some more useful and encouraging results, since this three-dimensional equivalent to the planar ternary collision (ternary test case 9) of the previous section, yielded mostly low discrepancy results (below 10 %, except for one sphere's 34 % discrepancy in the linear velocity Z-component) when the rigid body and DEM approaches were compared. The single worst discrepancy seems to be attributable to the same momentum-imbalance problem as had arisen in the DEM solution of ternary test case 9, which in turn seems to be related to the selection of the very sensitive damping coefficients in the tangential and normal direction (Hoomans, 1999:55-58). The rigid body test results for quaternary test case 7 did show that the developed formulation and solution algorithm does provide a reliable and accurate alternative method for the quaternary concurrent collision as shown set up in Figure 5.6. The remaining test cases 8, 9 and 10 indicated that the rigid body based solver yielded the same result whenever hit in exactly the same fashion, only with the direction of the impact swung through 120 °C round the Z-axis each time. The qualitative effects expected did indeed appear to have manifested themselves for the rigid body approach in each case, and there are only discrepancies relative to the DEM results (Polson, 2004) due to the non-conservancy of momentum, non-concurrency of collisions and, again, also the tangential and normal damping coefficients used.

5.1.4 Five-Body Concurrent Collisions

Five body (quintenary) collision tests are yet a little more complicated than smaller systems, but they do present the possibility of seven concurrent collisions occurring, involving at least one of the rigid bodies in all of the four active collisions present. This facilitates studying the capability of the collision resolution algorithm to handle the solution of all variables correctly for seven concurrent, four co-located collisions. The correct identification of slippages where and whenever they occur also needs to be done as in the case of simpler configurations and is potentially a very tricky process. Furthermore, the setups used are the first true three-dimensional indirect or implicit collision test cases to be evaluated using the impulse based rigid body approach developed.

5.1.4.1 Setup and Execution

The general three-dimensional quintenary collision tests, test cases 1 through 3, were performed using a general setup as shown in Figure 5.7. For test case 1, sphere 1 had a velocity vector of (0, 0, -1) m/s in the Cartesian coordinate system, whilst all other spheres were stationary, while in test cases 2 and 3, sphere 1 respectively had velocities of (1, 0.5, -0.5) m/s and (1, -1, -1) m/s. As before, the available linear and non-linear frictional solution methods were used to solve for the post-collision linear and angular velocities of each sphere and normal and tangential impulse magnitudes and tangential unit vector coordinates at each contact for each test case.

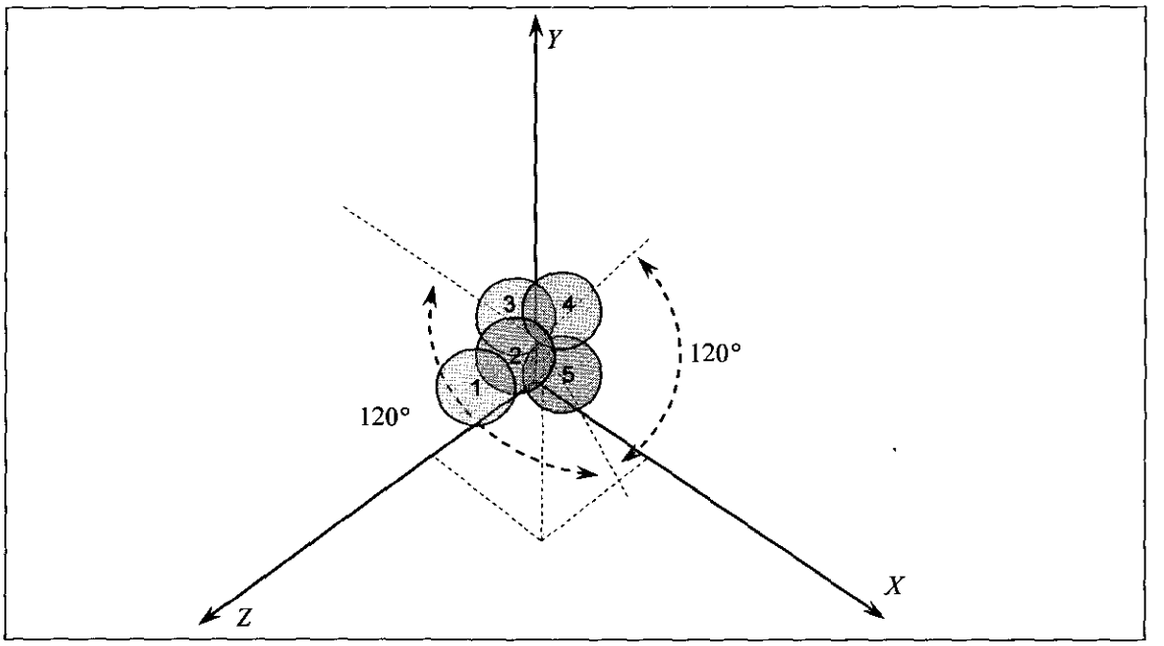


Figure 5.7: General Quintenary Collision Setup.

5.1.4.2 Benchmarks

The quintenary benchmark tests should at least yield results symmetrical round an axis of radial symmetry for the perfectly perpendicular impact test. A brief summary of qualitative and some quantitative phenomena to be observed are now given.

For the full three-dimensional quintenary tests– as generally described by Figure 5.7, the results should exhibit the following phenomena:

- Translations
    - Should be of equal magnitude but with radially mirrored directions round the symmetry line through sphere 1, parallel to the Z-axis, for spheres 3, 4 and 5 in test case 1.

- Should be opposite in Z-direction for sphere 1 compared to its original velocity for all test cases.
- Should have equal Z-components for spheres 3, 4 and 5 in test case 1.

- Rotations
  - Should be of zero magnitude for spheres 1 and 2 in test case 1.
  - Should be of equal magnitude and radialy mirrored in three directions round the symmetry line for spheres 3, 4 and 5 in test case 1.
- Contacts
  - Should initially number seven in total.
  - Should have three inactive contacts, between spheres 3 and 4, spheres 3 and 5 and spheres 4 and 5 in test case 1.
  - Should have only four active contacts in test case 1, between spheres 1 and 2, spheres 2 and 3, spheres 2 and 4 and spheres 2 and 5.
  - Normal impulses should always be compressive.
  - Tangential impulse directions and magnitudes should be radially symmetric round the axis of symmetry for test case 1.

As with the previous tests, there should be no energy gains (though there might and should be energy losses, due to slippage and incomplete restitution) and momentum should always be conserved.

5.1.4.3 Results

The various results for the quintenary collision test cases can be seen in Appendix B, section B.4, listing both linear and non-linear approach results, as well as the independent DEM results (Polson, 2004).

5.1.4.4 Discussion

Quintenary test runs mostly indicated that the linear and non-linear collision resolution methods yielded matching results where the tangential impulses are expected to be symmetric – again barring the occasional difference in sign for exact zero values. Non-coplanar or non-radially symmetric results, i.e. those for test cases 2 and 3, did differ significantly, indicating as in the quaternary case that the linear method can no longer be considered a good approximation to the non-linear solution as in earlier test-configurations and setups. Inspection of the numerical results obtained using the non-linear solver revealed that the resulting linear and angular velocities and acting impulses appear to be oriented in intuitively correct directions and are of

acceptable magnitude. Since the various governing equations for momentum balance, velocity changes and tangential impulse orientations are satisfied, it can be assumed with some degree of confidence that the results are physically correct too, at least where quintenary collisions are concerned. Unfortunately, as with the previous tests, no alternative exact analytical results are as yet available for verification.

As with the greater part of the DEM results (Polson, 2004) for the quaternary test cases, the quintenary test case DEM results are not strictly equivalent or comparable to the rigid body concurrent collision model results, since partial non-concurrency is one of the inevitable effects of the soft-sphere approach used. Comparison of the rigid body model and DEM results for the first quintenary test did indicate that the concurrent collision model yields results different from one another, illustrating once again that true concurrent collision modelling is not possible when using DEM, in particular when the collision involves more than one "layer" of initially stationary bodies in contact. Judging by the sum of linear momentum, momentum conservation is also shown to be hard to maintain using the DEM/soft-sphere approach. The second and third test cases further illustrated that the lack of concurrency exacerbates the already existing problem of momentum non-conservancy and differing momentum distribution in general, due to successive collisions that could and should have been handled as one single cluster collision. The large discrepancies in some of the linear and rotational velocities can once again be ascribed to the factors as summarised and explained by Hoomans (1999:55-58).

5.1.5 Six-Body Concurrent Collisions

Six body (hexenary) collision tests are only slighty more complicated than smaller systems, but they do present the possibility of ten concurrent collisions, four of these occurring co-located as in the case of quintenary collisions, involving at least one of the rigid bodies in a four contact interaction. This facilitates studying the capability of the collision resolution algorithm to handle the solution of all variables correctly for ten concurrent, four co-located collisions. The setups used were also the first true three-dimensional three-layer indirect or implicit collision test cases to be evaluated using the impulse based algorithm developed.

5.1.5.1 Setup and Execution

The general three-dimensional hexenary collision tests, test cases 1 through 3, were performed using a general setup as shown in Figure 5.8. For the only test case, case 1, sphere 1 had a velocity vector of (0, 0, -1) m/s in the Cartesian coordinate system, whilst all other spheres were stationary. Again the available linear and non-linear frictional solution methods were used to

solve for the post-collision linear and angular velocities of each sphere and normal and tangential impulse magnitudes and tangential unit vector coordinates at each contact for each test case.

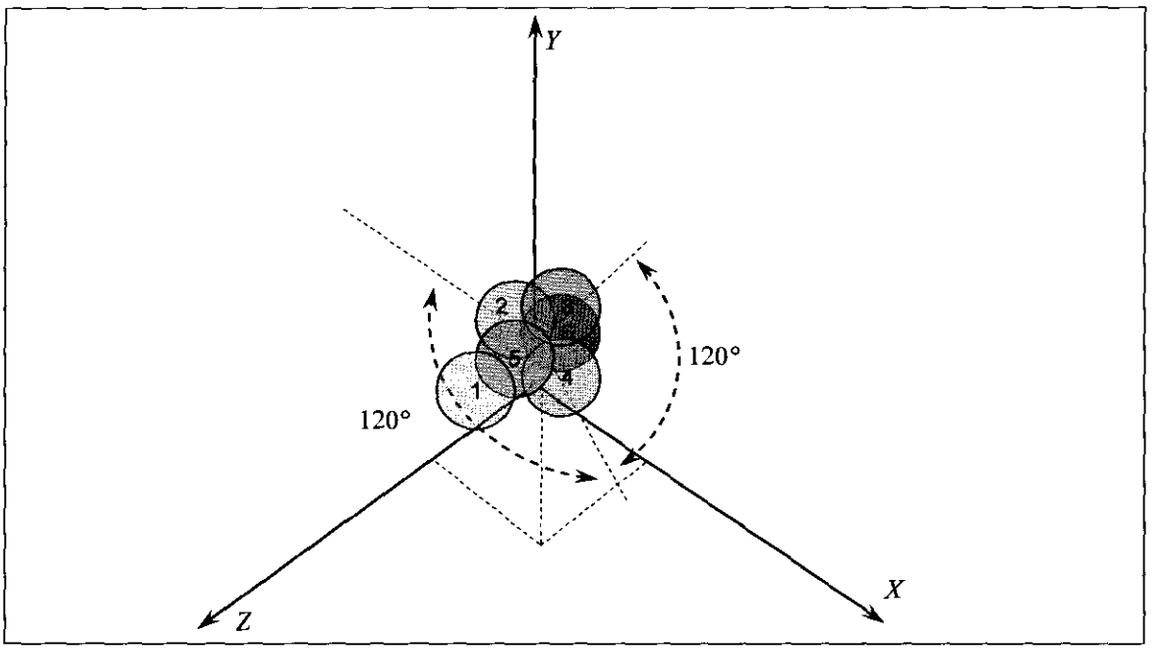


Figure 5.8: General Hexenary Collision Setup.

5.1.5.2 Benchmarks

As with the previous three-dimensional tests, the hexenary benchmark tests should at least yield results symmetrical round an axis of radial symmetry for the perfectly perpendicular impact test. A brief summary of qualitative and some quantitative phenomena to be observed are now given.

For the full three-dimensional hexenary test – as generally described by Figure 5.8, the results should exhibit the following phenomena:

- Translations
    - Should be of equal magnitude but with radially mirrored directions round the symmetry line through sphere 1, parallel to the Z-axis, for spheres 3, 4 and 5 in test case 1.
    - Should be opposite in Z-direction for sphere 1 compared to its original velocity for all test cases.
    - Should have equal Z-components for spheres 3, 4 and 5 in test case 1.
- Rotations
    - Should be of zero magnitude for spheres 1, 2 and 6 in test case 1.
    - Should be of equal magnitude and radialy mirrored in three directions round the symmetry line for spheres 3, 4 and 5 in test case 1.

- Contacts
    - Should initially number ten in total.
    - Should have at least three inactive contacts, between spheres 3 and 4, spheres 3 and 5 and spheres 4 and 5 in test case 1.
    - Normal impulses should always be compressive.
    - Tangential impulse directions and magnitudes should be radially symmetric round the axis of symmetry for test case 1.

As with the previous tests, energy can only ever be lost due to incomplete restitution and slippage, while momentum should always be conserved.

5.1.5.3 Results

The various results for the hexenary collision test cases can be seen in Appendix B, section B.5, listing non-linear approach results, as well as the independent DEM results (Polson, 2004).

5.1.5.4 Discussion

Hexenary test run data indicate that the linear and non-linear collision resolution methods yielded the same results where the tangential impulses were radially symmetric, apart from the occasional difference in sign for exact zero values. Closer inspection of numerical results obtained revealed that the resulting linear and angular velocities and acting impulses appear to be oriented in intuitively correct directions and are of acceptable magnitude. In addition to this, the various governing equations for momentum balance, velocity changes and tangential impulse orientations are yet again satisfied, indicating that it can be accepted with reasonable certainty that the hexenary collision results are physically correct. As with all other tests after the binary cases, no alternative exact analytical results are as yet available for verification.

As in quintenary collision cases, the additional DEM results for the hexenary case were again not strictly comparable or equivalent to those obtained using the rigid body approach, but still it does serve as an indicator of the possible advantages posed by the fully concurrent collision rigid body system approach. The most important advantage of the rigid body approach still is the proper conservation of momentum and the exact determination of normal and tangential impulse magnitudes and directions. The discrepancies apparent in the DEM results are again attributable to non-concurrency of collisions due to the nature of contact force modelling and the damping parameter problem (Hoomans, 1999:55-58) as before.

5.1.6 Seven-Body Concurrent Collisions

The chosen configuration's seven body (heptenary) collision tests are again more complicated than smaller systems, and they present the possibility of ten concurrent collisions occurring, involving at least one of the rigid bodies in a three contact interaction. This facilitates studying the capability of the collision resolution algorithm to handle the solution of all variables correctly for ten concurrent collisions in a snooker or billiards type of setup. The setups used yielded the first true three-dimensional three-layer indirect or implicit collision test cases of this kind to be evaluated using the impulse based algorithm developed.

5.1.6.1 Setup and Execution

The heptenary collision tests, cases 1 through 4 were performed using a general setup as shown in Figure 5.9. For the first test case, case 1, sphere 1 had a linear velocity vector of (1, 0, 0) m/s in the Cartesian coordinate system, whilst all six other spheres arranged in a triangular stack were stationary. Test case 2 had the same setup as case 1, but the projectile sphere (Sphere 1) was given an initial velocity of (1, 0, 10) m/s. Int test cases 3 and 4, sphere 1 respectively had a linear velocity of (1, 0.5, 1) and (1, -0.5, 1) m/s, with the rest of the spheres again being stationary and located as indicated in Figure 5.9.

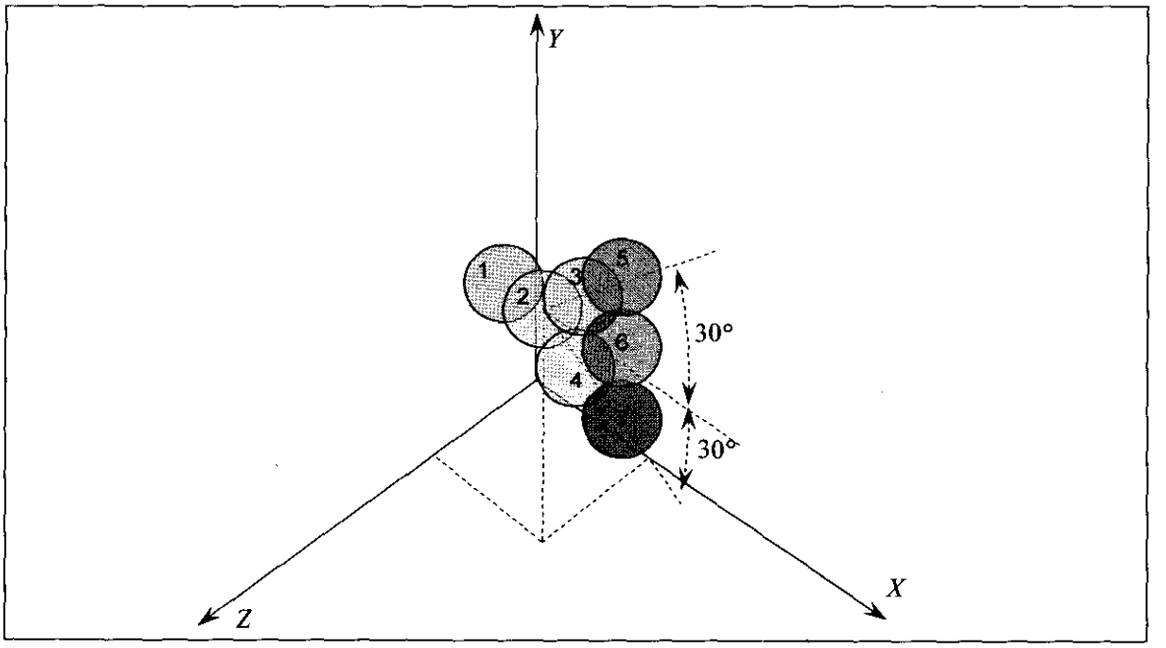


Figure 5.9: General Planar Heptanary Collision Setup.

Here too, the available linear and non-linear frictional solution methods were used to solve for the post-collision linear and angular velocities of each sphere and normal and tangential impulse magnitudes and tangential unit vector coordinates at each contact for each test case.

5.1.6.2 Benchmarks

The heptenary benchmark tests should at least yield results symmetrical round the centreline parallel to the X-axis and running through sphere 1. A brief summary of qualitative and some quantitative phenomena to be observed are now given.

- Translations
    - Should be mirrored in direction and magnitude round the symmetry line through sphere 1, parallel to the X-axis in test cases 1 and 2.
    - Should have exactly the same magnitudes in test case 3 and 4, only with the directions in case 3 flipped round the symmetry line through sphere 1 parallel to the X-axis in case 4.
    - Should be opposite in Z-direction for sphere 1 compared to its original velocity for all test cases.
    - Should have equal Z-components for spheres 3, 4 and 5 in test case 1.
- Rotations
    - Should be of zero magnitude for spheres 1, 2 and 6 in test case 1.
    - Should be of equal magnitude and mirrored round the symmetry line parallel to the X-axis running through sphere 1.
- Contacts
    - Should initially number ten in total.
    - Normal impulses should always be compressive.
    - Tangential impulse directions and magnitudes should be symmetric round the axis of symmetry for test case 1, which runs through sphere 1 and parallel to the X-axis.

As with all the other previous tests, there should be no energy gains (though energy losses, due to slippage and incomplete restitution are likely) and momentum should always be conserved.

5.1.6.3 Results

The various results for the heptenary collision test cases can be seen in Appendix B, section B.6, listing non-linear approach results, as well as the independent DEM results (Polson, 2004).

5.1.6.4 Discussion

Heptenary test run result data indicate that the linear and non-linear collision resolution methods yielded different results where the tangential impulses should have been symmetric for the coplanar test case (heptenary test case 1). Upon closer inspection of the numerical results

obtained for the heptenary test cases using the non-linear collision resolution algorithm, it can be seen that the resulting linear and angular velocities and acting impulses appear to be oriented in intuitively correct directions and are of acceptable magnitude. Furthermore, since the various governing equations for momentum balance, velocity changes and tangential impulse orientations are satisfied, it can be assumed with reasonable certainty that heptenary collision results are physically correct. Again no alternative exact analytical results are as yet available for verification.

An alternative DEM solution had again been obtained for each heptenary test case, and these, as could be expected, are again not equivalent or comparable to the concurrent contact approach, apart from also yielding symmetrical answers where expected (i.e. test cases 1 and 2), and demonstrating that test cases 3 and 4 are indeed mirror images of one another relative to the centre line running through sphere 1 and parallel to the X axis. The DEM results (Polson, 2004) exhibit the typical sequential momentum transfer to be expected from non-concurrent collision handling, combined with the unacceptable X-direction linear momentum loss in the vicinity of 40 % that was experienced. The large discrepancies of the DEM results relative to the rigid body approach results can once again be attributed to the partial non-concurrency of collisions and the sensitivity of the DEM solution to contact parameters used (Hoomans, 1999:55-58).

5.1.7 Eight-Body Concurrent Collisions

The last but not the least, eight body (octenary) collision tests are again more complicated than smaller systems and they present the possibility of fifteen concurrent collisions occurring, involving at least one of the rigid bodies in a co-located six contact interaction. This facilitates studying the capability of the collision resolution algorithm to handle the solution of all variables correctly for thirteen concurrent collisions. The setups used were also the first true three-dimensional three-layer indirect or implicit collision test cases to be evaluated using the impulse based algorithm developed. This test is very strict and highly crucial and should indicate any irregularities in the formulations as well as the solution technique, however small these might be, since it has strict radially symmetric results requirements to meet.

5.1.7.1 Setup and Execution

The general three-dimensional octenary collision test was performed using a general setup as shown in Figure 5.10. For the only test case, case 1, sphere 1 had a velocity vector of (0, 0, -1) m/s in the Cartesian coordinate system, whilst all other spheres were stationary.

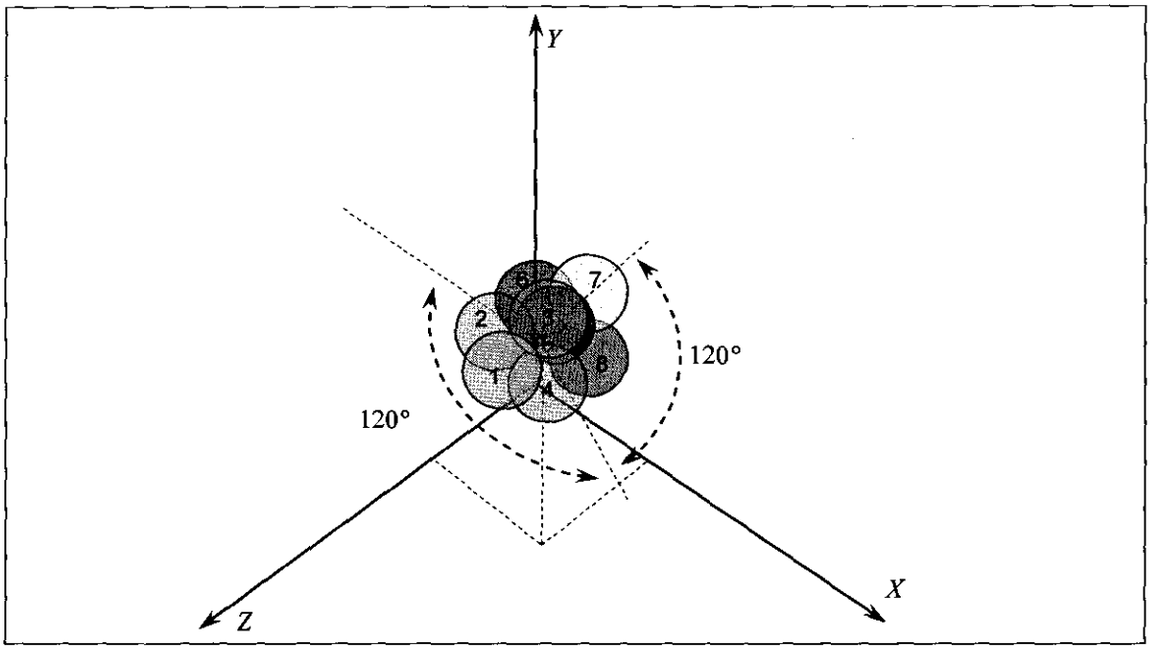


Figure 5.10: General Octenary Collision Setup.

The available linear and non-linear frictional solution methods were again used to solve for the post-collision linear and angular velocities of each sphere and normal and tangential impulse magnitudes and tangential unit vector coordinates at each contact for each test case.

5.1.7.2 Benchmarks

The octenary benchmark test should at least yield results radially symmetric round the centreline parallel to the Z-axis and running through sphere 1. A brief summary of qualitative and some quantitative phenomena to be observed are now given.

- Translations
    - Should be radially mirrored in direction and magnitude round the centreline through sphere 1, parallel to the Z-axis in test case 1.
    - Should only have a Z-component for sphere 5 in test case 1.
    - Should be opposite in Z-direction for sphere 1 compared to its original velocity for all test cases.
    - Should have equal Z-components for spheres 2, 3 and 4 and again for spheres 6, 7 and 8 in test case 1.
- Rotations
    - Should be of zero magnitude for spheres 1, and 5 in test case 1.
    - Should be of equal magnitude and radially mirrored in three directions round the axis of symmetry for spheres 2, 3 and 4 and again for spheres 6, 7 and 8 in test case 1.

- Contacts
    - Should initially number fifteen in total.
    - Should at least be inactive between spheres 2 and 3, spheres 2 and 4, spheres 3 and 4, spheres 6 and 7, spheres 6 and 8 and spheres 7 and 8.
    - Normal impulses should always be compressive.
    - Tangential impulse directions and magnitudes should be radially symmetric round the axis of symmetry for test case 1, which runs through sphere 1 and parallel to the Z-axis.

There could and should be some energy losses, due to slippage and incomplete restitution, and momentum should always be conserved.

5.1.7.3 Results

The various results for the octenary collision test cases can be seen in Appendix B, section B.7, listing non-linear approach results, as well as the independent DEM results (Polson, 2004).

5.1.7.4 Discussion

The octenary test runs indicate that the linear and non-linear collision resolution methods yielded different results for the supposedly radially symmetric test case. Upon closer inspection of the numerical results obtained using the non-linear collision resolution algorithm, resulting linear and angular velocities and acting impulses appear to be oriented in intuitively correct directions and of acceptable magnitude. And since the various governing equations for momentum balance, velocity changes and tangential impulse orientations are satisfied, it can be safely assumed that the results are physically correct too, at least where octenary collisions are concerned. No alternative exact analytical results are as yet available for verification of octenary test runs either.

The alternative DEM solution had again been obtained for each test case, and these, as could be expected, are again not equivalent or comparable to the concurrent contact approach, apart from also yielding radially symmetric answers where expected. The rigid body concurrent collision approach appears to have yielded a physically correct and intuitively sound set of results, with translational and rotational directions matching both that which was expected and that which had been indicated by the DEM results (Polson, 2004), though the momentum distribution is considerably different from that which would have been obtained when contacts were to be considered as fully concurrent. The DEM results exhibit the typical sequential momentum transfer to be expected from non-concurrent collision handling, combined with the unacceptable

Z-direction linear momentum loss in the vicinity of 31 % that was experienced. As with the previous test cases, the large discrepancies of the DEM results relative to the rigid body approach results can be attributed to the partial non-concurrency of collisions and the sensitivity of the DEM solution to contact parameters used (Hoomans, 1999:55-58).

5.1.8 More-Body Concurrent Collisions and Related Findings

Some further more-body concurrent collision tests were tentatively run, but it was found that difficulties and restrictions related to the obligatory matrix solution, to be done during any Newton-Raphson multi-variable solution process, arose due to ill conditioned matrices. Ill conditioning appears to be related to the fact that the number of off-diagonal terms related to any particular rigid body, increases by five for each of the contacts involving that particular body. It was found that none of the more conventional, and usually robust, sparse matrix solution techniques, such as Bi-CGM, CGSQUARE or BiCGSTAB could be employed. A matrix solution technique found to be able to solve the linear systems involved, was the Pivoting Gauss-Elimination, a full matrix based technique employed for the time being, since this study was more focussed on the development of the basic theory needed for the exact solution of multiple concurrent contacts than on the solution techniques for such sets of equations.

The Newton-Raphson iteration itself takes excessively longer and more numerous sub-iterations to reach satisfactory levels of convergence for sub-iterations as the number of contacts and rigid bodies involved increases. It also became apparent that the initial guess values for all the variables solved for are of crucial importance, as could be expected from the Newton-Raphson iteration used, which is always sensitive to starting values. The initital guesses for the directions of the tangential unit vectors are the most crucial and several informal tests revealed that bad guesses can prevent the algorithm from ever converging or yielding the right results. After various trials and errors, it was found that starting values of 1.0e-12 for all normal and tangential impulses yielded the most stable iterations, along with guessed unit tangentials with initial directions determined using the average linear and angular momentum of the system as a guideline (see section 3.1.1.2 for more detail on the related theory). Perhaps even more crucial are the initial guess values for the linear and angular velocity components, which are obtained using the linear solution method with the aforementioned initial guess values for the unit tangents. One of the obvious problems that might arise in almost any system is that when there is no resultant linear or angular momentum in a system to determine the probable direction of tangential impulses from, they are nearly impossible to predict at present. Most of the other

problems described above, apart from the tangential unit vector prediction, appear to occur only in assemblies containing more than eight rigid bodies as far as could be determined, and thus the limit for the present study had been set at octenary clusters.

5.2 SUMMARY OF DISCUSSIONS

The results obtained by running the various test cases were encouraging and indicated that the developed non-linear rigid body collision resolution algorithm can at least be used to solve for simple systems with multiple concurrent, co-located contacts. Firstly, the results for the simpler test cases – i.e. binary, ternary and quaternary – indicated that momentum conservation is maintained with very high accuracy, and that the directions and magnitudes of post-collisional linear and angular velocities are consistent with what was expected qualitatively and quantitatively.

In further support of the results for binary, ternary and quaternary tests, the alternative DEM-based solutions mostly yielded results of the same order of magnitude, having the same vector directions, especially where linear velocities are concerned. The larger discrepancies found can be explained by virtue of the remarks made by Hoomans (1999:55-58) – e.g. ternary test cases 1 through 7 and 10 and quaternary test cases 1 through 6 and 8 through 10. Apparently also related to the contact parameter problem (Hoomans, 1999:55-58), DEM experiences some inherent difficulty with the balancing of momentum – e.g. ternary test cases 9 and 10, and quaternary test cases 7 through 10.

The more complex test cases unfortunately fared in uncharted waters and only approximate comparisons could be made with the data obtained from either the linear solution or the DEM solution benchmark-results (Polson, 2004). The linear contact resolution algorithm, in particular, yielded totally improbable answers that did not even vaguely resemble the expected qualitative, let alone comparable numerical, results. In contrast to the alternative results, those obtained using the non-linear contact resolution algorithm were representative of the expected qualitative results for rigid bodies in concurrent contact. It could be induced from the success and accuracy of the simpler test case results that the more complex test cases can also be expected to maintain momentum balance, physical realism and accuracy when the non-linear rigid body approach is used. The general directions of motion seem to be affirmed by the DEM results, but the momentum distribution and the momentum imbalances are too large for the results to be of any help in either confirming or disproving the results obtained using the rigid body concurrent collision non-linear solution. What does make the more complex non-linear results acceptable,

are the facts that momentum balance is still maintained, qualitative benchmark results as described for each test case were indeed reflected, and the magnitudes and distribution of velocities appear to have been logical extensions of similar simpler test cases (e.g. quintenary test case 1 has a realistic redistribution of momentum compared to and also somewhat similar to that of quaternary test case 7 and in turn hexenary test case 1 has further redistributed momentum compared to both quintenary test case 1 and quaternary test case 7).

The test results obtained from the heptenary and octenary test cases are also of purely academical interest at the moment, though they also did satisfy the momentum conservation requirements and did yield results consistent with what was expected qualitatively. In all test cases requiring radial symmetry of results (quaternary 7, quintenary 1, hexenary 1 and octenary 1), the quantitative results were indeed radially symmetric and accurate up to the tenth decimal digit, and no momentum imbalances were ever introduced by the solutions obtained using the non-linear collision resolution algorithm. Lastly, the concurrent contact clusters numbering more than eight participating rigid bodies had occasionally yielded the expected qualitative results, but the solver could not stably or uniformly handle all permutations of the test cases, and thus the limit is set at octenary collisions for the time being. Judging by tentative experiments it does appear that any simultaneous collision of more than three layers of impacted rigid bodies might not be physically realistic, possibly due to wave propagation effects (Stronge, 2003). Several numerical aspects, such as importance of good initial guess values for Newton-Raphson iterations, quick, accurate and robust matrix solution techniques and general physically based constraint enforcement, were brought to the fore by the test runs. This concludes the summary of the results obtained, and in the next chapter, the final conclusions and summary of contributions are now made and recommendations can be done with respect to future work.

Chapter 6 Conclusions and Recommendations

In this chapter, the conclusion and then the general contributions made by and capabilities, reliability and accuracy to be expected from the present study are summarised. Following the summary of contributions and the motivations for their validity, recommendations based on the various shortcomings uncovered are given. This includes suggestions regarding the mathematical aspects, practical application and further investigation of the physical phenomena being modelled.

6.1 Conclusion

This study set out on a quest to find a rigid body system interaction formulation suitable for engineering purposes, i.e. accurate, simple, general and quick. A thorough literature survey was done, reviewing and evaluating sources from various relevant fields of interest regarding their suitability and applicability. The formal research topic was then formulated as *the development of a more physically based approach to the solution of systems containing multiple concurrent rigid body collisions.* The usual basic theory for sum and transfer of momentum was employed, but the specification of tangential impulse direction was done by requiring that it be in the exact direction of the post-collisional slip-velocity at each contact, not by maximising energy loss as is usually the case in alternative methods. In addition to the novel tangential impulse direction specification, the contact breaking and tangential slippage inequality constraints were handled in a more physically based manner than hitherto encountered. The test results obtained indicated that the non-linear mathematical formulation for the multiple-concurrent-collision-problem is physically realistic, and that the solution of the sets of constrained non-linear equations involved is the only obstacle to be overcome by future research for this formulation to be practical in a wide range of applications. The rudimentary physically based solution algorithm can presently handle up to octenary (eight-body) multiple concurrent collisions, with up to six co-located contacts, but indications are that these limitations with respect to number of rigid bodies and co-located collisions can be overcome when a more suitable sparse matrix solution technique can be found or developed.

6.2 CONTRIBUTIONS

Several conclusions can be made regarding the contributions made to the basic theory, algorithms, and software components used in numerical rigid body modelling. In addition to the new theory and techniques, various differences from DEM, advantages, disadvantages and limitations of the currently existing RBSS technique were demonstrated by the results obtained from running the test cases. The following sections briefly highlight the contributions.

