Contact Mechanics using Green's Functions for Interactive Simulated Environments

by

C. Ullrich

B.Math., University of Waterloo, 1995

A THESIS SUBMITTED IN PARTIAL FULFILLMENT OF

THE REQUIREMENTS FOR THE DEGREE OF

Master of Science

in

THE FACULTY OF GRADUATE STUDIES

(Institute of Applied Mathematics)

we accept this thesis as conforming
to the required standard

The University of British Columbia

August 1998

Abstract

This thesis is concerned with the determination of contact models for linear elastic solids which are both accurate and computable in a real time environment. We review previous models based on restitution coefficients and penalty functions, along with their efficiency and accuracy. A new model is proposed which is both rapidly computable and based on easily measurable material properties. In addition, we consider the role that friction plays in contact modeling and examine the various friction based effects that each model can produce. We conclude by outlining a simulation system in which this model can be used.

Contents

List of Tables

List of Figures

Acknowledgements

I would like to thank my advisor Dr. D. K. Pai for introducing me to this problem and for numerous enlightening discussions. Dr. Pai's passion for interactive simulation served as both motivation and inspiration.

I would also like to thank my parents for providing support at those critical moments, to allow me to pursue this research.

Finally, I would like to thank my thesis committee, Dr. Ascher and Dr. Peirce for taking the time to read my thesis.

C. Ullrich

The University of British Columbia

August 1998

Chapter 1

Introduction

Contact modeling for real time simulation has traditionally involved choosing how best to approximate more accurate mechanical models. A good example of this is finite elements, which is not realizable in real time on almost all computers, yet provides highly accurate contact modeling. This thesis considers a middle ground approach which yields good accuracy while minimizing computational expense.

In the past virtual reality simulations were constrained to high end equipment or relegated to poor performance. Today there are a number of VR systems that give interactive performance [39], [17] but make no attempt to achieve physical accuracy. On the other hand, there are also quite a few high end engineering simulation environments [12], [33] which provide excellent accuracy but must be run off line.

Emerging application areas such as rapid prototyping, multi-user engineering and high latency telerobotics can all benefit from a physical simulation environment in which object behaviours are not only realistic but also interactive.

1.1 Contact Models

The main focus of this thesis is to consider the state of the art in interactive collision force computation. Collision forces occur on a time scale vastly different from any human sense organs.

In fact, an entire school of thought on collision forces models them as instantaneous. This is the impulsive model, in which velocities of colliding objects change discontinuously in time over a collision event. Though the impulsive model has the advantage of fast computation, it can suffer from problems related to energetic consistency, and parameter determination.

A second school tries to model collision forces on their natural time scale. Highly simplified models, based mostly on linear approximations are commonly used to speed up the calculations. Full scale modeling of elastic response of solids using an algorithm like finite elements or boundary elements is not currently possible at interactive rates, though there is at least one group pursuing this lofty goal [35].

The model introduced in this thesis attempts to leverage the accuracy of the boundary element method by precomputing as much information as possible. We trade off space for time by storing Green's functions for a large number of points on the surface of each object to be simulated. We pay considerable attention to the accuracy of this model since we would like to use this approach in a realistic interactive simulation environment.

1.2 Simulation

Finally this thesis considers the object oriented design of an interactive simulation environment. It is the intention of this author to produce a flexible environment in

which the contact model presented in this thesis could be implemented. However we also realize that in the future, more advanced or computationally demanding algorithms may become feasible not just for contact force determination, but for many of the other aspects of interactive simulation.

The simulation environment has as primary goals interactive running speed and realistic physics. This environment differs significantly from previous simulators in that it has explicit support for the integration of live data from sensor arrays, simulated data and (multi) user input.

Such a simulation environment would find use in a wide variety of application areas, from research to engineering design to telerobotics and entertainment.

1.3 Overview

The thesis begins with a review of the usual contact algorithms including impulsive, linear force and quasi-static continuum models. We deliberately do not consider issues related to tangential forces such as friction, which we reserve for a later chapter. Following this, we introduce the contact map algorithm for generating Green's functions and using them in a simulation environment. Chapter 4 presents some artificial and real simulation results from the contact map algorithm. In chapter 5, we review the current knowledge of tangential surface forces, primarily friction. We then present some numerical results comparing the contact model with friction to actual experimental results. In the appendices, we review the mathematics of the boundary element method and discuss the proposed simulation environment.

Chapter 2

Contact Model Review

2.1 Goals of Contact Modeling

When two solids come into contact, many different forces act on each body. The forces arise from local effects near the contact region such as elastic response and friction but also include global effects which are functions of geometry, mass distribution and others. Currently, a full theoretical understanding of all these effects is incomplete and experimental data are very limited. In addition, simulation of contact using more than one or two of these properties is usually computationally expensive.

A primary goal of contact modeling is to produce an acceptable tradeoff between the cost of computing contact forces and the accuracy of the results. In this section, we discuss the theory of contact and some of the more popular models that have been proposed to resolve this process.

2.1.1 Real Solids

Realistic modeling of solids which undergo deformation and respond to collision related forces is typically done with the elastic continuum model and various extensions for thermal, plastic and cracking effects. We postpone a detailed analysis of the elastic continuum model of solids for the moment, and consider here only the processes which occur near the surface of contacting bodies.

Hertz Model

The first model we consider was proposed by Hertz over a hundred years ago. We present this model in a separate section because it introduces the basic approach that many contact models take. Full details of this model are available in many books, for example [37].

The Hertz theory relates to two elastic bodies coming into contact at a point. Hertz made four basic assumptions when formulating his theory;

1) small strains (surface displacements within the elastic limit)

2) the two contact surfaces are continuous and do not conform

3) the contact region is much smaller than the local radius of curvature

4) there is no friction.

Consider two bodies which come into contact at a point. Define a co-ordinate system with origin at the contact point, and z-axis normal to the contact. Each body has a local radius of curvature (e.g., Gaussian curvature) R_{1x}, R_{1y} and R_{2x}, R_{2y} for bodies 1 and 2 in the x and y directions respectively. Further, we assume that the principle axes of both bodies are aligned and coincident with the z co-ordinate axis.

Hence for x and y close to the contact point, we can express the undeformed

surfaces as

$$z_k = \frac{x^2}{2R_{kx}} + \frac{y^2}{2R_{ky}} \tag{2.1}$$

for $k = 1, 2$. Define the relative radii of curvature by $R_x = \frac{R_{1x}R_{2x}}{R_{1x}+R_{2x}}$ and $R_y = \frac{R_{1y}R_{2y}}{R_{1y}+R_{2y}}$. Then we can write the separation between the two bodies as

$$h(x, y) = z_1 + z_2 = \frac{x^2}{2R_x} + \frac{y^2}{2R_y}. \tag{2.2}$$

Note that level sets of h are ellipses.

We assume further that there is some local deformation around the contact point of maximum magnitude δ_k along the z axis. Thus, within an elliptical contact region, we have

$$w_1(x, y) + w_2(x, y) = \delta_1 + \delta_2 - h(x, y), \tag{2.3}$$

where $w_k(x, y)$ represents the vertical deformation of a point on body k at position (x, y).

By assuming the contact region is elliptical the displacement can be expressed as a function of the contact pressure,

$$w_j(x, y) = \frac{1 - \nu^2}{\pi E_j} \int\int_S \frac{p(\xi, \eta)d\xi d\eta}{\sqrt{(x - \xi)^2 + (y - \eta)^2}}. \tag{2.4}$$

Where ν is the Poisson ratio and E is Young's modulus of elasticity. Hertz showed that an ellipsoidal pressure distribution could satisfy 2.4 in this case.

Combining the above equations, we arrive at the Hertz model of contact,

$$\frac{E^*}{\pi} \int\int_S \frac{p(\xi, \eta)d\xi d\eta}{\sqrt{(x - \xi)^2 + (y - \eta)^2}} = \delta - \frac{x^2}{2R_x} + \frac{y^2}{2R_y}, \tag{2.5}$$

where $\delta = \delta_1 + \delta_2$ and $E^* = \frac{1-\nu_1^2}{E_1} + \frac{1-\nu_2^2}{E_2}$. This equation relates the pressure to the displacement in the contact zone.

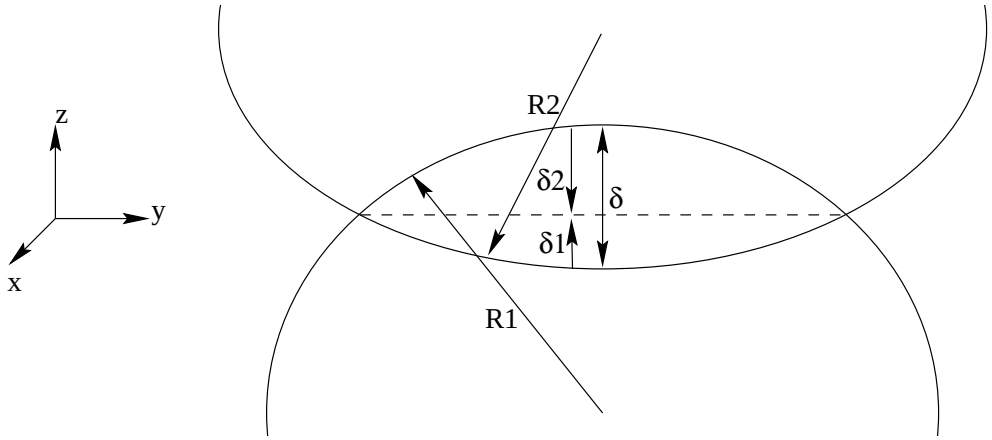


Figure 2.1: Schematic of Hertz contact scenario.

In the case where the contacting bodies are spherical, as shown in 2.1, it is relatively straightforward to show that the total pressure P is related to the displacement δ by,

$$P = \frac{4}{3} E^* R \delta^{\frac{3}{2}}. \tag{2.6}$$

When the bodies are non-spherical, it becomes difficult or impossible to construct analytic solutions, although perturbations exist for mildly elliptic situations. Most authors simply refer to the relationship $f = k\delta^{\frac{3}{2}}$ with force f and elastic constant k as the Hertz model.

The Hertz model is a cornerstone of modern contact theory yet it does not account for many commonly observed collision phenomenon such as energy loss, friction and permanent deformation. We discuss these effects and models for them below.

2.1.2 Rigid Bodies

In light of the large number of collision phenomenon, most researchers have focused on a few of the key principles. This thesis will be no exception, we will consider physical contact between objects which are accurately modeled as rigid bodies. By rigid bodies, we mean that collision forces are small enough that significant global deformation does not occur.

The rigid body assumption restricts our consideration to collisions which are of relatively low velocity and between hard objects. In this narrow band, there are a large number of models and theories which attempt to bridge the gap between the scientific discipline of tribology and practical engineering. For an enlightening discussion of this connection, see [54].

2.2 Contact Model Classification

Contact models can be classified into two groups, those which attempt to model contact forces explicitly and those which only attempt to model the effects of a collision. The former are called force response rigid models, while the latter are impulse response rigid. This taxonomy was originally introduced by Chatterjee [14].

2.3 Impulse Response Rigid Models

Impulse response rigid models only attempt to model the physics immediately before and immediately after a collision. The forces generated during contact are not modeled, only their net effect on object velocity. We express this with,

$$\Delta p = \int_0^{t_f} f(t)dt. \tag{2.7}$$

Where the applied impulse is Δp and the collision starts at time $t = 0$ and runs until the bodies separate at t_f. One of the fundamental assumptions of impulse response rigid models is that t_f is very small relative to the rest of the simulated processes.

The category of impulse response rigid models is further subdivided by three different definitions of a dimensionless coefficient which relates pre-contact states to post-contact states. This quantity is generally known as the coefficient of restitution. We briefly describe the three definitions here, however we must postpone an analysis of their relative merits until after we discuss friction in 5.

2.3.1 Newton

The first person to consider this problem in detail was Newton, who proposed that a dimensionless constant $e \in [0, 1]$ relate the velocities of bodies before and after a

collision,

$$e = \frac{v_N^+}{v_N^-}. \tag{2.8}$$

Where v_N^+ is to post collision normal velocity and v_N^- is the pre-collision normal velocity. A choice of $e = 1$ corresponds to a perfectly elastic collision whereas setting $e = 0$ indicates that all kinetic energy is dissipated during a collision.

2.3.2 Poisson

Poisson revised Newton's model by dividing an impulsive collision event into two phases, the compression phase and the restitution phase. The transition from compression to restitution is defined as the point where the normal velocity is zero. The resulting coefficient is expressed as a ratio of normal momenta,

$$e = \frac{p_N^+}{p_N^-} \tag{2.9}$$

Where p_N^+ is the integrated normal force of restitution from the time of maximum compression to the final contact time. Also, p_N^- is the integrated normal force of compression, from the time of initial contact to the time of maximum compression. Keller presents a detailed analysis of this model for 3D impacts in [40].

2.3.3 Stronge

The third widely used definition was proposed in 1991 by Stronge [58] and relates the work done during compression to the work done during restitution. Precisely,

$$e^2 = \frac{W_r}{W_c}, \tag{2.10}$$

where W_r and W_c is the work done by the normal force during compression and restitution respectively. This model was proposed after it was discovered that the above two models generated energy in certain types of frictional collisions.

2.3.4 Discussion

It is important to note that the above three laws give exactly the same result in contact configurations which do not involve friction during the collision. The latter two were constructed to resolve apparent violations of conservation of energy noted by [40], [8] and others.

It is also well known that experimentally, the coefficient of restitution is a function of impact velocity (varying approximately as the fourth root, see [25]), impact geometry [56] as well as a host of other surface and material properties. These functional dependencies lead one to question the merit of just a single restitution coefficient for two colliding bodies.

We discuss the impulse response rigid models in more detail in 5 on frictional contact where the differences are much more clear.

2.4 Force Response Rigid Models

Models which attempt to accurately generate the forces that occur during a collision are called force response rigid. These models attempt to be more physically accurate than impulse models but typically come at the price of increased computational cost. Also, when these models are coupled with a friction model, we are ensured that no anomalous energy will be generated by the collision, unlike impulse models.

2.4.1 Penalty Models

The simplest force response rigid models are penalty models, in which the force is a function of the current penetration depth of two bodies. This corresponds to the term δ in the discussion of the Hertz contact law. A simple analysis shows that

models which depend only on the local value of this parameter cannot dissipate energy through collision processes. The addition of a viscous term depending on $\dot{\epsilon}$ makes energy dissipation possible. Penalty models can thus be expressed in the following functional form,

$$f = \begin{cases} f(\epsilon, \dot{\epsilon}) & \text{if } \epsilon \geq 0 \\ 0 & \text{otherwise} \end{cases} \tag{2.11}$$

Note that this model depends only on the local penetration depth and its derivative (i.e., f only depends on the value at time t). This is an important distinction, because in the next chapter we consider non-local laws of this type.

The Hertz model presented above can be considered as a nonlinear example of this type of law, with no dependence on $\dot{\epsilon}$. We consider here further examples, the bilinear model of Stronge [59], the harmonic penalty model and the nonlinear model of Hunt and Crossley [32]. We analyze each of these models in the context of a one dimensional collision. In this case Newton's second law gives,

$$m\ddot{\epsilon} + f(\epsilon, \dot{\epsilon}) = 0. \tag{2.12}$$

2.4.2 Bilinear

The bilinear model is a penalty realization of the energetic coefficient of restitution 2.10 defined above. In this model, there are two spring constants, one for compression k_1 and one for restitution k_2. Given a coefficient of restitution e, we define $k_2 = \frac{k_1}{e^2}$ which for $e \in [0, 1]$ gives a higher spring constant for unloading, see figure 2.2. Contact is broken when the force returns to zero. The energy lost to collision is equal to the area between the two lines which is consistent with 2.10.

As noted by [14], this method suffers from numerical difficulties when e is close to 0 because the unloading slope can be arbitrarily steep. Chatterjee suggests

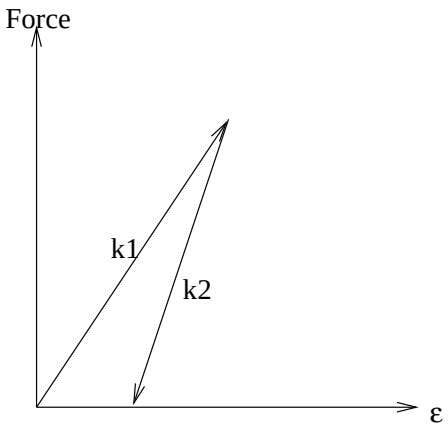


Figure 2.2: Bilinear force model, showing a sample collision path.

a possible alternative, where the loading and unloading lines are constrained to pass through the origin, and a 'jump' occurs at the point of maximum compression. In this case, $k_2 < k_1$.

2.4.3 Harmonic Penalty

It is also possible to achieve energy loss by including a so called dissipation term in the penalty law. This easily seen in the simple 2nd order linear ODE,

$$m\ddot{\epsilon} + c\dot{\epsilon} + k\epsilon = 0 \tag{2.13}$$

The velocity dependent term $c\dot{x}$ controls the amount of damping in the solution. Adding a damping term to the penalty law removes the need to adjust the spring constant k during the course of the collision. This approach is used by many researchers as it is simple and easy to implement. A complete simulation system based on this type of model, but generalized to 3D, is presented in [26][27] and we examine it in chapter 5.

Chatterjee analyzes this model and constructs an approximation to the co-

efficient of restitution. Define $\beta = \frac{c}{2\sqrt{km}}$, then a good approximation (maximum relative error less than .06) to e is given by,

$$e = \frac{1}{1 + 2.5\beta + 4\beta^2}. \tag{2.14}$$

One major drawback of this model is that due to the non-zero initial relative velocity, a force discontinuity occurs at the beginning and end of contact. The size of this discontinuity is proportional to the impact velocity. This may be a concern for simulation environments in which this type of force model is used to generate haptic feedback, since the discontinuity will certainly be felt by a user.

2.4.4 Hunt and Crossley

The issue of force discontinuity is one of the motivations behind the seminal work of Hunt and Crossley [32]. They consider a modification to the the linear equation above,

$$m\ddot{\epsilon} + (\lambda\epsilon^m)\dot{\epsilon} + k\epsilon^n = 0. \tag{2.15}$$

Using this modification, the initial force is zero for zero displacement, independent of the initial velocity. The nonlinearities in this equation mean an analytic solution is in general impossible. The restitution coefficient, as defined by Newton can be approximated for the case $m = n = 1$, see [14]. Hunt and Crossley derive a more general expression in the case when λ is small (see [32] eqn. 14).

Relating the energy loss through a collision to the various collision parameters can only be done numerically. We performed this integration for a variety of λ and k values in the case $m = n = 1$, see figure 2.3.

In the case where $m = n$, the authors construct an explicit method for

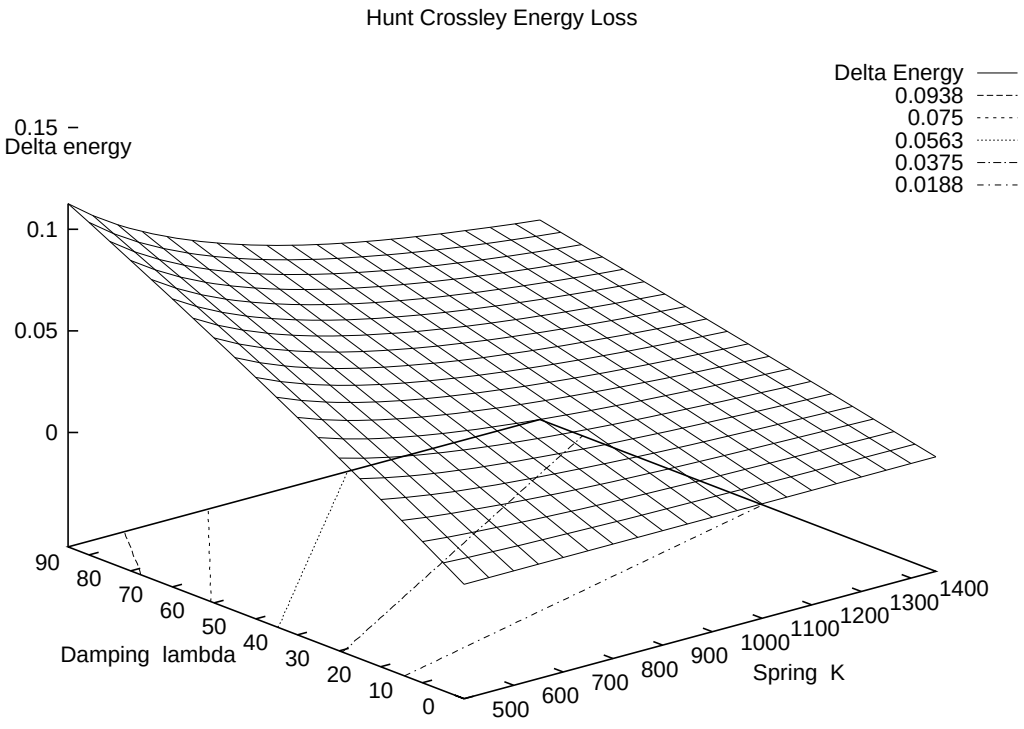


Figure 2.3: Fraction of energy lost during single impact collision. In this case $m = n = 1$.

determining λ based on a velocity dependent restitution definition.

$$e = 1 - \alpha v_i. \tag{2.16}$$

Where v_i is the impact velocity and α is a small coupling constant. In this case we can write

$$\lambda = \frac{3}{2}\alpha k. \tag{2.17}$$

For further details, we refer the interested reader to [32].