6.2.1 Basic Theory

Extensions were made to basic rigid body impact and momentum transfer theory, simplifying the approach to representing multiple concurrent impact phenomena using a collection of rigid body momentum conservation and velocity change equations. The particular contribution details are as follows:

- Implicit, non-linear formulations were developed, linking rigid body velocity changes, normal and tangential impulse magnitudes and tangential impulse directions at each contact – see Figure 4.9 for the total mathematical problem statement and equations (3.15) through (3.17) and (3.19) through (3.25) for explanations of its origin.
- Contact breaking handled by inequality constraints for each normal impulse.
- Tangential slippage specified by inequality constraints for each tangential impulse.
- Unique specification of tangential impulse direction using equation (3.23) for the orthonormal unit tangential vector, forcing the tangential impulse to always point in the direction of the post-collision sliding velocity at each contact and avoiding the use of complicated and possibly erroneous impact energy formulations.
- Avoidance of using a friction cone approximation by directly applying the friction factor in the direction of the sliding velocity where necessary (valid for isotropic friction surfaces).
- Independent normal and tangential restitution factors, making it possible to model contact behaviour more realistically whenever experimental results would indicate a correlation problem for data obtained.

6.2.2 Algorithms

The constraint-based contact resolution algorithm developed is able to handle non-linear sets of equations, which might be handy whenever impact-equations other than the simple Newton-restitution equations would be used to specify the momentum transfer implicitly. The detailed collection of contributions can be summarised as follows:

- Physically based constraint enforcement, evaluating normal and tangential impulse magnitudes and then only removing or altering the contact or contacts that exceed their natural limits the most, thus making it more intuitive in eliminating breaking contacts and applying slipping equations where necessary.
- Handles non-linear interdependencies using a customised Newton-Raphson-based iteration.
- Initialises variables to be solved using Newton-Raphson iterations by intelligent predictive guessing of the unit tangential directions, based on combined linear and angular velocity of the system of contacting bodies, and also uses tentative velocity distribution calculations obtained from the linear resolution approach.

6.2.3 Software Components

The software components developed in this study provide various independently functioning and re-usable tools relevant to geometrical calculations, data storage, grid representation and rigid body simulation in general. In more detail, the following components are now available in future work:

- Reusable generic geometrical mathematics utilities.
- Reusable generic Object Oriented data storage for all data used.
- Reusable generic Cartesian coordinate system based sorting bins and related locating and sorting routines for spheres, vertices, lines/edges and triangular facets.
- Reusable generic independent rigid-body-system object with associated contact finding and concurrent contact resolution solver routines.

6.2.4 Test Case Results

The test case results revealed several traits of RBSS, mostly advantageous, though some drawbacks were also encountered, as should well have been expected. Some of these traits, be they negative or positive, are contributions to the understanding of RBSS itself, as well as the main alternatives used in engineering contexts, such as DEM (Polson, 2004). The results listed and summarised in Chapter 5 are now recapitulated and their meanings within the context of contributions are highlighted.

Table 6.1: The Various Properties of DEM and RBSS compared

Property	DEM (PFC3D)	RBSS
1. Multiple concurrent, co-located contacts handled accurately.	Yes, incapable of modelling true concurrency for highly rigid bodies.	Yes, capable of modelling highly rigid bodies using truly concurrent contacts. Limited to eight or less bodies at present.
2. Modelling parameters to be set for materials.	At least four problem-specific and sensitive parameters: - Normal spring. - Tangential spring. - Global damping. - Friction factor.	At least four problem-specific and sensitive parameters: - Normal restitution. - Tangential restitution. - Static friction factor. - Dynamic friction factor.
3. High speed collisions handled easily.	Yes, but only when using the correct spring and damping constants, and very problem specific, prone to energy-related problems.	Yes, no apparent restrictions on numerical stability for reasonable approach velocities (tested up to 10 m/s in informal setups).
4. Modelling of very large systems of rigid bodies.	Yes, but no true concurrency or rigidity.	Not at present, limited to eight bodies.
5. Strict enforcement of momentum balance.	No explicit momentum balance enforcment.	Yes, part of the mathematical problem statement.
6. Tangential collisions handled reliably.	No, dependent on and highly influenced by the spring and damping constants chosen.	Yes, part of the mathematical problem statement.

Looking at the various positive results listed in Table 6.1 above, as well as the various individual test results listed and discussed in Chapter 5, it can be seen that several small steps had been made in the direction of true concurrent contact modelling with engineering level accuracy. There still remains a lot of future work to be done on this subject. The suggestions or recommendations for further work that follow in the next section can be seen as a particular outcome and contribution of this study, since the identification of shortcomings and problems is indeed a step forward too.

6.3 Suggestions for Further Work

The suggestions for further work listed in this section highlight the various shortcomings and problems that were uncovered or identified by this study and also suggests some topics for further investigation of the currently existing methods' capabilities. The same categories are used here as for the summary of contributions, since these are the main areas where further work would be required or suggested.

6.3.1 Basic Theory

The basic theory applicable and relevant to multiple-concurrent-contact modelling using the RBSS approach can be extended or modified in various ways and spans a multitude of disciplines and research foci. In the following subsections, the suggested work pertaining to these research foci are listed.

6.3.1.1 RBSS Theory

The following suggestions are made regarding the basic RBSS theory:

- Investigation of alternative momentum or energy transfer expressions to account for relative velocity changes at each contact in both tangential and normal directions.
- Investigation as to whether better expressions are available for enforcing the direction of the unit tangential vector.

6.3.2 Algorithms

The following suggestions are made regarding the various algorithms necessary for and used in the collision resolution algorithm:

6.3.2.1 Matrix Solution

The following suggestions are made regarding sparse matrix solution techniques:

- Search for or investigate more robust sparse matrix solution methods or theory to handle very ill conditioned matrices encountered in and generated by the Newton-Raphson based RBSS non-linear solution technique.

6.3.2.2 Constrained Non-Linear Equation Set Solution

The following suggestions are made regarding the constrained non-linear equation solution techniques:

- Search for or investigate ways of obtaining better initial guess value for the Newton-Raphson algorithm currently employed.
- Find better ways to simultaneously predict and apply the positive normal impulse and contact slippage constraints at the correct contacts.
- Search for or investigate better or alternative, faster multiple-variable non-linear equation solution techniques (as opposed to the normal Newton-Raphson iteration used) able to handle larger sets of equations.

- Search for or investigate better or alternative, faster large constrained multiple-variable non-linear equation solution techniques.

6.3.3 Software Components

The following suggestions are made regarding the software components developed:

- Develop a graphical interface able to visually do model-setup (material, vertex, edge, facet and sphere property specification), display model solutions (spatial arrangement) in real-time.
- Identification of clusters of rigid bodies which can all be solved separately.
- Support for more complex rigid bodies such as various types of geometric primitive solids (cubes, prisms, pyramids, more-faceted polyhedra).
- Coupled fluid-grid solutions for fluid-particle interaction investigations.

6.3.4 Test Cases

The following suggestions are made regarding future test cases:

- Do "two-layered" tightly-packed sphere assembly tests, where one layer has initial velocity, and the other is stationary.
- Do "corner" reflection tests for sphere interaction with facets and edges.
- Do true thirteen ball "tightly packed cluster" tests.

Bibliography

ADREANI, R., FRIEDLANDER, A., MELLO, M.P. & SANTOS, S.A. 2002. **Mixed nonlinear complementarity problems via nonlinear optimization - numerical results in multi-rigid-body contact problems with Coulomb friction.** [Web:] http://www.ime.unicamp.br/rel_pesq/2002/ps/rp50-02.pdf [Date of access: December 2003]

ALANELLI, M. & HADJIDIMOS, A. 2004. **Block Gauss elimination followed by a classical iterative method for the solution of linear systems.** *Journal of Computational and Applied Mathematics*, 163(2): 381-400, Feb.

ANDERSON, D.A., TANNEHILL, J.C. & PLETCHER, R.H. 1984. **Computational fluid mechanics and heat transfer.** New York : Hemisphere Publishing Corporation. 599 p.

ANITESCU, M. & HART, G.D. 2000a. **A constraint-stabilized time-stepping approach for rigid multibody dynamics with joints, contact and friction.** [Web:] http://www-unix.mcs.anl.gov/~anitescu/PUBLICATIONS/submit1.pdf [Date of access: January 2004]

ANITESCU, M. & HART, G.D. 2000b. **Solving nonconvex problems of multibody dynamics with joints, contact and small friction by successive convex relaxation.** [Web:] http://www-unix.mcs.anl.gov/~anitescu/PUBLICATIONS/prague1.pdf [Date of access: January 2004]

ANITESCU, M. & HART, G.D. 2002. **A fixed-point iteration approach for multibody dynamics with contact and small friction.** [Web:] http://www-unix.mcs.anl.gov/~anitescu/PUBLICATIONS/prague10.pdf [Date of access: January 2004]

ANITESCU, M. 2002. **A fixed time step approach for multi-body dynamics with contact and friction.** [Web:] http://www-unix.mcs.anl.gov/~anitescu/PUBLICATIONS/iros2.pdf [Date of access: January 2004]

ANITESCU, M. 2003. **On using the elastic mode in nonlinear programming approaches to mathematical programs with complementarity constraints.** [Web:] http://www-unix.mcs.anl.gov/~anitescu/PUBLICATIONS/mpecrevfinal1.pdf [Date of access: January 2004]

ANITESCU, M., MILLER, A. & HART, G.D. 2003. **Constraint stabilisation for time-stepping approaches for rigid multibody dynamics with joints, contact and friction.** [Web:] http://www-unix.mcs.anl.gov/~anitescu/PUBLICATIONS/andrew2.pdf [Date of access: January 2004]

ANONYMOUS. 2002. **PFC3D (Particle Flow Code in 3D): Theory and background manual. Version 3.0.** Minneapolis : Itasca Consulting Group Inc.

ÅSTRÔM, K., CIPOLLA, R. & GIBLIN, P.J. 1996. **Generalised epipolar constraints.** [Web:] http://www.maths.lth.se/matematiklth/personal/kalle/publi/astrom_cipolla_giblin.eccv96.ps [Date of access: March 2004]

AXELSSON, O. & KARÁTSON, J. 2004. **Conditioning analysis of separate displacement preconditioners for some nonlinear elasticity systems.** *Mathematics and Computers in Simulation*, 64(6): 649-668, Mar.

AXELSSON, O. 2003. **Iteration number for the conjugate gradient method.** *Mathematics and Computers in Simulation*, 61(3-6): 421-435, Jan.

BARAFF, D. & WITKIN, A. 1997. **Partitioned dynamics.** [Web:] http://vdream.kist.re.kr/~zinook/RigidBody/partition-tr.pdf [Date of access: January 2004]

BARAFF, D. 1989. **Analytical methods for dynamic simulation of non-penetrating rigid bodies.** *Computer Graphics*, 23(3): 223-232, Jul.

BARAFF, D. 1993. **Non-penetrating rigid body simulation.** [Web:] http://www.cs.unc.edu/~dm/UNC/PHYSICS/Papers/baraff93.ps.gz [Date of access: January 2004]

BARAFF, D. 1994. **Fast contact force computation for nonpenetrating rigid bodies.** [Web:] http://www-2.cs.cmu.edu/afs/cs/user/baraff/www/papers/sig94.pdf [Date of access: January 2004]

BARAFF, D. 1997. **Rigid body simulation - SIGGRAPH'97 notes.** [Web:] http://www-imagis.imag.fr/Membres/Gilles.Debunne/Enseignement/DEA/PDF/rigid.pdf [Date of access: January 2004]

BARAFF, D. 1999. **Physically based modelling.** [Web:] http://graphics.stanford.edu/courses/cs448b-00-winter/papers/phys_model.pdf [Date of access: January 2004]

BARRET, R., BERRY, M., CHAN, T., DEMMEL, J., DONATO, J., DONGARRA, J., EIJKHOUT, V., ROMINE, C. & VAN DER VORST, H.A. 1994. **Templates for the solution of linear systems: building blocks for iterative methods.** Philadelphia, Pasadena : SIAM. 104 p.

BENEDETTI, F. 2002. **Physical response to collision between deformable objects.** [Web:] http://vrlab.epfl.ch/public/STUDENTS_PROJECTS/Fabiana.Benedetti/fbenedetti_2002.pdf [Date of access: January 2004].

BORESI, A.P., SCHMIDT, R.J. & SIDEBOTTOM, O.M. 1993. **Advanced mechanics of materials (Fifth edition).** New York : John Wiley & Sons, Inc. 811 p.

BRUYNICKX, H. & KHATIB, O. 2000. **Gauss' principle and the dynamics of redundant and constrained manipulators.** [Web:] http://www.robotics.stanford.edu/groups/manips/publications/files/icra00d.pdf [Date of access: February 2004]

BUCK, M. & SCHÖMER, E. 1998. **Interactive rigid body manipulation with obstacle contacts.** [Web:] http://www-hotz.cs.uni-sb.de/home/schoemer/publications/WSCG98.pdf [Date of access: December 2003]

BÜCKER, H.M. 2002. **Iteratively solving large sparse linear systems on parallel computers.** (*In* Grotendorst, J., Marx, D. & Muramatsu, A. *eds*. Quantum simulations of complex many-body systems: from theory to algorithms. Julich : John von Neumann Institute for Computing. p. 521-548.)

BUROV, A.A. 2003. **A transformation of the equations of mechanics.** *Journal of Applied Mathematics and Mechanics*, 67(5): 639-646.

CAFIERO, R. & LUDING, S. 2000. **Mean field theory for a driven granular gas of frictional particles.** [Web:] http://arxiv.org/PS_cache/cond-mat/pdf/0011/0011484.pdf [Date of access: December 2003]

CEANGA, V. & HURMUZLU, Y. 2001. **A new look at an old problem: Newton's cradle.** [Web:] http://cyborg.seas.smu.edu/syslab/papers/paper2_3.pdf [Date of access: December 2003]

CELLEDONI, E. & OWREN, B. 2003. **Lie group methods for rigid body dynamics and time integration on manifolds.** *Computer Methods in Applied Mechanics and Engineering*, 192 (3-4): 421-438, Jan.

CHAMORET, D., SAILLARD, P., RASSINEUX, A. & BERGEAU, J-M., 2004. **New smoothing procedures in contact mechanics.** *Journal of Computational and Applied Mathematics*, 168(1-2): 107-116, Jul.

CHATTERJEE, A. & RUINA, A. 1997. **Two interpretations of rigidity in rigid body collisions.** [Web:] http://tam.cornell.edu/~ruina/hplab/downloads/collision_papers/latest_rigidity_paper.pdf [Date of access: November 2003]

CHATTERJEE, A. & RUINA, A. 1998. **A new algebraic rigid body collision law based on impulse space considerations.** [Web:] http://tam.cornell.edu/~ruina/hplab/downloads/collision_papers/latest_alg_law.pdf [Date of access: November 2003]

CHEN, H-D., PARDALOS, P.M. & SAUNDERS, M.A. 1993. **The simplex algorithm with a new primal and dual pivot rule.** [Web:] http://dimacs.rutgers.edu/techps/1993/93-39.ps [Date of access: February 2004]

CHEN, Z., EWING, R.E., KUZNETZOV, Y.A., LAZAROV, R.D. & MALIASSOV, S. 2000. **Multilevel preconditioners for mixed methods for second order elliptic problems.** [Web:] http://www.isc.tamu.edu/iscpubs/9414.ps [Date of access: February 2004]

CHENEY, W. & KINCAID, D. 1999. **Numerical mathematics and computing (Fourth edition).** New York : Brooks/Cole Publishing Company. 671 p.

CLINE, M.B. 2002. **Rigid body simulation with contact and constraints.** Vancouver, Canada: University of British Columbia – Department of Computer Science (Dissertation – M.Sc.). 101 p.

COHOON, J.P. & DAVIDSON, J.W. 1999. **C++ program design: an introduction to programming and object-oriented design (Second edition).** Singapore : McGraw-Hill Book Company. 896 p.

CUNDALL, P.A. & STRACK, O.D.L. 1979. **A discrete numerical model for granular assemblies.** *Géotechnique,* 29(1): 47-65, Jun.

DINGLIANA, J. & O'SULLIVAN, C. 2003. **Graceful degradation of collision handling in physically based animation.** [Web:] http://www.cms.livjm.ac.uk/library/CMSSEM026/papersPresentation/AnimationPresentation/Nethercott/eggracef.pdf [Date of access: January 2004]

DURBIN, J.R. 1992. **Modern algebra: an introduction (Third edition).** New York : John Wiley & Sons, Inc. 348 p.

EGAN, K.T. 2003. **Techniques for real-time rigid body simulation.** Providence, Rhode Island: Brown University – Department of Computer Science (Dissertation – Hons.Sc.). 38 p.

EL KAHOUI, M., WEBER, A. & EBERHARDT, B. 2001. **Improved algorithms for linear complementarity problems arising from collision response.** *Mathematics and Computers in Simulation,* 56(1): 69-93, Mar.

ELLIS, R. & GULICK, D. 1994. **Calculus with analytic geometry (Fifth edition).** Fort Worth : Saunders College Publishing – Harcourt Brace College Publishers. 1113 p.

ENGELSSON, V. 2000. **Tools for design, interactive simulation, and visualization of object-oriented models in scientific computing.** Linköping : Linköpings Universitet (Dissertation - D.Phil.), 343 p.

ERLEBEN, K. 2001. **En introducerende lærebog i dynamisk simulation af stive legemer.** [Web:] http://www.diku.dk/hjemmesider/ansatte/kenny/speciale.pdf [Date of access: December 2003]

ERLEBEN, K. 2002. **Module based design for rigid body simulators.** [Web:] http://www.diku.dk/publikationer/tekniske.rapporter/2002/02-06.ps [Date of access: December 2003].

FASSHAUER, G.E. 2004. **Toward approximate moving least squares approximation with irregularly spaced centers.** *Computer Methods in Applied Mechanics and Engineering,* 193(12-14): 1231-1243, Mar.

FAURE, F. 1997. **Interactive solid animation using linearized displacement constraints.** [Web:] http://vdream.kist.re.kr/~zinook/RigidBody/faure97.pdf [Date of access: January 2004]

FORTIN, J.J. & COOREVITS, P. 2004. **Selecting contact particles in dynamics granular mechanics systems.** *Journal of Computational and Applied Mathematics,* 168(1-2): 375-382, Jul.

FRIEDMAN, A., KLANDER, L., MICHAELIS, M. & SCHILDT, H. 1999. **C/C++ annotated archives.** New York : McGraw-Hill Book Company. 751 p.

GAMMA, E., HELM, R., JOHNSON, R. & VLISSADES, J. 1995. **Design patterns: elements of reusable object oriented software.** New York : Addison-Wesley Publishing Company. 395 p.

GHORESHI, F. & HOSSEINI, S.M. 2004. **The Tau method and a new preconditioner.** *Journal of Computational and Applied Mathematics*, 163(2): 351-379, Feb.

GIANG, T., BRADSHAW, G. & O'SULLIVAN, C. 2003. **Complementarity based multiple point collision resolution.** [Web:] http://isg.cs.tcd.ie/giangt/EGIrl2003.pdf [Date of access: January 2004]

GILARDI, G. & SHARF, I. 2002. **Literature survey of contact dynamics modelling.** *Mechanism and Machine Theory*, 37(10): 1213-1239, Oct.

GLOCKER, C. 1997. **Formulation of rigid body systems with nonsmooth and multivalued interactions.** *Nonlinear Analysis*, 30(8):4887-4892, Dec.

GOLDBERG, K., MIRTICH, B., ZHUANG, Y., CRAIG, J., CARLISLE, B. & CANNY, J. 1999. **Part pose statistics - estimators and experiments.** [Web:] http://www.ieor.berkeley.edu/~goldberg/pubs/eps.pdf [Date of access: January 2004]

GOLDSTEIN, H. 1980. **Classical mechanics (Second edition).** New York : Addison-Wesley Publishing Company. 672 p.

GONZÁLEZ, P., PENA, T.F. & CABALEIRO, J.C. 2004. **Parallel sparse approximate preconditioners applied to the solution of BEM systems.** *Engineering Analysis with Boundary Elements*, 28(9): 1061-1068, Sept.

GU, R.J., MURTY, P. & ZHENG, Q. 2002. **Use of penalty variable in finite element analysis of contacting objects.** *Computers & Structures*, 80(31): 2449-2459, Dec.

GUN, H. 2004. **An effective BE algorithm for 3D elastoplastic frictional contact problems.** *Engineering Analysis with Boundary Elements*, 28(7): 859-867, Jul.

HASEGAWA, S., FUJII, N., KOIKE, Y. & SATO, M. 2003. **Real-time rigid body simulation based on volumetric penalty method.** [Web:] http://sklab-www.pi.titech.ac.jp/~hase/doc/PBMHaptic.pdf [Date of access: January 2004]

HELLER, S. 1995. **Efficient C/C++ programming: smaller, faster, better (Second edition).** New York : AP Professional. 415 p.

HENRICI, P. 1982. **Essentials of numerical analysis, with pocket calculator demonstrations.** New York : John Wiley & Sons, Inc. 409 p.

HERRMANN, H.J. & LUDING, S. 1998. **Modelling granular media on the computer.** Stuttgart : Institute for Computer Applications. 47 p.

HIBBELER, R.C. 1995. **Engineering mechanics: dynamics (Seventh edition).** London : Prentice-Hall International, Inc. 624 p.

HOLMLUND, K. 2002. **Physically based modelling for computer graphics – VR, Vis-Sim & games.** [Web:] http://www.cs.umu.se/kurser/TDBD12/HT02/lectures/vrphysics.pdf [Date of access: January 2004]

HOMEIER, H.H.H. 2004. **A modified Newton method with cubic convergence: the multivariate case.** *Journal of Computational and Applied Mathematics*, 169(1): 161-169, Aug.

HOOMANS, B.P.B. 1999. **Granular dynamics of gas-solid two-phase flows.** Twente : Universiteit Twente (Dissertation - D.Phil.). 241 p.

HOROWITZ, B & AFONSO, S.M.B. 2002. **Quadratic programming solver for structural optimisation using SQP algorithm.** *Advances In Engineering Software*, 33: 669-674.

HSIAO, C.H. 2004a. **Haar wavelet approach to linear stiff systems.** *Mathematics and Computers in Simulation*, 64(5): 561-567, Feb.

HSIAO, C.H. 2004b. **Haar wavelet direct method for solving variational problems.** *Mathematics and Computers in Simulation*, 64(5): 569-585, Feb.

HUSTRULID, A.I. 1995. **Parallel implementation of the discrete element method.** [Web:] http://ppl.mines.edu/dempaper/dempaper.html [Date of access: July 1997]

IVANOV, A.P. 1995. **On multiple impact.** *Journal of Applied Mathematics and Mechanics*, 59(6): 887-902.

IVANOV, A.P. 2003. **The stability of mechanical systems with positional non-conservative forces.** *Journal of Applied Mathematics and Mechanics*, 67(5): 625-629.

JÄNICKE, L. & KOST, A. 1996. **Numerical modelling for anisotropic magnetic media including saturation effects.** [Web:] http://www.aet.tu-cottbus.de/personen/jaenicke/abstracts/download/cefc96.ps [Date of access: March 2004]

JING, L. 2003. **A review of techniques, advances and outstanding issues in numerical modelling for rock mechanics and rock engineering.** *International Journal of Rock Mechanics and Mining Sciences*, 40(3): 283-353, Apr.

JOHNSON, L.W., RIESS, R.D. & ARNOLD, J.T. 1993. **Introduction to linear algebra (Third edition).** Reading, Massachusetts: Addison-Wesley Publishing Company. 569 p.

KAWACHI, K., SUZUKI, H. & KIMURA, F. 1997. **Simulation of rigid body motion with impulsive friction force.** [Web:] http://www.dh.aist.go.jp/~kawachi/isatp97/isatp97kawachi.pdf [Date of access: January 2004]

KELLER, H., STOLZ, H., ZIEGLER, A. & BRÄUNL, T. 1998. **Virtual mechanics - simulation and animation of rigid body systems.** [Web:] http://www.hitl.washington.edu/projects/greenspace/pulkka/aero/docu.english.ps [Date of access: January 2004]

KLEIN, J. 2003. **BREVE: A 3D environment for the simulation of decentralized systems and artificial life.** [Web:] http://www.spiderland.org/breveSavers/breve-klein-alife2002.pdf [Date of access: December 2003].

KRAUS, P.R. & KUMAR, V. 1997. **Compliant contact models for rigid body collisions.** [Web:] http://www.cis.upenn.edu/~pkraus/publications/IEEE97.ps [Date of access: January 2004]

KRAUS, P.R., FREDRIKSSON, A. & KUMAR, V. 1997. **Modeling of frictional contacts for dynamic simulation.** [Web:] http://www.cis.upenn.edu/~pkraus/publications/IROS97.ps [Date of access: January 2004]

KRAUS, P.R., KUMAR, V. & DUPONT, P. 1998. **Analysis of frictional contact models for dynamic simulation.** [Web:] http://www.cis.upenn.edu/~pkraus/publications/ICRA98_5.ps [Date of access: January 2004]

KREYSZIG, E. 1993. **Advanced engineering mathematics (Seventh edition).** New York : John Wiley & Sons, Inc. 1397 p.

KRY, P.G. & PAI, D.K. 2002. **Continuous contact simulation for smooth surfaces.** [Web:] http://www.cs.ubc.ca/~pgkry/pubs/tog.pdf [Date of access: January 2004]

LADEVEZE, P., NOUY, A. & LOISEAU, O. 2002. **A multiscale computational approach for contact problems.** *Computer Methods in Applied Mechanics and Engineering*, 191(43): 4869-4891, Sept.

LAKOS, J. 1996. **Large scale C++ software design.** New York :Addison-Wesly. 852 p.

LAURIE, D.P. 1983. **Solution of large systems of linear equations.** (*In* Laurie, D.P., *ed.* Numerical solution of partial differential equations: theory, tools and case studies. Basel : Birkhäuser Verlag. p. 204-224)

LEE, P.U., RUSPINI, D.C. & KHATIB, O. 1994. **Dynamic simulation of interactive robotic environment.** [Web:] http://citeseer.nj.nec.com/cache/papers/cs/4590/http://robotics.stanford.edu/groups/manips/publications/icra94.pdf/lee94dynamic.pdf [Date of access: February 2004]

LIPPMAN, S.B. & LAJOIE, J. 1998. **C++ primer (Third edition).** Reading, Massachusetts : Addison-Wesley Longman, Inc. 1236 p.

LUCAS, B.D. & KANADE, T. 1981. **An iterative image registration technique with an application to stereo vision.** (*In* Proceedings of Imaging Understanding Workshop, p. 121-130)

LUDING, S. & McNAMARA, S. 1998. **How to handle the inelastic collapse of a dissipative hard sphere gas with the TC model.** [Web:] http://arxiv.org/PS_cache/cond-mat/pdf/9810/9810009.pdf [Date of access: January 2004]

LUH, W. 2000. **Analytical methods for dynamic simulation of non-penetrating rigid bodies.** [Web:] http://graphics.stanford.edu/courses/cs448b-00-winter/critiques/luh.pdf [Date of access: January 2004]

MAROS, I. 2003. **A generalized dual phase-2 simplex algorithm.** *European Journal of Operational Research*, 149(1): 1-16, Aug.

MARTIN, J. & ODELL, J.J. 1992. **Object-oriented analysis and design.** New Jersey : Prentice-Hall International, Inc. 513 p.

McALLESTER, D. 1996. **The rise of nonlinear mathematical programming.** [Web:] http://www.acm.org/pubs/articles/journals/surveys/1996-28-4es/a68-mcallester/a68-mcallester.ps [Date of access: January 2004]

MILENKOVIC, V.J. & SCHMIDL, H. 2001. **Optimisation-based animation.** [Web:] http://graphics.stanford.edu/courses/cs468-01-fall/Papers/milenkovic_schmidl.pdf [Date of access: February 2003]

MIRTICH, B.V. & CANNY, J. 1994. **Impulse-based dynamic simulation.** [Web:] http://www.cs.berkeley.edu/~jfc/papers/94/ibds94.pdf [Date of access: January 2004]

MIRTICH, B.V. & CANNY, J. 1995. **Impulse-based simulation of rigid bodies.** [Web:] http://www.cs.berkeley.edu/~jfc/papers/95/ibsrb95.pdf [Date of access: January 2004]

MIRTICH, B.V. 1996. **Impulse-based dynamic simulation of rigid body systems.** Berkeley : Department of Computer Science, University of California at Berkeley. (Dissertation - D.Phil.) 246 p.

MIRTICH, B.V. 1998. **Rigid body contact - collision detection to force computation.** [Web:] http://www.merl.com/papers/docs/TR98-01.pdf [Date of access: January 2004]

MIRTICH, B.V. 1999. **Hybrid simulation: combining constraints and impulses.** [Web:] http://graphics.stanford.edu/courses/cs468-03-winter/Papers/hybridImpulse.pdf [Date of access: December 2003]

MIRTICH, B.V. 2000. **Timewarp rigid body simulation.** [Web:] http://vdream.kist.re.kr/~zinook/RigidBody/mirtich.pdf [Date of access: December 2003]

MISHRA, B.K. & RAJAMANI, R.K. 1992. **The discrete element method for the simulation of ball mills.** *Applied mathematical modelling,* 16(11): 598-604, Nov.

MISHRA, B.K. & RAJAMANI, R.K. 1993. **Numerical simulation of charge motion in ball mills – lifter bar effect.** *Minerals and metallurgical processing,* 5: 84-90, May.

MOORE, J.B. & WEISS, H. 1979. **Recursive prediction error methods for adaptive estimation.** [Web:] http://www.syseng.anu.edu.au/ftp/Publications/by_author/John_Moore/JOUR/054.pdf [Date of access: February 2004].

NEETHLING, R.S. 1998. **A numerical algorithm for the simulation of systems of rigid bodies.** Potchefstroom : PU for CHE. (Master's thesis) 84 p.

NIELSEN, K. 1995. **Software development with C++: maximizing reuse with object technology.** New York : AP Professional. 450 p.

NOOR, M.A. 2003. **New extragradient-type methods for general variational inequalities.** *Journal of Mathematical Analysis and Applications*, 277(2): 379-394, Jan.

OLIVER, I. 1993. **Programming classics: implementing the world's best algorithms.** Sydney : Prentice-Hall of Australia Pty. Ltd. 386 p.

OPPE, T.C., JOUBERT, W.D. & KINCAID, D.R. 2002. **NSPCG user's guide version 1.0 - a package for solving large sparse linear systems by various iterative methods.** [Web:] ftp://ftp.ma.utexas.edu/pub/CNA/kincaid/cna216.ps [Date of access: February 2004]

OUYANG, J. & LI, J. 1999. **Particle-motion-resolved discrete model for simulating gas-solid fluidization.** *Chemical Engineering Science*, 54(13-14): 2077 – 2083, Jul.

PANDOLFI, A., KANE, C., MARSDEN, J.E. & ORTIZ, M. 2002. **Time-discretized variational formulation of non-smooth frictional contact.** *International Journal for Numerical Methods in Engineering*, 53: 1801-1829.

PANG, J.S. & TRINKLE, J.C. 1996. **Complementarity formulations and existence of solutions of dynamic multi-rigid-body contact problems with Coulomb friction.** *Mathematical Programming*, 73:199-226, May.

PENG, J., ANITESCU, M. & AKELLA, S. 2004. **Optimal control of multiple robot systems with friction using MPCC.** [Web:] http://www-unix.mcs.anl.gov/~anitescu/PUBLICATIONS/icra04_final.pdf [Date of access: January 2004]

PETTERSSON, S. 2003. **Implementation and evaluation of a polynomial-based division algorithm.** Linköping : Linköpings Universitet (Master's thesis). 89 p.

POLSON, A. (Alexander.Polson@pbmr.co.za) 2004. Resultate. [E-mail to:] Neethling, R.S. (mgirn@puk.ac.za) Nov. 5.

POPOVIC, J. 2001. **Interactive design of rigid-body simulations for computer animation.** Pittsburgh : Carnegie Mellon University (Dissertation - D.Phil.). 86 p.

POPOVIC, J., SEITZ, S.M., ERDMANN, M., POPOVIC, Z. & WITKIN, A. 2000. **Interactive manipulation of rigid body simulations.** [Web:] http://grail.cs.washington.edu/pub/papers/sigg00-rbedit.pdf [Date of access: January 2004]

PRESS, W.H., TEUKOLSKY, S.A., VETTERLING, W.T. & FLANNERY, B.P. 1992. **Numerical recipes in C - the art of scientific computing (Second edition).** Cambridge : Cambridge University Press. 1032 p.

RAGHUNATHAN, A.U. & BIEGLER, L.T. 2003. **Interior point methods for mathematical programs with complementarity constraints (MPCCs).** [Web:] http://dynopt.cheme.cmu.edu/papers/preprint/paper23.pdf [Date of access: February 2004]

REDDY, J.N. 1993. **An introduction to the finite element method (Second edition).** Singapore : McGraw-Hill Book Company. 684 p.

REDON, S., KHEDDAR, A. & COQUILLART, S. 2002. **Gauss' least constraints principle and rigid body simulations.** [Web:] http://www-rocq.inria.fr/~redon/papers/icra2002.pdf [Date of access: January 2004]

RENOUF, M., DUBOIS, F. & ALART, P. 2004. **A parallel version of the non smooth contact dynamics algorithm applied to the simulation of granular media.** *Journal of Computational and Applied Mathematics*, 168 (1-2): 375-382, Jul.

RORRES, C. & ANTON, H. 1979. **Applications of linear algebra.** New York : John Wiley & Sons, Inc. 295 p.

RUSPINI, D.C. & KHATIB, O. 1997. **Collision or contact models for the dynamic simulation of complex environments.** [Web:] http://robotics.stanford.edu/groups/manips/publications/files/iros97b.pdf [Date of access: January 2004]

SALAS, S.L. 1995. **Salas & Hille's calculus: one and several variables/revised by Garret J. Etgen (Seventh edition).** New York : John Wiley & Sons, Inc. 1369 p.

SAUER, J. & SCHÖMER, E. 1998. **A constraint-based approach to rigid body dynamics for virtual reality applications.** [Web:] http://www.mpi-sb.mpg.de/~schoemer/publications/VRST98.pdf [Date of access: January 2004]

SAUER, J., SCHÖMER, E. & LENNERZ, C. 1998. **Real-time rigid body simulations of some 'classical mechanics toys'.** [Web:] http://www.mpi-sb.mpg.de/~schoemer/publications/ESS98.pdf [Date of access: January 2004]

SCHMIDL, H. & MILENKOVIC, V.J. 2003. **A fast impulsive contact suite for rigid body simulation.** [Web:] http://www.cs.unc.edu/~schmidl/papers/fics-tvcg.pdf. [Date of access: October 2003]

SHIN, J.Y., JEON, Y.J., MAENG, D.J., KIM, J.S. & ROC, S.T. 2002. **Analysis of the dynamic characteristics of a combined-cycle power plant.** *Energy*, 27(12): 1085-1098, Dec.