2.4.5 Three Dimensions

It is important to understand that the conclusions of the one dimensional collision analysis above do not necessarily extend naturally to three dimensions. The reason for this is that in three dimensions, we must deal with mass matrices, spring and damping tensors. The interaction between these three quantities becomes more subtle and even more dependent on the particular impact configuration. To reduce this problem somewhat, Goyal [26] considers only the spring and damping constants in the normal and tangential directions at the collision point only.

In higher dimensions, the number of external model parameters in the force response rigid models presented thus far becomes almost prohibitive. Simplifying assumptions such as Goyal's are required to have a workable simulation. However since the parameters of all the force models have somewhat tenuous relation to the physics of contact, one cannot expect accurate solutions to result from these models.

2.4.6 Discussion

In all of the models discussed so far, with the exception of the Hertz model, no mechanism is given to determine the collision parameters. Chatterjee tries to relate

all penalty model parameters to the coefficient of restitution. Though this may seem like a reasonable goal, in practice the coefficient of restitution is a difficult quantity to measure, requiring specialized equipment and high speed measuring devices for even simple scenarios (see [56], [20]).

Including the important dependencies of e on geometry, velocity and material properties, we quickly end up with a large number of scenarios in which e must be determined. For general simulation environments, a huge database would be needed, which would grow at least as the square of the number of objects. It is highly doubtful that e could even be measured for all possible combinations, even with an advanced facility such as the ACME testbed [48].

Given the requirement that a simulator run at interactive speeds and produce realistic and accurate data, we can evaluate the models presented thus far. Although impulsive models work very quickly in practice, the restitution coefficient has many subtle dependencies and many measurement related difficulties. The force rigid models require that collisions be explicitly integrated, however there is some difficulty in determining the coefficients of the force relations especially in more than one dimension. Accuracy will also have to be traded off to achieve reasonable integration speed on all but the fastest computers.

Given these options, we look beyond these simple models to the much more sophisticated models that are currently used in mechanical and civil engineering for performing precise contact analysis.

2.5 Quasi-Static Continuum Models

The framework for the study of mechanical contact when only small deformations are present is the continuum theory of linear elastic solids. This model consists

a system of PDE's together with appropriate boundary and initial conditions and can accurately model many contact and low energy collision scenarios. As we shall see however the numerical solution of these problems is both complex and time consuming, effectively removing them from consideration, for interactive simulation, as well.

2.5.1 Linear Elastic Solids in Equilibrium

We begin by reviewing the relevant definitions in the subject of continuum mechanics as they relate to elastic solids. Consider a point set $\Omega \in \Re^3$ with elements $\mathbf{x} = (x_1, x_2, x_3)$. For most applications, we will assume Ω is compact. Now define transformed coordinates of each point by $y_i = \chi_i(\mathbf{x})$ and define the displacement vector by $\mathbf{u} = \mathbf{y} - \mathbf{x}$. Assume further that the transformation χ is invertible and at least C^1.

From these fundamentals, we define two important quantities, the strain tensor $\epsilon_{ij} = \frac{1}{2}(u_{i,j} + u_{j,i})$ and the stress tensor $\sigma_{ij}(\mathbf{x}) = \hat{\sigma}_{ij}(\mathbf{x}, \epsilon_{ij})$. The function $\hat{\sigma}_{ij}$ encapsulates the material properties and is known as the constitutive relation. When Ω is a linear elastic solid, the constitutive relation is

$$\sigma_{ij} = E_{ijkl}\epsilon_{kl}. \tag{2.18}$$

Where E_{iljk} is the Hooke tensor, completely analogous to the spring constant k in the previous section, but with 81 components.

From a force balance we see that,

$$\sigma_{ij,j}(\mathbf{x}) + f_i(\mathbf{x}) = 0. \tag{2.19}$$

Where $f_i(\mathbf{x})$ are body forces which act over the entire region Ω (e.g., gravity). This equation is a force balance and is valid in the absence of time dependent forces

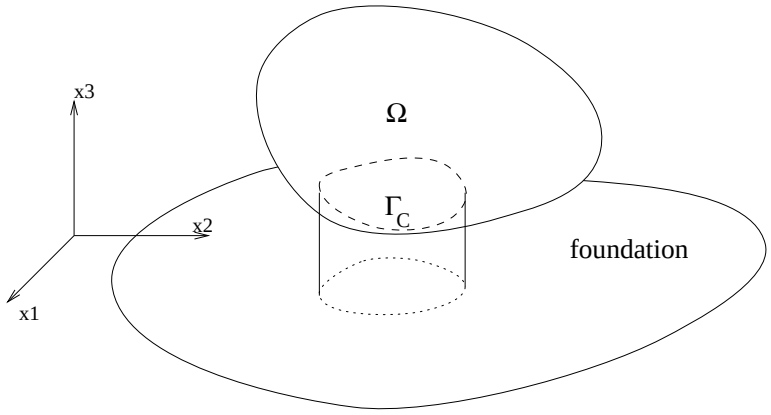


Figure 2.4: Schematic of Signorini problem configuration.

which act on time scales smaller than the time for a sound wave to traverse the solid (equilibrium).

Full details of the theory of elastic solids can be found in many books, for example [63]. In addition, we consider a dynamic version of this theory in the next chapter.

2.5.2 Signorini Problem

The study of elastic solids in contact situations was pioneered by Signorini in 1933 [53] when he formulated a general constrained elastostatic problem. Signorini considered the problem of a linear elastic solid coming into contact with a rigid, semi-infinite foundation, see figure 2.5.2.

Consider an elastic solid Ω which is free to move in the vertical direction in 3-space. Below the solid is a semi-infinite foundation. We let Γ_C be the region on the surface of Ω which will contact the foundation if Ω is translated vertically. Assume that a chart $\phi(x_1, x_2)$ parameterizes this region and that a chart $\psi(x_1, x_2)$ parameterizes a similar region on the surface of the foundation. Using the displacement

vector $\mathbf{u}$, we can express the kinematic constraint as

$$\phi(x_1, x_2)+u_3(x_1, x_2, \phi(x_1, x_2)) \leq \psi(x_1+u_1(x_1, x_2, \phi(x_1, x_2)), x_2+u_2(x_1, x_2, \phi(x_1, x_2))). \tag{2.20}$$

Equation 2.20 simply states that the two surfaces cannot penetrate each other. For simplicity we rewrite this using the invertability of the transformation χ,

$$\hat{\phi}(y_1, y_2) + \hat{u}_3(y_1, y_2) \leq \hat{\psi}(y_1, y_2). \tag{2.21}$$

By considering small increments to the displacement u and linearizing, 2.21 is rewritten as

$$u_n(\mathbf{x}) - g(\mathbf{x}) \leq 0. \tag{2.22}$$

Here

$$g(\mathbf{x}) = \frac{\psi(y_1, y_2) - \phi(y_1, y_2)}{\sqrt{1 + (\frac{\partial \phi}{\partial y_1})^2 + (\frac{\partial \phi}{\partial y_2})^2}} \tag{2.23}$$

is the normalized separation and $u_n(\mathbf{x}) = \mathbf{u}(\mathbf{x}) \cdot \mathbf{n}(\mathbf{x})$ is the normal displacement.

Let $\tau_n(\mathbf{x})$ and $\tau_{T_i}(\mathbf{x})$ represent the normal and tangential surface stress in the contact region Γ_C respectively. Then for frictionless contact we require,

$$\tau_{T_i}(\mathbf{x}) = \mathbf{0} \tag{2.24}$$

$$\tau_n(\mathbf{x}) \leq \mathbf{0}. \tag{2.25}$$

Finally, because the contact stresses only occur when there is actual contact, we have the linear complementarity condition,

$$\tau_n(\mathbf{x})(u_n(\mathbf{x}) - g(\mathbf{x})) = 0. \tag{2.26}$$

The Signorini problem is the set of relations 2.19, 2.21, 2.24, 2.26 and any boundary conditions on $\partial\Omega - \Gamma_C$.

There are a number of techniques for solving such problems in general. In [41] the problem is formulated as a variational principle which leads to weak solution techniques such as FEM (Finite Element Method). The authors of [2] also use an integral form but apply instead a boundary integral (or BEM, Boundary Element Method) technique. The BEM approach has the advantage that only the boundary Γ needs to be discretized. We discuss this approach in detail in the next section and the following chapter.

The solution to the Signorini problem and its extensions (two body contact, dynamics) provide high accuracy and depend only on easily known material parameters such as the shear modulus G and the Poisson ratio ν. However the numerics of this approach are formidable and computationally intensive. The author has never seen reference to any system which claims to compute such models at or near real time.

2.5.3 Flexibility Matrices

Finally we consider an approach to solving the contact problem which is similar to the method presented in the next chapter. The boundary element method (see Appendix A) for the two contacting bodies above yields a dense system of linear equations,

$$\mathbf{A}\mathbf{y} = \mathbf{b}. \tag{2.27}$$

Where $\mathbf{A}$ is a discretization of the boundary integrals, $\mathbf{y}$ contains the unknown displacement or traction as appropriate and $\mathbf{b}$ is a vector of known values of the displacement or traction at each boundary node.

The flexibility matrix approach [62] to solving 2.27 rapidly, is to precompute the elastic response of each node and then solve the boundary integral only in

the contact region. Two matrices are needed to accomplish this, the displacement matrix represents the deformation produced by an incremental external load in the contact region. The flexibility matrix gives the displacement when a unit normal or tangential force is applied on the boundary.

To calculate the displacement matrix for each body, external forces are applied to the appropriate boundary elements, fixed displacement boundary conditions are applied to those elements which are initially in contact and free boundary conditions are applied elsewhere. Then using a BEM solution technique, the displacements of the elements close to the contact region are calculated. The displacement matrix $\mathbf{D}$ is $N \times 3$ for three dimensional contact problems ([62] consider only the 2D case) and is used as follows,

$$\mathbf{S} = \mathbf{DP}. \tag{2.28}$$

Where $\mathbf{P}$ is the applied surface traction, and $\mathbf{S}$ is the resulting surface displacement. Generation of the flexibility matrix is somewhat more complex. This matrix relates a normal or tangential force at the contact surface to the resulting displacement (of all elements). The contact surface is the average of the surface elements weighted by the Young's moduli of the two contacting bodies,

$$\theta_s = \theta_1 - (\theta_1 - \theta_2)\frac{E_2}{E_1 + E_2}. \tag{2.29}$$

This model assumes a reasonable prior knowledge of this contact angle to compute the flexibility matrices for each object and that only small changes will occur during the collision. Using this information, the boundary element method is successively applied with a normal and tangential force for each element and free boundary conditions for all others. The result is a rank 4 tensor,

$$\mathbf{F} = [F_{kl}^{ij}]. \tag{2.30}$$

Where i is the node displaced, j is the node at which a force is applied, k is the direction of the displacement component and l is the direction of the applied traction. Multiplication by a traction matrix $\mathbf{T}$ gives a displacement matrix $\mathbf{S}$,

$$\mathbf{FT} = \mathbf{S}. \tag{2.31}$$

Using an iterative, incremental algorithm the authors [62] show how to use eqns. 2.28, 2.31 to rapidly solve contact problems. We refer to their paper for further details.