SCHÄFER, B.C., QUIGLEY, S.F. & CHAN, A.H.C. 2004. **Acceleration of the Discrete Element Method (DEM) on a reconfigurable co-processor.** *Computers & Structures*, 82(20-21): 1707-1718, Aug.

SONG, P., KRAUS, P. & KUMAR, V. 1999. **Analysis of rigid body dynamic models for simulation of systems with frictional contacts.** [Web:] http://www.cis.upenn.edu/~pkraus/publications/jam99.ps [Date of access: January 2004]

STEWART, D.E. & TRINKLE, J.C. 1995. **Dynamics, friction and complementarity problems.** [Web:] http://citeseer.nj.nec.com/cache/papers/cs/2516/http://www.cs.tamu.edu/faculty/trink/Papers/iccp95StewTrink.pdf/dynamics-friction-and-complementarity.pdf [Date of access: January 2004]

STEWART, D.E. & TRINKLE, J.C. 2002. **An implicit time-stepping scheme for rigid body dynamics with coulomb friction.** [Web:] http://www.cs.rpi.edu/~trink/Papers/STicra.pdf [Date of access: January 2004]

STEWART, J. 1998a. **Calculus: concepts and contexts, single variable.** New York : Brooks/Cole Publishing Company. 762 p.

STEWART, J. 1998b. **Multivariable calculus: concepts and contexts.** New York : Brooks/Cole Publishing Company. 285 p.

STOECKER, W.F. 1989. **Design of thermal systems (Third edition).** Singapore : McGraw-Hill, Inc. 565 p.

STRONGE, W.J. 2003. **Chain reaction from impact on aggregate of elasto-plastic 'rigid' bodies.** *International Journal of Impact Engineering*, 28(3): 291-302, Mar.

STROUSTRUP, B. 2000. **The C++ programming language (Special edition).** Reading, Massachusetts : Addison Wesley. 1040 p.

THOMPSON, D. 1995. **The concise Oxford dictionary.** Oxford: Clarendon Press. 1672 p.

TRINKLE, J.C. 2003. **Formulation of multibody dynamics as complementarity problems.** [Web:] http://www.cs.rpi.edu/~trink/Papers/Tdetc.pdf [Date of access: February 2004]

TRINKLE, J.C., TZITZOURIS, J.A. & PANG, J.S. 2001. **Dynamic multi-rigid-body systems with concurrent distributed contacts: theory and examples.** [Web:] http://www.cs.rpi.edu/~trink/Papers/TTPrs.pdf [Date of access: December 2003]

TZITZOURIS, J.A. 2001. **Numerical resolution of frictional multi-rigid-body systems via fully implicit time-stepping and nonlinear complementarity.** Baltimore : Johns Hopkins University. (Dissertation - D.Phil.) 171 p.

VAN HEERDEN, E. 2000. **Coding standards and practices for Flownet Object Oriented redevelopment.** Potchefstroom : M-Tech Industrial. 81 p. (RP PBMR-0023)

WALL, G. 1991. **On the optimization of refrigeration machinery.** *International Journal of Refrigeration*, 14:336-340.

WOODHOUSE, N.M.J. 1987. **Introduction to analytical dynamics.** Oxford : Clarendon Press. 169 p.

XIU, N., ZHANG, J. & NOOR, M.A. 2001. **Tangent projection equations and general variational inequalities.** *Journal of Computational and Applied Mathematics*, 152(1-2): 559-585, Mar.

YANG, L.T. 2003. **Accuracy of preconditioned CG-type methods for least squares problems.** *Computers & Mathematics with Applications*, 45(1-3): 77-96, Jan.

YUN, J.H. & KIM, S.W. 2004. **Convergence of two-stage iterative methods using incomplete factorization.** *Journal of Computational and Applied Mathematics*, 166(2): 565-580, Apr.

ZHANG, J. 2002. **A sparse approximate inverse preconditioner for parallel preconditioning of general sparse matrices.** *Applied Mathematics and Computation*, 130(1): 63-85, Jul.

ZHOU, H., FLAMANT, G., GAUTHIER, D. & LU, J. 2002. **Lagrangian approach for simulating the gas-particle flow structure in a circulating fluidized bed riser.** *International Journal of Multiphase Flow*, 28(11): 1801-1821, Nov.

Appendix A
Example of the Setup of the Impulse Calculation Matrices

No general form exists for the matrix used in the algorithm to calculate normal and tangential impulses and post-collision velocities, however simple the concept may be. The best, and probably the only, way to illustrate what it looks like is through an example. Figure A.1 will be used as visual aid to the example.

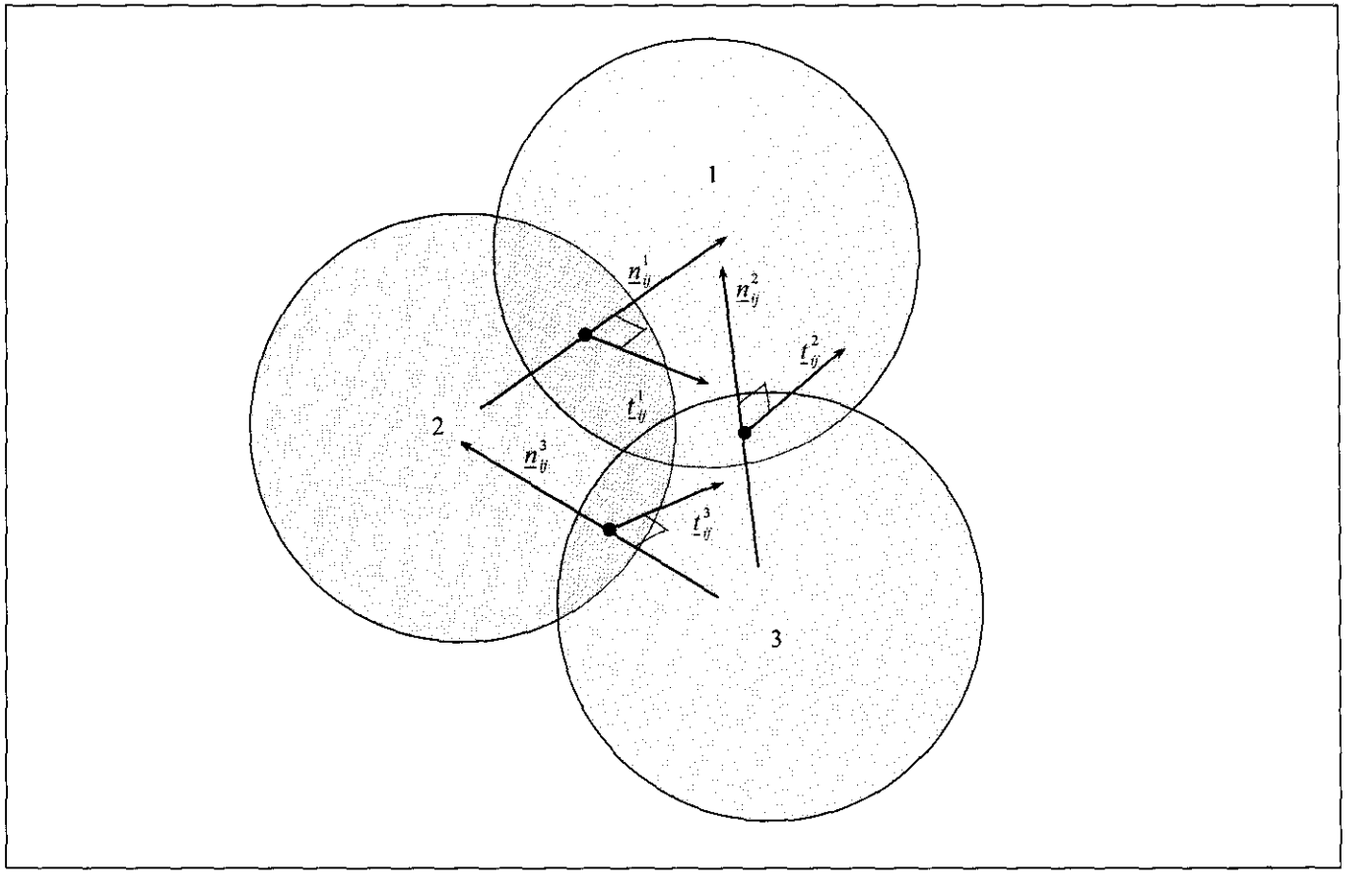


Figure A.1 Three Spheres in Cluster Collision.

In the system shown in Figure A.1 there are 3 spheres present. There are also, at this particular time step, 3 concurrent contacts/collisions in the system. Each sphere i (1 to 3) had a pre-collision velocity of $\underline{\dot{x}}_i^- = (\dot{x}_{i,x}^-, \dot{x}_{i,y}^-, \dot{x}_{i,z}^-)$ and will have a post-collision velocity of $\underline{\dot{x}}_i^+ = (\dot{x}_{i,x}^+, \dot{x}_{i,y}^+, \dot{x}_{i,z}^+)$. The matrix setup for both the linear and the non-linear solution method is illustrated by virtue of the ternary configuration depicted in Figure A.1, first with the linear and then the non-linear approach.

A.1 LINEAR APPROACH

Grouping all related coefficients in the equations in Figure 4.9, and assuming the tangential unit vectors remain fixed, yields a composite matrix of the form shown in equation (A.1). A composite vector consisting of original momentum terms, $\underline{F}_i$, and prescribed post-collision relative velocity terms, $\underline{F}_j$, will be the augmenting vector for this problem and the matrix-vector equation thus obtained can be directly solved for the variables, i.e. $\underline{X}_i$ and $\underline{X}_j$, respectively representing post collision velocities and normal and tangential impulses. Some iteration might be required, removing contacts with negative impulses and substituting tangential slip condition equations where tangential impulses exceed the friction factor limits, and then recalculating until no more anomalous contacts are found. *Note that all superscripts refer to contact numbers and that nothing is raised to any power anywhere!!!*

$$\left[\begin{array}{c|c} M_{ii} & M_{ij} \\ \hline M_{ji} & M_{jj} \end{array}\right]\left\{\begin{array}{c} \underline{X}_i \\ \hline \underline{X}_j \end{array}\right\} = \left\{\begin{array}{c} \underline{F}_i \\ \hline \underline{F}_j \end{array}\right\} \tag{A.1}$$

$$\underline{X}_i = \left[\dot{x}_{1,x}^+ \;\; \dot{x}_{1,y}^+ \;\; \dot{x}_{1,z}^+ \;\; \theta_{1,x}^+ \;\; \theta_{1,y}^+ \;\; \theta_{1,z}^+ \;\; \dot{x}_{2,x}^+ \;\; \dot{x}_{2,y}^+ \;\; \dot{x}_{2,z}^+ \;\; \theta_{2,x}^+ \;\; \theta_{2,y}^+ \;\; \theta_{2,z}^+ \;\; \dot{x}_{3,x}^+ \;\; \dot{x}_{3,y}^+ \;\; \dot{x}_{3,z}^+ \;\; \theta_{3,x}^+ \;\; \theta_{3,y}^+ \;\; \theta_{3,z}^+\right]^T \tag{A.2}$$

$$\underline{X}_j = \left[\lambda_n^1 \;\; \lambda_t^1 \;\; \lambda_n^2 \;\; \lambda_t^2 \;\; \lambda_n^3 \;\; \lambda_t^3\right]^T \tag{A.3}$$

$$\underline{F}_i = \left[m_1\dot{x}_{1,x}^- \;\; m_1\dot{x}_{1,y}^- \;\; m_1\dot{x}_{1,z}^- \;\; I_1\theta_{1,x}^- \;\; I_1\theta_{1,y}^- \;\; I_1\theta_{1,z}^- \;\; m_2\dot{x}_{2,x}^- \;\; m_2\dot{x}_{2,y}^- \;\; m_2\dot{x}_{2,z}^- \;\; I_2\theta_{2,x}^- \;\; I_2\theta_{2,y}^- \;\; I_2\theta_{2,z}^- \;\; m_3\dot{x}_{3,x}^- \;\; m_3\dot{x}_{3,y}^- \;\; m_3\dot{x}_{3,z}^- \;\; I_3\theta_{3,x}^- \;\; I_3\theta_{3,y}^- \;\; I_3\theta_{3,z}^-\right]^T \tag{A.4}$$

$$\underline{F}_j = \left[-\varepsilon_n^1\underline{\dot{x}}_x^{1,-}\bullet\underline{n}^1 \;\; -\varepsilon_t^1\underline{\dot{x}}_x^{1,-}\bullet\underline{t}^1 \;\; -\varepsilon_n^2\underline{\dot{x}}_x^{2,-}\bullet\underline{n}^2 \;\; -\varepsilon_t^2\underline{\dot{x}}_x^{2,-}\bullet\underline{t}^2 \;\; -\varepsilon_n^3\underline{\dot{x}}_x^{3,-}\bullet\underline{n}^3 \;\; -\varepsilon_t^3\underline{\dot{x}}_x^{3,-}\bullet\underline{t}^3\right]^T \tag{A.5}$$

In both the linear and non-linear solution methods, the first sub-matrix, M_{ii} – seen in equation (A.6), looks the same. However, in the linear case it represents all the linear and angular inertia for each of the three rigid bodies in the system, whilst in the non-linear case it represents the various Jacobian coefficients, i.e. derivatives, relevant to the linear and angular momentum terms, which incidentally look the same as the inertia terms in the linear approach, due to their linear nature.

$$M_{ii} = \begin{bmatrix}
m_1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & m_1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & m_1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & I_1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & I_1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & I_1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & m_2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & m_2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & m_2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & I_2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & I_2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & I_2 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & m_3 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & m_3 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & m_3 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & I_3 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & I_3 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & I_3
\end{bmatrix} \tag{A.6}$$

The second sub-matrix, M_{ij}, represents all the linear and angular impulses acting upon each of the three rigid bodies in the system, and can be seen in equation (A.7).

$$M_{ij} = \begin{bmatrix} -n_x^1 & -t_x^1 & -n_x^2 & -t_x^2 & 0 & 0 \\ -n_y^1 & -t_y^1 & -n_y^2 & -t_y^2 & 0 & 0 \\ -n_z^1 & -t_z^1 & -n_z^2 & -t_z^2 & 0 & 0 \\ 0 & r_1 o_x^1 & 0 & r_1 o_x^2 & 0 & 0 \\ 0 & r_1 o_y^1 & 0 & r_1 o_y^2 & 0 & 0 \\ 0 & r_1 o_z^1 & 0 & r_1 o_z^2 & 0 & 0 \\ n_x^1 & t_x^1 & 0 & 0 & -n_x^3 & -t_x^3 \\ n_y^1 & t_y^1 & 0 & 0 & -n_y^3 & -t_y^3 \\ n_z^1 & t_z^1 & 0 & 0 & -n_z^3 & -t_z^3 \\ 0 & r_2 o_x^1 & 0 & 0 & 0 & r_2 o_x^3 \\ 0 & r_2 o_y^1 & 0 & 0 & 0 & r_2 o_y^3 \\ 0 & r_2 o_z^1 & 0 & 0 & 0 & r_2 o_z^3 \\ 0 & 0 & n_x^2 & t_x^2 & n_x^3 & t_x^3 \\ 0 & 0 & n_y^2 & t_y^2 & n_y^3 & t_y^3 \\ 0 & 0 & n_z^2 & t_z^2 & n_z^3 & t_z^3 \\ 0 & 0 & 0 & r_3 o_x^2 & 0 & r_3 o_x^3 \\ 0 & 0 & 0 & r_3 o_y^2 & 0 & r_3 o_y^3 \\ 0 & 0 & 0 & r_3 o_z^2 & 0 & r_3 o_z^3 \end{bmatrix} \quad \text{(A.7)}$$

The third sub-matrix, M_{ji}, represents the various coefficients relevant to all the velocity components for the one normal and one tangential Newton collision equations for each contact, as seen in equation (A.8).

$$M_{ji} = \begin{bmatrix} n_x^1 & n_y^1 & n_z^1 & 0 & 0 & 0 & -n_x^1 & -n_y^1 & -n_z^1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ t_x^1 & t_y^1 & t_z^1 & -r_1o_x^1 & -r_1o_y^1 & -r_1o_z^1 & -t_x^1 & -t_y^1 & -t_z^1 & -r_2o_x^1 & -r_2o_y^1 & -r_2o_z^1 & 0 & 0 & 0 & 0 & 0 & 0 \\ n_x^2 & n_y^2 & n_z^2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -n_x^2 & -n_y^2 & -n_z^2 & 0 & 0 & 0 \\ t_x^2 & t_y^2 & t_z^2 & -r_1o_x^2 & -r_1o_y^2 & -r_1o_z^2 & 0 & 0 & 0 & 0 & 0 & 0 & -t_x^2 & -t_y^2 & -t_z^2 & -r_3o_x^2 & -r_3o_y^2 & -r_3o_z^2 \\ 0 & 0 & 0 & 0 & 0 & 0 & n_x^3 & n_y^3 & n_z^3 & 0 & 0 & 0 & -n_x^3 & -n_y^3 & -n_z^3 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & t_x^3 & t_y^3 & t_z^3 & -r_2o_x^3 & -r_2o_y^3 & -r_2o_z^3 & -t_x^3 & -t_y^3 & -t_z^3 & -r_3o_x^3 & -r_3o_y^3 & -r_3o_z^3 \end{bmatrix} \tag{A.8}$$

The last sub matrix, M_{jj} – shown in equation (A.9), represents all the coefficients related to the tangential and normal impulses.

$$M_{jj} = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \tag{A.9}$$

The empty matrix above is no mistake, since the basic/original form of the M_{jj} matrix for the linear approach is such, as all the collision equations have no explicit impulse terms. Do take note, however, that the situation changes slightly as soon as slip is detected somewhere in the system, since the tangential collision equation is then replaced by the direct relationship between normal and tangential impulse magnitudes via the dynamic friction factor as shown in equation (3.24).

For argument's sake, assume that contact 2 should be slipping first (determined by calculating the current tangential vs. normal impulse ratio, which should be the maximum of the ratios for all contacts *and* larger than the static friction coefficient, μ_{stat}, at that particular contact): then the M_{ji} and M_{jj} matrices and $\underline{F}_j$ vector would have to change as shown in equations (A.10), (A.11) and (A.12).

$$M_{ji} = \begin{bmatrix} n_x^1 & n_y^1 & n_z^1 & 0 & 0 & 0 & -n_x^1 & -n_y^1 & -n_z^1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ t_x^1 & t_y^1 & t_z^1 & -r_1 o_x^1 & -r_1 o_y^1 & -r_1 o_z^1 & -t_x^1 & -t_y^1 & -t_z^1 & -r_2 o_x^1 & -r_2 o_y^1 & -r_2 o_z^1 & 0 & 0 & 0 & 0 & 0 & 0 \\ n_x^2 & n_y^2 & n_z^2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -n_x^2 & -n_y^2 & -n_z^2 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & n_x^3 & n_y^3 & n_z^3 & 0 & 0 & 0 & -n_x^3 & -n_y^3 & -n_z^3 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & t_x^3 & t_y^3 & t_z^3 & -r_2 o_x^3 & -r_2 o_y^3 & -r_2 o_z^3 & -t_x^3 & -t_y^3 & -t_z^3 & -r_3 o_x^3 & -r_3 o_y^3 & -r_3 o_z^3 \end{bmatrix} \tag{A.10}$$

$$M_{jj} = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ -\mu_{dyn}^2 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \tag{A.11}$$

$$\underline{F}_j = \left[-\varepsilon_n^1 \dot{\underline{x}}_x^{1,-} \bullet \underline{n}^1 \;\; -\varepsilon_t^1 \dot{\underline{x}}_x^{1,-} \bullet \underline{t}^1 \;\; -\varepsilon_n^2 \dot{\underline{x}}_x^{2,-} \bullet \underline{n}^2 \;\; 0 \;\; -\varepsilon_n^3 \dot{\underline{x}}_x^{3,-} \bullet \underline{n}^3 \;\; -\varepsilon_t^3 \dot{\underline{x}}_x^{3,-} \bullet \underline{t}^3 \right]^T \tag{A.12}$$

The matrix-vector equation can then be solved again and checked for any more anomalous normal or tangential impulses, and if found, the tangential impulse equation or equations need to change again where necessary, until no more anomalous impulses are present, at which stage it is assumed that the solution has converged.

A.2 NON-LINEAR APPROACH

Part of the reason for this study is the problem that the chosen tangential unit vector might not be exactly correct in which case some form of iterative procedure needs to be employed to implicitly determine the correct values of the Cartesian components. Moreover, some of the variables, such as the tangential impulse and the tangential unit vector to be solved for, occur in product form, thus necessitating the use of a non-linear solution method. Enter the Newton-Raphson-iteration based algorithm suggested and developed in the course of this study. Grouping all the related coefficients in equations (4.19) through (4.26) in a fashion similar to the linear approach, a general composite Jacobian matrix as seen in equation (A.13) can be constituted.

$$\left[\begin{array}{c|c} M_{ii} & M_{ij} \\ \hline M_{ji} & M_{jj} \end{array}\right]\left\{\begin{array}{c} \underline{\Delta X}_i \\ \hline \underline{\Delta X}_j \end{array}\right\} = \left\{\begin{array}{c} \underline{F}_i \\ \hline \underline{F}_j \end{array}\right\} \tag{A.13}$$

The starting values (guess values) to be used for the unknowns in the Newton-Raphson iteration is obtained by first running the linear approach solution once (without doing anomalous impulse checking) and then using only the velocity field thus obtained, setting all impulse starting values to zero and using the tangential unit vectors determined from combined average angular and linear velocities at each contact for the whole contacting cluster, i.e. the fixed tangential unit vectors used in the linear approach, each calculated using the general equation (3.14).

The sub-vectors seen on the left hand side, to be solved for using some form of matrix solution technique, represent the Newton-Raphson adjustors for the velocities, $\underline{\Delta X}_i$, and the two impulse and three tangential components for each contact, $\underline{\Delta X}_j$, seen in equations (A.14) and (A.15).

$$\underline{\Delta X}_i = \left[\Delta\dot{x}^+_{1,x}\ \Delta\dot{x}^+_{1,y}\ \Delta\dot{x}^+_{1,z}\ \Delta\theta^+_{1,x}\ \Delta\theta^+_{1,y}\ \Delta\theta^+_{1,z}\ \Delta\dot{x}^+_{2,x}\ \Delta\dot{x}^+_{2,y}\ \Delta\dot{x}^+_{2,z}\ \Delta\theta^+_{2,x}\ \Delta\theta^+_{2,y}\ \Delta\theta^+_{2,z}\ \Delta\dot{x}^+_{3,x}\ \Delta\dot{x}^+_{3,y}\ \Delta\dot{x}^+_{3,z}\ \Delta\theta^+_{3,x}\ \Delta\theta^+_{3,y}\ \Delta\theta^+_{3,z}\right]^T \tag{A.14}$$

$$\underline{\Delta X}_j = \left[\Delta\lambda^1_n\ \Delta\lambda^1_t\ \Delta t^1_x\ \Delta t^1_y\ \Delta t^1_z\ \Delta\lambda^2_n\ \Delta\lambda^2_t\ \Delta t^2_x\ \Delta t^2_y\ \Delta t^2_z\ \Delta\lambda^3_n\ \Delta\lambda^3_t\ \Delta t^3_x\ \Delta t^3_y\ \Delta\lambda^3_t\right]^T \tag{A.15}$$

The sub-vectors seen on the right hand side in equation (A.13) here represent the root-equations for the momentum terms, $\underline{F}_i$, and three collision equations and two tangential unit vector requirements initially imposed for each contact, $\underline{F}_j$, as shown in equations (A.16) and (A.17).

$$\underline{F}_i = \begin{bmatrix} m_1(\dot{x}_{1,x}^- - \dot{x}_{1,x}^+) + (\lambda_n^1 n_x^1 + \lambda_t^1 t_x^1) + (\lambda_n^2 n_x^2 + \lambda_t^2 t_x^2) \\ m_1(\dot{x}_{1,y}^- - \dot{x}_{1,y}^+) + (\lambda_n^1 n_y^1 + \lambda_t^1 t_y^1) + (\lambda_n^2 n_y^2 + \lambda_t^2 t_y^2) \\ m_1(\dot{x}_{1,z}^- - \dot{x}_{1,z}^+) + (\lambda_n^1 n_z^1 + \lambda_t^1 t_z^1) + (\lambda_n^2 n_z^2 + \lambda_t^2 t_z^2) \\ I_1(\dot{\theta}_{1,x}^- - \dot{\theta}_{1,x}^+) - r_1\lambda_t^1 o_x^1 - r_1\lambda_t^2 o_x^2 \\ I_1(\dot{\theta}_{1,y}^- - \dot{\theta}_{1,y}^+) - r_1\lambda_t^1 o_y^1 - r_1\lambda_t^2 o_y^2 \\ I_1(\dot{\theta}_{1,z}^- - \dot{\theta}_{1,z}^+) - r_1\lambda_t^1 o_z^1 - r_1\lambda_t^2 o_z^2 \\ m_2(\dot{x}_{2,x}^- - \dot{x}_{2,x}^+) - (\lambda_n^1 n_x^1 + \lambda_t^1 t_x^1) + (\lambda_n^3 n_x^3 + \lambda_t^3 t_x^3) \\ m_2(\dot{x}_{2,y}^- - \dot{x}_{2,y}^+) - (\lambda_n^1 n_y^1 + \lambda_t^1 t_y^1) + (\lambda_n^3 n_y^3 + \lambda_t^3 t_y^3) \\ m_2(\dot{x}_{2,z}^- - \dot{x}_{2,z}^+) - (\lambda_n^1 n_z^1 + \lambda_t^1 t_z^1) + (\lambda_n^3 n_z^3 + \lambda_t^3 t_z^3) \\ I_2(\dot{\theta}_{2,x}^- - \dot{\theta}_{2,x}^+) - r_2\lambda_t^1 o_x^1 - r_2\lambda_t^3 o_x^3 \\ I_2(\dot{\theta}_{2,y}^- - \dot{\theta}_{2,y}^+) - r_2\lambda_t^1 o_y^1 - r_2\lambda_t^3 o_y^3 \\ I_2(\dot{\theta}_{2,z}^- - \dot{\theta}_{2,z}^+) - r_2\lambda_t^1 o_z^1 - r_2\lambda_t^3 o_z^3 \\ m_3(\dot{x}_{3,x}^- - \dot{x}_{3,x}^+) - (\lambda_n^2 n_x^2 + \lambda_t^2 t_x^2) - (\lambda_n^3 n_x^3 + \lambda_t^3 t_x^3) \\ m_3(\dot{x}_{3,y}^- - \dot{x}_{3,y}^+) - (\lambda_n^2 n_y^2 + \lambda_t^2 t_y^2) - (\lambda_n^3 n_y^3 + \lambda_t^3 t_y^3) \\ m_3(\dot{x}_{3,z}^- - \dot{x}_{3,z}^+) - (\lambda_n^2 n_z^2 + \lambda_t^2 t_z^2) - (\lambda_n^3 n_z^3 + \lambda_t^3 t_z^3) \\ I_3(\dot{\theta}_{3,x}^- - \dot{\theta}_{3,x}^+) - r_3\lambda_t^2 o_x^2 - r_3\lambda_t^3 o_x^3 \\ I_3(\dot{\theta}_{3,y}^- - \dot{\theta}_{3,y}^+) - r_3\lambda_t^2 o_y^2 - r_3\lambda_t^3 o_y^3 \\ I_3(\dot{\theta}_{3,z}^- - \dot{\theta}_{3,z}^+) - r_3\lambda_t^2 o_z^2 - r_3\lambda_t^3 o_z^3 \end{bmatrix}. \tag{A.16}$$

$$\underline{F}_j = \begin{bmatrix} -(\dot{\underline{x}}^{1,+} + \varepsilon_n^1 \dot{\underline{x}}^{1,-}) \bullet \underline{n}^1 \\ -(\dot{\underline{x}}^{1,+} + \varepsilon_t^1 \dot{\underline{x}}^{1,-}) \bullet \underline{t}^1 \\ -\dot{\underline{x}}^{1,+} \bullet \underline{o}^1 \\ -\underline{n}^1 \bullet \underline{t}^1 \\ 1 - \underline{t}^1 \bullet \underline{t}^1 \\ -(\dot{\underline{x}}^{2,+} + \varepsilon_n^2 \dot{\underline{x}}^{2,-}) \bullet \underline{n}^2 \\ -(\dot{\underline{x}}^{2,+} + \varepsilon_t^2 \dot{\underline{x}}^{2,-}) \bullet \underline{t}^2 \\ -\dot{\underline{x}}^{2,+} \bullet \underline{o}^2 \\ -\underline{n}^2 \bullet \underline{t}^2 \\ 1 - \underline{t}^2 \bullet \underline{t}^2 \\ -(\dot{\underline{x}}^{3,+} + \varepsilon_n^3 \dot{\underline{x}}^{3,-}) \bullet \underline{n}^3 \\ -(\dot{\underline{x}}^{3,+} + \varepsilon_t^3 \dot{\underline{x}}^{3,-}) \bullet \underline{t}^3 \\ -\dot{\underline{x}}^{3,+} \bullet \underline{o}^3 \\ -\underline{n}^3 \bullet \underline{t}^3 \\ 1 - \underline{t}^3 \bullet \underline{t}^3 \end{bmatrix} \tag{A.17}$$

As stated earlier, the sub-matrix, M_{ii} – seen in equation (A.6), looks exactly the same as the one for the linear approach, and the reader is thus kindly referred thereto.

The second sub-matrix, M_{ij}, for the non-linear approach represents the various Jacobian coefficients relevant to all the normal and tangential impulses acting upon each of the three rigid bodies in the system and the three Cartesian components of each unknown tangential unit vector and can be seen in equation (A.18).

$$M_{ij} = \begin{bmatrix}
-n_x^1 & -t_x^1 & -\lambda_t^1 & 0 & 0 & -n_x^2 & -t_x^2 & -\lambda_t^2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
-n_y^1 & -t_y^1 & 0 & -\lambda_t^1 & 0 & -n_y^2 & -t_y^2 & 0 & -\lambda_t^2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
-n_z^1 & -t_z^1 & 0 & 0 & -\lambda_t^1 & -n_z^2 & -t_z^2 & 0 & 0 & -\lambda_t^2 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & r_1 o_x^1 & 0 & -r_1\lambda_t^1 n_z^1 & r_1\lambda_t^1 n_y^1 & 0 & r_1 o_x^2 & 0 & -r_1\lambda_t^2 n_z^2 & r_1\lambda_t^2 n_y^2 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & r_1 o_y^1 & r_1\lambda_t^1 n_z^1 & 0 & -r_1\lambda_t^1 n_x^1 & 0 & r_1 o_y^2 & r_1\lambda_t^2 n_z^2 & 0 & -r_1\lambda_t^2 n_x^2 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & r_1 o_z^1 & -r_1\lambda_t^1 n_y^1 & r_1\lambda_t^1 n_x^1 & 0 & 0 & r_1 o_z^2 & -r_1\lambda_t^2 n_y^2 & r_1\lambda_t^2 n_x^2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
n_x^1 & t_x^1 & \lambda_t^1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -n_x^3 & -t_x^3 & -\lambda_t^3 & 0 & 0 & \\
n_y^1 & t_y^1 & 0 & \lambda_t^1 & 0 & 0 & 0 & 0 & 0 & 0 & -n_y^3 & -t_y^3 & 0 & -\lambda_t^3 & 0 & \\
n_z^1 & t_z^1 & 0 & 0 & \lambda_t^1 & 0 & 0 & 0 & 0 & 0 & -n_z^3 & -t_z^3 & 0 & 0 & -\lambda_t^3 & \\
0 & r_2 o_x^1 & 0 & -r_2\lambda_t^1 n_z^1 & r_2\lambda_t^1 n_y^1 & 0 & 0 & 0 & 0 & 0 & 0 & r_2 o_x^3 & 0 & -r_2\lambda_t^3 n_z^3 & r_2\lambda_t^3 n_y^3 & \\
0 & r_2 o_y^1 & r_2\lambda_t^1 n_z^1 & 0 & -r_2\lambda_t^1 n_x^1 & 0 & 0 & 0 & 0 & 0 & 0 & r_2 o_y^3 & r_2\lambda_t^3 n_z^3 & 0 & -r_2\lambda_t^3 n_x^3 & \\
0 & r_2 o_z^1 & -r_2\lambda_t^1 n_y^1 & r_2\lambda_t^1 n_x^1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & r_2 o_z^3 & -r_2\lambda_t^3 n_y^3 & r_2\lambda_t^3 n_x^3 & 0 & \\
0 & 0 & 0 & 0 & 0 & n_x^2 & t_x^2 & \lambda_t^2 & 0 & 0 & n_x^3 & t_x^3 & \lambda_t^3 & 0 & 0 & \\
0 & 0 & 0 & 0 & 0 & n_y^2 & t_y^2 & 0 & \lambda_t^2 & 0 & n_y^3 & t_y^3 & 0 & \lambda_t^3 & 0 & \\
0 & 0 & 0 & 0 & 0 & n_z^2 & t_z^2 & 0 & 0 & \lambda_t^2 & n_z^3 & t_z^3 & 0 & 0 & \lambda_t^3 & \\
0 & 0 & 0 & 0 & 0 & 0 & r_3 o_x^2 & 0 & -r_3\lambda_t^2 n_z^2 & r_3\lambda_t^2 n_y^2 & 0 & r_3 o_x^3 & 0 & -r_3\lambda_t^3 n_z^3 & r_3\lambda_t^3 n_y^3 & \\
0 & 0 & 0 & 0 & 0 & 0 & r_3 o_y^2 & r_3\lambda_t^2 n_z^2 & 0 & -r_3\lambda_t^2 n_x^2 & 0 & r_3 o_y^3 & r_3\lambda_t^3 n_z^3 & 0 & -r_3\lambda_t^3 n_x^3 & \\
0 & 0 & 0 & 0 & 0 & 0 & r_3 o_z^2 & -r_3\lambda_t^2 n_y^2 & r_3\lambda_t^2 n_x^2 & 0 & 0 & r_3 o_z^3 & -r_3\lambda_t^3 n_y^3 & r_3\lambda_t^3 n_x^3 & 0 &
\end{bmatrix} \quad \text{(A.18)}$$

The third sub-matrix, M_{ji}, represents the various Jacobian coefficients relevant to all the velocity components for the one normal and two tangential collision equations of Newton for each contact, as can be seen in equation (A.19).

$$M_{ji} = \begin{bmatrix}
n_x^1 & n_y^1 & n_z^1 & 0 & 0 & 0 & -n_x^1 & -n_y^1 & -n_z^1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
t_x^1 & t_y^1 & t_z^1 & -r_1 o_x^1 & -r_1 o_y^1 & -r_1 o_z^1 & -t_x^1 & -t_y^1 & -t_z^1 & -r_2 o_x^1 & -r_2 o_y^1 & -r_2 o_z^1 & 0 & 0 & 0 & 0 & 0 & 0 \\
o_x^1 & o_y^1 & o_z^1 & r_1 t_x^1 & r_1 t_y^1 & r_1 t_z^1 & -o_x^1 & -o_y^1 & -o_z^1 & r_2 t_x^1 & r_2 t_y^1 & r_2 t_z^1 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
n_x^2 & n_y^2 & n_z^2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -n_x^2 & -n_y^2 & -n_z^2 & 0 & 0 & 0 \\
t_x^2 & t_y^2 & t_z^2 & -r_1 o_x^2 & -r_1 o_y^2 & -r_1 o_z^2 & 0 & 0 & 0 & 0 & 0 & 0 & -t_x^2 & -t_y^2 & -t_z^2 & -r_3 o_x^2 & -r_3 o_y^2 & -r_3 o_z^2 \\
o_x^2 & o_y^2 & o_z^2 & r_1 t_x^2 & r_1 t_y^2 & r_1 t_z^2 & 0 & 0 & 0 & 0 & 0 & 0 & -o_x^2 & -o_y^2 & -o_z^2 & r_3 t_x^2 & r_3 t_y^2 & r_3 t_z^2 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & n_x^3 & n_y^3 & n_z^3 & 0 & 0 & 0 & -n_x^3 & -n_y^3 & -n_z^3 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & t_x^3 & t_y^3 & t_z^3 & -r_2 o_x^3 & -r_2 o_y^3 & -r_2 o_z^3 & -t_x^3 & -t_y^3 & -t_z^3 & -r_3 o_x^3 & -r_3 o_y^3 & -r_3 o_z^3 \\
0 & 0 & 0 & 0 & 0 & 0 & o_x^3 & o_y^3 & o_z^3 & r_2 t_x^3 & r_2 t_y^3 & r_2 t_z^3 & -o_x^3 & -o_y^3 & -o_z^3 & r_3 t_x^3 & r_3 t_y^3 & r_3 t_z^3 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0
\end{bmatrix} \quad \text{(A.19)}$$

For the non-linear approach, the last sub matrix, M_{jj} – shown in equation (A.20), represents all the Jacobian coefficients related to the tangential and normal impulses as well as the three tangential coordinates for each contact.

$$M_{jj} = \begin{bmatrix}
0\,0 & 0 & 0 & 0 & 0\,0 & 0 & 0 & 0 & 0\,0 & 0 & 0 & 0 \\
0\,0 & \dot{x}_x^{1,+} & \dot{x}_y^{1,+} & \dot{x}_z^{1,+} & 0\,0 & 0 & 0 & 0 & 0\,0 & 0 & 0 & 0 \\
0\,0 & \begin{pmatrix} \dot{x}_y^{1,+} n_z^1 \\ -\dot{x}_z^{1,+} n_y^1 \end{pmatrix} & \begin{pmatrix} \dot{x}_z^{1,+} n_x^1 \\ -\dot{x}_x^{1,+} n_z^1 \end{pmatrix} & \begin{pmatrix} \dot{x}_x^{1,+} n_y^1 \\ -\dot{x}_y^{1,+} n_x^1 \end{pmatrix} & 0\,0 & 0 & 0 & 0 & 0\,0 & 0 & 0 & 0 \\
0\,0 & n_x^1 & n_y^1 & n_z^1 & 0\,0 & 0 & 0 & 0 & 0\,0 & 0 & 0 & 0 \\
0\,0 & 2t_x^1 & 2t_y^1 & 2t_z^1 & 0\,0 & 0 & 0 & 0 & 0\,0 & 0 & 0 & 0 \\
0\,0 & 0 & 0 & 0 & 0\,0 & 0 & 0 & 0 & 0\,0 & 0 & 0 & 0 \\
0\,0 & 0 & 0 & 0 & 0\,0 & \dot{x}_x^{2,+} & \dot{x}_y^{2,+} & \dot{x}_z^{2,+} & 0\,0 & 0 & 0 & 0 \\
0\,0 & 0 & 0 & 0 & 0\,0 & \begin{pmatrix} \dot{x}_y^{2,+} n_z^2 \\ -\dot{x}_z^{2,+} n_y^2 \end{pmatrix} & \begin{pmatrix} \dot{x}_z^{2,+} n_x^2 \\ -\dot{x}_x^{2,+} n_z^2 \end{pmatrix} & \begin{pmatrix} \dot{x}_x^{2,+} n_y^2 \\ -\dot{x}_y^{2,+} n_x^2 \end{pmatrix} & 0\,0 & 0 & 0 & 0 \\
0\,0 & 0 & 0 & 0 & 0\,0 & n_x^2 & n_y^2 & n_z^2 & 0\,0 & 0 & 0 & 0 \\
0\,0 & 0 & 0 & 0 & 0\,0 & 2t_x^2 & 2t_y^2 & 2t_z^2 & 0\,0 & 0 & 0 & 0 \\
0\,0 & 0 & 0 & 0 & 0\,0 & 0 & 0 & 0 & 0\,0 & 0 & 0 & 0 \\
0\,0 & 0 & 0 & 0 & 0\,0 & 0 & 0 & 0 & 0\,0 & \dot{x}_x^{3,+} & \dot{x}_y^{3,+} & \dot{x}_z^{3,+} \\
0\,0 & 0 & 0 & 0 & 0\,0 & 0 & 0 & 0 & 0\,0 & \begin{pmatrix} \dot{x}_y^{3,+} n_z^3 \\ -\dot{x}_z^{3,+} n_y^3 \end{pmatrix} & \begin{pmatrix} \dot{x}_z^{3,+} n_x^3 \\ -\dot{x}_x^{3,+} n_z^3 \end{pmatrix} & \begin{pmatrix} \dot{x}_x^{3,+} n_y^3 \\ -\dot{x}_y^{3,+} n_x^3 \end{pmatrix} \\
0\,0 & 0 & 0 & 0 & 0\,0 & 0 & 0 & 0 & 0\,0 & n_x^3 & n_y^3 & n_z^3 \\
0\,0 & 0 & 0 & 0 & 0\,0 & 0 & 0 & 0 & 0\,0 & 2t_x^3 & 2t_y^3 & 2t_z^3
\end{bmatrix} \tag{A.20}$$

Note here, as in the case of the linear approach, there are originally no impulse-related components in the Jacobian entries, which will change as soon as the relevant first tangential collision equation or equations are replaced by the direct relationship as specified by equation (4.18).

Using the same example as in the non-linear case, with contact 2 supposed to slip tangentially, the sub-matrices M_{ji} and M_{jj} and sub-vector $\underline{F}_j$ are to alter as respectively depicted in equations (A.21), (A.22) and (A.23). This alteration only takes place after a sufficient level of convergence had been reached for the non-linear solution iterations with the previous set of equations and subsequent checking of the tangential-to-normal impulse ratios had indicated such a condition. Note that only row seven in each of the matrices and the sub-vector, associated with the first tangential restitution equation for contact 2, changes and the rest remain exactly the same. If another contact is discovered to be exceeding its tangential slip limit too, that contact will need to undergo the same type of change to the row of its first tangential restitution equation.

$$M_{ji} = \begin{bmatrix}
n_x^1 & n_y^1 & n_z^1 & 0 & 0 & 0 & -n_x^1 & -n_y^1 & -n_z^1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
t_x^1 & t_y^1 & t_z^1 & -r_1 o_x^1 & -r_1 o_y^1 & -r_1 o_z^1 & -t_x^1 & -t_y^1 & -t_z^1 & -r_2 o_x^1 & -r_2 o_y^1 & -r_2 o_z^1 & 0 & 0 & 0 & 0 & 0 & 0 \\
o_x^1 & o_y^1 & o_z^1 & r_1 t_x^1 & r_1 t_y^1 & r_1 t_z^1 & -o_x^1 & -o_y^1 & -o_z^1 & r_2 t_x^1 & r_2 t_y^1 & r_2 t_z^1 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
n_x^2 & n_y^2 & n_z^2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -n_x^2 & -n_y^2 & -n_z^2 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
o_x^2 & o_y^2 & o_z^2 & r_1 t_x^2 & r_1 t_y^2 & r_1 t_z^2 & 0 & 0 & 0 & 0 & 0 & 0 & -o_x^2 & -o_y^2 & -o_z^2 & r_3 t_x^2 & r_3 t_y^2 & r_3 t_z^2 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & n_x^3 & n_y^3 & n_z^3 & 0 & 0 & 0 & -n_x^3 & -n_y^3 & -n_z^3 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & t_x^3 & t_y^3 & t_z^3 & -r_2 o_x^3 & -r_2 o_y^3 & -r_2 o_z^3 & -t_x^3 & -t_y^3 & -t_z^3 & -r_3 o_x^3 & -r_3 o_y^3 & -r_3 o_z^3 \\
0 & 0 & 0 & 0 & 0 & 0 & o_x^3 & o_y^3 & o_z^3 & r_2 t_x^3 & r_2 t_y^3 & r_2 t_z^3 & -o_x^3 & -o_y^3 & -o_z^3 & r_3 t_x^3 & r_3 t_y^3 & r_3 t_z^3 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0
\end{bmatrix} \quad \text{(A.21)}$$

$$
M_{jj} = \begin{bmatrix}
0\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0\ 0 & 0 & 0 & 0 \\
0\ 0 & \dot{x}_x^{1,+} & \dot{x}_y^{1,+} & \dot{x}_z^{1,+} & 0 & 0 & 0 & 0 & 0 & 0\ 0 & 0 & 0 & 0 \\
0\ 0 & \begin{pmatrix} \dot{x}_y^{1,+} n_z^1 \\ -\dot{x}_z^{1,+} n_y^1 \end{pmatrix} & \begin{pmatrix} \dot{x}_z^{1,+} n_x^1 \\ -\dot{x}_x^{1,+} n_z^1 \end{pmatrix} & \begin{pmatrix} \dot{x}_x^{1,+} n_y^1 \\ -\dot{x}_y^{1,+} n_x^1 \end{pmatrix} & 0 & 0 & 0 & 0 & 0 & 0\ 0 & 0 & 0 & 0 \\
0\ 0 & n_x^1 & n_y^1 & n_z^1 & 0 & 0 & 0 & 0 & 0 & 0\ 0 & 0 & 0 & 0 \\
0\ 0 & 2t_x^1 & 2t_y^1 & 2t_z^1 & 0 & 0 & 0 & 0 & 0 & 0\ 0 & 0 & 0 & 0 \\
0\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0\ 0 & 0 & 0 & 0 \\
0\ 0 & 0 & 0 & 0 & -\mu_{dyn}^2 & 1 & 0 & 0 & 0 & 0\ 0 & 0 & 0 & 0 \\
0\ 0 & 0 & 0 & 0 & 0 & 0 & \begin{pmatrix} \dot{x}_y^{2,+} n_z^2 \\ -\dot{x}_z^{2,+} n_y^2 \end{pmatrix} & \begin{pmatrix} \dot{x}_z^{2,+} n_x^2 \\ -\dot{x}_x^{2,+} n_z^2 \end{pmatrix} & \begin{pmatrix} \dot{x}_x^{2,+} n_y^2 \\ -\dot{x}_y^{2,+} n_x^2 \end{pmatrix} & 0\ 0 & 0 & 0 & 0 \\
0\ 0 & 0 & 0 & 0 & 0 & 0 & n_x^2 & n_y^2 & n_z^2 & 0\ 0 & 0 & 0 & 0 \\
0\ 0 & 0 & 0 & 0 & 0 & 0 & 2t_x^2 & 2t_y^2 & 2t_z^2 & 0\ 0 & 0 & 0 & 0 \\
0\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0\ 0 & 0 & 0 & 0 \\
0\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0\ 0 & \dot{x}_x^{3,+} & \dot{x}_y^{3,+} & \dot{x}_z^{3,+} \\
0\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0\ 0 & \begin{pmatrix} \dot{x}_y^{3,+} n_z^3 \\ -\dot{x}_z^{3,+} n_y^3 \end{pmatrix} & \begin{pmatrix} \dot{x}_z^{3,+} n_x^3 \\ -\dot{x}_x^{3,+} n_z^3 \end{pmatrix} & \begin{pmatrix} \dot{x}_x^{3,+} n_y^3 \\ -\dot{x}_y^{3,+} n_x^3 \end{pmatrix} \\
0\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0\ 0 & n_x^3 & n_y^3 & n_z^3 \\
0\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0\ 0 & 2t_x^3 & 2t_y^3 & 2t_z^3
\end{bmatrix} \tag{A.22}
$$

$$\underline{F}_j = \begin{bmatrix} -(\dot{\underline{x}}^{1,+} + \varepsilon_n^1 \dot{\underline{x}}^{1,-}) \bullet \underline{n}^1 \\ -(\dot{\underline{x}}^{1,+} + \varepsilon_t^1 \dot{\underline{x}}^{1,-}) \bullet \underline{t}^1 \\ -\dot{\underline{x}}^{1,+} \bullet \underline{o}^1 \\ -\underline{n}^1 \bullet \underline{t}^1 \\ 1 - \underline{t}^1 \bullet \underline{t}^1 \\ -(\dot{\underline{x}}^{2,+} + \varepsilon_n^2 \dot{\underline{x}}^{2,-}) \bullet \underline{n}^2 \\ \mu_{dyn}^2 \lambda_n^2 - \lambda_t^2 \\ -\dot{\underline{x}}^{2,+} \bullet \underline{o}^2 \\ -\underline{n}^2 \bullet \underline{t}^2 \\ 1 - \underline{t}^2 \bullet \underline{t}^2 \\ -(\dot{\underline{x}}^{3,+} + \varepsilon_n^3 \dot{\underline{x}}^{3,-}) \bullet \underline{n}^3 \\ -(\dot{\underline{x}}^{3,+} + \varepsilon_t^3 \dot{\underline{x}}^{3,-}) \bullet \underline{t}^3 \\ -\dot{\underline{x}}^{3,+} \bullet \underline{o}^3 \\ -\underline{n}^3 \bullet \underline{t}^3 \\ 1 - \underline{t}^3 \bullet \underline{t}^3 \end{bmatrix} \tag{A.23}$$

As in the linear approach, the process of non-linear equation solution, anomalous impulse checking and matrix re-constitution (due to contact removals and equation replacements) is repeated until no further anomalies are detected, in which case the solution process can be terminated, since the correct set of variable values is most likely to have been obtained.

Appendix B
Test Case Results

B.1 BINARY TEST RESULTS

The various binary test results as referred to in section 5.1.1.3 can be seen in this section, with the linear and angular velocities shown in columns V_X, V_Y, V_Z and Theta_X, Theta_Y, Theta_Z of the result tables. The normal and tangential impulse magnitudes are in columns Lambda_X and Lambda_Y and the three direction coordinates of the unit tangential vector can further be seen in columns Tangent_X, Tangent_Y and Tangent_Z.

B.1.1 Binary Test 1

Post-Collision Rigid Body Data:

Number	Pos_X	Pos_Y	Pos_Z	V_X	V_Y	V_Z	Omega_X	Omega_Y	Omega_Z
1	1.5000000000	1.5000000000	1.5000000000	-0.9000000000	-0.9000000000	0.5969491347	-28.5000000000	28.5000000000	0.0000000000
2	1.5353553391	1.5353553391	1.5000000000	0.9000000000	0.9000000000	-0.5969491347	-28.5000000000	28.5000000000	0.0000000000

Contact Data:

Number	Active	Slipping	Object1	Type1	Object2	Type2	Normal_X	Normal_Y	Normal_Z	Lambda_N	Tangent_X	Tangent_Y	Tangent_Z	Lambda_T
1	true	true	2	19	1	19	-0.7071067812	-0.7071067812	0.0000000000	1.3805333130	0.0000000000	0.0000000000	-1.0000000000	0.2070799969

Solution Data:
2 iterations, converged
Energy:
Total energy before collision = 1.54134390 Joules
Total energy after collision = 1.22407021 Joules
Energy discrepancy = -20.58422435 %

Figure B.1: Binary Test 1 - Linear Solution.

Post-Collision Rigid Body Data:

Number	Pos_X	Pos_Y	Pos_Z	V_X	V_Y	V_Z	Omega_X	Omega_Y	Omega_Z
1	1.5000000000	1.5000000000	1.5000000000	-0.9000000000	-0.9000000000	0.5969491347	-28.5000000000	28.5000000000	-0.0000000000
2	1.5353553391	1.5353553391	1.5000000000	0.9000000000	0.9000000000	-0.5969491347	-28.5000000000	28.5000000000	0.0000000000

Contact Data:

Number	Active	Slipping	Object1	Type1	Object2	Type2	Normal_X	Normal_Y	Normal_Z	Lambda_N	Tangent_X	Tangent_Y	Tangent_Z	Lambda_T
1	true	true	2	19	1	19	-0.7071067812	-0.7071067812	0.0000000000	1.3805333130	0.0000000000	0.0000000000	-1.0000000000	0.2070799969

Solution Data:
2 iterations, converged
Energy:
Total energy before collision = 1.54134390 Joules
Total energy after collision = 1.22407021 Joules
Energy discrepancy = -20.58422435 %

Figure B.2: Binary Test 1 – Non-Linear Solution.

Number	V_X	V_Y	V_Z	Omega_X	Omega_Y	Omega_Z
1	-9.98715794636e-001	-9.98715794636e-001	5.35824439258e-001	-2.70878829360e+001	2.70878829360e+001	0.00000000000e+000
2	9.98715794636e-001	9.98715794636e-001	-5.35824439258e-001	-2.70878829360e+001	2.70878829360e+001	0.00000000000e+000

Total energy before collision = 1.54134389567e+000
Total energy after collision = 1.36093011036e+000
Energy discrepancy = -1.17049664135e+001 %

Figure B.3: Binary Test 1 – PFC3D (DEM-Approach) Solution.

B.1.2 Binary Test 2

```
Post-Collision Rigid Body Data:
 Number          Pos_X          Pos_Y          Pos_Z            V_X            V_Y            V_Z          Omega_X          Omega_Y          Omega_Z
      1   1.5000000000   1.5000000000   1.5000000000  -1.1808153390  -0.4136135653   0.6344035473  -18.2798226356   31.6615815583  -13.3817589226
      2   1.5433012702   1.5250000000   1.5000000000   1.1808153390   0.4136135653  -0.6344035473  -18.2798226356   31.6615815583  -13.3817589226

Contact Data:
 Number      Active    Slipping     Object1       Type1     Object2       Type2       Normal_X       Normal_Y       Normal_Z       Lambda_N      Tangent_X      Tangent_Y      Tangent_Z      Lambda_T
      1        true        true           2          19           1          19  -0.8660254038  -0.5000000000   0.0000000000   1.3334927811   0.1718618847  -0.2976735161  -0.9390708016   0.2000239172
Solution Data:
  2 iterations, converged
  Energy:
    Total energy before collision =      1.54134390 Joules
    Total energy after collision =      1.20573614 Joules
    Energy discrepancy =    -21.77371031 %
```

Figure B.4: Binary Test 2 - Linear Solution.

```
Post-Collision Rigid Body Data:
 Number          Pos_X          Pos_Y          Pos_Z            V_X            V_Y            V_Z          Omega_X          Omega_Y          Omega_Z
      1   1.5000000000   1.5000000000   1.5000000000  -1.1808153390  -0.4136135653   0.6344035473  -18.2798226356   31.6615815583  -13.3817589226
      2   1.5433012702   1.5250000000   1.5000000000   1.1808153390   0.4136135653  -0.6344035473  -18.2798226356   31.6615815583  -13.3817589226

Contact Data:
 Number      Active    Slipping     Object1       Type1     Object2       Type2       Normal_X       Normal_Y       Normal_Z       Lambda_N      Tangent_X      Tangent_Y      Tangent_Z      Lambda_T
      1        true        true           2          19           1          19  -0.8660254038  -0.5000000000   0.0000000000   1.3334927811   0.1718618847  -0.2976735161  -0.9390708016   0.2000239172
Solution Data:
  2 iterations, converged
  Energy:
    Total energy before collision =      1.54134390 Joules
    Total energy after collision =      1.20573614 Joules
    Energy discrepancy =    -21.77371031 %
```

Figure B.5: Binary Test 2 - Non-Linear Solution.

```
Number              V_X              V_Y              V_Z          Omega_X          Omega_Y          Omega_Z
     1 -1.29414190158e+000 -4.86324302936e-001 5.79585626107e-001 -1.73607844622e+001 3.00695744929e+001 -1.27793524638e+001
     2 1.29414190158e+000 4.86324302936e-001 -5.79585626107e-001 -1.73607844622e+001 3.00695744929e+001 -1.27793524638e+001

Total energy before collision = 1.54134389567e+000
Total energy after collision  = 1.33041411053e+000
Energy discrepancy = -1.36847971261e+001 %
```

Figure B.6: Binary Test 2 – PFC3D (DEM-Approach) Solution.

B.1.3 Binary Test 3

```
Post-Collision Rigid Body Data:
 Number          Pos_X          Pos_Y          Pos_Z            V_X            V_Y            V_Z        Omega_X        Omega_Y        Omega_Z
      1  1.5000000000   1.5000000000   1.5000000000  -1.1955655136   0.2030655136   0.7150000000  -7.3763427854  27.5288860492  -20.1525432638
      2  1.5482962913   1.5129409523   1.5000000000   1.1955655136  -0.2030655136  -0.7150000000  -7.3763427854  27.5288860492  -20.1525432638

Contact Data:
 Number     Active   Slipping    Object1      Type1    Object2      Type2       Normal_X       Normal_Y       Normal_Z       Lambda_N      Tangent_X      Tangent_Y      Tangent_Z      Lambda_T
      1       true       true          2         19          1         19  -0.9659258263  -0.2588190451   0.0000000000   1.1955769198   0.1494292454  -0.5576775358  -0.8164965809   0.1793365380
Solution Data:
  2 iterations, converged
  Energy:
    Total energy before collision =     1.54134390 Joules
    Total energy after collision =     1.17472562 Joules
    Energy discrepancy =   -23.78562500 %
```

Figure B.7: Binary Test 3 - Linear Solution.

```
Post-Collision Rigid Body Data:
 Number          Pos_X          Pos_Y          Pos_Z            V_X            V_Y            V_Z        Omega_X        Omega_Y        Omega_Z
      1  1.5000000000   1.5000000000   1.5000000000  -1.1955655136   0.2030655136   0.7150000000  -7.3763427854  27.5288860492  -20.1525432638
      2  1.5482962913   1.5129409523   1.5000000000   1.1955655136  -0.2030655136  -0.7150000000  -7.3763427854  27.5288860492  -20.1525432638

Contact Data:
 Number     Active   Slipping    Object1      Type1    Object2      Type2       Normal_X       Normal_Y       Normal_Z       Lambda_N      Tangent_X      Tangent_Y      Tangent_Z      Lambda_T
      1       true       true          2         19          1         19  -0.9659258263  -0.2588190451   0.0000000000   1.1955769198   0.1494292454  -0.5576775358  -0.8164965809   0.1793365380
Solution Data:
  2 iterations, converged
  Energy:
    Total energy before collision =     1.54134390 Joules
    Total energy after collision =     1.17472562 Joules
    Energy discrepancy =   -23.78562500 %
```

Figure B.8: Binary Test 3 – Non-Linear Solution.

```
Number              V_X              V_Y              V_Z          Omega_X          Omega_Y          Omega_Z
     1 -1.30139086670e+000 8.27928516529e-002 6.72717575531e-001 -7.00685017276e+000 2.61510684836e+001 -1.89252806734e+001
     2 1.30139086670e+000 -8.27928516529e-002 -6.72717575531e-001 -7.00685017276e+000 2.61510684836e+001 -1.89252806734e+001

Total energy before collision = 1.54134389567e+000
Total energy after collision  = 1.24633423240e+000
Energy discrepancy = -1.91397691389e+001 %
```

Figure B.9: Binary Test 3 – PFC3D (DEM-Approach) Solution.

B.1.4 Binary Test 4

```
Post-Collision Rigid Body Data:
 Number          Pos_X          Pos_Y          Pos_Z            V_X            V_Y            V_Z          Omega_X          Omega_Y          Omega_Z
      1   1.5000000000   1.5000000000   1.5000000000  -0.9000000000   0.7984745674   0.7984745674   0.0000000000  20.1525432638 -20.1525432638
      2   1.5500000000   1.5000000000   1.5000000000   0.9000000000  -0.7984745674  -0.7984745674   0.0000000000  20.1525432638 -20.1525432638

Contact Data:
 Number     Active   Slipping    Object1      Type1    Object2      Type2       Normal_X       Normal_Y       Normal_Z       Lambda_N      Tangent_X      Tangent_Y      Tangent_Z       Lambda_T
      1       true       true          2         19          1         19  -1.0000000000   0.0000000000   0.0000000000   0.9761844673   0.0000000000  -0.7071067812  -0.7071067812   0.1464276701
Solution Data:
  2 iterations, converged
  Energy:
    Total energy before collision =     1.54134390 Joules
    Total energy after collision =     1.17562706 Joules
    Energy discrepancy =   -23.72714102 %
```

Figure B.10: Binary Test 4 - Linear Solution.

```
Post-Collision Rigid Body Data:
 Number          Pos_X          Pos_Y          Pos_Z            V_X            V_Y            V_Z          Omega_X          Omega_Y          Omega_Z
      1   1.5000000000   1.5000000000   1.5000000000  -0.9000000000   0.7984745674   0.7984745674   0.0000000000  20.1525432638 -20.1525432638
      2   1.5500000000   1.5000000000   1.5000000000   0.9000000000  -0.7984745674  -0.7984745674   0.0000000000  20.1525432638 -20.1525432638

Contact Data:
 Number     Active   Slipping    Object1      Type1    Object2      Type2       Normal_X       Normal_Y       Normal_Z       Lambda_N      Tangent_X      Tangent_Y      Tangent_Z       Lambda_T
      1       true       true          2         19          1         19  -1.0000000000   0.0000000000   0.0000000000   0.9761844673   0.0000000000  -0.7071067812  -0.7071067812   0.1464276701
Solution Data:
  2 iterations, converged
  Energy:
    Total energy before collision =     1.54134390 Joules
    Total energy after collision =     1.17562706 Joules
    Energy discrepancy =   -23.72714102 %
```

Figure B.11: Binary Test 4 – Non-Linear Solution.

```
Number                V_X                V_Y                V_Z            Omega_X            Omega_Y            Omega_Z
     1 -9.95841862549e-001 7.69250019517e-001 7.69250019517e-001 0.00000000000e+000 1.91298959330e+001 -1.91298959330e+001
     2 9.95841862549e-001 -7.69250019517e-001 -7.69250019517e-001 0.00000000000e+000 1.91298959330e+001 -1.91298959330e+001

Total energy before collision = 1.54134389567e+000
Total energy after collision  = 1.21158295607e+000
Energy discrepancy = -2.13943780179e+001 %
```

Figure B.12: Binary Test 4 – PFC3D (DEM-Approach) Solution.

B.1.5 Binary Test 5

```
Post-Collision Rigid Body Data:
 Number          Pos_X          Pos_Y          Pos_Z              V_X              V_Y              V_Z          Omega_X          Omega_Y          Omega_Z
      1  1.5000000000   1.5000000000   1.5000000000   0.5969491347  -0.9000000000  -0.9000000000   0.0000000000 -28.5000000000  28.5000000000
      2  1.5000000000   1.5353553391   1.5353553391  -0.5969491347   0.9000000000   0.9000000000   0.0000000000 -28.5000000000  28.5000000000

Contact Data:
 Number     Active   Slipping    Object1      Type1    Object2      Type2       Normal_X       Normal_Y       Normal_Z       Lambda_N      Tangent_X      Tangent_Y      Tangent_Z      Lambda_T
      1       true       true          2         19          1         19   0.0000000000  -0.7071067812  -0.7071067812   1.3805333130  -1.0000000000   0.0000000000   0.0000000000   0.2070799969
Solution Data:
  2 iterations, converged
  Energy:
    Total energy before collision =     1.54134390 Joules
    Total energy after collision =     1.22407021 Joules
    Energy discrepancy =   -20.58422435 %
```

Figure B.13: Binary Test 5- Linear Solution.

```
Post-Collision Rigid Body Data:
 Number          Pos_X          Pos_Y          Pos_Z              V_X              V_Y              V_Z          Omega_X          Omega_Y          Omega_Z
      1  1.5000000000   1.5000000000   1.5000000000   0.5969491347  -0.9000000000  -0.9000000000   0.0000000000 -28.5000000000  28.5000000000
      2  1.5000000000   1.5353553391   1.5353553391  -0.5969491347   0.9000000000   0.9000000000   0.0000000000 -28.5000000000  28.5000000000

Contact Data:
 Number     Active   Slipping    Object1      Type1    Object2      Type2       Normal_X       Normal_Y       Normal_Z       Lambda_N      Tangent_X      Tangent_Y      Tangent_Z      Lambda_T
      1       true       true          2         19          1         19   0.0000000000  -0.7071067812  -0.7071067812   1.3805333130  -1.0000000000   0.0000000000   0.0000000000   0.2070799969
Solution Data:
  2 iterations, converged
  Energy:
    Total energy before collision =     1.54134390 Joules
    Total energy after collision =     1.22407021 Joules
    Energy discrepancy =   -20.58422435 %
```

Figure B.14: Binary Test 5– Non-Linear Solution.

```
Number                V_X                V_Y                V_Z            Omega_X            Omega_Y            Omega_Z
     1 5.35824439258e-001 -9.98715794636e-001 -9.98715794636e-001 0.00000000000e+000 -2.70878829360e+001 2.70878829360e+001
     2 -5.35824439258e-001 9.98715794636e-001 9.98715794636e-001 0.00000000000e+000 -2.70878829360e+001 2.70878829360e+001

Total energy before collision = 1.54134389567e+000
Total energy after collision  = 1.36093011036e+000
Energy discrepancy = -1.17049664135e+001 %
```

Figure B.15: Binary Test 5 – PFC3D (DEM-Approach) Solution.

B.1.6 Binary Test 6

```
Post-Collision Rigid Body Data:
 Number          Pos_X          Pos_Y          Pos_Z           V_X           V_Y           V_Z       Omega_X       Omega_Y       Omega_Z
      1   1.5000000000   1.5000000000   1.5000000000  -0.9000000000   0.5969491347  -0.9000000000  28.5000000000   0.0000000000  -28.5000000000
      2   1.5353553391   1.5000000000   1.5353553391   0.9000000000  -0.5969491347   0.9000000000  28.5000000000   0.0000000000  -28.5000000000

Contact Data:
 Number     Active   Slipping    Object1      Type1    Object2      Type2      Normal_X      Normal_Y      Normal_Z      Lambda_N     Tangent_X     Tangent_Y     Tangent_Z     Lambda_T
      1       true       true          2         19          1         19  -0.7071067812   0.0000000000  -0.7071067812   1.3805333130   0.0000000000  -1.0000000000   0.0000000000   0.2070799969
Solution Data:
  2 iterations, converged
  Energy:
    Total energy before collision =     1.54134390 Joules
    Total energy after collision =     1.22407021 Joules
    Energy discrepancy =   -20.58422435 %
```

Figure B.16: Binary Test 6 Linear Solution.

```
Post-Collision Rigid Body Data:
 Number          Pos_X          Pos_Y          Pos_Z           V_X           V_Y           V_Z       Omega_X       Omega_Y       Omega_Z
      1   1.5000000000   1.5000000000   1.5000000000  -0.9000000000   0.5969491347  -0.9000000000  28.5000000000   0.0000000000  -28.5000000000
      2   1.5353553391   1.5000000000   1.5353553391   0.9000000000  -0.5969491347   0.9000000000  28.5000000000   0.0000000000  -28.5000000000

Contact Data:
 Number     Active   Slipping    Object1      Type1    Object2      Type2      Normal_X      Normal_Y      Normal_Z      Lambda_N     Tangent_X     Tangent_Y     Tangent_Z     Lambda_T
      1       true       true          2         19          1         19  -0.7071067812   0.0000000000  -0.7071067812   1.3805333130   0.0000000000  -1.0000000000   0.0000000000   0.2070799969
Solution Data:
  2 iterations, converged
  Energy:
    Total energy before collision =     1.54134390 Joules
    Total energy after collision =     1.22407021 Joules
    Energy discrepancy =   -20.58422435 %
```

Figure B.17: Binary Test 6 Non-Linear Solution.

```
Number              V_X              V_Y              V_Z          Omega_X          Omega_Y          Omega_Z
     1 -9.98715794636e-001 5.35824439258e-001 -9.98715794636e-001 2.70878829360e+001 0.00000000000e+000 -2.70878829360e+001
     2 9.98715794636e-001 -5.35824439258e-001 9.98715794636e-001 2.70878829360e+001 0.00000000000e+000 -2.70878829360e+001

Total energy before collision = 1.54134389567e+000
Total energy after collision  = 1.36093011036e+000
Energy discrepancy = -1.17049664135e+001 %
```

Figure B.18: Binary Test 6 – PFC3D (DEM-Approach) Solution.

B.2 TERNARY TEST RESULTS

The various ternary test results as referred to in section 5.1.2.3 can be seen in this section, with the various columns representing the quantities as described in section B.1 above.

B.2.1 Ternary Test 1

Post-Collision Rigid Body Data:

Number	Pos_X	Pos_Y	Pos_Z	V_X	V_Y	V_Z	Omega_X	Omega_Y	Omega_Z
1	1.5000000000	1.5000000000	1.5000000000	-0.9000000000	-0.9000000000	0.7764705882	-15.8059162853	15.8059162853	0.0000000000
2	1.5353553391	1.5353553391	1.5000000000	0.0000000000	0.0000000000	0.0000000000	-31.6118325707	31.6118325707	0.0000000000
3	1.5707106781	1.5707106781	1.5000000000	0.9000000000	0.9000000000	-0.7764705882	-15.8059162853	15.8059162853	0.0000000000

Contact Data:

Number	Active	Slipping	Object1	Type1	Object2	Type2	Normal_X	Normal_Y	Normal_Z	Lambda_N	Tangent_X	Tangent_Y	Tangent_Z	Lambda_T
1	true	false	2	19	1	19	-0.7071067812	-0.7071067812	0.0000000000	1.3805333130	0.0000000000	0.0000000000	-1.0000000000	0.1148452314
2	true	false	3	19	2	19	-0.7071067812	-0.7071067812	0.0000000000	1.3805333130	0.0000000000	0.0000000000	-1.0000000000	0.1148452314

Solution Data:
1 iteration, converged
Energy:
Total energy before collision = 1.54134390 Joules
Total energy after collision = 1.33462248 Joules
Energy discrepancy = -13.41176471 %

Figure B.19: Ternary Test 1 Linear Solution.