This approach provides a fast and extremely accurate method for solving quasi-static contact problems. However this algorithm requires a fairly significant amount of knowledge about the type of contact which is occurring to design the deformation and flexibility matrices. The authors do not comment on the ease with which the various choices are made. In addition the coupling of the two bodies through the contact angle means that two different flexibility matrices will have to be computed for each object pair. For environments with a large number of objects, with complex objects, or both the storage requirements and the pre-computation times could become prohibitive.

2.6 Impact Loss in Elastic Solids

The Signorini and flexibility approach discussed above both assume that the contact forces are applied in a quasi static manner. This assumption is justified in situations where loading and unloading in the contact region take place over a long time period (relative to the wave speed in each body). For collision situations this assumption is often violated. Collision events typically modeled with impulsive laws take place on the scale of tens to hundreds of microseconds.

In [56], the authors demonstrate collision events in which as much as 50% of the initial kinetic energy is lost to elastic waves during a collision event. Similar results are reported in [60].

Examining the contact models presented thus far, it is clear none explicitly consider this effect. One reason for this omission is that energy loss to elastic waves is very dependent on the contact geometry and the geometry of the solids in contact. Stronge reports that energy loss for a collision between two cylinders at 44% but only 10% for a similar collision involving a sphere. Any of the models presented which include damping effects could include elastic wave loss implicitly. This approach would necessitate a field of coefficients covering the surface of each body, which could also be a function of the contactors.

The next chapter presents an alternative to the models presented thus far which also includes effects related to geometry and elastic waves.

2.7 Summary

In this chapter, we considered contact models that can be used either for rapid, inaccurate solution of impact configurations or for slow accurate solution of contact configurations. The impulsive and penalty models are not suited for long term contact situations because their assumptions lead to either discontinuous dynamics or very stiff ODE's. By comparison, continuum models discussed above are of little use in impact situations in which the internal dynamics of a solid are relevant, due to their quasi-static nature, but provide very accurate solutions for bodies which come to rest against each other.

The inaccuracies of the impulsive and penalty models is a result of the severe assumptions on the colliding bodies. The critical dependence of contact mechanics

on geometry is only explicitly included in the Hertz model. In the next chapter we attempt to construct a model that addresses this issue directly.

Chapter 3

Contact Response Maps

3.1 Goals

This chapter is concerned with a new approach to contact modeling which tries to incorporate geometric and transient effects, without greatly sacrificing performance. The approach is to determine the response of a linear elastic solid to a small deformation at each point on the surface and save all these response curves as a *contact map*. This information can then be used during a simulation to more accurately model the response.

One of the primary goals of this approach is to remove arbitrary and/or difficult to measure parameters from the collision model. Nearly all the models discussed in the previous chapter suffer from this parameter indeterminacy problem. In addition, none of the non-continuum models make a direct attempt to handle the dependency on global object geometry of the dynamic collision forces. The contact map model attempts to address these deficiencies while remaining rapidly computable.

We begin by discussing the materials to which this model is suited, then discuss the dynamic equations which govern them. We take a short detour into boundary elements to describe one possible way to compute a contact map. Following this, we discuss a time stepping algorithm for handling collisions that uses contact maps. We conclude with some remarks on the algorithm consistency and energetics.

3.2 Contact Map Model

We restrict our attention in this chapter to linear elastic solids which do not deform significantly during a collision process. This category includes most *hard* objects like steel, aluminum and cast iron but excludes deformable materials like rubber and clay. The classification of an object into the *hard* category is also a function of velocity since, for example most plastics do not deform significantly at low speed. Materials of this type are modeled faithfully [63] by the dynamic Navier equation discussed in the next section.

3.3 Dynamic Linear Elastic Solids

In situations where dynamic effects are important, 2.19 is modified to include accelerations,

$$\sigma_{ij,j}(t,\mathbf{x}) + \mathbf{f_i}(\mathbf{t},\mathbf{x}) = \rho\ddot{\mathbf{u}}_\mathbf{i}(\mathbf{t},\mathbf{x}). \tag{3.1}$$

Where $\ddot{u}_i = \frac{d^2}{dt^2}u_i(t,\mathbf{x})$ and ρ is the mass density. This modification changes the equation into a hyperbolic PDE and greatly complicates the solution process.

When the solid Ω is linear, elastic, homogeneous and isotropic the 81 components of E_{ijkl} reduce to just 2 constant values. In general these two parameters

are given by the Young's modulus E and the Poisson ratio ν. In 3.1, this gives rise to two distinct wave speeds which we denote c_1 and c_2. In terms of the material constants, they are

$$c_1^2 = \frac{E(1-\nu)}{\rho(1+\nu)(1-2\nu)} \tag{3.2}$$

and

$$c_2^2 = \frac{E}{2\rho(1+\nu)}. \tag{3.3}$$

Waves of speed c_1 are known as pressure or p waves and those with speed c_2 are shear or s waves. Using these constants, we rewrite 3.1 in terms of displacements to obtain the Navier equation for elastic solids,

$$(c_1^2 - c_2^2)u_{j,ji} + c_2^2 u_{i,jj} + \frac{f_i}{\rho} = \ddot{u}_i. \tag{3.4}$$

We note that other types of waves exist in elastic solids, Raleigh waves (surface waves) are one notable type. In our model, these other wave types are either implicit or are not modeled.

3.4 Boundary Elements

There has been a great deal of work over the past two decades in the boundary element community on the solution of elastostatic and elastodynamic problems [1]. Boundary element (BEM) solution techniques are ideally suited to contact problems because all the unknown quantities are on the boundaries of the colliding bodies. In contrast, FEM (finite element) approaches deal with the entire domain of both bodies and solve for stresses and displacements at every point.

The basic idea of BEM is to reformulate the partial differential equation as an integral equation. There are several ways of doing this for the Navier field equation. We briefly outline one approach and refer the interested reader to [22] or [1].

Denote the solution to Eq. 3.4 by u, with stress field σ and body forces b. Consider a second solution u^* to the field equation 3.4 *with different initial conditions* for the same geometry, with stress field σ^* and body forces b^*. The weighted residual of the two solutions can be written using the Graffi reciprocal relation [22],

$$\int_\Omega (\sigma_{kj,j} * u_k^*) d\Omega + \int_\Omega \rho(b_k * u_k^*) d\Omega - \int_\Omega \rho(\ddot{u}_k * u_k^*) d\Omega = 0, \tag{3.5}$$

where $*$ denotes a convolution product in time. Using the divergence theorem we can rewrite this as

$$\begin{aligned} &\int_\Gamma (p_k * u_k^*) d\Gamma + \int_\Omega \rho(b_k * u_k^* + u_{0k}\dot{u}_k^* + \dot{u}_{0k} u_k^*) d\Omega \\ &= \int_\Gamma (p_k^* * u_k) d\Gamma + \int_\Omega \rho(b_k^* * u_k + u_{0k}^* \dot{u}_k + \dot{u}_{0k}^* u_k) d\Omega, \end{aligned} \tag{3.6}$$

where $u_{0k} = u_k(x,0)$, $\dot{u}_{0k} = \dot{u}_k(x,0)$ are initial displacements and velocities respectively, and p_k is the k'th component of the surface traction. Now, we generate the reciprocal solution u^* as a solution of the field equations when an impulsive body load is applied at the point x^i in the direction l, i.e.,

$$\rho b_k^* = \delta(t)\delta(x - x^i)\delta_{lk}. \tag{3.7}$$

Further, assume the initial conditions are identically zero and that the body forces of the primary solution may be neglected. Then we can write Eq. 3.6 as

$$u_l(x^i, t) = \int_\Gamma (u_{lk}^* * p_k) - (u_k * p_{lk}^*) d\Gamma, \tag{3.8}$$

which gives the displacement at point x^i in direction l as a function of the boundary. Note that this same expression can be obtained from a variational principle (see [2]).

Because the integrals in 3.8 are difficult or impossible to solve analytically, numerical approximation is almost always introduced at this point. Observe that we need only to discretize the boundary, reducing the dimension of the problem by one. In contrast, the FEM approach requires the entire domain Ω to be discretized. Typical BEM algorithms divide up the boundary into piecewise‘ constant, linear or quadratic elements. Corners require special treatment. Also, it is not uncommon for the resulting integrals to be highly singular, in which case, the Cauchy principal value is used (see Appendix A).

As a result of the discretization of the boundary and approximate evaluation of the integrals, we obtain a system of linear algebraic equations for the unknown boundary displacements or tractions. Although the system is much smaller than a similar FEM system, the matrix presents no special symmetry or sparseness properties and is usually solved using dense matrix techniques such as Gaussian elimination.

When the integral 3.8 is discretized, the resulting linear equations can be rearranged so that for nodes on which displacements are given, the tractions are solved for and for nodes with given traction, the displacement is solved for. The result of this procedure is a linear system of the form,

$$\mathbf{A}\mathbf{x} = \mathbf{b}. \tag{3.9}$$

Where $\mathbf{A}$ is a matrix of boundary integrals over each boundary element, $\mathbf{b}$ is the known traction or displacement at each boundary element and $\mathbf{x}$ is the unknown traction or displacement at each boundary element. This gives kn linear equations for a body with n boundary elements and k is a small constant that depends on the order of the integral approximations and the dimensionality of the problem.

Mullen and Rencis [46] review a number of iterative schemes for solving

boundary element equations in the form of eqn. (3.9). Due to the dense structure of the equations, relaxation schemes such as Gauss-Sidel and Jacobi do not compare favorably with direct Gauss elimination. In fact, the only iterative methods which compared favorably with Gauss elimination were the conjugant gradient (Lanczos) type schemes [4].

We end this section by noting that although boundary element techniques are particularly well suited to contact problems, we are by no means restricted to their use. Highly accurate simulation of elastic bodies can be done with finite elements or, given the proper equipment actual physical experimentation can be done.

3.5 Contact Map Determination

We would like to characterize the forces which occur on the surface of a linear elastic solid when it is deformed in various ways. Surface forces are given by the stress tensor evaluated on the surface,

$$\mathbf{p} = \sigma \cdot \mathbf{n}. \tag{3.10}$$

Where $\mathbf{n}$ is the local surface normal. The traction $\mathbf{p}$ can be thought of as a surface pressure. In the following, we assume that bodies in contact have the same sized boundary elements or are integral multiples of each other.