Post-Collision Rigid Body Data:

Number	Pos_X	Pos_Y	Pos_Z	V_X	V_Y	V_Z	Omega_X	Omega_Y	Omega_Z
1	1.5000000000	1.5000000000	1.5000000000	-0.9000000000	-0.9000000000	0.7764705882	-15.8059162853	15.8059162853	0.0000000000
2	1.5353553391	1.5353553391	1.5000000000	-0.0000000000	0.0000000000	0.0000000000	-31.6118325707	31.6118325707	-0.0000000000
3	1.5707106781	1.5707106781	1.5000000000	0.9000000000	0.9000000000	-0.7764705882	-15.8059162853	15.8059162853	-0.0000000000

Contact Data:

Number	Active	Slipping	Object1	Type1	Object2	Type2	Normal_X	Normal_Y	Normal_Z	Lambda_N	Tangent_X	Tangent_Y	Tangent_Z	Lambda_T
1	true	false	2	19	1	19	-0.7071067812	-0.7071067812	0.0000000000	1.3805333130	0.0000000000	0.0000000000	-1.0000000000	0.1148452314
2	true	false	3	19	2	19	-0.7071067812	-0.7071067812	0.0000000000	1.3805333130	0.0000000000	-0.0000000000	-1.0000000000	0.1148452314

Solution Data:
1 iteration, converged
Energy:
Total energy before collision = 1.54134390 Joules
Total energy after collision = 1.33462248 Joules
Energy discrepancy = -13.41176471 %

Figure B.20: Ternary Test 1 Non-Linear Solution.

Number	V_X	V_Y	V_Z	Omega_X	Omega_Y	Omega_Z
1	-9.25786600724e-001	-9.25786600724e-001	7.74422362059e-001	-1.13824598691e+001	1.13824598691e+001	0.00000000000e+000
2	-1.79475512016e-011	-1.79475512016e-011	-8.06590176888e-013	-2.27649197382e+001	2.27649197382e+001	0.00000000000e+000
3	9.25786600740e-001	9.25786600740e-001	-7.74422362059e-001	-1.13824598692e+001	1.13824598692e+001	0.00000000000e+000

Total energy before collision = 1.54134389567e+000
Total energy after collision = 1.28868291943e+000
Energy discrepancy = -1.63922585315e+001 %

Figure B.21: Ternary Test 1 – PFC3D (DEM-Approach) Solution.

B.2.2 Ternary Test 2

```
Post-Collision Rigid Body Data:
 Number          Pos_X          Pos_Y          Pos_Z            V_X            V_Y            V_Z          Omega_X          Omega_Y          Omega_Z
      1   1.5000000000   1.5000000000   1.5000000000  -1.2068154120  -0.3685801179   0.7764705882  -11.1764705882   19.3582149081   -8.1817443199
      2   1.5433012702   1.5250000000   1.5000000000   0.0000000000  -0.0000000000   0.0000000000  -22.3529411765   38.7164298162  -16.3634886398
      3   1.5866025404   1.5500000000   1.5000000000   1.2068154120   0.3685801179  -0.7764705882  -11.1764705882   19.3582149081   -8.1817443199

Contact Data:
 Number      Active   Slipping    Object1      Type1    Object2      Type2       Normal_X       Normal_Y       Normal_Z       Lambda_N      Tangent_X      Tangent_Y      Tangent_Z       Lambda_T
      1        true      false          2         19          1         19  -0.8660254038  -0.5000000000   0.0000000000   1.3334927811   0.1718618847  -0.2976735161  -0.9390708016   0.1222966695
      2        true      false          3         19          2         19  -0.8660254038  -0.5000000000   0.0000000000   1.3334927811   0.1718618847  -0.2976735161  -0.9390708016   0.1222966695
Solution Data:
  1 iteration, converged
  Energy:
    Total energy before collision =     1.54134390 Joules
    Total energy after collision =     1.34616224 Joules
    Energy discrepancy =   -12.66308314 %
```

Figure B.22: Ternary Test 2 Linear Solution.

```
Post-Collision Rigid Body Data:
 Number          Pos_X          Pos_Y          Pos_Z            V_X            V_Y            V_Z          Omega_X          Omega_Y          Omega_Z
      1   1.5000000000   1.5000000000   1.5000000000  -1.2068154120  -0.3685801179   0.7764705882  -11.1764705882   19.3582149081   -8.1817443199
      2   1.5433012702   1.5250000000   1.5000000000   0.0000000000  -0.0000000000   0.0000000000  -22.3529411765   38.7164298162  -16.3634886398
      3   1.5866025404   1.5500000000   1.5000000000   1.2068154120   0.3685801179  -0.7764705882  -11.1764705882   19.3582149081   -8.1817443199

Contact Data:
 Number      Active   Slipping    Object1      Type1    Object2      Type2       Normal_X       Normal_Y       Normal_Z       Lambda_N      Tangent_X      Tangent_Y      Tangent_Z       Lambda_T
      1        true      false          2         19          1         19  -0.8660254038  -0.5000000000   0.0000000000   1.3334927811   0.1718618847  -0.2976735161  -0.9390708016   0.1222966695
      2        true      false          3         19          2         19  -0.8660254038  -0.5000000000   0.0000000000   1.3334927811   0.1718618847  -0.2976735161  -0.9390708016   0.1222966695
Solution Data:
  1 iteration, converged
  Energy:
    Total energy before collision =     1.54134390 Joules
    Total energy after collision =     1.34616224 Joules
    Energy discrepancy =   -12.66308314 %
```

Figure B.23: Ternary Test 2 Non-Linear Solution.

```
Number                V_X                V_Y                V_Z            Omega_X            Omega_Y            Omega_Z
     1 -1.18110224922e+000 -4.32747624041e-001  7.72165865207e-001 -8.35342299594e+000  1.44728106018e+001 -4.58001057811e+000
     2 -7.51734805098e-012 -4.61543149507e-012 -6.40048924357e-013 -1.67068459919e+001  2.89456212031e+001 -9.16002115625e+000
     3  1.18110224923e+000  4.32747624046e-001 -7.72165865206e-001 -8.35342299598e+000  1.44728106016e+001 -4.58001057814e+000

Total energy before collision = 1.54134389567e+000
Total energy after collision  = 1.23496427349e+000
Energy discrepancy = -1.98774344285e+001 %
```

Figure B.24: Ternary Test 2 – PFC3D (DEM-Approach) Solution.

B.2.3 Ternary Test 3

Post-Collision Rigid Body Data:

Number	Pos_X	Pos_Y	Pos_Z	V_X	V_Y	V_Z	Omega_X	Omega_Y	Omega_Z
1	1.5000000000	1.5000000000	1.5000000000	-1.2068154120	0.2450507061	0.7764705882	-5.7853668905	21.5912831759	-15.8059162853
2	1.5482962913	1.5129409523	1.5000000000	0.0000000000	0.0000000000	-0.0000000000	-11.5707337811	43.1825663517	-31.6118325707
3	1.5965925826	1.5258819045	1.5000000000	1.2068154120	-0.2450507061	-0.7764705882	-5.7853668905	21.5912831759	-15.8059162853

Contact Data:

Number	Active	Slipping	Object1	Type1	Object2	Type2	Normal_X	Normal_Y	Normal_Z	Lambda_N	Tangent_X	Tangent_Y	Tangent_Z	Lambda_T
1	true	false	2	19	1	19	-0.9659258263	-0.2588190451	0.0000000000	1.1955769198	0.1494292454	-0.5576775358	-0.8164965809	0.1406561082
2	true	false	3	19	2	19	-0.9659258263	-0.2588190451	0.0000000000	1.1955769198	0.1494292454	-0.5576775358	-0.8164965809	0.1406561082

Solution Data:
1 iteration, converged
Energy:
Total energy before collision = 1.54134390 Joules
Total energy after collision = 1.37768944 Joules
Energy discrepancy = -10.61764706 %

Figure B.25: Ternary Test 3 Linear Solution.

Post-Collision Rigid Body Data:

Number	Pos_X	Pos_Y	Pos_Z	V_X	V_Y	V_Z	Omega_X	Omega_Y	Omega_Z
1	1.5000000000	1.5000000000	1.5000000000	-1.2068154120	0.2450507061	0.7764705882	-5.7853668905	21.5912831759	-15.8059162853
2	1.5482962913	1.5129409523	1.5000000000	0.0000000000	0.0000000000	-0.0000000000	-11.5707337811	43.1825663517	-31.6118325707
3	1.5965925826	1.5258819045	1.5000000000	1.2068154120	-0.2450507061	-0.7764705882	-5.7853668905	21.5912831759	-15.8059162853

Contact Data:

Number	Active	Slipping	Object1	Type1	Object2	Type2	Normal_X	Normal_Y	Normal_Z	Lambda_N	Tangent_X	Tangent_Y	Tangent_Z	Lambda_T
1	true	false	2	19	1	19	-0.9659258263	-0.2588190451	0.0000000000	1.1955769198	0.1494292454	-0.5576775358	-0.8164965809	0.1406561082
2	true	false	3	19	2	19	-0.9659258263	-0.2588190451	0.0000000000	1.1955769198	0.1494292454	-0.5576775358	-0.8164965809	0.1406561082

Solution Data:
1 iteration, converged
Energy:
Total energy before collision = 1.54134390 Joules
Total energy after collision = 1.37768944 Joules
Energy discrepancy = -10.61764706 %

Figure B.26: Ternary Test 3 Non-Linear Solution.

Number	V_X	V_Y	V_Z	Omega_X	Omega_Y	Omega_Z
1	-1.23356431171e+000	1.43151581601e-001	7.71892428038e-001	-4.57741202539e+000	1.70892753974e+001	-1.13930732572e+001
2	-1.38071736894e-011	-4.90698533893e-012	-1.94786698093e-012	-9.15482405070e+000	3.41785507950e+001	-2.27861465144e+001
3	1.23356431172e+000	-1.43151581595e-001	-7.71892428036e-001	-4.57741202537e+000	1.70892753976e+001	-1.13930732573e+001

Total energy before collision = 1.54134389567e+000
Total energy after collision = 1.26908595601e+000
Energy discrepancy = -1.76636726185e+001 %

Figure B.27: Ternary Test 3 – PFC3D (DEM-Approach) Solution.

B.2.4 Ternary Test 4

```
Post-Collision Rigid Body Data:
 Number          Pos_X          Pos_Y          Pos_Z          V_X            V_Y            V_Z            Omega_X        Omega_Y        Omega_Z
      1  1.5000000000   1.5000000000   1.5000000000  -0.9000000000   0.7984745674   0.7984745674   0.0000000000  20.1525432638 -20.1525432638
      2  1.5500000000   1.5000000000   1.5000000000  -0.0000000000   0.0000000000   0.0000000000   0.0000000000  40.3050865276 -40.3050865276
      3  1.6000000000   1.5000000000   1.5000000000   0.9000000000  -0.7984745674  -0.7984745674   0.0000000000  20.1525432638 -20.1525432638

Contact Data:
 Number      Active    Slipping     Object1       Type1     Object2       Type2       Normal_X       Normal_Y       Normal_Z       Lambda_N      Tangent_X      Tangent_Y      Tangent_Z      Lambda_T
      1        true        true           2          19           1          19  -1.0000000000   0.0000000000   0.0000000000   0.9761844673   0.0000000000  -0.7071067812  -0.7071067812   0.1464276701
      2        true        true           3          19           2          19  -1.0000000000   0.0000000000   0.0000000000   0.9761844673   0.0000000000  -0.7071067812  -0.7071067812   0.1464276701
Solution Data:
  2 iterations, converged
  Energy:
    Total energy before collision =     1.54134390 Joules
    Total energy after collision =     1.38428649 Joules
    Energy discrepancy =    -10.18964102 %
```

Figure B.28: Ternary Test 4 Linear Solution.

```
Post-Collision Rigid Body Data:
 Number          Pos_X          Pos_Y          Pos_Z          V_X            V_Y            V_Z            Omega_X        Omega_Y        Omega_Z
      1  1.5000000000   1.5000000000   1.5000000000  -0.9000000000   0.7984745674   0.7984745674   0.0000000000  20.1525432638 -20.1525432638
      2  1.5500000000   1.5000000000   1.5000000000  -0.0000000000   0.0000000000   0.0000000000   0.0000000000  40.3050865276 -40.3050865276
      3  1.6000000000   1.5000000000   1.5000000000   0.9000000000  -0.7984745674  -0.7984745674   0.0000000000  20.1525432638 -20.1525432638

Contact Data:
 Number      Active    Slipping     Object1       Type1     Object2       Type2       Normal_X       Normal_Y       Normal_Z       Lambda_N      Tangent_X      Tangent_Y      Tangent_Z      Lambda_T
      1        true        true           2          19           1          19  -1.0000000000   0.0000000000   0.0000000000   0.9761844673   0.0000000000  -0.7071067812  -0.7071067812   0.1464276701
      2        true        true           3          19           2          19  -1.0000000000   0.0000000000   0.0000000000   0.9761844673   0.0000000000  -0.7071067812  -0.7071067812   0.1464276701
Solution Data:
  2 iterations, converged
  Energy:
    Total energy before collision =     1.54134390 Joules
    Total energy after collision =     1.38428649 Joules
    Energy discrepancy =    -10.18964102 %
```

Figure B.29: Ternary Test 4 Non-Linear Solution.

```
Number                V_X                V_Y                V_Z            Omega_X            Omega_Y            Omega_Z
     1 -9.29966944084e-001 7.93804805099e-001 7.93804805099e-001 0.00000000000e+000 1.65589216703e+001 -1.65589216703e+001
     2 0.00000000000e+000 0.00000000000e+000 0.00000000000e+000 0.00000000000e+000 3.31178433406e+001 -3.31178433406e+001
     3 9.29966944084e-001 -7.93804805099e-001 -7.93804805099e-001 0.00000000000e+000 1.65589216703e+001 -1.65589216703e+001

Total energy before collision = 1.54134389567e+000
Total energy after collision  = 1.30314845548e+000
Energy discrepancy = -1.54537505130e+001 %
```

Figure B.30: Ternary Test 4 – PFC3D (DEM-Approach) Solution.

B.2.5 Ternary Test 5

```
Post-Collision Rigid Body Data:
 Number          Pos_X          Pos_Y          Pos_Z            V_X            V_Y            V_Z          Omega_X          Omega_Y          Omega_Z
      1  1.5000000000   1.5000000000   1.5000000000   0.7764705882  -0.9000000000  -0.9000000000   0.0000000000  -15.8059162853   15.8059162853
      2  1.5000000000   1.5353553391   1.5353553391   0.0000000000   0.0000000000   0.0000000000   0.0000000000  -31.6118325707   31.6118325707
      3  1.5000000000   1.5707106781   1.5707106781  -0.7764705882   0.9000000000   0.9000000000   0.0000000000  -15.8059162853   15.8059162853

Contact Data:
 Number      Active    Slipping     Object1       Type1     Object2       Type2        Normal_X        Normal_Y        Normal_Z        Lambda_N       Tangent_X       Tangent_Y       Tangent_Z        Lambda_T
      1        true       false           2          19           1          19   0.0000000000  -0.7071067812  -0.7071067812   1.3805333130  -1.0000000000   0.0000000000   0.0000000000   0.1148452314
      2        true       false           3          19           2          19   0.0000000000  -0.7071067812  -0.7071067812   1.3805333130  -1.0000000000   0.0000000000   0.0000000000   0.1148452314
Solution Data:
  1 iteration, converged
  Energy:
    Total energy before collision =     1.54134390 Joules
    Total energy after collision =     1.33462248 Joules
    Energy discrepancy =   -13.41176471 %
```

Figure B.31: Ternary Test 5 Linear Solution.

```
Post-Collision Rigid Body Data:
 Number          Pos_X          Pos_Y          Pos_Z            V_X            V_Y            V_Z          Omega_X          Omega_Y          Omega_Z
      1  1.5000000000   1.5000000000   1.5000000000   0.7764705882  -0.9000000000  -0.9000000000   0.0000000000  -15.8059162853   15.8059162853
      2  1.5000000000   1.5353553391   1.5353553391   0.0000000000  -0.0000000000   0.0000000000  -0.0000000000  -31.6118325707   31.6118325707
      3  1.5000000000   1.5707106781   1.5707106781  -0.7764705882   0.9000000000   0.9000000000  -0.0000000000  -15.8059162853   15.8059162853

Contact Data:
 Number      Active    Slipping     Object1       Type1     Object2       Type2        Normal_X        Normal_Y        Normal_Z        Lambda_N       Tangent_X       Tangent_Y       Tangent_Z        Lambda_T
      1        true       false           2          19           1          19   0.0000000000  -0.7071067812  -0.7071067812   1.3805333130  -1.0000000000   0.0000000000   0.0000000000   0.1148452314
      2        true       false           3          19           2          19   0.0000000000  -0.7071067812  -0.7071067812   1.3805333130  -1.0000000000   0.0000000000  -0.0000000000   0.1148452314
Solution Data:
  1 iteration, converged
  Energy:
    Total energy before collision =     1.54134390 Joules
    Total energy after collision =     1.33462248 Joules
    Energy discrepancy =   -13.41176471 %
```

Figure B.32: Ternary Test 5 Non-Linear Solution.

```
Number                V_X                V_Y                V_Z            Omega_X            Omega_Y            Omega_Z
     1 7.74422362059e-001 -9.25786600724e-001 -9.25786600724e-001 0.00000000000e+000 -1.13824598691e+001 1.13824598691e+001
     2 -8.06590176888e-013 -1.79476133232e-011 -1.79476133232e-011 0.00000000000e+000 -2.27649197382e+001 2.27649197382e+001
     3 -7.74422362059e-001 9.25786600740e-001 9.25786600740e-001 0.00000000000e+000 -1.13824598692e+001 1.13824598692e+001

Total energy before collision = 1.54134389567e+000
Total energy after collision  = 1.28868281943e+000
Energy discrepancy = -1.63922585315e+001 %
```

Figure B.33: Ternary Test 5 – PFC3D (DEM-Approach) Solution.

B.2.6 Ternary Test 6

Post-Collision Rigid Body Data:

Number	Pos_X	Pos_Y	Pos_Z	V_X	V_Y	V_Z	Omega_X	Omega_Y	Omega_Z
1	1.5000000000	1.5000000000	1.5000000000	-0.9000000000	0.7764705882	-0.9000000000	15.8059162853	0.0000000000	-15.8059162853
2	1.5353553391	1.5000000000	1.5353553391	0.0000000000	0.0000000000	0.0000000000	31.6118325707	0.0000000000	-31.6118325707
3	1.5707106781	1.5000000000	1.5707106781	0.9000000000	-0.7764705882	0.9000000000	15.8059162853	0.0000000000	-15.8059162853

Contact Data:

Number	Active	Slipping	Object1	Type1	Object2	Type2	Normal_X	Normal_Y	Normal_Z	Lambda_N	Tangent_X	Tangent_Y	Tangent_Z	Lambda_T
1	true	false	2	19	1	19	-0.7071067812	0.0000000000	-0.7071067812	1.3805333130	0.0000000000	-1.0000000000	0.0000000000	0.1148452314
2	true	false	3	19	2	19	-0.7071067812	0.0000000000	-0.7071067812	1.3805333130	0.0000000000	-1.0000000000	0.0000000000	0.1148452314

Solution Data:
1 iteration, converged
Energy:
Total energy before collision = 1.54134390 Joules
Total energy after collision = 1.33462248 Joules
Energy discrepancy = -13.41176471 %

Figure B.34: Ternary Test 6 Linear Solution.

Post-Collision Rigid Body Data:

Number	Pos_X	Pos_Y	Pos_Z	V_X	V_Y	V_Z	Omega_X	Omega_Y	Omega_Z
1	1.5000000000	1.5000000000	1.5000000000	-0.9000000000	0.7764705882	-0.9000000000	15.8059162853	-0.0000000000	-15.8059162853
2	1.5353553391	1.5000000000	1.5353553391	-0.0000000000	0.0000000000	0.0000000000	31.6118325707	0.0000000000	-31.6118325707
3	1.5707106781	1.5000000000	1.5707106781	0.9000000000	-0.7764705882	0.9000000000	15.8059162853	0.0000000000	-15.8059162853

Contact Data:

Number	Active	Slipping	Object1	Type1	Object2	Type2	Normal_X	Normal_Y	Normal_Z	Lambda_N	Tangent_X	Tangent_Y	Tangent_Z	Lambda_T
1	true	false	2	19	1	19	-0.7071067812	0.0000000000	-0.7071067812	1.3805333130	0.0000000000	-1.0000000000	0.0000000000	0.1148452314
2	true	false	3	19	2	19	-0.7071067812	0.0000000000	-0.7071067812	1.3805333130	0.0000000000	-1.0000000000	-0.0000000000	0.1148452314

Solution Data:
1 iteration, converged
Energy:
Total energy before collision = 1.54134390 Joules
Total energy after collision = 1.33462248 Joules
Energy discrepancy = -13.41176471 %

Figure B.35: Ternary Test 6 Non-Linear Solution.

Number	V_X	V_Y	V_Z	Omega_X	Omega_Y	Omega_Z
1	-9.25786600724e-001	7.74422362059e-001	-9.25786600724e-001	1.13824598691e+001	0.00000000000e+000	-1.13824598691e+001
2	-1.79475512016e-011	-8.06590176888e-013	-1.79475512016e-011	2.27649197382e+001	0.00000000000e+000	-2.27649197382e+001
3	9.25786600740e-001	-7.74422362059e-001	9.25786600740e-001	1.13824598692e+001	0.00000000000e+000	-1.13824598692e+001

Total energy before collision = 1.54134389567e+000
Total energy after collision = 1.28868281943e+000
Energy discrepancy = -1.63922585315e+001 %

Figure B.36: Ternary Test 6 – PFC3D (DEM-Approach) Solution.

B.2.7 Ternary Test 7

Post-Collision Rigid Body Data:

Number	Pos_X	Pos_Y	Pos_Z	V_X	V_Y	V_Z	Omega_X	Omega_Y	Omega_Z
1	1.5000000000	1.5000000000	1.5000000000	-0.9000000000	0.7984745674	0.7984745674	0.0000000000	20.1525432638	-20.1525432638
2	1.5500000000	1.5000000000	1.5000000000	-0.0000000000	0.4030508653	0.0000000000	0.0000000000	40.3050865276	-0.0000000000
3	1.6000000000	1.5000000000	1.5000000000	0.9000000000	0.7984745674	-0.7984745674	0.0000000000	20.1525432638	20.1525432638

Contact Data:

Number	Active	Slipping	Object1	Type1	Object2	Type2	Normal_X	Normal_Y	Normal_Z	Lambda_N	Tangent_X	Tangent_Y	Tangent_Z	Lambda_T
1	true	true	2	19	1	19	-1.0000000000	0.0000000000	0.0000000000	0.9761844673	0.0000000000	-0.7071067812	-0.7071067812	0.1464276701
2	true	true	3	19	2	19	-1.0000000000	0.0000000000	0.0000000000	0.9761844673	0.0000000000	0.7071067812	-0.7071067812	0.1464276701

Solution Data:
2 iterations, converged
Energy:
Total energy before collision = 1.54134390 Joules
Total energy after collision = 1.32168866 Joules
Energy discrepancy = -14.25089102 %

Figure B.37: Ternary Test 7 Linear Solution.

Post-Collision Rigid Body Data:

Number	Pos_X	Pos_Y	Pos_Z	V_X	V_Y	V_Z	Omega_X	Omega_Y	Omega_Z
1	1.5000000000	1.5000000000	1.5000000000	-0.9000000000	0.7528499416	0.8580780192	0.0000000000	14.1921980834	-24.7150058378
2	1.5500000000	1.5000000000	1.5000000000	-0.0000000000	0.4943001168	-0.0000000000	0.0000000000	28.3843961667	0.0000000000
3	1.6000000000	1.5000000000	1.5000000000	0.9000000000	0.7528499416	-0.8580780192	0.0000000000	14.1921980834	24.7150058378

Contact Data:

Number	Active	Slipping	Object1	Type1	Object2	Type2	Normal_X	Normal_Y	Normal_Z	Lambda_N	Tangent_X	Tangent_Y	Tangent_Z	Lambda_T
1	true	true	2	19	1	19	-1.0000000000	0.0000000000	0.0000000000	0.9761844673	0.0000000000	-0.8671931873	-0.4979718626	0.1464276701
2	true	true	3	19	2	19	-1.0000000000	0.0000000000	0.0000000000	0.9761844673	0.0000000000	0.8671931873	-0.4979718626	0.1464276701

Solution Data:
3 iterations, converged
Energy:
Total energy before collision = 1.54134390 Joules
Total energy after collision = 1.30450046 Joules
Energy discrepancy = -15.36603442 %

Figure B.38: Ternary Test 7 Non-Linear Solution.

Number	V_X	V_Y	V_Z	Omega_X	Omega_Y	Omega_Z
1	-9.30487864286e-001	7.48587318595e-001	8.19343169379e-001	-8.21827250067e-003	1.46894406279e+001	-2.02052238432e+001
2	0.00000000000e+000	4.01488873718e-001	0.00000000000e+000	0.00000000000e+000	2.93788812558e+001	0.00000000000e+000
3	9.30487864286e-001	7.48587318595e-001	-8.19343169379e-001	8.21827250067e-003	1.46894406279e+001	2.02052238432e+001

Total energy before collision = 1.54134389567e+000
Total energy after collision = 1.25465807226e+000
Energy discrepancy = -1.85997313264e+001 %

Figure B.39: Ternary Test 7 – PFC3D (DEM-Approach) Solution.

B.2.8 Ternary Test 8

```
Post-Collision Rigid Body Data:
 Number          Pos_X          Pos_Y          Pos_Z            V_X            V_Y            V_Z          Omega_X          Omega_Y          Omega_Z
      1  1.5000000000   1.5000000000   1.5000000000  -0.9000000000   0.7984745674   0.7984745674   0.0000000000  20.1525432638 -20.1525432638
      2  1.5500000000   1.5000000000   1.5000000000  -0.0000000000   0.4030508653   0.4030508653   0.0000000000   0.0000000000  -0.0000000000
      3  1.6000000000   1.5000000000   1.5000000000   0.9000000000   0.7984745674   0.7984745674   0.0000000000 -20.1525432638  20.1525432638

Contact Data:
 Number      Active    Slipping     Object1       Type1     Object2       Type2       Normal_X       Normal_Y       Normal_Z       Lambda_N      Tangent_X      Tangent_Y      Tangent_Z      Lambda_T
      1        true        true           2          19           1          19  -1.0000000000   0.0000000000   0.0000000000   0.9761844673   0.0000000000  -0.7071067812  -0.7071067812   0.1464276701
      2        true        true           3          19           2          19  -1.0000000000   0.0000000000   0.0000000000   0.9761844673   0.0000000000   0.7071067812   0.7071067812   0.1464276701
Solution Data:
  2 iterations, converged
  Energy:
    Total energy before collision =     1.54134390 Joules
    Total energy after collision =     1.25909083 Joules
    Energy discrepancy =   -18.31214102 %
```

Figure B.40: Ternary Test 8 Linear Solution.

```
Post-Collision Rigid Body Data:
 Number          Pos_X          Pos_Y          Pos_Z            V_X            V_Y            V_Z          Omega_X          Omega_Y          Omega_Z
      1  1.5000000000   1.5000000000   1.5000000000  -0.9000000000   0.7984745674   0.7984745674   0.0000000000  20.1525432638 -20.1525432638
      2  1.5500000000   1.5000000000   1.5000000000  -0.0000000000   0.4030508653   0.4030508653   0.0000000000   0.0000000000  -0.0000000000
      3  1.6000000000   1.5000000000   1.5000000000   0.9000000000   0.7984745674   0.7984745674   0.0000000000 -20.1525432638  20.1525432638

Contact Data:
 Number      Active    Slipping     Object1       Type1     Object2       Type2       Normal_X       Normal_Y       Normal_Z       Lambda_N      Tangent_X      Tangent_Y      Tangent_Z      Lambda_T
      1        true        true           2          19           1          19  -1.0000000000   0.0000000000   0.0000000000   0.9761844673   0.0000000000  -0.7071067812  -0.7071067812   0.1464276701
      2        true        true           3          19           2          19  -1.0000000000   0.0000000000   0.0000000000   0.9761844673   0.0000000000   0.7071067812   0.7071067812   0.1464276701
Solution Data:
  2 iterations, converged
  Energy:
    Total energy before collision =     1.54134390 Joules
    Total energy after collision =     1.25909083 Joules
    Energy discrepancy =   -18.31214102 %
```

Figure B.41: Ternary Test 8 Non-Linear Solution.

```
Number                V_X                V_Y                V_Z            Omega_X            Omega_Y            Omega_Z
     1 -9.30961441931e-001 7.76929963724e-001 7.76929963724e-001 0.00000000000e+000 1.79907126356e+001 -1.79907126356e+001
     2 0.00000000000e+000 3.57160175070e-001 3.57160175070e-001 0.00000000000e+000 0.00000000000e+000 0.00000000000e+000
     3 9.30961441931e-001 7.76929963724e-001 7.76929963724e-001 0.00000000000e+000 -1.79907126356e+001 1.79907126356e+001

Total energy before collision = 1.54134389567e+000
Total energy after collision  = 1.21423260082e+000
Energy discrepancy = -2.12224731783e+001 %
```

Figure B.42: Ternary Test 8 – PFC3D (DEM-Approach) Solution.

B.2.9 Ternary Test 9

Post-Collision Rigid Body Data:

Number	Pos_X	Pos_Y	Pos_Z	V_X	V_Y	V_Z	Omega_X	Omega_Y	Omega_Z
1	1.5000000000	1.5000000000	1.5000000000	-0.1775401167	0.0000000000	0.0000000000	0.0000000000	0.0000000000	0.0000000000
2	1.5433012702	1.5250000000	1.5000000000	0.5887700584	0.2315575692	0.0000000000	0.0000000000	0.0000000000	9.3850291819
3	1.5433012702	1.4750000000	1.5000000000	0.5887700584	-0.2315575692	0.0000000000	0.0000000000	0.0000000000	-9.3850291819

Contact Data:

Number	Active	Slipping	Object1	Type1	Object2	Type2	Normal_X	Normal_Y	Normal_Z	Lambda_N	Tangent_X	Tangent_Y	Tangent_Z	Lambda_T
1	true	true	2	19	1	19	-0.8660254038	-0.5000000000	0.0000000000	0.3214568320	-0.5000000000	0.8660254038	0.0000000000	0.0482185248
2	true	true	3	19	1	19	-0.8660254038	0.5000000000	0.0000000000	0.3214568320	-0.5000000000	-0.8660254038	0.0000000000	0.0482185248
3	false	false	3	19	2	19	0.0000000000	1.0000000000	0.0000000000	0.0000000000	-1.0000000000	0.0000000000	0.0000000000	0.0000000000

Solution Data:
3 iterations, converged
Energy:
Total energy before collision = 0.25689065 Joules
Total energy after collision = 0.22506140 Joules
Energy discrepancy = -12.39019416 %

Figure B.43: Ternary Test 9 Linear Solution.

Post-Collision Rigid Body Data:

Number	Pos_X	Pos_Y	Pos_Z	V_X	V_Y	V_Z	Omega_X	Omega_Y	Omega_Z
1	1.5000000000	1.5000000000	1.5000000000	-0.1775401167	0.0000000000	0.0000000000	0.0000000000	0.0000000000	0.0000000000
2	1.5433012702	1.5250000000	1.5000000000	0.5887700584	0.2315575692	0.0000000000	0.0000000000	0.0000000000	9.3850291819
3	1.5433012702	1.4750000000	1.5000000000	0.5887700584	-0.2315575692	0.0000000000	0.0000000000	0.0000000000	-9.3850291819

Contact Data:

Number	Active	Slipping	Object1	Type1	Object2	Type2	Normal_X	Normal_Y	Normal_Z	Lambda_N	Tangent_X	Tangent_Y	Tangent_Z	Lambda_T
1	true	true	2	19	1	19	-0.8660254038	-0.5000000000	0.0000000000	0.3214568320	-0.5000000000	0.8660254038	0.0000000000	0.0482185248
2	true	true	3	19	1	19	-0.8660254038	0.5000000000	0.0000000000	0.3214568320	-0.5000000000	-0.8660254038	0.0000000000	0.0482185248
3	false	false	3	19	2	19	0.0000000000	1.0000000000	0.0000000000	0.0000000000	-1.0000000000	0.0000000000	0.0000000000	0.0000000000

Solution Data:
3 iterations, converged
Energy:
Total energy before collision = 0.25689065 Joules
Total energy after collision = 0.22506140 Joules
Energy discrepancy = -12.39019416 %

Figure B.44: Ternary Test 9 Non-Linear Solution.

Number	V_X	V_Y	V_Z	Omega_X	Omega_Y	Omega_Z
1	-2.59003400795e-001	0.00000000000e+000	0.00000000000e+000	0.00000000000e+000	0.00000000000e+000	0.00000000000e+000
2	5.59020832642e-001	2.21060725633e-001	0.00000000000e+000	0.00000000000e+000	0.00000000000e+000	8.81618953196e+000
3	5.59020832642e-001	-2.21060725633e-001	0.00000000000e+000	0.00000000000e+000	0.00000000000e+000	-8.81618953196e+000

Total energy before collision = 2.56890649278e-001
Total energy after collision = 2.12882617626e-001
Energy discrepancy = -1.71310367955e+001 %

Figure B.45: Ternary Test 9 – PFC3D (DEM-Approach) Solution.

B.2.10 Ternary Test 10

```
Post-Collision Rigid Body Data:
 Number          Pos_X          Pos_Y          Pos_Z           V_X           V_Y           V_Z       Omega_X       Omega_Y       Omega_Z
      1   1.5000000000   1.5000000000   1.5000000000  -0.1419669704   0.0000000000   9.8033029597  -0.0000000000  17.0344633783   0.0000000000
      2   1.5433012702   1.5250000000   1.5000000000   0.5709834852   0.3239793144   0.0983485202  -4.9174260085   8.5172316891   0.4917426008
      3   1.5433012702   1.4750000000   1.5000000000   0.5709834852  -0.3239793144   0.0983485202   4.9174260085   8.5172316891  -0.4917426008

Contact Data:
 Number     Active   Slipping    Object1      Type1    Object2      Type2      Normal_X      Normal_Y      Normal_Z      Lambda_N     Tangent_X     Tangent_Y     Tangent_Z     Lambda_T
      1       true       true          2         19          1         19  -0.8660254038  -0.5000000000   0.0000000000   0.3372850201  -0.0249688085   0.0432472449  -0.9987523389   0.0505927530
      2       true       true          3         19          1         19  -0.8660254038   0.5000000000   0.0000000000   0.3372850201  -0.0249688085  -0.0432472449  -0.9987523389   0.0505927530
      3      false      false          3         19          2         19   0.0000000000   1.0000000000   0.0000000000   0.0000000000  -0.2576626506   0.0000000000  -0.9662349396   0.0000000000
Solution Data:
  3 iterations, converged
  Energy:
    Total energy before collision =    25.94595558 Joules
    Total energy after collision =    24.95108082 Joules
    Energy discrepancy =    -3.83441171 %
```

Figure B.46: Ternary Test 10 Linear Solution.

```
Post-Collision Rigid Body Data:
 Number          Pos_X          Pos_Y          Pos_Z           V_X           V_Y           V_Z       Omega_X       Omega_Y       Omega_Z
      1   1.5000000000   1.5000000000   1.5000000000  -0.1397074292   0.0000000000   9.8024756126  -0.0000000000  17.1061137523  -0.0000000000
      2   1.5433012702   1.5250000000   1.5000000000   0.5698537146   0.3298497748   0.0987621937  -4.9381096841   8.5530568661  -0.0731427081
      3   1.5433012702   1.4750000000   1.5000000000   0.5698537146  -0.3298497748   0.0987621937   4.9381096841   8.5530568661   0.0731427081

Contact Data:
 Number     Active   Slipping    Object1      Type1    Object2      Type2      Normal_X      Normal_Y      Normal_Z      Lambda_N     Tangent_X     Tangent_Y     Tangent_Z     Lambda_T
      1       true       true          2         19          1         19  -0.8660254038  -0.5000000000   0.0000000000   0.3382903977   0.0037028695  -0.0064135581  -0.9999725771   0.0507435597
      2       true       true          3         19          1         19  -0.8660254038   0.5000000000   0.0000000000   0.3382903977   0.0037028695   0.0064135581  -0.9999725771   0.0507435597
      3      false      false          3         19          2         19   0.0000000000   1.0000000000   0.0000000000   0.0000000000  -0.2576626506   0.0000000000  -0.9662349396   0.0000000000
Solution Data:
  4 iterations, converged
  Energy:
    Total energy before collision =    25.94595558 Joules
    Total energy after collision =    24.94033354 Joules
    Energy discrepancy =    -3.94500018 %
```

Figure B.47: Ternary Test 10 Non-Linear Solution.

```
Number                V_X                V_Y                V_Z            Omega_X            Omega_Y            Omega_Z
     1 -1.45056762017e-001 0.00000000000e+000 9.79719990202e+000 0.00000000000e+000 1.51908939985e+001 0.00000000000e+000
     2 5.08672293462e-001 2.90266291889e-001 8.43904386939e-002 -4.38579854042e+000 7.59544794926e+000 3.08248933120e-001
     3 5.08672293462e-001 -2.90266291889e-001 8.43904386939e-002 4.38579854042e+000 7.59544794926e+000 -3.08248933120e-001

Total energy before collision = 2.59459555771e+001
Total energy after collision  = 2.48676869472e+001
Energy discrepancy = -4.15582546816e+000 %
```

Figure B.48: Ternary Test 10 – PFC3D (DEM-Approach) Solution.

B.3 QUATERNARY TEST RESULTS

The various quaternary test results as referred to in section 5.1.3.3 can be seen in this section, with the quantities again as described in section B.1.

B.3.1 Quaternary Test 1

Post-Collision Rigid Body Data:

Number	Pos_X	Pos_Y	Pos_Z	V_X	V_Y	V_Z	Omega_X	Omega_Y	Omega_Z
1	1.5000000000	1.5000000000	1.5000000000	-0.9000000000	-0.9000000000	0.7011235955	-21.1337532355	21.1337532355	0.0000000000
2	1.5353553391	1.5353553391	1.5000000000	-0.0000000000	0.0000000000	0.4269662921	-12.0764304203	12.0764304203	0.0000000000
3	1.5707106781	1.5707106781	1.5000000000	0.0000000000	-0.0000000000	-0.4269662921	-12.0764304203	12.0764304203	0.0000000000
4	1.6060660172	1.6060660172	1.5000000000	0.9000000000	0.9000000000	-0.7011235955	-21.1337532355	21.1337532355	0.0000000000

Contact Data:

Number	Active	Slipping	Object1	Type1	Object2	Type2	Normal_X	Normal_Y	Normal_Z	Lambda_N	Tangent_X	Tangent_Y	Tangent_Z	Lambda_T
1	true	false	2	19	1	19	-0.7071067812	-0.7071067812	0.0000000000	1.3805333130	0.0000000000	0.0000000000	-1.0000000000	0.1535571072
2	true	false	3	19	2	19	-0.7071067812	-0.7071067812	0.0000000000	1.3805333130	0.0000000000	0.0000000000	-1.0000000000	-0.0658101888
3	true	false	4	19	3	19	-0.7071067812	-0.7071067812	0.0000000000	1.3805333130	0.0000000000	0.0000000000	-1.0000000000	0.1535571072

Solution Data:
1 iteration, converged
Energy:
Total energy before collision = 1.54134390 Joules
Total energy after collision = 1.33075129 Joules
Energy discrepancy = -13.66292135 %

Figure B.49: Quaternary Test 1 Linear Solution.

Post-Collision Rigid Body Data:

Number	Pos_X	Pos_Y	Pos_Z	V_X	V_Y	V_Z	Omega_X	Omega_Y	Omega_Z
1	1.5000000000	1.5000000000	1.5000000000	-0.9000000000	-0.9000000000	0.7011235955	-21.1337532355	21.1337532355	0.0000000000
2	1.5353553391	1.5353553391	1.5000000000	-0.0000000000	0.0000000000	0.4269662921	-12.0764304203	12.0764304203	0.0000000000
3	1.5707106781	1.5707106781	1.5000000000	0.0000000000	-0.0000000000	-0.4269662921	-12.0764304203	12.0764304203	0.0000000000
4	1.6060660172	1.6060660172	1.5000000000	0.9000000000	0.9000000000	-0.7011235955	-21.1337532355	21.1337532355	0.0000000000

Contact Data:

Number	Active	Slipping	Object1	Type1	Object2	Type2	Normal_X	Normal_Y	Normal_Z	Lambda_N	Tangent_X	Tangent_Y	Tangent_Z	Lambda_T
1	true	false	2	19	1	19	-0.7071067812	-0.7071067812	0.0000000000	1.3805333130	0.0000000000	0.0000000000	-1.0000000000	0.1535571072
2	true	false	3	19	2	19	-0.7071067812	-0.7071067812	0.0000000000	1.3805333130	0.0000000000	0.0000000000	-1.0000000000	-0.0658101888
3	true	false	4	19	3	19	-0.7071067812	-0.7071067812	0.0000000000	1.3805333130	0.0000000000	0.0000000000	-1.0000000000	0.1535571072

Solution Data:
1 iteration, converged
Energy:
Total energy before collision = 1.54134390 Joules
Total energy after collision = 1.33075129 Joules
Energy discrepancy = -13.66292135 %

Figure B.50: Quaternary Test 1 Non-Linear Solution.

Number	V_X	V_Y	V_Z	Omega_X	Omega_Y	Omega_Z
1	-8.14543122215e-001	-8.14543122215e-001	7.98167749057e-001	-9.30291075735e+000	9.30291075735e+000	0.00000000000e+000
2	-2.90532648230e-001	-2.90532648230e-001	3.04536920558e-001	3.18058866204e+000	-3.18058866204e+000	0.00000000000e+000
3	2.90532648233e-001	2.90532648233e-001	-3.04536920555e-001	3.18058866225e+000	-3.18058866225e+000	0.00000000000e+000
4	8.14543122213e-001	8.14543122213e-001	-7.98167749060e-001	-9.30291075710e+000	9.30291075710e+000	0.00000000000e+000

Total energy before collision = 1.54134389567e+000
Total energy after collision = 1.16829966319e+000
Energy discrepancy = -2.42025308905e+001 %

Figure B.51: Quaternary Test 1 – PFC3D (DEM-Approach) Solution.

B.3.2 Quaternary Test 2

```
Post-Collision Rigid Body Data:
 Number          Pos_X          Pos_Y          Pos_Z            V_X            V_Y            V_Z        Omega_X        Omega_Y        Omega_Z
      1   1.5000000000   1.5000000000   1.5000000000  -1.1930259553  -0.3924641575   0.7011235955  -14.9438202247   25.8834558884  -10.9396356637
      2   1.5433012702   1.5250000000   1.5000000000  -0.0781402547   0.1353428913   0.4269662921   -8.5393258427   14.7905462219   -6.2512203792
      3   1.5866025404   1.5500000000   1.5000000000   0.0781402547  -0.1353428913  -0.4269662921   -8.5393258427   14.7905462219   -6.2512203792
      4   1.6299038106   1.5750000000   1.5000000000   1.1930259553   0.3924641575  -0.7011235955  -14.9438202247   25.8834558884  -10.9396356637

Contact Data:
 Number      Active    Slipping     Object1       Type1     Object2       Type2      Normal_X      Normal_Y      Normal_Z      Lambda_N      Tangent_X     Tangent_Y     Tangent_Z     Lambda_T
      1        true       false           2          19           1          19  -0.8660254038  -0.5000000000   0.0000000000   1.3334927811   0.1718618847  -0.2976735161  -0.9390708016   0.1635202659
      2        true       false           3          19           2          19  -0.8660254038  -0.5000000000   0.0000000000   1.3334927811   0.1718618847  -0.2976735161  -0.9390708016  -0.0700801140
      3        true       false           4          19           3          19  -0.8660254038  -0.5000000000   0.0000000000   1.3334927811   0.1718618847  -0.2976735161  -0.9390708016   0.1635202659
Solution Data:
  1 iteration, converged
  Energy:
    Total energy before collision =     1.54134390 Joules
    Total energy after collision =     1.34177241 Joules
    Energy discrepancy =   -12.94788839 %
```

Figure B.52: Quaternary Test 2 Linear Solution.

```
Post-Collision Rigid Body Data:
 Number          Pos_X          Pos_Y          Pos_Z            V_X            V_Y            V_Z        Omega_X        Omega_Y        Omega_Z
      1   1.5000000000   1.5000000000   1.5000000000  -1.1930259553  -0.3924641575   0.7011235955  -14.9438202247   25.8834558884  -10.9396356637
      2   1.5433012702   1.5250000000   1.5000000000  -0.0781402547   0.1353428913   0.4269662921   -8.5393258427   14.7905462219   -6.2512203792
      3   1.5866025404   1.5500000000   1.5000000000   0.0781402547  -0.1353428913  -0.4269662921   -8.5393258427   14.7905462219   -6.2512203792
      4   1.6299038106   1.5750000000   1.5000000000   1.1930259553   0.3924641575  -0.7011235955  -14.9438202247   25.8834558884  -10.9396356637

Contact Data:
 Number      Active    Slipping     Object1       Type1     Object2       Type2      Normal_X      Normal_Y      Normal_Z      Lambda_N      Tangent_X     Tangent_Y     Tangent_Z     Lambda_T
      1        true       false           2          19           1          19  -0.8660254038  -0.5000000000   0.0000000000   1.3334927811   0.1718618847  -0.2976735161  -0.9390708016   0.1635202659
      2        true       false           3          19           2          19  -0.8660254038  -0.5000000000   0.0000000000   1.3334927811   0.1718618847  -0.2976735161  -0.9390708016   0.0700801140
      3        true       false           4          19           3          19  -0.8660254038  -0.5000000000   0.0000000000   1.3334927811   0.1718618847  -0.2976735161  -0.9390708016   0.1635202659
Solution Data:
  1 iteration, converged
  Energy:
    Total energy before collision =     1.54134390 Joules
    Total energy after collision =     1.34177241 Joules
    Energy discrepancy =   -12.94788839 %
```

Figure B.53: Quaternary Test 2 Non-Linear Solution.

```
Number                V_X                V_Y                V_Z            Omega_X            Omega_Y            Omega_Z
     1 -1.10807592270e+000 -3.57157285256e-001 7.89714059644e-001 -7.28944063782e+000 1.26325848553e+001 -2.88182413714e+000
     2 -3.33654893410e-001 -8.43063090453e-002 3.07088968780e-001 1.24849273159e+000 -2.15684317188e+000 2.11208664282e+000
     3 3.33654893410e-001 8.43063090453e-002 -3.07088968780e-001 1.24849273159e+000 -2.15684317188e+000 2.11208664282e+000
     4 1.10807592270e+000 3.57157285256e-001 -7.89714059644e-001 -7.28944063782e+000 1.26325848553e+001 -2.88182413714e+000

Total energy before collision = 1.54134389567e+000
Total energy after collision  = 1.15585497791e+000
Energy discrepancy = -2.50099227589e+001 %
```

Figure B.54: Quaternary Test 2 – PFC3D (DEM-Approach) Solution.

B.3.3 Quaternary Test 3

```
Post-Collision Rigid Body Data:
 Number          Pos_X          Pos_Y          Pos_Z            V_X            V_Y            V_Z          Omega_X          Omega_Y          Omega_Z
      1   1.5000000000   1.5000000000   1.5000000000  -1.1955655136   0.2030655136   0.7150000000  -7.3763427854   27.5288860492  -20.1525432638
      2   1.5482962913   1.5129409523   1.5000000000  -0.0745123143   0.2780837429   0.4071428571  -4.2150530202   15.7307920281  -11.5157390079
      3   1.5965925826   1.5258819045   1.5000000000   0.0745123143  -0.2780837429  -0.4071428571  -4.2150530202   15.7307920281  -11.5157390079
      4   1.6448888739   1.5388228568   1.5000000000   1.1955655136  -0.2030655136  -0.7150000000  -7.3763427854   27.5288860492  -20.1525432638

Contact Data:
 Number      Active    Slipping     Object1       Type1     Object2       Type2       Normal_X       Normal_Y       Normal_Z     Lambda_N     Tangent_X     Tangent_Y     Tangent_Z     Lambda_T
      1        true        true           2          19           1          19  -0.9659258263  -0.2588190451   0.0000000000   1.1955769198   0.1494292454  -0.5576775358  -0.8164965809   0.1793365380
      2        true       false           3          19           2          19  -0.9659258263  -0.2588190451   0.0000000000   1.1955769198   0.1494292454  -0.5576775358  -0.8164965809  -0.0768585163
      3        true        true           4          19           3          19  -0.9659258263  -0.2588190451   0.0000000000   1.1955769198   0.1494292454  -0.5576775358  -0.8164965809   0.1793365380
Solution Data:
  2 iterations, converged
  Energy:
    Total energy before collision =     1.54134390 Joules
    Total energy after collision =     1.35357656 Joules
    Energy discrepancy =   -12.18205357 %
```

Figure B.55: Quaternary Test 3 Linear Solution.

```
Post-Collision Rigid Body Data:
 Number          Pos_X          Pos_Y          Pos_Z            V_X            V_Y            V_Z          Omega_X          Omega_Y          Omega_Z
      1   1.5000000000   1.5000000000   1.5000000000  -1.1955655136   0.2030655136   0.7150000000  -7.3763427854   27.5288860492  -20.1525432638
      2   1.5482962913   1.5129409523   1.5000000000  -0.0745123143   0.2780837429   0.4071428571  -4.2150530202   15.7307920281  -11.5157390079
      3   1.5965925826   1.5258819045   1.5000000000   0.0745123143  -0.2780837429  -0.4071428571  -4.2150530202   15.7307920281  -11.5157390079
      4   1.6448888739   1.5388228568   1.5000000000   1.1955655136  -0.2030655136  -0.7150000000  -7.3763427854   27.5288860492  -20.1525432638

Contact Data:
 Number      Active    Slipping     Object1       Type1     Object2       Type2       Normal_X       Normal_Y       Normal_Z     Lambda_N     Tangent_X     Tangent_Y     Tangent_Z     Lambda_T
      1        true        true           2          19           1          19  -0.9659258263  -0.2588190451   0.0000000000   1.1955769198   0.1494292454  -0.5576775358  -0.8164965809   0.1793365380
      2        true       false           3          19           2          19  -0.9659258263  -0.2588190451   0.0000000000   1.1955769198  -0.1494292454   0.5576775358   0.8164965809   0.0768585163
      3        true        true           4          19           3          19  -0.9659258263  -0.2588190451   0.0000000000   1.1955769198   0.1494292454  -0.5576775358  -0.8164965809   0.1793365380
Solution Data:
  2 iterations, converged
  Energy:
    Total energy before collision =     1.54134390 Joules
    Total energy after collision =     1.35357656 Joules
    Energy discrepancy =   -12.18205357 %
```

Figure B.56: Quaternary Test 3 Non-Linear Solution.

```
Number                 V_X                   V_Y                   V_Z               Omega_X               Omega_Y               Omega_Z
     1 -1.09369184629e+000  2.25142639114e-001  7.76340989235e-001 -4.58928306535e+000  1.71310332625e+001 -1.12680441705e+001
     2 -3.87688253476e-001  7.75518808036e-002  3.00715195690e-001 -1.11570770456e+000  4.17115569618e+000 -2.20701219501e+000
     3  3.87688253471e-001 -7.75518808058e-002 -3.00715195690e-001 -1.11570770457e+000  4.17115569624e+000 -2.20701219499e+000
     4  1.09369184630e+000 -2.25142639113e-001 -7.76340989234e-001 -4.58928306536e+000  1.71310332625e+001 -1.12680441704e+001

Total energy before collision = 1.54134389567e+000
Total energy after collision  = 1.13676908620e+000
Energy discrepancy = -2.62481857947e+001 %
```

Figure B.57: Quaternary Test 3 – PFC3D (DEM-Approach) Solution.

B.3.4 Quaternary Test 4

Post-Collision Rigid Body Data:

Number	Pos_X	Pos_Y	Pos_Z	V_X	V_Y	V_Z	Omega_X	Omega_Y	Omega_Z
1	1.5000000000	1.5000000000	1.5000000000	-0.9000000000	0.7984745674	0.7984745674	0.0000000000	20.1525432638	-20.1525432638
2	1.5500000000	1.5000000000	1.5000000000	0.0000000000	0.2878934752	0.2878934752	0.0000000000	11.5157390079	-11.5157390079
3	1.6000000000	1.5000000000	1.5000000000	0.0000000000	-0.2878934752	-0.2878934752	0.0000000000	11.5157390079	-11.5157390079
4	1.6500000000	1.5000000000	1.5000000000	0.9000000000	-0.7984745674	-0.7984745674	0.0000000000	20.1525432638	-20.1525432638

Contact Data:

Number	Active	Slipping	Object1	Type1	Object2	Type2	Normal_X	Normal_Y	Normal_Z	Lambda_N	Tangent_X	Tangent_Y	Tangent_Z	Lambda_T
1	true	true	2	19	1	19	-1.0000000000	0.0000000000	0.0000000000	0.9761844673	0.0000000000	-0.7071067812	-0.7071067812	0.1464276701
2	true	false	3	19	2	19	-1.0000000000	0.0000000000	0.0000000000	0.9761844673	0.0000000000	-0.7071067812	-0.7071067812	-0.0627547158
3	true	true	4	19	3	19	-1.0000000000	0.0000000000	0.0000000000	0.9761844673	0.0000000000	-0.7071067812	-0.7071067812	0.1464276701

Solution Data:
2 iterations, converged
Energy:
Total energy before collision = 1.54134390 Joules
Total energy after collision = 1.29486102 Joules
Energy discrepancy = -15.99142673 %

Figure B.58: Quaternary Test 4 Linear Solution.

Post-Collision Rigid Body Data:

Number	Pos_X	Pos_Y	Pos_Z	V_X	V_Y	V_Z	Omega_X	Omega_Y	Omega_Z
1	1.5000000000	1.5000000000	1.5000000000	-0.9000000000	0.7984745674	0.7984745674	0.0000000000	20.1525432638	-20.1525432638
2	1.5500000000	1.5000000000	1.5000000000	0.0000000000	0.2878934752	0.2878934752	0.0000000000	11.5157390079	-11.5157390079
3	1.6000000000	1.5000000000	1.5000000000	0.0000000000	-0.2878934752	-0.2878934752	0.0000000000	11.5157390079	-11.5157390079
4	1.6500000000	1.5000000000	1.5000000000	0.9000000000	-0.7984745674	-0.7984745674	0.0000000000	20.1525432638	-20.1525432638

Contact Data:

Number	Active	Slipping	Object1	Type1	Object2	Type2	Normal_X	Normal_Y	Normal_Z	Lambda_N	Tangent_X	Tangent_Y	Tangent_Z	Lambda_T
1	true	true	2	19	1	19	-1.0000000000	0.0000000000	0.0000000000	0.9761844673	0.0000000000	-0.7071067812	-0.7071067812	0.1464276701
2	true	false	3	19	2	19	-1.0000000000	0.0000000000	0.0000000000	0.9761844673	0.0000000000	0.7071067812	0.7071067812	0.0627547158
3	true	true	4	19	3	19	-1.0000000000	0.0000000000	0.0000000000	0.9761844673	0.0000000000	-0.7071067812	-0.7071067812	0.1464276701

Solution Data:
2 iterations, converged
Energy:
Total energy before collision = 1.54134390 Joules
Total energy after collision = 1.29486102 Joules
Energy discrepancy = -15.99142673 %

Figure B.59: Quaternary Test 4 Non-Linear Solution.

Number	V_X	V_Y	V_Z	Omega_X	Omega_Y	Omega_Z
1	-7.98661229937e-001	7.88265979864e-001	7.88265979864e-001	0.00000000000e+000	1.71581908012e+001	-1.71581908012e+001
2	-3.09651869853e-001	2.60523662498e-001	2.60523662498e-001	0.00000000000e+000	7.60548880414e+000	-7.60548880414e+000
3	3.09651869849e-001	-2.60523662498e-001	-2.60523662498e-001	0.00000000000e+000	7.60548880420e+000	-7.60548880420e+000
4	7.98661229941e-001	-7.88265979864e-001	-7.88265979864e-001	0.00000000000e+000	1.71581908012e+001	-1.71581908012e+001

Total energy before collision = 1.54134389567e+000
Total energy after collision = 1.17570588260e+000
Energy discrepancy = -2.37220268683e+001 %

Figure B.60: Quaternary Test 4 – PFC3D (DEM-Approach) Solution.

B.3.5 Quaternary Test 5

Post-Collision Rigid Body Data:

Number	Pos_X	Pos_Y	Pos_Z	V_X	V_Y	V_Z	Omega_X	Omega_Y	Omega_Z
1	1.5000000000	1.5000000000	1.5000000000	-0.3765701520	0.0000000000	0.0000000000	0.0000000000	0.0000000000	-0.0000000000
2	1.5500000000	1.5000000000	1.5000000000	0.5234298480	0.0000000000	0.0000000000	0.0000000000	0.0000000000	0.0000000000
3	1.5933012702	1.5250000000	1.5000000000	0.4265701520	0.1677659148	0.0000000000	0.0000000000	0.0000000000	6.7995531826
4	1.5933012702	1.4750000000	1.5000000000	0.4265701520	-0.1677659148	0.0000000000	0.0000000000	0.0000000000	-6.7995531826

Contact Data:

Number	Active	Slipping	Object1	Type1	Object2	Type2	Normal_X	Normal_Y	Normal_Z	Lambda_N	Tangent_X	Tangent_Y	Tangent_Z	Lambda_T
1	true	false	2	19	1	19	-1.0000000000	0.0000000000	0.0000000000	0.7072560002	0.0000000000	1.0000000000	0.0000000000	0.0000000000
2	true	true	3	19	2	19	-0.8660254038	-0.5000000000	0.0000000000	0.2328988843	-0.5000000000	0.8660254038	0.0000000000	0.0349348326
3	true	true	4	19	2	19	-0.8660254038	0.5000000000	0.0000000000	0.2328988843	-0.5000000000	-0.8660254038	0.0000000000	0.0349348326
4	false	false	4	19	3	19	0.0000000000	1.0000000000	0.0000000000	0.0000000000	-1.0000000000	0.0000000000	0.0000000000	0.0000000000

Solution Data:
3 iterations, converged
Energy:
Total energy before collision = 0.25689065 Joules
Total energy after collision = 0.22069883 Joules
Energy discrepancy = -14.08841596 %

Figure B.61: Quaternary Test 5 Linear Solution.

Post-Collision Rigid Body Data:

Number	Pos_X	Pos_Y	Pos_Z	V_X	V_Y	V_Z	Omega_X	Omega_Y	Omega_Z
1	1.5000000000	1.5000000000	1.5000000000	-0.3765701520	0.0000000000	0.0000000000	0.0000000000	0.0000000000	-0.0000000000
2	1.5500000000	1.5000000000	1.5000000000	0.5234298480	-0.0000000000	0.0000000000	0.0000000000	0.0000000000	0.0000000000
3	1.5933012702	1.5250000000	1.5000000000	0.4265701520	0.1677659148	0.0000000000	0.0000000000	0.0000000000	6.7995531826
4	1.5933012702	1.4750000000	1.5000000000	0.4265701520	-0.1677659148	0.0000000000	0.0000000000	0.0000000000	-6.7995531826

Contact Data:

Number	Active	Slipping	Object1	Type1	Object2	Type2	Normal_X	Normal_Y	Normal_Z	Lambda_N	Tangent_X	Tangent_Y	Tangent_Z	Lambda_T
1	true	false	2	19	1	19	-1.0000000000	0.0000000000	0.0000000000	0.7072560002	0.0000000000	1.0000000000	0.0000000000	0.0000000000
2	true	true	3	19	2	19	-0.8660254038	-0.5000000000	0.0000000000	0.2328988843	-0.5000000000	0.8660254038	0.0000000000	0.0349348326
3	true	true	4	19	2	19	-0.8660254038	0.5000000000	0.0000000000	0.2328988843	-0.5000000000	-0.8660254038	0.0000000000	0.0349348326
4	false	false	4	19	3	19	0.0000000000	1.0000000000	0.0000000000	0.0000000000	-1.0000000000	0.0000000000	0.0000000000	0.0000000000

Solution Data:
3 iterations, converged
Energy:
Total energy before collision = 0.25689065 Joules
Total energy after collision = 0.22069883 Joules
Energy discrepancy = -14.08841596 %

Figure B.62: Quaternary Test 5 Non-Linear Solution.

Number	V_X	V_Y	V_Z	Omega_X	Omega_Y	Omega_Z
1	-2.607313518 32e-001	0.00000000000e+000	0.00000000000e+000	0.00000000000e+000	0.00000000000e+000	0.00000000000e+000
2	-1.005683544 73e-001	0.00000000000e+000	0.00000000000e+000	0.00000000000e+000	0.00000000000e+000	0.00000000000e+000
3	5.507584699 00e-001	2.169739361 78e-001	0.00000000000e+000	0.00000000000e+000	0.00000000000e+000	8.862775265 24e+000
4	5.507584699 00e-001	-2.169739361 78e-001	0.00000000000e+000	0.00000000000e+000	0.00000000000e+000	-8.862775265 24e+000

Total energy before collision = 2.56890649278e-001
Total energy after collision = 2.10186487246e-001
Energy discrepancy = -1.81805613256e+001 %

Figure B.63: Quaternary Test 5 – PFC3D (DEM-Approach) Solution.

B.3.6 Quaternary Test 6

Post-Collision Rigid Body Data:

Number	Pos_X	Pos_Y	Pos_Z	V_X	V_Y	V_Z	Omega_X	Omega_Y	Omega_Z
1	1.5000000000	1.5000000000	1.5000000000	-0.3585072378	0.0000000000	9.7962239143	0.0000000000	20.3776085673	0.0000000000
2	1.5500000000	1.5000000000	1.5000000000	0.5414927622	-0.0000000000	0.2638418508	-0.0000000000	15.1757607185	-0.0000000000
3	1.5933012702	1.5250000000	1.5000000000	0.4085072378	0.2303376849	-0.0300328826	1.5016441279	-2.6009239244	0.4775332365
4	1.5933012702	1.4750000000	1.5000000000	0.4085072378	-0.2303376849	-0.0300328826	-1.5016441279	-2.6009239244	-0.4775332365

Contact Data:

Number	Active	Slipping	Object1	Type1	Object2	Type2	Normal_X	Normal_Y	Normal_Z	Lambda_N	Tangent_X	Tangent_Y	Tangent_Z	Lambda_T
1	true	true	2	19	1	19	-1.0000000000	0.0000000000	0.0000000000	0.6979756127	0.0000000000	0.0000000000	-1.0000000000	0.1046963419
2	true	false	3	19	2	19	-0.8660254038	-0.5000000000	0.0000000000	0.2409359356	-0.0785154100	0.1359926792	0.9875937027	0.0156241715
3	true	false	4	19	2	19	-0.8660254038	0.5000000000	0.0000000000	0.2409359356	-0.0785154100	-0.1359926792	0.9875937027	0.0156241715
4	false	false	4	19	3	19	0.0000000000	1.0000000000	0.0000000000	0.0000000000	-0.0385639334	0.0000000000	0.9992561349	0.0000000000

Solution Data:
3 iterations, converged
Energy:
Total energy before collision = 25.94595558 Joules
Total energy after collision = 24.93510117 Joules
Energy discrepancy = -3.89599992 %

Figure B.64: Quaternary Test 6 Linear Solution.

Post-Collision Rigid Body Data:

Number	Pos_X	Pos_Y	Pos_Z	V_X	V_Y	V_Z	Omega_X	Omega_Y	Omega_Z
1	1.5000000000	1.5000000000	1.5000000000	-0.3487297649	-0.0000000000	9.7976905353	0.0000000000	20.2309464736	-0.0000000000
2	1.5500000000	1.5000000000	1.5000000000	0.5512702351	0.0000000000	0.0717531629	-0.0000000000	31.5374538719	0.0000000000
3	1.5933012702	1.5250000000	1.5000000000	0.3987297649	0.2642078446	0.0652781509	-3.2639075450	5.6532536991	-2.9445822824
4	1.5933012702	1.4750000000	1.5000000000	0.3987297649	-0.2642078446	0.0652781509	3.2639075450	5.6532536991	2.9445822824

Contact Data:

Number	Active	Slipping	Object1	Type1	Object2	Type2	Normal_X	Normal_Y	Normal_Z	Lambda_N	Tangent_X	Tangent_Y	Tangent_Z	Lambda_T
1	true	true	2	19	1	19	-1.0000000000	0.0000000000	0.0000000000	0.6929521300	0.0000000000	-0.0000000000	-1.0000000000	0.1039428195
2	true	true	3	19	2	19	-0.8660254038	-0.5000000000	0.0000000000	0.2452863992	0.2055924983	-0.3560966528	-0.9115519176	0.0367929599
3	true	true	4	89	2	19	-0.8660254038	0.5000000000	0.0000000000	0.2452863992	0.2055924983	0.3560966528	-0.9115519176	0.0367929599
4	false	false	4	89	3	19	0.0000000000	1.0000000000	0.0000000000	0.0000000000	-0.0385639334	0.0000000000	0.9992561349	0.0000000000

Solution Data:
4 iterations, converged
Energy:
Total energy before collision = 25.94595558 Joules
Total energy after collision = 24.98727090 Joules
Energy discrepancy = -3.69492914 %

Figure B.65: Quaternary Test 6 Non-Linear Solution.

Number	V_X	V_Y	V_Z	Omega_X	Omega_Y	Omega_Z
1	-1.48611701739e-001	0.00000000000e+000	9.82606646183e+000	0.00000000000e+000	1.48421875344e+001	0.00000000000e+000
2	-1.22013950340e-001	0.00000000000e+000	2.04273005056e-001	0.00000000000e+000	8.84724308634e+000	0.00000000000e+000
3	5.07001934386e-001	2.07050326972e-001	-3.20336126613e-002	1.59984677456e+000	-2.76859183285e+000	7.52543754150e+000
4	5.07001934386e-001	-2.07050326972e-001	-3.20336126613e-002	-1.59984677456e+000	-2.76859183285e+000	-7.52543754150e+000

Total energy before collision = 2.59459555771e+001
Total energy after collision = 2.50057989416e+001
Energy discrepancy = -3.62351902085e+000 %

Figure B.66: Quaternary Test 6 – PFC3D (DEM-Approach) Solution.

B.3.7 Quaternary Test 7

Post-Collision Rigid Body Data:

Number	Pos_X	Pos_Y	Pos_Z	V_X	V_Y	V_Z	Omega_X	Omega_Y	Omega_Z
1	1.5288675135	1.5000000000	1.5408248290	0.0000000000	-0.0000000000	0.3084925589	-0.0000000000	0.0000000000	0.0000000000
2	1.5000000000	1.5000000000	1.5000000000	-0.2196885377	0.0000000000	-0.4361641863	0.0000000000	7.2444570483	0.0000000000
3	1.5433012702	1.5250000000	1.5000000000	0.1098442688	0.1902558546	-0.4361641863	6.2738838405	-3.6222285242	0.0000000000
4	1.5433012702	1.4750000000	1.5000000000	0.1098442688	-0.1902558546	-0.4361641863	-6.2738838405	-3.6222285242	0.0000000000

Contact Data:

Number	Active	Slipping	Object1	Type1	Object2	Type2	Normal_X	Normal_Y	Normal_Z	Lambda_N	Tangent_X	Tangent_Y	Tangent_Z	Lambda_T
1	true	true	2	19	1	19	0.5773502692	0.0000000000	0.8164965809	0.2481377700	-0.8164965809	0.0000000000	0.5773502692	0.0372206655
2	true	true	3	19	1	19	-0.2886751346	-0.5000000000	0.8164965809	0.2481377700	0.4082482905	0.7071067812	0.5773502692	0.0372206655
3	false	false	3	19	2	19	-0.8660254038	-0.5000000000	0.0000000000	0.0000000000	0.0000000000	-0.0000000000	1.0000000000	0.0000000000
4	true	true	4	19	1	19	-0.2886751346	0.5000000000	0.8164965809	0.2481377700	0.4082482905	-0.7071067812	0.5773502692	0.0372206655
5	false	false	4	19	2	19	-0.8660254038	0.5000000000	0.0000000000	0.0000000000	-0.0000000000	-0.0000000000	1.0000000000	0.0000000000
6	false	false	4	19	3	19	0.0000000000	1.0000000000	0.0000000000	0.0000000000	0.0000000000	0.0000000000	1.0000000000	0.0000000000

Solution Data:
3 iterations, converged
Energy:
Total energy before collision = 0.25689065 Joules
Total energy after collision = 0.21836631 Joules
Energy discrepancy = -14.99639696 %

Figure B.67: Quaternary Test 7 Linear Solution.

Post-Collision Rigid Body Data:

Number	Pos_X	Pos_Y	Pos_Z	V_X	V_Y	V_Z	Omega_X	Omega_Y	Omega_Z
1	1.5288675135	1.5000000000	1.5408248290	0.0000000000	0.0000000000	0.3084925589	0.0000000000	-0.0000000000	-0.0000000000
2	1.5000000000	1.5000000000	1.5000000000	-0.2196885377	-0.0000000000	-0.4361641863	0.0000000000	7.2444570483	-0.0000000000
3	1.5433012702	1.5250000000	1.5000000000	0.1098442688	0.1902558546	-0.4361641863	6.2738838405	-3.6222285242	-0.0000000000
4	1.5433012702	1.4750000000	1.5000000000	0.1098442688	-0.1902558546	-0.4361641863	-6.2738838405	-3.6222285242	0.0000000000

Contact Data:

Number	Active	Slipping	Object1	Type1	Object2	Type2	Normal_X	Normal_Y	Normal_Z	Lambda_N	Tangent_X	Tangent_Y	Tangent_Z	Lambda_T
1	true	true	2	19	1	19	0.5773502692	0.0000000000	0.8164965809	0.2481377700	-0.8164965809	0.0000000000	0.5773502692	0.0372206655
2	true	true	3	19	1	19	-0.2886751346	-0.5000000000	0.8164965809	0.2481377700	0.4082482905	0.7071067812	0.5773502692	0.0372206655
3	false	false	3	19	2	19	-0.8660254038	-0.5000000000	0.0000000000	0.0000000000	0.0000000000	-0.0000000000	1.0000000000	0.0000000000
4	true	true	4	19	1	19	-0.2886751346	0.5000000000	0.8164965809	0.2481377700	0.4082482905	-0.7071067812	0.5773502692	0.0372206655
5	false	false	4	19	2	19	-0.8660254038	0.5000000000	0.0000000000	0.0000000000	-0.0000000000	-0.0000000000	1.0000000000	0.0000000000
6	false	false	4	19	3	19	0.0000000000	1.0000000000	0.0000000000	0.0000000000	0.0000000000	0.0000000000	1.0000000000	0.0000000000

Solution Data:
3 iterations, converged
Energy:
Total energy before collision = 0.25689065 Joules
Total energy after collision = 0.21836631 Joules
Energy discrepancy = -14.99639696 %

Figure B.68: Quaternary Test 7 Non-Linear Solution.