In addition, we assume that the boundary of the body Ω has been discretized into a finite set of nodes $\{\mathbf{x}_1, ..., \mathbf{x}_n\}$. To determine the forces which will occur at a particular boundary node $\mathbf{x}_j$, (see fig. 3.1) we consider the traction generated by a normal step displacement at that point (we denote this boundary segment by Γ_1) on the surface,

$$d_j(t) = -DH(t)\mathbf{n_j}. \tag{3.11}$$

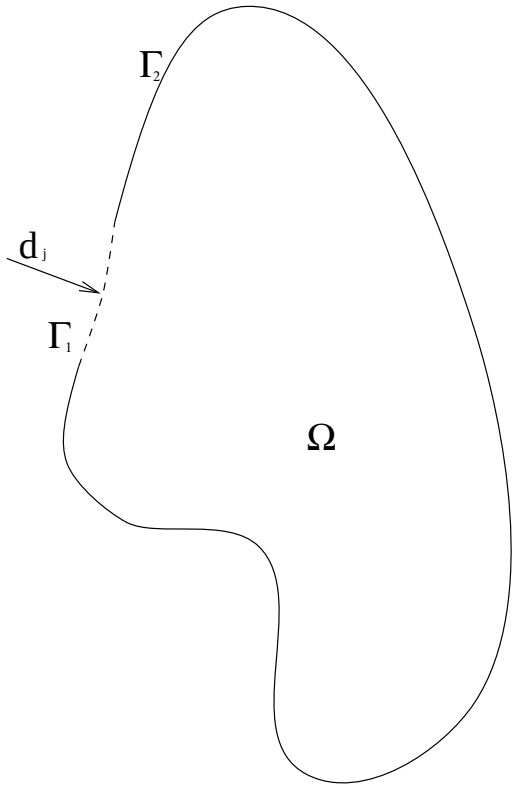


Figure 3.1: Boundary configuration for contact map determination.

Where D is a constant scale factor, $H(t)$ is the Heaviside function and $\mathbf{n_j}$ is the unit normal at point $\mathbf{x}_j$. All other points on the surface (denoted by Γ_2) are free, with Neumann condition,

$$p_k(t) = 0 \quad \forall \quad \mathbf{x}_k \neq \mathbf{x}_j. \tag{3.12}$$

Here p_k is the surface traction at point $\mathbf{x}_k$.

Because of the dual nature of boundary conditions in BEM it is an ideal candidate for determining the response with mixed boundary conditions such as these.

Once the step response has been determined at various points on the boundary Γ, we compute the impulse response by numerically differentiating the response function. We discuss how this is done in some detail in the next section. The result is essentially the numerical determination of the Green's function for the IBVP consisting of eq. (3.4), (3.11), (3.12) and zero initial condition. We denote the computed Green's function for body A by

$$g^A(x,t). \tag{3.13}$$

Where $x \in \{\mathbf{x_1}, \mathbf{x_2}, ..., \mathbf{x_n}\}$, is the set of surface nodes in the discretized boundary. For simplicity, we drop the explicit x reference in what follows by assuming that the contact point does not change significantly during a collision event. For collision type impacts with a time scale in the tens to hundreds of micro-seconds this assumption is reasonable. For sustained contact events, some motion may occur and tangential elastic forces will need to be taken into account.

We note that given the well posed nature of the above IBVP, we are assured of the existence of the Green's function. Also, since the problem is linear uniqueness follows directly.

At this point we are ready to define a contact map.

Definition 1 *The contact map C_A for a linear elastic solid A is a set of functions $\{g^A(\mathbf{x_1}, \mathbf{t}), ..., \mathbf{g^A}(\mathbf{x_n}, \mathbf{t})\}$ defined at a discrete set of nodal points $x \in \{\mathbf{x_1}, \mathbf{x_2}, ..., \mathbf{x_n}\}$ located on the surface of A, and on the time interval $t \in (0, T]$. The functions $g^A(\mathbf{x_j}, \mathbf{t})$ correspond to the Green's function solution of eq. (3.4) for A due to a normal displacement $\delta(t, \mathbf{x} - \mathbf{x_j})$.*

As is well known from the study of differential equations, the Green's function allows us to solve the PDE for any boundary function by a convolution [23]. We pursue this technique below.

3.5.1 Linear Systems Theory

Note that the equation for elastic bodies (3.4) is a linear PDE. This linearity allows us to consider the response of an elastic body in terms of linear systems theory [52], which will lead to a technique for using contact maps.

In terms of systems theory, contact maps characterize the input-output response of a solid with no explicit reference to the internal state. Each boundary node on the body corresponds to an input 'terminal' and the corresponding tractions are output 'terminals'. In this sense, the elastic dynamics of the body constitute a black box system which we can only investigate and characterize using these 'terminals'. This approach is known as *lumped systems theory* and is discussed in detail in [38].

Linear systems theory uses three basic functions to analyze input-output systems, the delta function $\delta(t)$, the Heaviside step function $H(t)$ and the one parameter family of sinusoids $e^{i\omega t}$. For our present purpose, we consider only the first

two. We use the T subscript to denote the discrete versions of these functions:

$$\delta_T(t) = \begin{cases} 1 & for\ t \in [0,T] \\ 0 & otherwise \end{cases}, \tag{3.14}$$

$$H_T(t) = \begin{cases} 0 & for\ t < 0 \\ 1 & for\ t > 0 \end{cases}. \tag{3.15}$$

The subscript is necessary because the discrete delta function contains an explicit time scale, in contrast to the actual delta function.

The discrete response $h(t)$ of a linear system to a step function also gives the the impulse or delta function response $g(t)$ by backward differencing:

$$g(kT) = \frac{h(kT) - h((k-1)T)}{T}, \tag{3.16}$$

for integral k. Note that this relationship is exact in the sense that sampling the continuous impulse response will produce the same result(see [21] for a proof) as (3.16).

Given the impulse response, we can use the properties of the delta function and the system linearity to construct the response to arbitrary inputs. This is done with the discrete analogue of the convolution product,

$$F(nT) = \sum_{k=0}^{n-1} g((n-k)T)y(kT)T. \tag{3.17}$$

Where $y(kT)$ is the actual system input. We incorporate this very useful result into our time stepping algorithm in the next section.

3.6 Simulation Using Contact Maps

Once the contact maps have been determined for a number of bodies, we naturally would like to do simulations in which objects may collide with each other. A full

featured simulation environment is discussed in Appendix B, but we need only consider an isolated collision event here. For our purposes, the simulation engine needs to provide us with accurate inter-object distances in the sense of minimum translation distances (MTD's).

Definition 1 *The MTD is a real valued function of two geometric objects given by* $sgn\{\mathbf{l} \cdot \mathbf{n}\}|\mathbf{l}|$. *Where* $\mathbf{l}$ *is the shortest (in a Euclidian sense) vector joining the surfaces of the two objects, and* $\mathbf{n}$ *is the surface normal at the point* $\mathbf{l}$ *is measured from.*

The MTD between two objects is the minimum distance that one object must be translated for the two objects to just touch [13]. We discuss algorithms for extremely fast calculation of this distance in Appendix B.

Once it has been determined that two objects are colliding, we must find the closest nodes at which we know the contact map on each object. This can be quickly done if the map is stored in a topologically consistent data structure. After the relevant maps have been found, we integrate the collision using a convolution technique described next.

3.6.1 Convolution Integration

Given the relevant step response curves, we develop a procedure for resolving collisions in a rapid manner. The idea of this approach is to consider the penetrations as occurring in discrete steps and use the impulse response to calculate the behavior of the solids at each step. In addition, we must ensure that the principle of action-reaction (Newton's third law) is satisfied at each step.

In this algorithm, we consider the collision of two bodies A and B. Define

the penetration depth ϵ of A as,

$$\epsilon^A = (\tilde{\mathbf{u}}^A - \mathbf{u}^{\mathbf{A}}) \cdot \mathbf{n}. \tag{3.18}$$

Where $\tilde{\mathbf{u}}$ is the co-ordinate of a point on the undeformed boundary, $\mathbf{u}$ is the co-ordinate of the same point on the deformed boundary and $\mathbf{n}$ is the contact normal. The penetration depth of B is define similarly. We can think of ϵ as the surface strain of body A. Assume that contact begins when time $t = 0$. We compute the force at time $\tau > 0$ using a convolution product with the penetration trajectory,

$$f(\tau) = \int_0^\tau g(\tau - t)\epsilon(t)dt. \tag{3.19}$$

In this section, we assume that the specific impulse response map $f_s(t)$ has been determined by the methods described above, experimentation or by another method. Further, we assume that the simulator has selected the correct contact map based on the contact location.

Now we approximate the penetration depths by a series of impulses and thus compute the force up to time t_k as

$$\tilde{f}^A_{k+1} = \sum_{j=0}^{k} g^A(j\Delta t)\epsilon^A_{k-j}. \tag{3.20}$$

Where we define $\tilde{f}^A_{k+1} = \tilde{f}^A(t_{k+1})$. Equation (3.20) is essentially the endpoint approximation to (3.19) and has a constant error order independent of Δt. We use this quadrature formula only to simplify the algebra. In practice, a better approximation would be used such as midpoint or Simpson's rule. Note that if the time step Δt is the same as the constant T used in (3.16) we do not need to multiply (3.20) by Δt in the discretized equation. A similar calculation is performed for for $\tilde{f}^B(t_{k+1})$.

Given the forces $\tilde{f}^A_{k+1}$ and $\tilde{f}^B_{k+1}$ we compute the updated positions of the two bodies using a standard ODE integrator, such as RK4 (Runge-Kutta-4, see [10]). This will give us the updated values for the boundary coordinates $\tilde{\mathbf{u}}^A$ and $\tilde{\mathbf{u}}^B$.

With the updated boundary coordinates, we compute the total penetration depth along the contact normal as

$$\epsilon_{k+1} = (\tilde{\mathbf{u}}^A_{k+1} - \tilde{\mathbf{u}}^B_{k+1}) \cdot \mathbf{n}. \tag{3.21}$$

To determine the updated positions of the deformed boundary points, we use the principle of action reaction,

$$\mathbf{f^A_{k+1}} + \mathbf{f^B_{k+1}} = \mathbf{0}. \tag{3.22}$$

We have in the normal direction

$$f^A_{k+1} = \tilde{f}^A_{k+1} + g^A(\Delta t)\epsilon^A_{k+1}, \tag{3.23}$$

$$f^B_{k+1} = \tilde{f}^B_{k+1} + g^B(\Delta t)\epsilon^B_{k+1}. \tag{3.24}$$

for both A and B. And also,

$$\epsilon^A_{k+1} - \epsilon^B_{k+1} = \epsilon_{k+1}. \tag{3.25}$$

Equations 3.23 and 3.25 form a system in two unknowns δ^A_{k+1} and δ^B_{k+1}. Solving the system yields,

$$\epsilon^A_{k+1} = \frac{g^B(\Delta t)\epsilon_{k+1} - \tilde{f}^A_{k+1} - \tilde{f}^B_{k+1}}{g^A(\Delta t) + g^B(\Delta t)} \tag{3.26}$$

$$\epsilon^B_{k+1} = -\frac{g^A(\Delta t)\epsilon_{k+1} + \tilde{f}^A_{k+1} + \tilde{f}^B_{k+1}}{g^A(\Delta t) + g^B(\Delta t)} \tag{3.27}$$

which gives the co-ordinates of the deformed boundary for the next time step.