Number	V_X	V_Y	V_Z	Omega_X	Omega_Y	Omega_Z
1	5.23283520711e-012	0.00000000000e+000	4.24174892165e-001	0.00000000000e+000	1.66674879200e-010	0.00000000000e+000
2	-2.16989530293e-001	0.00000000000e+000	-4.21804801850e-001	0.00000000000e+000	6.61380854347e+000	0.00000000000e+000
3	1.08494765144e-001	1.87918445586e-001	-4.21804801839e-001	5.72772621432e+000	-3.30690427165e+000	1.73601960872e-011
4	1.08494765144e-001	-1.87918445586e-001	-4.21804801839e-001	-5.72772621432e+000	-3.30690427165e+000	-1.73601960872e-011

Total energy before collision = 2.56890649278e-001
Total energy after collision = 2.28052663358e-001
Energy discrepancy = -1.12257826436e+001 %

Figure B.69: Quaternary Test 7 – PFC3D (DEM-Approach) Solution.

B.3.8 Quaternary Test 8

Post-Collision Rigid Body Data:

Number	Pos_X	Pos_Y	Pos_Z	V_X	V_Y	V_Z	Omega_X	Omega_Y	Omega_Z
1	1.5288675135	1.5000000000	1.5408248290	0.0303386031	0.0000000000	0.3081356513	-0.0000000000	6.2870396081	-0.0000000000
2	1.5000000000	1.5000000000	1.5000000000	-0.1882174513	0.0000000000	-0.3736822702	0.0000000000	6.2066653823	0.0000000000
3	1.5433012702	1.5250000000	1.5000000000	0.1289394241	0.1985667571	-0.4672266906	7.1484266988	-2.9598128871	0.7148426699
4	1.5433012702	1.4750000000	1.5000000000	0.1289394241	-0.1985667571	-0.4672266906	-7.1484266988	-2.9598128871	-0.7148426699

Contact Data:

Number	Active	Slipping	Object1	Type1	Object2	Type2	Normal_X	Normal_Y	Normal_Z	Lambda_N	Tangent_X	Tangent_Y	Tangent_Z	Lambda_T
1	true	true	2	19	1	19	0.5773502692	0.0000000000	0.8164965809	0.2125912400	-0.8164965809	0.0000000000	0.5773502692	0.0318886860
2	true	true	3	19	1	19	-0.2886751346	-0.5000000000	0.8164965809	0.2661356194	0.2650295626	0.7777469110	0.5699728707	0.0399203429
3	false	false	3	19	2	19	-0.8660254038	-0.5000000000	0.0000000000	0.0000000000	-0.0115623520	0.0200265810	0.9997325883	0.0000000000
4	true	true	4	19	1	19	-0.2886751346	0.5000000000	0.8164965809	0.2661356194	0.2650295626	-0.7777469110	0.5699728707	0.0399203429
5	false	false	4	19	2	19	-0.8660254038	0.5000000000	0.0000000000	0.0000000000	-0.0115623520	-0.0200265810	0.9997325883	0.0000000000
6	false	false	4	19	3	19	0.0000000000	1.0000000000	0.0000000000	0.0000000000	-0.0411719824	0.0000000000	0.9991520744	0.0000000000

Solution Data:
5 iterations, converged
Energy:
Total energy before collision = 0.25945956 Joules
Total energy after collision = 0.22079214 Joules
Energy discrepancy = -14.90305906 %

Figure B.70: Quaternary Test 8 Linear Solution.

Post-Collision Rigid Body Data:

Number	Pos_X	Pos_Y	Pos_Z	V_X	V_Y	V_Z	Omega_X	Omega_Y	Omega_Z
1	1.5288675135	1.5000000000	1.5408248290	0.0540308917	-0.0000000000	0.2986429678	-0.0000000000	-1.8370308200	0.0000000000
2	1.5000000000	1.5000000000	1.5000000000	-0.1855209451	0.0000000000	-0.3683286936	-0.0000000000	6.1177452989	0.0000000000
3	1.5433012702	1.5250000000	1.5000000000	0.1157450267	0.2387443491	-0.4651571371	3.7644623809	-3.9773880595	-1.1047043751
4	1.5433012702	1.4750000000	1.5000000000	0.1157450267	-0.2387443491	-0.4651571371	-3.7644623809	-3.9773880595	1.1047043751

Contact Data:

Number	Active	Slipping	Object1	Type1	Object2	Type2	Normal_X	Normal_Y	Normal_Z	Lambda_N	Tangent_X	Tangent_Y	Tangent_Z	Lambda_T
1	true	true	2	19	1	19	0.5773502692	0.0000000000	0.8164965809	0.2095455416	-0.8164965809	-0.0000000000	0.5773502692	0.0314318312
2	true	false	3	19	1	19	-0.2886751346	-0.5000000000	0.8164965809	0.2736317493	0.6801652362	0.4930947230	0.5424329420	0.0287033957
3	false	false	3	19	2	19	-0.8660254038	-0.5000000000	0.0000000000	0.0000000000	-0.0115623520	0.0200265810	0.9997325883	0.0000000000
4	true	false	4	19	1	19	-0.2886751346	0.5000000000	0.8164965809	0.2736317493	0.6801652362	-0.4930947230	0.5424329420	0.0287033957
5	false	false	4	19	2	19	-0.8660254038	0.5000000000	0.0000000000	0.0000000000	-0.0115623520	-0.0200265810	0.9997325883	0.0000000000
6	false	false	4	19	3	19	0.0000000000	1.0000000000	0.0000000000	0.0000000000	-0.0411719824	0.0000000000	0.9991520744	0.0000000000

Solution Data:
12 iterations, converged
Energy:
Total energy before collision = 0.25945956 Joules
Total energy after collision = 0.22131922 Joules
Energy discrepancy = -14.69991712 %

Figure B.71: Quaternary Test 8 Non-Linear Solution.

Number	V_X	V_Y	V_Z	Omega_X	Omega_Y	Omega_Z
1	9.36457815547e-003	0.00000000000e+000	4.24694554690e-001	0.00000000000e+000	1.80674946564e+000	0.00000000000e+000
2	-1.88960132379e-001	0.00000000000e+000	-3.73416528941e-001	0.00000000000e+000	6.13054895868e+000	0.00000000000e+000
3	1.31304451706e-001	1.91882744017e-001	-4.46340646330e-001	6.65744233725e+000	-2.07516968625e+000	1.02704207108e+000
4	1.31304451706e-001	-1.91882744017e-001	-4.46340646330e-001	-6.65744233725e+000	-2.07516968625e+000	-1.02704207108e+000

Total energy before collision = 2.59459555771e-001
Total energy after collision = 2.30485365265e-001
Energy discrepancy = -1.11671317789e+001 %

Figure B.72: Quaternary Test 8 – PFC3D (DEM-Approach) Solution.

B.3.9 Quaternary Test 9

Post-Collision Rigid Body Data:

Number	Pos_X	Pos_Y	Pos_Z	V_X	V_Y	V_Z	Omega_X	Omega_Y	Omega_Z
1	1.5288675135	1.5000000000	1.5408248290	-0.0151693016	-0.0262740010	0.3081356513	0.2485835925	-0.1435198040	0.0000000000
2	1.5000000000	1.5000000000	1.5000000000	-0.2364335681	-0.0123814382	-0.4672266906	1.0109401997	7.6706255618	-0.7148426699
3	1.5433012702	1.5250000000	1.5000000000	0.0941087257	0.1630010943	-0.3736822702	5.3751298939	-3.1033326912	-0.0000000000
4	1.5433012702	1.4750000000	1.5000000000	0.1074941440	-0.2109481954	-0.4672266906	-6.1374865001	-4.7108126747	0.7148426699

Contact Data:

Number	Active	Slipping	Object1	Type1	Object2	Type2	Normal_X	Normal_Y	Normal_Z	Lambda_N	Tangent_X	Tangent_Y	Tangent_Z	Lambda_T
1	true	true	2	19	1	19	0.5773502692	0.0000000000	0.8164965809	0.2661356194	-0.6060633640	0.1593511215	0.5699728707	0.0399203429
2	true	true	3	19	1	19	-0.2886751346	-0.5000000000	0.8164965809	0.2125912400	0.4082482905	0.7071067812	0.5773502692	0.0318886860
3	false	false	3	19	2	19	-0.8660254038	-0.5000000000	0.0000000000	0.0000000000	-0.0115623520	0.0200265810	0.9997325883	0.0000000000
4	true	true	4	19	1	19	-0.2886751346	0.5000000000	0.8164965809	0.2661356194	0.5410338013	-0.6183957895	0.5699728707	0.0399203429
5	false	false	4	19	2	19	-0.8660254038	0.5000000000	0.0000000000	0.0000000000	0.0205859912	0.0356559827	0.9991520744	0.0000000000
6	false	false	4	19	3	19	0.0000000000	1.0000000000	0.0000000000	0.0000000000	0.0231247039	0.0000000000	0.9997325883	0.0000000000

Solution Data:
5 iterations, converged
Energy:
Total energy before collision = 0.25945956 Joules
Total energy after collision = 0.22079214 Joules
Energy discrepancy = -14.90305906 %

Figure B.73: Quaternary Test 9 Linear Solution.

Post-Collision Rigid Body Data:

Number	Pos_X	Pos_Y	Pos_Z	V_X	V_Y	V_Z	Omega_X	Omega_Y	Omega_Z
1	1.5288675135	1.5000000000	1.5408248290	-0.0270154459	-0.0467921248	0.2986429678	-1.5909153577	0.9185154100	0.0000000000
2	1.5000000000	1.5000000000	1.5000000000	-0.2646311847	0.0191340411	-0.4651571371	-1.5622879097	5.2488140832	1.1047043751
3	1.5433012702	1.5250000000	1.5000000000	0.0927604726	0.1606658514	-0.3683286936	5.2981228427	-3.0588726494	0.0000000000
4	1.5433012702	1.4750000000	1.5000000000	0.1488861580	-0.2196103080	-0.4651571371	-5.3267502907	-1.2714260238	-1.1047043751

Contact Data:

Number	Active	Slipping	Object1	Type1	Object2	Type2	Normal_X	Normal_Y	Normal_Z	Lambda_N	Tangent_X	Tangent_Y	Tangent_Z	Lambda_T
1	true	false	2	19	1	19	0.5773502692	0.0000000000	0.8164965809	0.2736317493	-0.7671151747	-0.3424930119	0.5424323420	0.0287033957
2	true	true	3	19	1	19	-0.2886751346	-0.5000000000	0.8164965809	0.2095455416	0.4082482905	0.7071067812	0.5773502692	0.0314318312
3	false	false	3	19	2	19	-0.8660254038	-0.5000000000	0.0000000000	0.0000000000	-0.0115623520	0.0200265810	0.9997325883	0.0000000000
4	true	false	4	19	1	19	-0.2886751346	0.5000000000	0.8164965809	0.2736317493	0.0869499584	-0.8355877348	0.5424323420	0.0287033957
5	false	false	4	19	2	19	-0.8660254038	0.5000000000	0.0000000000	0.0000000000	0.0205859912	0.0356559827	0.9991520744	0.0000000000
6	false	false	4	19	3	19	0.0000000000	1.0000000000	0.0000000000	0.0000000000	0.0231247039	0.0000000000	0.9997325883	0.0000000000

Solution Data:
12 iterations, converged
Energy:
Total energy before collision = 0.25945956 Joules
Total energy after collision = 0.22131922 Joules
Energy discrepancy = -14.69991712 %

Figure B.74: Quaternary Test 9 Non-Linear Solution.

Number	V_X	V_Y	V_Z	Omega_X	Omega_Y	Omega_Z
1	-4.68228907854e-003	-8.10996258379e-003	4.24694554686e-001	1.56469093539e+000	-9.03374732640e-001	1.05956739015e-011
2	-2.31927556719e-001	-1.77716187991e-002	-4.46340646331e-001	1.45235503062e+000	6.83188052756e+000	-1.02704207104e+000
3	9.44800461904e-002	1.63644274946e-001	-3.73416528941e-001	5.30921113726e+000	-3.06827447927e+000	7.72140368703e-012
4	1.00523105011e-001	-2.09654362814e-001	-4.46340646326e-001	-5.14248783859e+000	-4.68219369338e+000	1.02704207104e+000

Total energy before collision = 2.59459555771e-001
Total energy after collision = 2.30478706030e-001
Energy discrepancy = -1.11696983582e+001 %

Figure B.75: Quaternary Test 9 – PFC3D (DEM-Approach) Solution.

B.3.10 Quaternary Test 10

Post-Collision Rigid Body Data:

Number	Pos_X	Pos_Y	Pos_Z	V_X	V_Y	V_Z	Omega_X	Omega_Y	Omega_Z
1	1.5288675135	1.5000000000	1.5408248290	-0.0151693016	0.0262740010	0.3081356513	-0.2485835925	-0.1435199040	-0.0000000000
2	1.5000000000	1.5000000000	1.5000000000	-0.2364335681	0.0123814382	-0.4672266906	-1.0109401987	7.6706255618	0.7148426699
3	1.5433012702	1.5250000000	1.5000000000	0.1074941440	0.2109481954	-0.4672266906	6.1374865001	-4.7108126747	-0.7148426699
4	1.5433012702	1.4750000000	1.5000000000	0.0941097257	-0.1630010943	-0.3736822702	-5.3751298939	-3.1033326912	0.0000000000

Contact Data:

Number	Active	Slipping	Object1	Type1	Object2	Type2	Normal_X	Normal_Y	Normal_Z	Lambda_N	Tangent_X	Tangent_Y	Tangent_Z	Lambda_T
1	true	true	2	19	1	19	0.5773502692	0.0000000000	0.8164965809	0.2661356194	-0.8080633640	-0.1593511215	0.5699728707	0.0399203429
2	true	true	3	19	1	19	-0.2886751346	-0.5000000000	0.8164965809	0.2661356194	0.5410338013	0.6183957895	0.5699728707	0.0399203429
3	false	false	3	19	2	19	-0.8660254038	-0.5000000000	0.0000000000	0.0000000000	0.0205859912	-0.0356559827	0.9991520744	0.0000000000
4	true	true	4	19	1	19	-0.2886751346	0.5000000000	0.8164965809	0.2125912400	0.4082482905	-0.7071067812	0.5773502692	0.0318886860
5	false	false	4	19	2	19	-0.8660254038	0.5000000000	0.0000000000	0.0000000000	-0.0115623520	-0.0200265810	0.9997325883	0.0000000000
6	false	false	4	19	3	19	0.0000000000	1.0000000000	0.0000000000	0.0000000000	0.0231247039	0.0000000000	0.9997325883	0.0000000000

Solution Data:
5 iterations, converged
Energy:
Total energy before collision = 0.25945956 Joules
Total energy after collision = 0.22079214 Joules
Energy discrepancy = -14.90305906 %

Figure B.76: Quaternary Test 10 Linear Solution.

Post-Collision Rigid Body Data:

Number	Pos_X	Pos_Y	Pos_Z	V_X	V_Y	V_Z	Omega_X	Omega_Y	Omega_Z
1	1.5288675135	1.5000000000	1.5408248290	-0.0270154459	0.0467921248	0.2986429678	1.5909153577	0.9185154100	-0.0000000000
2	1.5000000000	1.5000000000	1.5000000000	-0.2646311847	-0.0191340411	-0.4651571371	1.5622879097	5.2488140832	-1.1047043751
3	1.5433012702	1.5250000000	1.5000000000	0.1488861580	0.2196103080	-0.4651571371	5.3267502907	-1.2714260238	1.1047043751
4	1.5433012702	1.4750000000	1.5000000000	0.0927604726	-0.1606658514	-0.3683286936	-5.2981228427	-3.0588726494	0.0000000000

Contact Data:

Number	Active	Slipping	Object1	Type1	Object2	Type2	Normal_X	Normal_Y	Normal_Z	Lambda_N	Tangent_X	Tangent_Y	Tangent_Z	Lambda_T
1	true	false	2	19	1	19	0.5773502692	0.0000000000	0.8164965809	0.2736317453	-0.7671151747	0.3424930119	0.5424323420	0.0287033957
2	true	false	3	19	1	19	-0.2886751346	-0.5000000000	0.8164965809	0.2736317453	0.0869499384	0.8358877348	0.5424323420	0.0287033957
3	false	false	3	19	2	19	-0.8660254038	-0.5000000000	0.0000000000	0.0000000000	0.0205859912	-0.0356559827	0.9991520744	0.0000000000
4	true	true	4	19	1	19	-0.2886751346	0.5000000000	0.8164965809	0.2095485416	0.4082482905	-0.7071067812	0.5773502692	0.0314318312
5	false	false	4	19	2	19	-0.8660254038	0.5000000000	0.0000000000	0.0000000000	-0.0115623520	-0.0200265810	0.9997325883	0.0000000000
6	false	false	4	19	3	19	0.0000000000	1.0000000000	0.0000000000	0.0000000000	0.0231247039	0.0000000000	0.9997325883	0.0000000000

Solution Data:
12 iterations, converged
Energy:
Total energy before collision = 0.25945956 Joules
Total energy after collision = 0.22131922 Joules
Energy discrepancy = -14.69991712 %

Figure B.77: Quaternary Test 10 Non-Linear Solution.

Number	V_X	V_Y	V_Z	Omega_X	Omega_Y	Omega_Z
1	-4.68228907854e-003	8.10996258379e-003	4.24694554686e-001	-1.56469093539e+000	-9.03374732640e-001	-1.05957252165e-011
2	-2.31927556718e-001	1.77716187991e-002	-4.46340646331e-001	-1.45235503062e+000	6.83188052756e+000	1.02704207104e+000
3	1.00529105011e-001	2.09654362814e-001	-4.46340646326e-001	5.14248783859e+000	-4.68219369338e+000	-1.02704207104e+000
4	9.44800661904e-002	-1.63644274946e-001	-3.73416528941e-001	-5.30921113726e+000	-3.06527447927e+000	-7.72141655501e-012

Total energy before collision = 2.59459555771e-001
Total energy after collision = 2.30478706030e-001
Energy discrepancy = -1.11696983582e+001 %

Figure B.78: Quaternary Test 10 – PFC3D (DEM-Approach) Solution.

B.4 QUINTENARY TEST RESULTS

The various quintenary test results as referred to in section 5.1.4.3 can be seen in this section, with the quantities again as described in section B.1.

B.4.1 Quintenary Test 1

Post-Collision Rigid Body Data:

Number	Pos_X	Pos_Y	Pos_Z	V_X	V_Y	V_Z	Omega_X	Omega_Y	Omega_Z
1	1.5288675135	1.5000000000	1.5908248290	-0.0000000000	0.0000000000	0.4489220223	-0.0000000000	0.0000000000	0.0000000000
2	1.5288675135	1.5000000000	1.5408248290	0.0000000000	0.0000000000	-0.4510779777	0.0000000000	-0.0000000000	0.0000000000
3	1.5000000000	1.5000000000	1.5000000000	-0.1675324002	-0.0000000000	-0.3326146815	0.0000000000	5.5245543983	-0.0000000000
4	1.5433012702	1.5250000000	1.5000000000	0.0837662001	0.1450873145	-0.3326146815	4.7844044535	-2.7622771991	0.0000000000
5	1.5433012702	1.4750000000	1.5000000000	0.0837662001	-0.1450873145	-0.3326146815	-4.7844044535	-2.7622771991	0.0000000000

Contact Data:

Number	Active	Slipping	Object1	Type1	Object2	Type2	Normal_X	Normal_Y	Normal_Z	Lambda_N	Tangent_X	Tangent_Y	Tangent_Z	Lambda_T
1	true	false	2	19	1	19	0.0000000000	0.0000000000	1.0000000000	0.7444290381	0.0000000000	-1.0000000000	0.0000000000	-0.0000000000
2	true	true	3	19	2	19	0.5773502692	0.0000000000	0.8164965809	0.1892275155	-0.8164965809	0.0000000000	0.5773502692	0.0283841273
3	true	true	4	19	2	19	-0.2886751346	-0.5000000000	0.8164965809	0.1892275155	0.4082482905	0.7071067812	0.5773502692	0.0283841273
4	true	true	5	19	2	19	-0.2886751346	0.5000000000	0.8164965809	0.1892275155	0.4082482905	-0.7071067812	0.5773502692	0.0283841273
5	false	false	4	19	3	19	-0.8660254038	-0.5000000000	0.0000000000	0.0000000000	-0.0000000000	0.0000000000	1.0000000000	0.0000000000
6	false	false	5	19	3	19	-0.8660254038	0.5000000000	0.0000000000	0.0000000000	0.0000000000	0.0000000000	1.0000000000	0.0000000000
7	false	false	5	19	4	19	0.0000000000	1.0000000000	0.0000000000	0.0000000000	0.0000000000	0.0000000000	1.0000000000	0.0000000000

Figure B.79: Quintenary Test 1 Linear Solution.

Post-Collision Rigid Body Data:

Number	Pos_X	Pos_Y	Pos_Z	V_X	V_Y	V_Z	Omega_X	Omega_Y	Omega_Z
1	1.5288675135	1.5000000000	1.5908248290	-0.0000000000	-0.0000000000	0.4489220223	-0.0000000000	-0.0000000000	0.0000000000
2	1.5288675135	1.5000000000	1.5408248290	0.0000000000	0.0000000000	-0.4510779777	0.0000000000	0.0000000000	0.0000000000
3	1.5000000000	1.5000000000	1.5000000000	-0.1675324002	-0.0000000000	-0.3326146815	0.0000000000	5.5245543983	-0.0000000000
4	1.5433012702	1.5250000000	1.5000000000	0.0837662001	0.1450873145	-0.3326146815	4.7844044535	-2.7622771991	0.0000000000
5	1.5433012702	1.4750000000	1.5000000000	0.0837662001	-0.1450873145	-0.3326146815	-4.7844044535	-2.7622771991	0.0000000000

Contact Data:

Number	Active	Slipping	Object1	Type1	Object2	Type2	Normal_X	Normal_Y	Normal_Z	Lambda_N	Tangent_X	Tangent_Y	Tangent_Z	Lambda_T
1	true	false	2	19	1	19	0.0000000000	0.0000000000	1.0000000000	0.7444290381	-0.0001110223	-0.9999999938	0.0000000000	0.0000000000
2	true	true	3	19	2	19	0.5773502692	0.0000000000	0.8164965809	0.1892275155	-0.8164965809	-0.0000000000	0.5773502692	0.0283841273
3	true	true	4	19	2	19	-0.2886751346	-0.5000000000	0.8164965809	0.1892275155	0.4082482905	0.7071067812	0.5773502692	0.0283841273
4	true	true	5	19	2	19	-0.2886751346	0.5000000000	0.8164965809	0.1892275155	0.4082482905	-0.7071067812	0.5773502692	0.0283841273
5	false	false	4	19	3	19	-0.8660254038	-0.5000000000	0.0000000000	0.0000000000	-0.0000000000	0.0000000000	1.0000000000	0.0000000000
6	false	false	5	19	3	19	-0.8660254038	0.5000000000	0.0000000000	0.0000000000	0.0000000000	0.0000000000	1.0000000000	0.0000000000
7	false	false	5	19	4	19	0.0000000000	1.0000000000	0.0000000000	0.0000000000	0.0000000000	0.0000000000	1.0000000000	0.0000000000

Figure B.80: Quintenary Test 1 Non-Linear Solution.

Number	V_X	V_Y	V_Z	Omega_X	Omega_Y	Omega_Z
1	8.17466160886e-006	0.00000000000e+000	4.07331649904e-001	0.00000000000e+000	-8.17406090564e-004	0.00000000000e+000
2	-8.99070559972e-005	0.00000000000e+000	8.52046195644e-002	0.00000000000e+000	7.98486637560e-003	0.00000000000e+000
3	-2.07516154092e-001	0.00000000000e+000	-4.11773297033e-001	0.00000000000e+000	6.83673808097e+000	0.00000000000e+000
4	1.03794418213e-001	1.79777205971e-001	-4.11721877975e-001	5.97616209633e+000	-3.45033879886e+000	-2.40440391789e-009
5	1.03794418213e-001	-1.79777205971e-001	-4.11721877975e-001	-5.97616209633e+000	-3.45033879886e+000	2.40440391789e-009

Total energy before collision = 2.56890649278e-001
Total energy after collision = 2.17460531530e-001
Energy discrepancy = -1.53489890965e+001 %

Figure B.81: Quintenary Test 1 – PFC3D (DEM-Approach) Solution.

B.4.2 Quintenary Test 2

Post-Collision Rigid Body Data:

Number	Pos_X	Pos_Y	Pos_Z	V_X	V_Y	V_Z	Omega_X	Omega_Y	Omega_Z
1	1.5288675135	1.5000000000	1.5908248290	0.9038724468	0.4519362234	0.2164923290	-4.8043766588	9.6127533175	0.0000000000
2	1.5288675135	1.5000000000	1.5408248290	0.0800332044	0.0363599732	-0.2335076710	-3.6724999782	6.1360851365	-0.1251913866
3	1.5000000000	1.5000000000	1.5000000000	-0.1007482184	0.0042409495	-0.1056759011	-0.3462720807	-2.1248563890	0.2448513964
4	1.5433012702	1.5250000000	1.5000000000	0.0568174488	0.0893541961	-0.2093859532	3.1735580972	-1.4053265870	0.2614389789
5	1.5433012702	1.4750000000	1.5000000000	0.0600250984	-0.0820913523	-0.1679228036	0.0535149650	-0.6314817018	

Contact Data:

Number	Active	Slipping	Object1	Type1	Object2	Type2	Normal_X	Normal_Y	Normal_Z	Lambda_N	Tangent_X	Tangent_Y	Tangent_Z	Lambda_T
1	true	true	2	19	1	19	0.0000000000	0.0000000000	1.0000000000	0.3681203592	-0.8944271910	-0.4472135955	0.0000000000	0.0552180539
2	true	true	3	19	2	19	0.5773502692	0.0000000000	0.8164965809	0.0742162341	0.8007042133	-0.1957272190	-0.5661833789	0.0111324351
3	true	true	4	19	2	19	-0.2886751346	-0.5000000000	0.8164965809	0.1192187361	0.2921096830	0.7661465291	0.5724433850	0.0178828104
4	true	false	5	19	2	19	-0.2886751346	0.5000000000	0.8164965809	0.1004349056	-0.1987906074	-0.8655195020	0.4597371923	0.0092897401
5	false	false	4	19	3	19	-0.8660254038	-0.5000000000	0.0000000000	0.0000000000	0.1966344124	-0.3405807928	0.9194235321	0.0000000000
6	false	false	5	19	3	19	-0.8660254038	0.5000000000	0.0000000000	0.0000000000	0.4374937942	0.7577614795	-0.4841453503	0.0000000000
7	false	false	5	19	4	19	0.0000000000	1.0000000000	0.0000000000	0.0000000000	0.7122129554	0.0000000000	0.7019634650	0.0000000000

Solution Data:
7 iterations, converged
Energy:
Total energy before collision = 0.38533597 Joules
Total energy after collision = 0.33189859 Joules
Energy discrepancy = -13.86773707 %

Figure B.82: Quintenary Test 2 Linear Solution.

Post-Collision Rigid Body Data:

Number	Pos_X	Pos_Y	Pos_Z	V_X	V_Y	V_Z	Omega_X	Omega_Y	Omega_Z
1	1.5288675135	1.5000000000	1.5908248290	0.9119318179	0.4544905481	0.1608787051	-4.5509451886	8.8068182131	0.0000000000
2	1.5288675135	1.5000000000	1.5408248290	-0.0374894054	0.0455013671	-0.2891212949	-1.0378516096	8.3268402631	-2.0582618836
3	1.5000000000	1.5000000000	1.5000000000	0.0000000000	0.0000000000	0.0000000000	0.0000000000	0.0000000000	0.0000000000
4	1.5433012702	1.5250000000	1.5000000000	0.0421985088	0.1107084137	-0.2210162422	2.0114916501	-2.9346955274	-1.0858569551
5	1.5433012702	1.4750000000	1.5000000000	0.0833590787	-0.1107005289	-0.1507411681	1.5016019290	2.4547175773	-0.9723049285

Contact Data:

Number	Active	Slipping	Object1	Type1	Object2	Type2	Normal_X	Normal_Y	Normal_Z	Lambda_N	Tangent_X	Tangent_Y	Tangent_Z	Lambda_T
1	true	true	2	19	1	19	0.0000000000	0.0000000000	1.0000000000	0.3395471193	-0.8883948138	-0.4590802269	0.0000000000	0.0509320679
2	false	false	3	19	2	19	0.5773502692	0.0000000000	0.8164965809	0.0000000000	0.8007042133	-0.1957272190	-0.5661833789	0.0000000000
3	true	true	4	19	2	19	-0.2886751346	-0.5000000000	0.8164965809	0.1274151793	0.7901093867	0.3572350058	0.4981067232	0.0191122769
4	true	true	5	19	2	19	-0.2886751346	0.5000000000	0.8164965809	0.1040574279	-0.8199175674	0.3112436003	-0.4804816376	0.0156056142
5	false	false	4	19	3	19	-0.8660254038	-0.5000000000	0.0000000000	0.0000000000	0.1966344124	-0.3405807928	0.9194235321	0.0000000000
6	false	false	5	19	3	19	-0.8660254038	0.5000000000	0.0000000000	0.0000000000	0.4374937942	0.7577614795	-0.4841453503	0.0000000000
7	false	false	5	19	4	19	0.0000000000	1.0000000000	0.0000000000	0.0000000000	0.7122129554	0.0000000000	0.7019634650	0.0000000000

Solution Data:
9 iterations, converged
Energy:
Total energy before collision = 0.38533597 Joules
Total energy after collision = 0.33522635 Joules
Energy discrepancy = -13.00413949 %

Figure B.83: Quintenary Test 2 Non-Linear Solution.

Number	V_X	V_Y	V_Z	Omega_X	Omega_Y	Omega_Z
1	9.98656233016e-001	4.49314970210e-001	2.00202512357e-001	-4.09112947792e+000	8.18021907381e+000	2.02571533025e-006
2	7.55979710133e-002	3.80940612044e-002	4.70397328364e-002	-2.34211958457e+000	4.60387277959e+000	-2.38621660992e-002
3	-9.30991935807e-002	-7.64566673158e-003	-1.60094271784e-001	6.23882855319e-001	1.59161353413e+000	-4.41291993305e-001
4	5.64403329076e-002	1.10233630821e-001	-2.44395997214e-001	3.25293335444e+000	-2.46950553131e+000	-3.62334296885e-001
5	4.24152716526e-002	-1.00149900922e-001	-2.07400333801e-001	-2.22173354139e+000	-2.54818964783e+000	7.75245101729e-001

Total energy before collision = 3.85335973917e-001
Total energy after collision = 3.23363056732e-001
Energy discrepancy = -1.60828267746e+001 %

Figure B.84: Quintenary Test 2 – PFC3D (DEM-Approach) Solution.

B.4.3 Quintenary Test 3

Post-Collision Rigid Body Data:

Number	Pos_X	Pos_Y	Pos_Z	V_X	V_Y	V_Z	Omega_X	Omega_Y	Omega_Z
1	1.5288675135	1.5000000000	1.5908248290	0.8474995271	-0.8474995271	0.4377892471	15.2500472915	15.2500472915	0.0000000000
2	1.5288675135	1.5000000000	1.5408248290	0.1103692630	-0.1358496119	-0.4622117529	9.1947853391	13.2250799990	0.7594608760
3	1.5000000000	1.5000000000	1.5000000000	-0.1726482225	0.0080507890	-0.2620881697	-0.6573441680	1.0349991935	0.4648125188
4	1.5433012702	1.5250000000	1.5000000000	0.1032084477	0.1615972715	-0.2825962716	0.9384516191	0.2688378395	0.4953609742
5	1.5433012702	1.4750000000	1.5000000000	0.1115739848	-0.1862989214	-0.4308920529	-6.3393594035	-8.3298044254	-0.2007126170

Contact Data:

Number	Active	Slipping	Object1	Type1	Object2	Type2	Normal_X	Normal_Y	Normal_Z	Lambda_N	Tangent_X	Tangent_Y	Tangent_Z	Lambda_T
1	true	true	2	19	1	19	0.0000000000	0.0000000000	1.0000000000	0.7387087126	-0.7071067812	0.7071067812	0.0000000000	0.1108863069
2	true	false	3	19	2	19	0.5773502692	0.0000000000	0.8164965809	0.1611591117	0.6444793879	0.6139784017	-0.4557157455	-0.0067369549
3	true	false	4	19	2	19	-0.2886751346	-0.5000000000	0.8164965809	0.1753691617	-0.4277787059	0.8303038828	0.3572125992	0.0056111081
4	true	true	5	19	2	19	-0.2886751346	0.5000000000	0.8164965809	0.2451661517	0.3657042364	-0.7305569134	0.5766688892	0.0367749228
5	false	false	4	19	3	19	-0.8660254038	-0.5000000000	0.0000000000	0.0000000000	0.4522573055	-0.7833326312	-0.4264426322	0.0000000000
6	false	false	5	19	3	19	-0.8660254038	0.5000000000	0.0000000000	0.0000000000	-0.2312448434	-0.4005278177	0.8866246611	0.0000000000
7	false	false	5	19	4	19	0.0000000000	1.0000000000	0.0000000000	0.0000000000	0.6419934792	0.0000000000	0.7667100969	0.0000000000

Solution Data:
  6 iterations, converged
  Energy:
    Total energy before collision =   0.77067195 Joules
    Total energy after collision =   0.64615240 Joules
    Energy discrepancy =   -16.15727062 %

Figure B.85: Quintenary Test 3 Linear Solution.

Post-Collision Rigid Body Data:

Number	Pos_X	Pos_Y	Pos_Z	V_X	V_Y	V_Z	Omega_X	Omega_Y	Omega_Z
1	1.5288675135	1.5000000000	1.5908248290	0.8590910351	-0.8384154869	0.4292949508	16.1584513120	14.0908964906	0.0000000000
2	1.5288675135	1.5000000000	1.5408248290	0.0942357335	-0.1449185005	-0.4707050492	8.3222496204	13.6611644013	2.3752858142
3	1.5000000000	1.5000000000	1.5000000000	-0.1888065884	-0.0026644599	-0.2705639039	0.2175522360	0.2050208868	-0.1538326613
4	1.5433012702	1.5250000000	1.5000000000	0.1389891945	0.1863163328	-0.2520432296	-2.6105033923	4.0725588977	1.5709704800
5	1.5433012702	1.4750000000	1.5000000000	0.0964906253	-0.2003178856	-0.4359827681	-5.4432515353	-4.7073118638	0.9581479955

Contact Data:

Number	Active	Slipping	Object1	Type1	Object2	Type2	Normal_X	Normal_Y	Normal_Z	Lambda_N	Tangent_X	Tangent_Y	Tangent_Z	Lambda_T
1	true	true	2	19	1	19	0.0000000000	0.0000000000	1.0000000000	0.7343480159	-0.6572422990	0.7536793486	0.0000000000	0.1101517524
2	true	false	3	19	2	19	0.5773502692	0.0000000000	0.8164965809	0.1695077727	-0.4979220120	0.7925342295	0.3520840312	0.0017273066
3	true	true	4	19	2	19	-0.2886751346	-0.5000000000	0.8164965809	0.1742095338	-0.8082255039	-0.3299125131	-0.4877800740	0.0261314301
4	true	true	5	19	2	19	-0.2886751346	0.5000000000	0.8164965809	0.2486659491	0.5954064790	-0.5740872108	0.5620631630	0.0372998924
5	false	false	4	19	3	19	-0.8660254038	-0.5000000000	0.0000000000	0.0000000000	0.4522573055	-0.7833326312	-0.4264426322	0.0000000000
6	false	false	5	19	3	19	-0.8660254038	0.5000000000	0.0000000000	0.0000000000	-0.2312448434	-0.4005278177	0.8866246611	0.0000000000
7	false	false	5	19	4	19	0.0000000000	1.0000000000	0.0000000000	0.0000000000	0.6419934792	0.0000000000	0.7667100969	0.0000000000

Solution Data:
  7 iterations, converged
  Energy:
    Total energy before collision =   0.77067195 Joules
    Total energy after collision =   0.65317518 Joules
    Energy discrepancy =   -15.24601630 %

Figure B.86: Quintenary Test 3 Non-Linear Solution.

Number	V_X	V_Y	V_Z	Omega_X	Omega_Y	Omega_Z
1	8.39814446776e-001	-8.39536093470e-001	4.02893249190e-001	1.29682649774e+001	1.29465590907e+001	-3.43640172990e-005
2	1.19252535862e-001	-1.20819100136e-001	9.52120866311e-002	7.32070103074e+000	7.41913981930e+000	1.32306504993e-001
3	-1.88406304953e-001	2.48965491049e-002	-3.40873634656e-001	-2.03020229794e+000	4.25921174199e+000	1.43652782600e+000
4	7.64528229244e-002	1.90158054530e-001	-3.83566510304e-001	3.71075349442e+000	-4.87871172670e+000	-1.67487650245e+000
5	1.21195165732e-001	-2.22768070427e-001	-5.02558271405e-001	-7.01094740266e+000	-4.65586673802e+000	3.72689545464e-001

Total energy before collision = 7.70671947834e-001
Total energy after collision = 6.20047936604e-001
Energy discrepancy = -1.95445042022e+001 %

Figure B.87: Quintenary Test 3 – PFC3D (DEM-Approach) Solution.

B.5 HEXENARY TEST RESULTS

The various hexenary test results as referred to in section 5.1.5.3 can be seen in this section, with the quantities again as described in section B.1.

B.5.1 Hexenary Test 1

Post-Collision Rigid Body Data:

Number	Pos_X	Pos_Y	Pos_Z	V_X	V_Y	V_Z	Omega_X	Omega_Y	Omega_Z
1	1.5289675135	1.5000000000	1.5908248290	0.0000000000	0.0000000000	0.4052161554	-0.0000000000	0.0000000000	0.0000000000
2	1.5000000000	1.5000000000	1.5000000000	-0.3022748745	-0.0000000000	-0.2810432311	-0.0000000000	-6.6093633326	0.0000000000
3	1.5433012702	1.5250000000	1.5000000000	0.1511374372	0.2617777202	-0.2810432311	-5.7238765488	3.3046816663	-0.0000000000
4	1.5433012702	1.4750000000	1.5000000000	0.1511374372	-0.2617777202	-0.2810432311	5.7238765488	3.3046816663	-0.0000000000
5	1.5288675135	1.5000000000	1.5408248290	-0.0000000000	-0.0000000000	-0.4947838446	0.0000000000	-0.0000000000	-0.0000000000
6	1.5288675135	1.5000000000	1.4591751710	-0.0000000000	-0.0000000000	-0.0673026175	-0.0000000000	0.0000000000	0.0000000000

Contact Data:

Number	Active	Slipping	Object1	Type1	Object2	Type2	Normal_X	Normal_Y	Normal_Z	Lambda_N	Tangent_X	Tangent_Y	Tangent_Z	Lambda_T
1	false	false	3	19	2	19	-0.8660254038	-0.5000000000	0.0000000000	0.0000000000	0.0000000000	-0.0000000000	1.0000000000	0.0000000000
2	true	false	5	19	1	19	0.0000000000	0.0000000000	1.0000000000	0.7219737811	0.9999999615	-0.0002775557	0.0000000000	0.0000000000
3	false	false	4	19	2	19	-0.8660254038	0.5000000000	0.0000000000	0.0000000000	0.0000000000	0.0000000000	1.0000000000	0.0000000000
4	true	true	5	19	2	19	-0.5773502692	0.0000000000	-0.8164965809	0.2136214614	-0.8164965809	-0.0000000000	0.5773502692	0.0320432192
5	true	true	6	19	2	19	-0.5773502692	0.0000000000	0.8164965809	0.0127630236	0.8164965809	-0.0000000000	0.5773502692	0.0019144535
6	false	false	4	19	3	19	0.0000000000	1.0000000000	0.0000000000	0.0000000000	0.0000000000	0.0000000000	1.0000000000	0.0000000000
7	true	true	5	19	3	19	0.2886751346	0.5000000000	-0.8164965809	0.2136214614	0.4082482905	0.7071067812	0.5773502692	0.0320432192
8	true	true	6	19	3	19	0.2886751346	0.5000000000	0.8164965809	0.0127630236	-0.4082482905	-0.7071067812	0.5773502692	0.0019144535
9	true	true	5	19	4	19	0.2886751346	-0.5000000000	-0.8164965809	0.2136214614	0.4082482905	-0.7071067812	0.5773502692	0.0320432192
10	true	true	6	19	4	19	0.2886751346	-0.5000000000	0.8164965809	0.0127630236	-0.4082482905	0.7071067812	0.5773502692	0.0019144535

Solution Data:
4 iterations, converged
Energy:
Total energy before collision = 0.25689065 Joules
Total energy after collision = 0.24593935 Joules
Energy discrepancy = -4.26302170 %

Figure B.88: Hexenary Test 1 Non-Linear Solution.

Number	V_X	V_Y	V_Z	Omega_X	Omega_Y	Omega_Z
1	1.79016458912e-012	0.00000000000e+000	4.07057036097e-001	0.00000000000e+000	-1.78972440936e-010	0.00000000000e+000
2	-2.61995016998e-001	0.00000000000e+000	-3.03768766978e-001	0.00000000000e+000	5.04474486495e+000	0.00000000000e+000
3	1.30997508496e-001	2.26894340381e-001	-3.03768766977e-001	4.36887720867e+000	-2.52237243242e+000	8.00538585556e-011
4	1.30997508496e-001	-2.26894340381e-001	-3.03768766977e-001	-4.36887720867e+000	-2.52237243242e+000	-8.00538585556e-011
5	3.07155765078e-013	0.00000000000e+000	9.36954138036e-002	0.00000000000e+000	3.01158092428e-011	0.00000000000e+000
6	3.43412619917e-012	0.00000000000e+000	-3.15910597180e-001	0.00000000000e+000	-1.11219132239e-010	0.00000000000e+000

Total energy before collision = 2.56890649278e-001
Total energy after collision = 1.99377856166e-001
Energy discrepancy = -2.23880445917e+001 %

Figure B.89: Hexenary Test 1 – PFC3D (DEM-Approach) Solution.

B.6 HEPTENARY TEST RESULTS

The various heptenary test results as referred to in section 5.1.6.3 can be seen in this section, with the quantities again as described in section B.1.

B.6.1 Heptenary Test 1

Post-Collision Rigid Body Data:

Number	Pos_X	Pos_Y	Pos_Z	V_X	V_Y	V_Z	Omega_X	Omega_Y	Omega_Z
1	1.4500000000	1.5000000000	1.5000000000	-0.5630264910	-0.0000000000	0.0000000000	0.0000000000	0.0000000000	-0.0000000000
2	1.5000000000	1.5000000000	1.5000000000	0.3369735090	-0.0000000000	-0.0000000000	-0.0000000000	-0.0000000000	0.0000000000
3	1.5433012702	1.5250000000	1.5000000000	0.2711605043	0.1139914679	0.0000000000	-0.0000000000	0.0000000000	5.2650403754
4	1.5433012702	1.4750000000	1.5000000000	0.2711605043	-0.1139914679	0.0000000000	0.0000000000	0.0000000000	-5.2650403754
5	1.5866025404	1.5500000000	1.5000000000	0.2391922369	0.1693621314	-0.0000000000	0.0000000000	-0.0000000000	-2.7075789790
6	1.5866025404	1.5000000000	1.5000000000	0.2053474996	-0.0000000000	0.0000000000	-0.0000000000	0.0000000000	-0.0000000000
7	1.5866025404	1.4500000000	1.5000000000	0.2391922369	-0.1693621314	-0.0000000000	-0.0000000000	-0.0000000000	2.7075789790

Contact Data:

Number	Active	Slipping	Object1	Type1	Object2	Type2	Normal_X	Normal_Y	Normal_Z	Lambda_N	Tangent_X	Tangent_Y	Tangent_Z	Lambda_T
1	true	false	2	19	1	19	-1.0000000000	0.0000000000	0.0000000000	0.8030537802	0.0000000000	-0.9058688383	-0.4238647329	0.0000000000
2	true	false	3	19	2	19	-0.8660254038	-0.5000000000	0.0000000000	0.3351821940	-0.5000000000	0.8660254038	-0.0000000000	0.0493705034
3	true	false	4	19	2	19	-0.8660254038	0.5000000000	0.0000000000	0.3351821940	-0.5000000000	-0.8660254038	-0.0000000000	0.0493705034
4	false	false	4	19	3	19	0.0000000000	1.0000000000	0.0000000000	0.0000000000	-1.0000000000	0.0000000000	0.0000000000	0.0000000000
5	true	true	6	19	3	19	-0.8660254038	0.5000000000	0.0000000000	0.0560578410	-0.5000000000	-0.8660254038	-0.0000000000	0.0084086762
6	true	true	6	19	4	19	-0.8660254038	-0.5000000000	0.0000000000	0.0560578410	-0.5000000000	0.8660254038	-0.0000000000	0.0084086762
7	true	false	7	19	4	19	-0.8660254038	0.5000000000	0.0000000000	0.1499355731	0.5000000000	0.8660254038	0.0000000000	0.0139110344
8	false	false	7	19	6	19	0.0000000000	1.0000000000	0.0000000000	0.0000000000	-1.0000000000	0.0000000000	0.0000000000	0.0000000000
9	true	false	5	19	3	19	-0.8660254038	-0.5000000000	0.0000000000	0.1499355731	0.5000000000	-0.8660254038	0.0000000000	0.0139110344
10	false	false	6	19	5	19	0.0000000000	1.0000000000	0.0000000000	0.0000000000	-1.0000000000	0.0000000000	0.0000000000	0.0000000000

Solution Data:
4 iterations, converged
Energy:
Total energy before collision = 0.25689065 Joules
Total energy after collision = 0.21452436 Joules
Energy discrepancy = -16.49195489 %

Figure B.90: Heptenary Test 1 Non-Linear Solution.

Number	V_X	V_Y	V_Z	Omega_X	Omega_Y	Omega_Z
1	-2.75953740226e-001	0.00000000000e+000	0.00000000000e+000	0.00000000000e+000	0.00000000000e+000	0.00000000000e+000
2	-2.22283465522e-001	0.00000000000e+000	0.00000000000e+000	0.00000000000e+000	0.00000000000e+000	0.00000000000e+000
3	-2.14135261953e-002	3.01704853521e-003	0.00000000000e+000	0.00000000000e+000	0.00000000000e+000	1.26150909485e+000
4	-2.14135261953e-002	-3.01704853521e-003	0.00000000000e+000	0.00000000000e+000	0.00000000000e+000	-1.26150909485e+000
5	3.87622625920e-001	2.78780854870e-001	0.00000000000e+000	0.00000000000e+000	0.00000000000e+000	-4.75972447741e+000
6	3.63545885430e-001	0.00000000000e+000	0.00000000000e+000	0.00000000000e+000	0.00000000000e+000	0.00000000000e+000
7	3.87622625920e-001	-2.78780854870e-001	0.00000000000e+000	0.00000000000e+000	0.00000000000e+000	4.75972447741e+000

Total energy before collision = 2.56890649278e-001
Total energy after collision = 1.86688759853e-001
Energy discrepancy = -2.73275378542e+001 %

Figure B.91: Heptenary Test 1 – PFC3D (DEM-Approach) Solution.

B.6.2 Heptenary Test 2

Post-Collision Rigid Body Data:

Number	Pos_X	Pos_Y	Pos_Z	V_X	V_Y	V_Z	Omega_X	Omega_Y	Omega_Z
1	1.4500000000	1.5000000000	1.5000000000	-0.5263888418	-0.0000000000	9.7710416737	0.0000000000	22.8958326274	-0.0000000000
2	1.5000000000	1.5000000000	1.5000000000	0.3736111582	-0.0000000000	0.0333073071	-0.0000000000	39.8397079148	0.0000000000
3	1.5433012702	1.5250000000	1.5000000000	0.2407130889	0.2301862082	0.1182580894	-3.7915368945	6.7024243260	-3.5963872207
4	1.5433012702	1.4750000000	1.5000000000	0.2407130889	-0.2301862082	0.1182580894	3.7915368945	6.7024243260	3.5963872207
5	1.5866025404	1.5500000000	1.5000000000	0.2817682431	0.1590765953	-0.0212136757	1.0606837875	-1.8371582107	0.3119748885
6	1.5866025404	1.5000000000	1.5000000000	0.1078150197	0.0000000000	0.0015621919	0.0000000000	0.1352897858	0.0000000000
7	1.5866025404	1.4500000000	1.5000000000	0.2817682431	-0.1590765953	-0.0212136757	-1.0606837875	-1.8371582107	-0.3119748885

Contact Data:

Number	Active	Slipping	Object1	Type1	Object2	Type2	Normal_X	Normal_Y	Normal_Z	Lambda_N	Tangent_X	Tangent_Y	Tangent_Z	Lambda_T
1	true	true	2	19	1	19	-1.0000000000	0.0000000000	0.0000000000	0.7842300413	0.0000000000	-0.0000000000	-1.0000000000	0.1176345062
2	true	true	3	19	2	19	-0.8660254038	-0.5000000000	0.0000000000	0.3510047142	0.1489285162	-0.2579517567	-0.9546104903	0.0526507071
3	true	true	4	19	2	19	-0.8660254038	0.5000000000	0.0000000000	0.3510047142	0.1489285162	0.2579517567	-0.9546104903	0.0526507071
4	false	false	4	19	3	19	0.0000000000	1.0000000000	0.0000000000	0.0000000000	-0.1077493899	0.0000000000	-0.9941780872	0.0000000000
5	true	true	6	19	3	19	-0.8660254038	0.5000000000	0.0000000000	0.0294421447	-0.4979313678	-0.8624424277	-0.0908703020	0.0044163217
6	true	true	6	19	4	19	-0.8660254038	-0.5000000000	0.0000000000	0.0294421447	-0.4979313678	0.8624424277	-0.0908703020	0.0044163217
7	true	false	7	19	4	19	-0.8660254038	0.5000000000	0.0000000000	0.1662374093	-0.0727490652	-0.1260050771	0.9893585265	0.0110164208
8	false	false	7	19	6	19	0.0000000000	1.0000000000	0.0000000000	0.0000000000	-0.0387888342	0.0000000000	0.9992474300	0.0000000000
9	true	false	5	19	3	19	-0.8660254038	-0.5000000000	0.0000000000	0.1662374093	-0.0727490652	0.1260050771	0.9893585265	0.0110164208
10	false	false	6	19	5	19	0.0000000000	1.0000000000	0.0000000000	0.0000000000	-0.0387888342	0.0000000000	0.9992474300	0.0000000000

Solution Data:
  6 iterations, converged
  Energy:
    Total energy before collision =   25.94595558 Joules
    Total energy after collision =   24.90016941 Joules
    Energy discrepancy =   -4.03063268 %

Figure B.92: Heptenary Test 2 Non-Linear Solution.

Number	V_X	V_Y	V_Z	Omega_X	Omega_Y	Omega_Z
1	-1.502401401 15e-001	0.0000000000e+000	9.8257879385 1e+000	0.0000000000e+000	1.4866562945 4e+001	0.0000000000e+000
2	-2.5744067487 1e-001	0.0000000000e+000	2.0667392408 0e-001	0.0000000000e+000	8.6412607530 0e+000	0.0000000000e+000
3	-2.1377222582 6e-002	1.1154278265 1e-002	-4.4429103687 2e-002	1.0206270695 6e+000	-1.7821451719 7e+000	-9.0092311269 7e-001
4	-2.1377222582 6e-002	-1.1154278265 1e-002	-4.4429103687 2e-002	-1.0206270695 6e+000	-1.7821451719 7e+000	-9.0092311269 7e-001
5	3.6098094423 8e-001	2.5664506554 1e-001	1.1401788294 9e-002	-5.6939751656 0e-001	9.8639415409 0e-001	-4.1752929891 2e+000
6	3.3472567630 6e-001	0.0000000000 e+000	-1.6613805760 5e-004	0.0000000000 e+000	-1.5049329631 1e-002	0.0000000000 e+000
7	3.6098094423 8e-001	-2.5664506554 1e-001	1.1401788294 9e-002	5.6939751656 0e-001	9.8639415409 0e-001	4.1752929891 2e+000

Total energy before collision = 2.59459555771e+001
Total energy after collision = 2.49885836721e+001
Energy discrepancy = -3.68986951398e+000 %

Figure B.93: Heptenary Test 2 – PFC3D (DEM-Approach) Solution.

B.6.3 Heptenary Test 3

Post-Collision Rigid Body Data:

Number	Pos_X	Pos_Y	Pos_Z	V_X	V_Y	V_Z	Omega_X	Omega_Y	Omega_Z
1	1.4500000000	1.5000000000	1.5000000000	-0.5080310293	0.4085721454	0.7930954494	0.0000000000	20.6904550645	-9.1427854585
2	1.5000000000	1.5000000000	1.5000000000	0.3919689707	0.0209485843	0.2108122053	-5.0493046686	20.3520424161	-14.2061177313
3	1.5433012702	1.5250000000	1.5000000000	0.2196162257	0.3194722956	0.0792449811	-0.9894741274	1.5444229088	-4.7283571280
4	1.5433012702	1.4750000000	1.5000000000	0.2579680881	-0.2111477526	-0.0636854887	-2.0604128697	-3.5687397732	3.4960471212
5	1.5866025404	1.5500000000	1.5000000000	0.3246741275	0.1375066717	-0.0297377493	1.4868874627	-2.5753646306	4.3252792851
6	1.5866025404	1.5000000000	1.5000000000	0.0241441918	-0.0190951985	-0.0009680115	-0.0484005727	-0.0838322510	0.4464831051
7	1.5866025404	1.4500000000	1.5000000000	0.2896594255	-0.1562567460	0.0112396156	0.5619307824	0.9732926654	-0.9507401240

Contact Data:

Number	Active	Slipping	Object1	Type1	Object2	Type2	Normal_X	Normal_Y	Normal_Z	Lambda_N	Tangent_X	Tangent_Y	Tangent_Z	Lambda_T
1	true	true	2	19	1	19	-1.0000000000	0.0000000000	0.0000000000	0.7747981405	0.0000000000	-0.4041820219	-0.9146785737	0.1162197211
2	true	true	3	19	2	19	-0.8660254038	-0.5000000000	0.0000000000	0.3654118644	0.4482495223	-0.7711947948	-0.4549051112	0.0548117797
3	true	false	4	19	2	19	-0.8660254038	0.5000000000	0.0000000000	0.5380493590	0.5229276390	0.5593270779	0.7634598614	0.0352948778
4	false	false	4	19	3	19	0.0000000000	1.0000000000	0.0000000000	0.0000000000	-0.7349501749	0.0000000000	-0.6781211104	0.0000000000
5	true	true	6	19	3	19	-0.8660254038	0.5000000000	0.0000000000	0.0156482795	0.4898472648	0.8463619897	0.2118853518	0.0023472419
6	false	false	6	19	4	19	-0.8660254038	-0.5000000000	0.0000000000	0.0000000000	-0.4752450850	0.8231486332	0.3107546248	0.0000000000
7	true	false	7	19	4	19	-0.8660254038	0.5000000000	0.0000000000	0.1690241795	-0.5229276390	-0.5593270779	-0.7634598614	0.0075631881
8	false	false	7	19	6	19	0.0000000000	1.0000000000	0.0000000000	0.0000000000	0.8145425544	0.0000000000	0.5801038071	0.0000000000
9	true	true	5	19	3	19	-0.8660254038	-0.5000000000	0.0000000000	0.1797871704	-0.4120144953	0.7136300392	0.5665476350	0.0269680756
10	false	false	6	19	5	19	0.0000000000	1.0000000000	0.0000000000	0.0000000000	-0.9089689237	0.0000000000	0.4168638816	0.0000000000

Solution Data:
  10 iterations, converged
  Energy:
    Total energy before collision =   0.57800396 Joules
    Total energy after collision =   0.53135320 Joules
    Energy discrepancy =   -8.07101114 %

Figure B.94: Heptenary Test 3 Non-Linear Solution.

Number	V_X	V_Y	V_Z	Omega_X	Omega_Y	Omega_Z
1	-2.68752671786e-001	4.07750182239e-001	8.18200216099e-001	-1.24758151066e-004	1.47501357621e+001	-7.48082284057e+000
2	-2.37286130715e-001	4.66472405282e-002	2.05071427955e-001	-4.94359855577e-001	9.32657142316+000	-3.36042060092e+000
3	-9.86435287282e-003	1.33604473105e-002	-3.35389595537e-002	7.18299465789e-001	-1.25730927741e+000	1.32817910729e+000
4	-2.32116237214e-002	-1.83129561449e-002	-4.69017493499e-002	-1.02940417164e+000	-1.79315448179e+000	1.76955303475e-001
5	4.20542346558e-001	3.01675751768e-001	8.94984992729e-003	-4.46822000178e-001	7.74094614020e-001	-5.09626834708e+000
6	3.51114981793e-001	-1.74666525026e-002	-1.39658047222e-004	-9.67766197863e-004	-1.27189224256e-002	-7.07527879168e-001
7	3.69290252036e-001	-2.57594191138e-001	1.24305395075e-002	6.20773892295e-001	1.07539131583e+000	3.94235811890e+000

Total energy before collision = 5.78003960875e-001
Total energy after collision = 4.40134281975e-001
Energy discrepancy = -2.38527221668e+001 %

Figure B.95: Heptenary Test 3 – PFC3D (DEM-Approach) Solution.

B.6.4 Heptenary Test 4

Post-Collision Rigid Body Data:

Number	Pos_X	Pos_Y	Pos_Z	V_X	V_Y	V_Z	Omega_X	Omega_Y	Omega_Z
1	1.4500000000	1.5000000000	1.5000000000	-0.5080310293	-0.4085721454	0.7930954494	0.0000000000	20.6904550645	9.1427854583
2	1.5000000000	1.5000000000	1.5000000000	0.3919689707	-0.0209485843	0.2108122033	5.0493044686	20.3520424161	14.2061177313
3	1.5433012702	1.5250000000	1.5000000000	0.2579680881	0.2111477526	-0.0636854887	2.0604128687	-3.5687397732	-3.4860471212
4	1.5433012702	1.4750000000	1.5000000000	0.2196162257	-0.3194722956	0.0792449811	0.9884741274	1.5444229088	4.7283571290
5	1.5866025404	1.5500000000	1.5000000000	0.2896594255	0.1562567460	0.0112386156	-0.5619307824	0.9732926654	0.9507401240
6	1.5866025404	1.5000000000	1.5000000000	0.0241441918	0.0190951985	-0.0009680115	0.0484005727	-0.0838322510	-0.4464831051
7	1.5866025404	1.4500000000	1.5000000000	0.3246741275	-0.1375066717	-0.0297377493	-1.4868874627	-2.5753646306	-4.3252792851

Contact Data:

Number	Active	Slipping	Object1	Type1	Object2	Type2	Normal_X	Normal_Y	Normal_Z	Lambda_N	Tangent_X	Tangent_Y	Tangent_Z	Lambda_T
1	true	true	2	19	1	19	-1.0000000000	0.0000000000	0.0000000000	0.7747981405	0.0000000000	0.4041820219	-0.9146785737	0.1162197211
2	true	false	3	19	2	19	-0.8660254038	-0.5000000000	0.0000000000	0.3380483590	0.3229276390	-0.5593270779	0.7634598614	0.0352948778
3	true	true	4	19	2	19	-0.8660254038	0.5000000000	0.0000000000	0.3654118644	0.4452495223	0.7711947948	-0.4549851112	0.0548117797
4	false	false	4	19	3	19	0.0000000000	1.0000000000	0.0000000000	0.0000000000	-0.7348501749	0.0000000000	-0.6781211104	0.0000000000
5	false	false	6	19	3	19	-0.8660254038	0.5000000000	0.0000000000	0.0000000000	-0.4752450850	-0.8231486332	0.3107546248	0.0000000000
6	true	true	6	19	4	19	-0.8660254038	-0.5000000000	0.0000000000	0.0136482795	0.4866472648	-0.8463618897	0.2118853518	0.0023472419
7	true	true	7	19	4	19	-0.8660254038	0.5000000000	0.0000000000	0.1797871704	-0.4120144953	-0.7136300392	0.5665476350	0.0269680756
8	false	false	7	19	6	19	0.0000000000	1.0000000000	0.0000000000	0.0000000000	-0.9089689237	0.0000000000	0.4168638816	0.0000000000
9	true	false	5	19	3	19	-0.8660254038	-0.5000000000	0.0000000000	0.1690241795	-0.3229276390	0.5593270779	-0.7634598614	0.0075631881
10	false	false	6	19	5	19	0.0000000000	1.0000000000	0.0000000000	0.0000000000	0.8145425544	0.0000000000	0.5801038071	0.0000000000

Solution Data:
10 iterations, converged
Energy:
Total energy before collision = 0.57800396 Joules
Total energy after collision = 0.53135320 Joules
Energy discrepancy = -8.07101114 %

Figure B.96: Heptenary Test 4 Non-Linear Solution.

Number	V_X	V_Y	V_Z	Omega_X	Omega_Y	Omega_Z
1	-2.68752671786e-001	-4.07750182239e-001	8.18200216099e-001	1.24758151066e-004	1.47501357621e+001	7.48082284057e+000
2	-2.37286130715e-001	-4.66472405282e-002	2.05071427955e-001	4.94359855577e-001	9.32657214231e+000	3.36042060092e+000
3	-2.32116237214e-002	1.93129561449e-002	-4.69017493499e-002	1.02940417164e+000	-1.79315448178e+000	-1.76955303475e-001
4	-9.86435287282e-003	-1.33604473105e-002	-3.35389595537e-002	-7.18299465789e-001	-1.25730927741e+000	-1.32817910728e+000
5	3.69290252036e-001	2.57594191158e-001	1.24305395075e-002	-6.20773892285e-001	1.07539131583e+000	-3.84235811890e+000
6	3.51114881793e-001	1.74666525026e-002	-1.39658047222e-004	9.67756197863e-004	-1.27189224256e-002	7.07527879168e-001
7	4.20542346558e-001	-3.01675751768e-001	8.94984992729e-003	4.46822001786-001	7.74094614820e-001	5.09626834708e+000

Total energy before collision = 5.78003960875e-001
Total energy after collision = 4.40134281975e-001
Energy discrepancy = -2.38527221660e+001 %

Figure B.97: Heptenary Test 4 – PFC3D (DEM-Approach) Solution.

B.7 OCTENARY TEST RESULTS

The various octenary test results as referred to in section 5.1.7.3 can be seen in this section, with the quantities again as described in section B.1.

B.7.1 Octenary Test 1

Post-Collision Rigid Body Data:

Number	Pos_X	Pos_Y	Pos_Z	V_X	V_Y	V_Z	Omega_X	Omega_Y	Omega_Z
1	1.5288675135	1.5000000000	1.5816496581	-0.0000000000	0.0000000000	0.3652209746	0.0000000000	-0.0000000000	0.0000000000
2	1.5000000000	1.5000000000	1.5408248290	-0.2936537191	0.0000000000	-0.5271344893	-0.0000000000	5.4335312970	0.0000000000
3	1.5433012702	1.5250000000	1.5408248290	0.1469268595	0.2543115806	-0.5271344893	4.7055761354	-2.7167656485	-0.0000000000
4	1.5433012702	1.4750000000	1.5408248290	0.1469268595	-0.2543115806	-0.5271344893	-4.7055761354	-2.7167656485	-0.0000000000
5	1.5288675135	1.5000000000	1.5000000000	-0.0000000000	-0.0000000000	-0.1194899533	-0.0000000000	0.0000000000	-0.0000000000
6	1.5000000000	1.5000000000	1.4591751710	-0.0443791088	0.0000000000	-0.0881091845	-0.0000000000	1.4634470743	0.0000000000
7	1.5433012702	1.5250000000	1.4591751710	0.0221895544	0.0384334356	-0.0881091845	1.2673823434	-0.7317235371	-0.0000000000
8	1.5433012702	1.4750000000	1.4591751710	0.0221895544	-0.0384334356	-0.0881091845	-1.2673823434	-0.7317235371	-0.0000000000

Contact Data:

Number	Active	Slipping	Object1	Type1	Object2	Type2	Normal_X	Normal_Y	Normal_Z	Lambda_N	Tangent_X	Tangent_Y	Tangent_Z	Lambda_T
1	true	true	6	19	5	19	0.5773502692	0.0000000000	0.8164965809	0.0501261159	-0.8164965809	0.0000000000	0.5773502692	0.0075189174
2	true	true	7	19	5	19	-0.2886751346	-0.5000000000	0.8164965809	0.0501261159	0.4082482905	0.7071067812	0.5773502692	0.0075189174
3	true	true	8	19	5	19	-0.2886751346	0.5000000000	0.8164965809	0.0501261159	0.4082482905	-0.7071067812	0.5773502692	0.0075189174
4	false	false	7	19	6	19	-0.8660254038	-0.5000000000	0.0000000000	0.0000000000	0.0000000000	-0.0000000000	1.0000000000	0.0000000000
5	false	false	8	19	6	19	-0.8660254038	0.5000000000	0.0000000000	0.0000000000	-0.0000000000	-0.0000000000	1.0000000000	0.0000000000
6	false	false	8	19	7	19	0.0000000000	1.0000000000	0.0000000000	0.0000000000	0.0000000000	0.0000000000	1.0000000000	0.0000000000
7	true	true	2	19	1	19	0.5773502692	0.0000000000	0.8164965809	0.2588955404	-0.8164965809	-0.0000000000	0.5773502692	0.0388343311
8	true	true	3	19	1	19	-0.2886751346	-0.5000000000	0.8164965809	0.2588955404	0.4082482905	0.7071067812	0.5773502692	0.0388343311
9	true	true	4	19	1	19	-0.2886751346	0.5000000000	0.8164965809	0.2588955404	0.4082482905	-0.7071067812	0.5773502692	0.0388343311
10	false	false	3	19	2	19	-0.8660254038	-0.5000000000	0.0000000000	0.0000000000	-0.0000000000	0.0000000000	1.0000000000	0.0000000000
11	false	false	4	19	2	19	-0.8660254038	0.5000000000	0.0000000000	0.0000000000	0.0000000000	0.0000000000	1.0000000000	0.0000000000
12	true	true	5	19	2	19	-0.5773502692	0.0000000000	0.8164965809	0.0727857560	0.8164965809	-0.0000000000	0.5773502692	0.0109178634
13	false	false	4	19	3	19	0.0000000000	1.0000000000	0.0000000000	0.0000000000	0.0000000000	0.0000000000	1.0000000000	0.0000000000
14	true	true	5	19	3	19	0.2886751346	0.5000000000	0.8164965809	0.0727857560	-0.4082482905	-0.7071067812	0.5773502692	0.0109178634
15	true	true	5	19	4	19	0.2886751346	-0.5000000000	0.8164965809	0.0727857560	-0.4082482905	0.7071067812	0.5773502692	0.0109178634

Solution Data:
  6 iterations, converged
  Energy:
    Total energy before collision = 0.25689065 Joules
    Total energy after collision = 0.20046707 Joules
    Energy discrepancy = -21.96404425 %

Figure B.98: Octenary Test 1 Non-Linear Solution.

Number	V_X	V_Y	V_Z	Omega_X	Omega_Y	Omega_Z
1	6.13023152559e-012	0.00000000000e+000	4.98589683268e-001	0.00000000000e+000	2.96784139006e-010	0.00000000000e+000
2	-3.45549275182e-001	0.00000000000e+000	-1.96192042735e-001	0.00000000000e+000	2.97893364351e+000	0.00000000000e+000
3	1.72771521450e-001	2.99248237354e-001	-1.96209636659e-001	2.58004409838e+000	-1.48964974295e+000	2.55306263352e-005
4	-8.69599648943e-006	0.00000000000e+000	-7.99165528156e-002	0.00000000000e+000	1.48503712692e-005	0.00000000000e+000
5	1.72771521450e-001	-2.99248237354e-001	-1.96209636659e-001	-2.58004409838e+000	-1.48964974295e+000	-2.55306263352e-005
6	-9.25861988947e-002	0.00000000000e+000	-1.83386999261e-001	0.00000000000e+000	3.05157573106e+000	0.00000000000e+000
7	4.62009269335e-002	7.99992677931e-002	-1.83378381932e-001	2.63813172961e+000	-1.52490022568e+000	6.66359949065e-004
8	4.62009269335e-002	-7.99992677931e-002	-1.83378381932e-001	-2.63813172961e+000	-1.52490022568e+000	-6.66359949065e-004

Total energy before collision = 2.56890649278e-001
Total energy after collision = 2.20235998506e-001
Energy discrepancy = -1.42685811552e+001 %

Figure B.99: Octenary Test 1 – PFC3D (DEM-Approach) Solution.