The time stepping algorithm is repeated until the value for the contact force has dropped to zero, at which point it is assumed that the bodies have separated.

Note that the actual separation distance is not likely to be zero at the same time as the force during object restitution. In this respect, this model is similar to the bilinear force model [59] and to the traditional spring damper model [26]. Further, note that in the case where the impulse response map has no memory (i.e. the step response is constant), the above algorithm is identical to a linear spring penalty model.

3.6.2 Complexity

The complexity of this contact algorithm is certainly greater than that of both the impulsive and penalty approaches ($O(1)$). This is due to the convolution product in eq. (3.20) whose per step cost grows linearly with the number of time steps. Although we do not pursue it in this thesis, we note that this complexity can be reduced to a constant order by replacing the convolution with the finite difference equation corresponding to the Green's function (see [52]).

3.7 Discussion

First note that the time step Δt in the above algorithm should match the time scale introduced in the calculation of the contact response map. Since the computed contact response map will have a large peak at the first sample point, using a different time scale will introduce extra error into the algorithm.

As presented, the integration algorithm is a zero order technique. This is due to the endpoint approximation to the convolution integral in (3.20). Higher order could be attained if the displacement were a continuous function.

Consider the traction response to a set of basis functions in an appropriate function space. One promising possibility would be to use a compactly supported

wavelet basis. Compact support would reduce the computational cost of the convolution product in equation (3.19). Another possibility it to use a standard Fourier basis. However, for harmonic bases it is simpler to Fourier transform the original equation (3.4) into:

$$(c_1^2 - c_2^2)\bar{u}_{j,ji} + c_2^2\bar{u}_{i,jj} + \frac{f_i}{\rho} = \omega\bar{u}_i. \tag{3.28}$$

Where $\bar{u}$ is the Fourier transformed displacement. Equation (3.28) is much easier to solve than its dynamic counterpart. Numerical techniques that speed up the solution of the corresponding BEM equation, such as multigrid [31] can be applied to (3.28) in a straightforward way.

The generalization of the contact map algorithm to arbitrary bases is beyond the scope of this thesis. We note however that for tangential motion or three dimensional impact, a more general approach is needed than the simple scalar step function approach. The reason for this is that we have more than just a one dimensional space to span. To accommodate tangential impact in 2D for example, we would need a matrix of step functions to span the space of allowable impacts.

3.7.1 Relation to Viscoelasticity

The convolution integral (3.19) used in simulation has a direct correspondence to the constitutive relation in the theory of viscoelasticity. In that theory, stress and strain are related by a Stieltjes integral expressed in convolution form as

$$\sigma_{ij}(t) = \int_0^t G_{ijkl}(t-\tau)d\epsilon_{kl}(\tau) \tag{3.29}$$

(see [15]). This integral expresses the *memory property* of viscoelastic materials. A discretized version of (3.29) would yield an algorithm similar to the one in the previous section, but using the step response map see ([64]).

In this theory, elastic materials can be identified as $G_{ijkl}(t-\tau) = E_{ijkl}\delta(t)$ which reduces (3.29) to the standard elastic constitutive equation. For a given material, the tensor G_{ijkl} is determined by applying a step strain to a half space of a given material and determining the resulting time dependent stress.

The relationship between (3.29) and (3.19) is readily apparent. However the contact map is calculated from the equations for a non-dissipative elastic solid, so the *memory property* can only be related to the boundary conditions. Viscoelastic theories also make the additional assumption that the magnitude of the tensor G_{ijkl} is a monotone decreasing function of time. In our case, the response can only be said to be bounded in time due to oscillations from reflected strain waves.

3.7.2 Energy Loss

In this section, we prove that for a broad class of contact maps, collisions will always dissipate energy. This is an important point, because many impulsive laws have energetic consistency problems especially in the presence of friction. We discuss impulsive consistency in chapter 5.

We first note that contact response maps consist of two main components, the transient part and the steady state. Since we have a step displacement at time $t = 0$, the force response has a singularity $(-\infty)$ at that point. Then on about the time scale of the wave speed c_p, the solid equilibrates and the force becomes just the characteristic solution for the geometry. A typical contact map is shown in figure 3.2.

Theorem 1 *The change in energy due elastic waves is negative after a collision when using the contact algorithm given above.*

Proof:

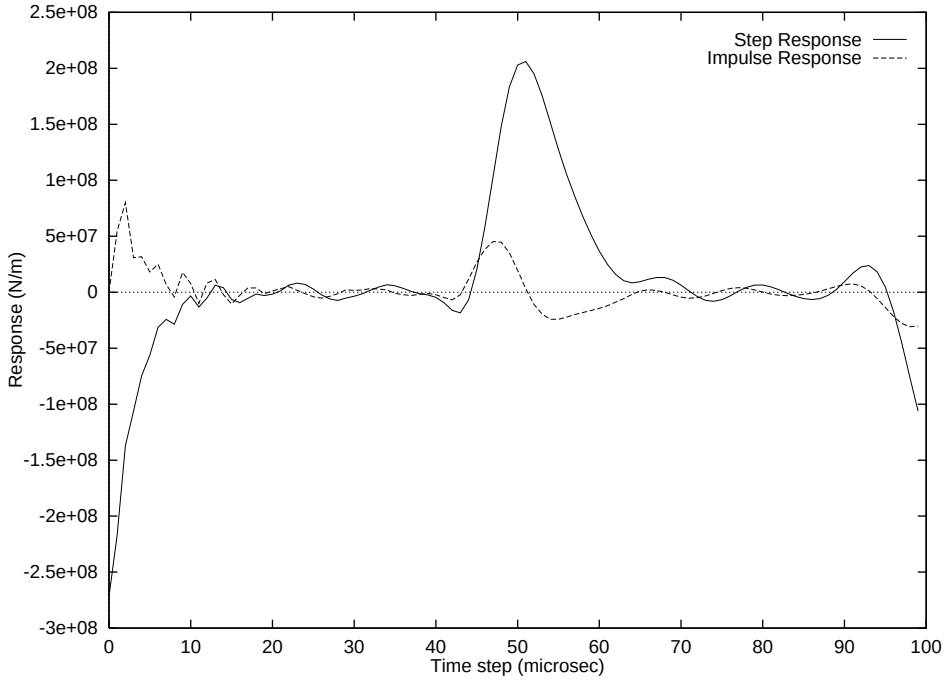


Figure 3.2: Traction response curves (step and impulsive) of a steel bar to a step displacement of 10^{-6} m normal to one end. The impulse response was computed using 3.16 but not scaled by $1/T$.

Consider the total energy of two bodies undergoing a collision,

$$E(t) = \int_0^t g^A(t-\tau)\epsilon^A(\tau)d\tau\epsilon^A(t) + \int_0^t g^B(t-\tau)\epsilon^B(\tau)d\tau\epsilon^B(t). \tag{3.30}$$

The two integral terms are the work done to arrive at the current strain depths ϵ^A and ϵ^B.

Taking the time derivative and using the fundamental theorem of calculus,

$$\dot{E}(t) = \int_0^t g^A(t-\tau)\epsilon^A(\tau)d\tau\frac{d\epsilon^A}{dt} + \int_0^t g^B(t-\tau)\epsilon^B(\tau)d\tau\frac{d\epsilon^B}{dt}. \tag{3.31}$$

The total force exerted by each body on the other is also given by

$$F(t) = \int_0^t g^A(t-\tau)\epsilon^A(\tau)d\tau = -\int_0^t g^B(t-\tau)\epsilon^B(\tau)d\tau. \tag{3.32}$$

To get

$$\dot{E}(t) = F(t)\frac{d(\epsilon^A - \epsilon^B)}{dt}. \tag{3.33}$$

Now integrating over the collision with $\epsilon = \epsilon^A - \epsilon^B$,

$$\int_0^T \dot{E}(t)dt = \Delta E(T) = \int_0^T F(t)\frac{d\epsilon}{dt}dt. \tag{3.34}$$

Which is simply a statement of work done. Integrating the right by parts gives

$$\Delta E(T) = [\epsilon(t)F(t)]_0^T - \int_0^T \epsilon(t)F'(t)dt. \tag{3.35}$$

And using the fact that $\epsilon(0) = \epsilon(T) = 0$, and expanding the convolution,

$$\Delta E(T) = -\int_0^T \epsilon(t)g^A(0)\epsilon^A(t)dt. \tag{3.36}$$

Now by definition of ϵ and ϵ^A, they always have the same sign. Thus for response maps with positive initial force response (to a delta boundary strain), the energy change due to elastic waves will be negative.

From the proof we can draw a few observations. First, note that since $F(t)$ is independent of the velocity, the energy loss is homogeneous of degree 1 in velocity [14]. Secondly the total amount of energy loss can be roughly estimated by the difference between the average force and the steady state force.

3.8 Summary

This chapter presented a complete algorithm for determining and using contact response maps. Contact response maps attempt to capture more of the *physics* of the objects using as few input parameters as possible. It is important to note that there is very little ambiguity in the determination of a contact map, since the density and Young's modulus are relatively easy to obtain for a wide variety of materials. The object geometry can be determined with a great deal of accuracy as well, using for example the ACME [48] apparatus.

Chapter 4

Numerical Experiments

4.1 Introduction

In this section, we exhibit some examples computed using the contact map algorithm introduced in the last chapter. We begin by showing the consistency of the algorithm with the energy conserving harmonic penalty model, then we give examples which have a damping component. We also discuss the possibility of fitting both impulsive and force response rigid models from Chapter 2.

4.2 Consistency

We begin by considering a contact map generated from a step response which is flat. A flat step response indicates that the body has no memory and behaves in a perfectly lumped fashion. This can be interpreted physically as an elastic solid with no transient behaviour and no wave properties. All the models of Chapter 2 belong to this category.

We model the collision of two unit spheres both with identical constant step

Step Size	L_{inf} Force Error
2×10^{-4}	2.45
1×10^{-4}	1.21
5×10^{-5}	.456
2×10^{-5}	.240

Table 4.1: Maximum force errors between contact map and harmonic penalty model.

response. These step response profiles were chosen arbitrarily and do not correspond to the solution of any elastic solid models. The constant was $4000N/m$, corresponding to a spring penalty model $f(\epsilon, \dot{\epsilon}) = k\epsilon$ with spring constant $k = 4000N/m$.

Table 4.1 clearly shows that the contact map algorithm is only of first order. In addition, our implementation uses a fixed step size 4th order Runge-Kutta [10] integrator for the ODE. Since this integrator is explicit, very small steps (on the order of $\frac{1}{k}$) must be taken to resolve the high frequency oscillations.

4.3 Gauging Energy Loss

In this section, we try to get a feel for the amount of energy loss associated with a contact map and in particular how the energy loss relates to the transient portion of the response. We collide two spheres, as before, but add successively larger transients to the step response. One of the spheres is modeled with a non-transient map as in the previous section. The step responses are shown in figure 4.1 and their corresponding impulse responses in 4.2. Our principal measure of energy loss is the relationship of inward to outward velocity for the spheres as in the Newtonian definition of restitution. Since this model is frictionless, the three definitions of the restitution coefficient are equivalent.

The computed restitution is shown in figure 4.3. Collision times for these

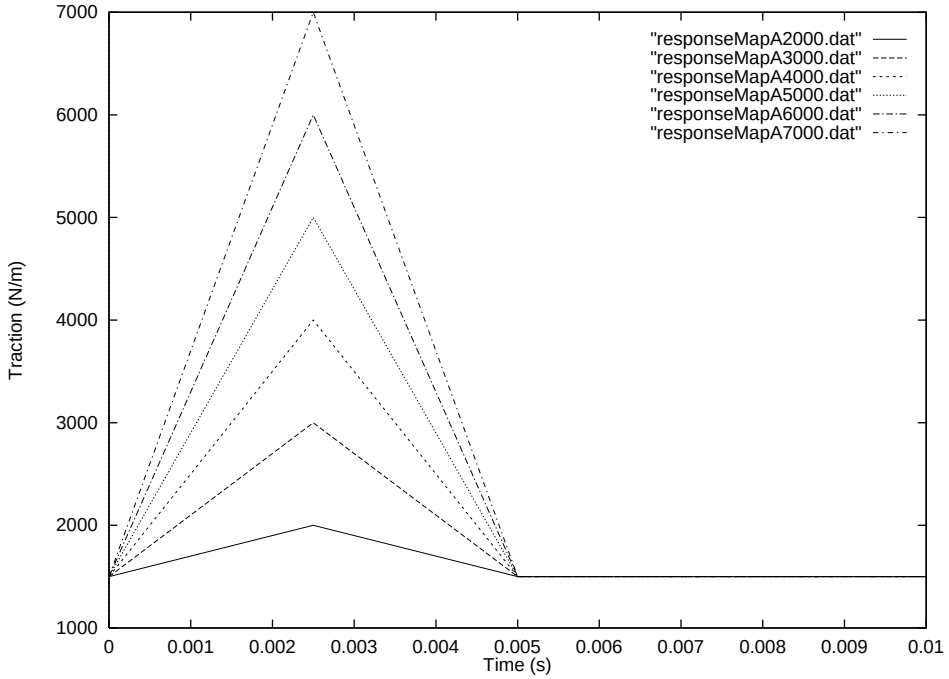


Figure 4.1: The step response curves used to gauge energy loss. The steady state behaviour is constant at 1500 N/m in all cases.

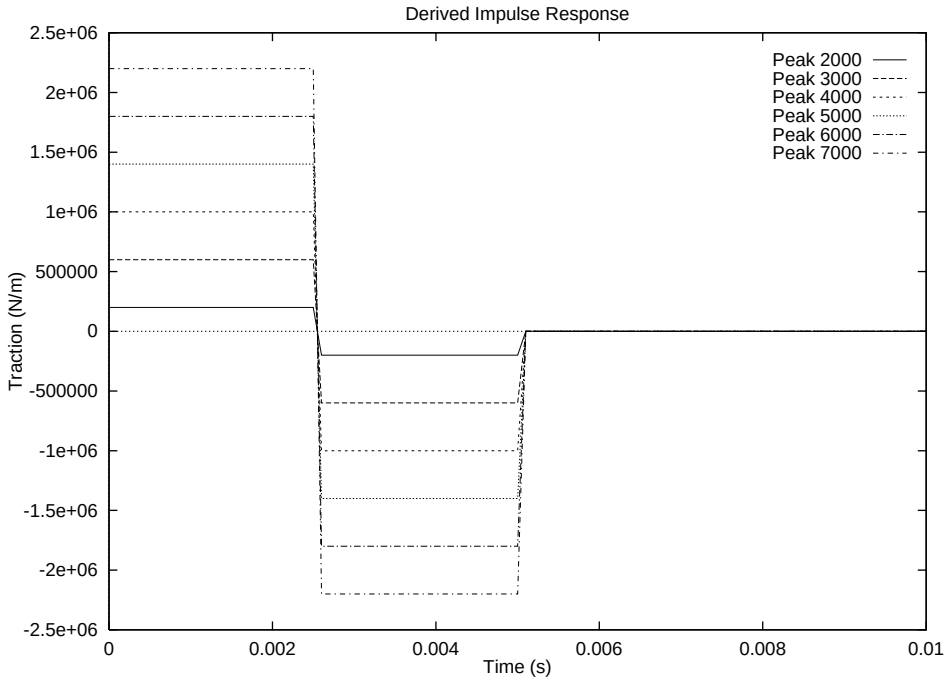


Figure 4.2: Derived impulse response curves used to gauge energy loss.

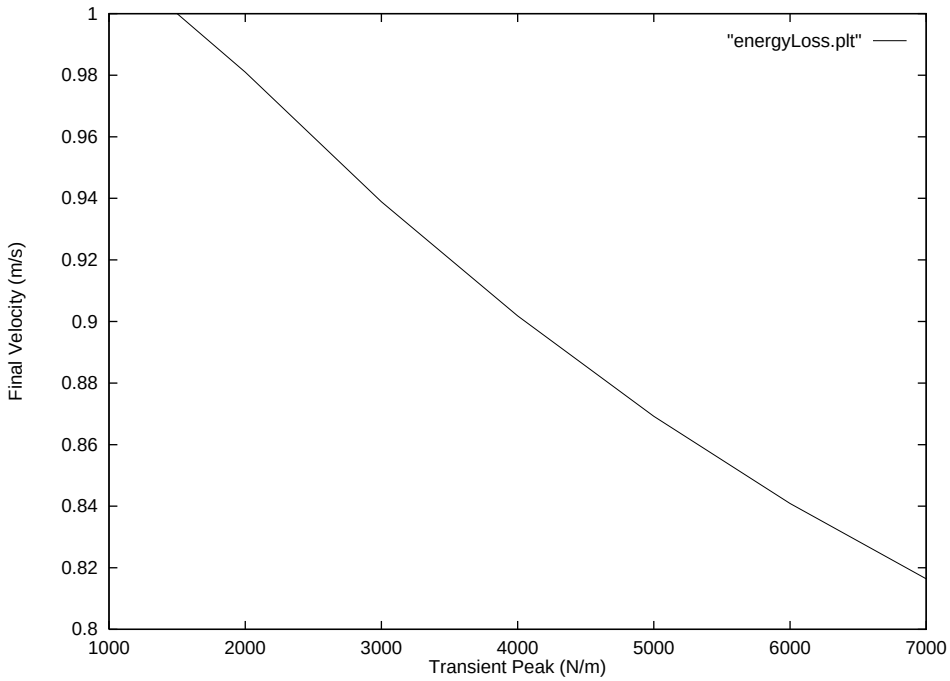


Figure 4.3: The computed restitution coefficient for each step response, as a function of the transient peak size.

cases was around .080s, so most of the force was the result of the steady state, and the transient acted as a damping mechanism.

We note that in these simulations, the step size for correct integration was set to 1×10^{-4}. This is quite small since it takes nearly 800 integration steps to resolve the collision in this case. One possible way to speed this up would be to reformulate the contact algorithm to handle variable step size integration. Another approach is to try and extend the algorithm itself to be 2nd or higher order accurate as discussed in the previous chapter.

A third technique for speeding up this approach is to use the results of these simulations to determine coefficients for a model from chapter 2. As an example, we consider the simple harmonic force with damping given by $f(\epsilon, \dot{\epsilon}) = c\dot{\epsilon} + k\epsilon$. Using

least squares on the computed trajectories from the *responseMapA6000* in figure 4.1, we get $k = 1591.3N/m$ and $c = 9.0728Ns/m$, a modestly damped system. Note however, that there is no viscous damping in this system. The non-zero value for c is due entirely to internal elastic energy storage. This approach might also benefit existing simulation systems that rely on explicit force models such as [26],[19].

Using this approach, more complex force models could be fit, such as [51]) the model of Hunt and Crossley eqn. (2.15) with a non-linear fit (e.g., Levenberg-Marqd.)

4.4 Real Contact Maps

The contact maps discussed in the last section were syntheticaly generated, in that they don't necessarily correspond to any real solid. In this section, we examine some maps generated using the boundary element technique discussed in the last chapter. Our geometry will be the often studied uniform isotropic bar with rounded ends being dropped at various angles.

The boundary of the bar is shown in figure 4.4 along with the nodal points which denote the boundary elements. The BEM algorithm used quadratic support functions to discretize the boundary [22]. We simulate dropping this bar by setting the normal displacement at successive boundary nodes at one end of the bar. All other boundary nodes are given Neumann conditions as discussed above.

The material properties of this bar, and its physical dimensions are as in [56]. We summarize them here for convenience in table 4.2.

The resulting derived impulse maps are shown in figures 4.5 through 4.12.

Note that the point at time $t = 0$ is undefined and can be chosen arbitrarily as 0. Maps for other angles can be determined by rounding to those given angles or

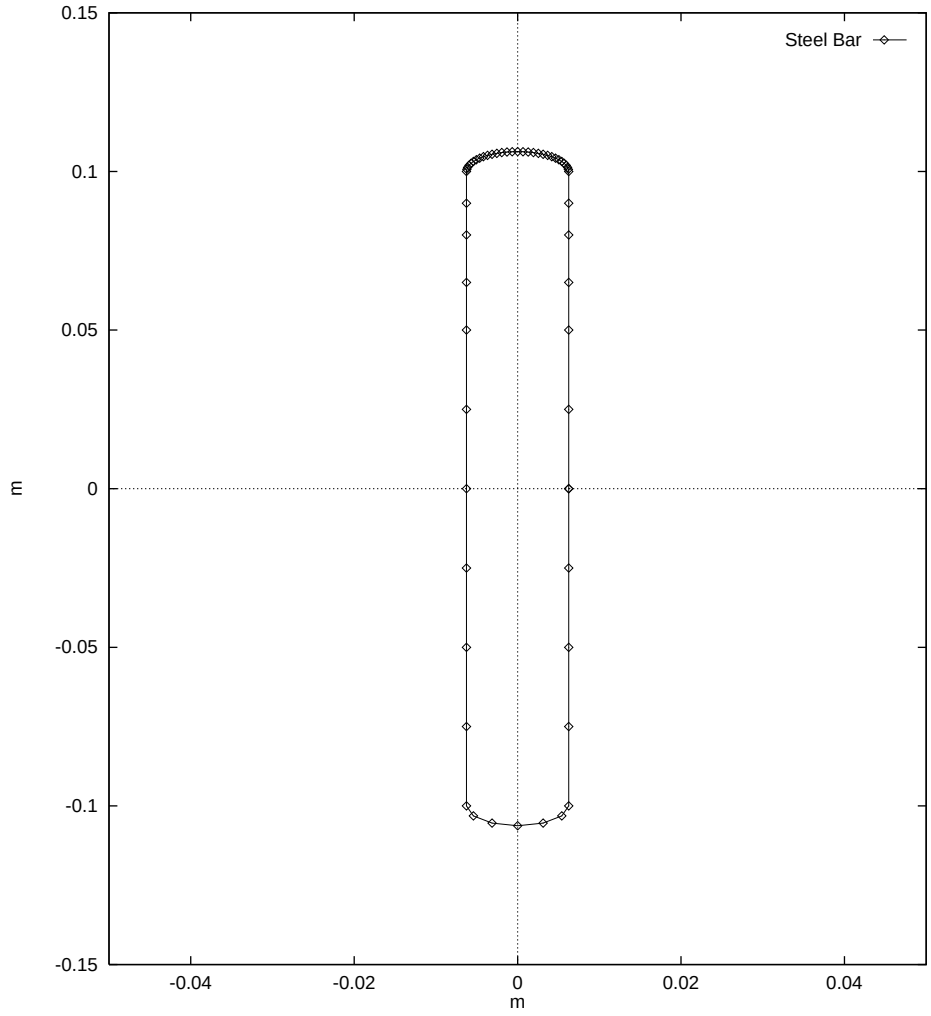


Figure 4.4: The bar used to determine contact maps

E	$2.1 \times 10^{11} N/m$
ρ	$7876.74 kg/m^3$
c_1	$9660. m/s$
c_2	$5163. m/s$

Table 4.2: Bar material properties.

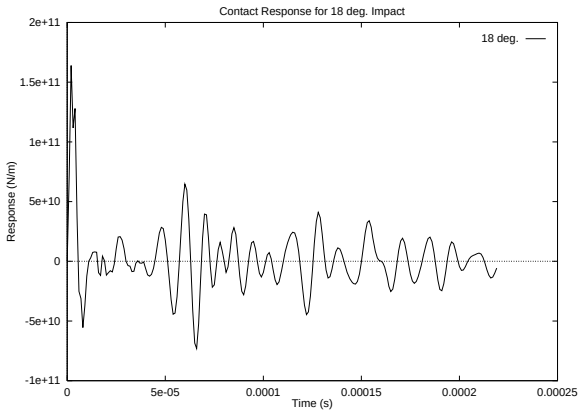


Figure 4.5: Derived impulse maps resulting from 18 deg. impact.

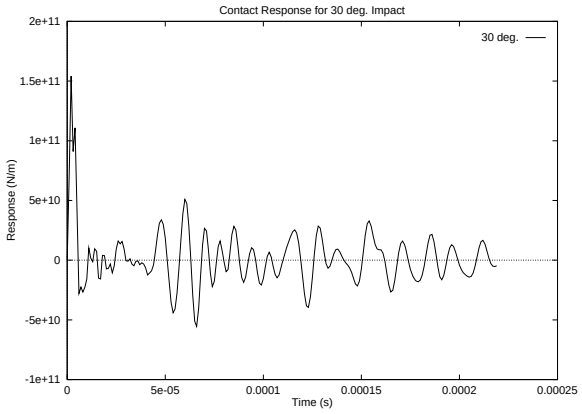


Figure 4.6: Derived impulse maps resulting from 30 deg. impact.

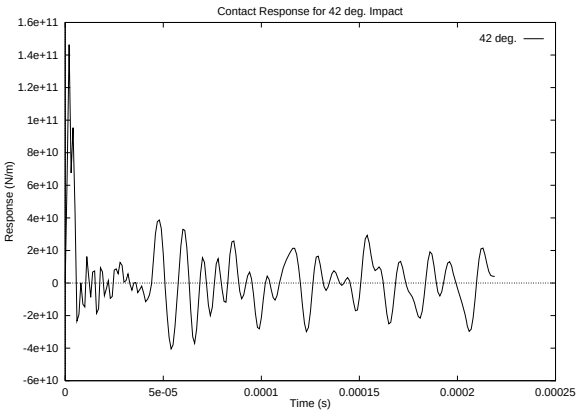


Figure 4.7: Derived impulse maps resulting from 42 deg. impact.

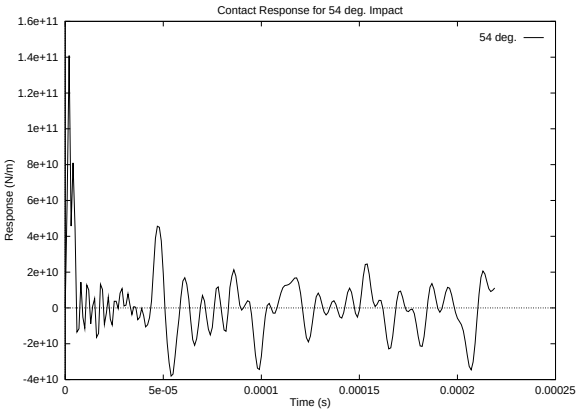


Figure 4.8: Derived impulse maps resulting from 54 deg. impact.

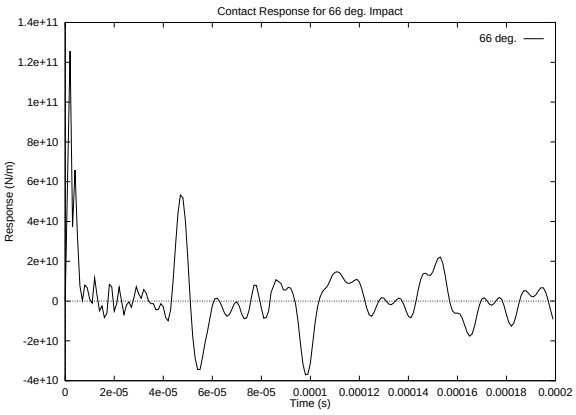


Figure 4.9: Derived impulse maps resulting from 66 deg. impact.

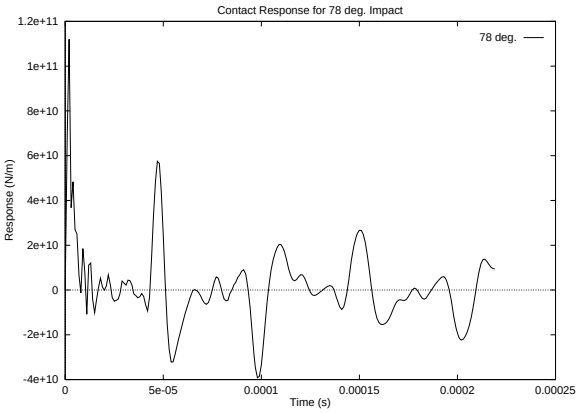


Figure 4.10: Derived impulse maps resulting from 78 deg. impact.

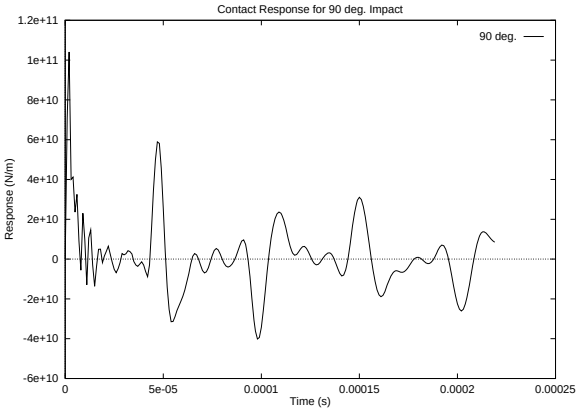


Figure 4.11: Derived impulse maps resulting from 90 deg. impact.

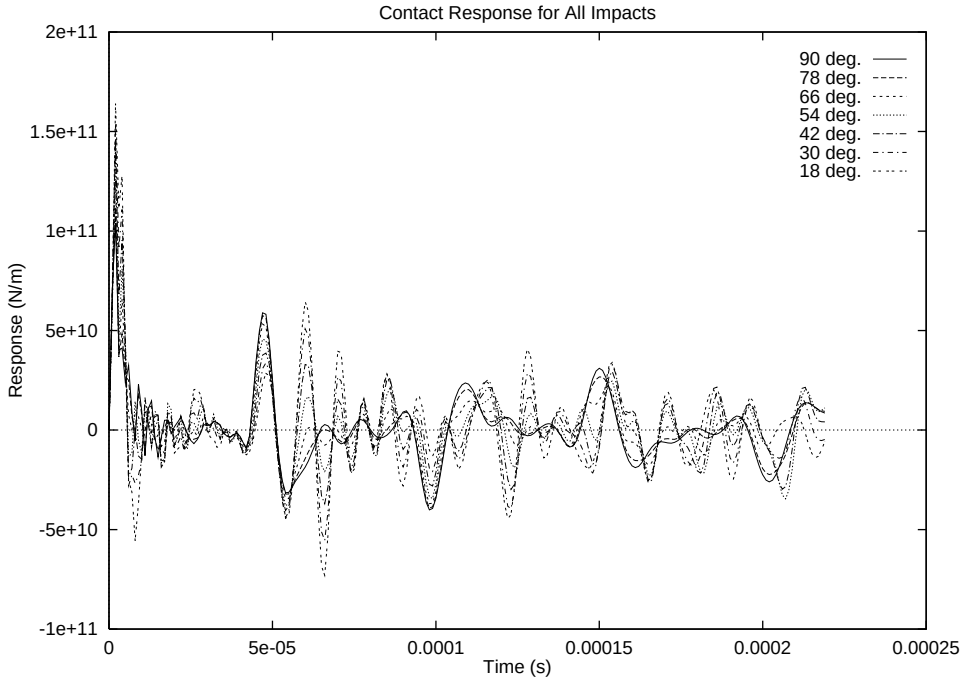


Figure 4.12: Derived impulse maps resulting from step displacement at each angle.

by some type of interpolation.

We now make a few observations on the contact response maps. The initial transient peak is on the same order as the Young's modulus of elasticity for the material ($2.1 \times 10^{11} N/m$). We expect this since in the early stages of compression, the solid should exhibit perfect Hookean response.

The return wave time of approximately $40\mu s$ is consistent with the wave speed in table 4.2 and a bar length of $0.2m$. As the impact angle decreases from normal, the pressure wave will be reflected more times in the interior before returning to the source point. This behaviour is manifested in 4.12 by multiple peaked response for $\theta = 18^\circ$ (fig. 4.5)and only a single wave pulse for $\theta = 90^\circ$ (fig. 4.11). Further for angles below about 66° (fig. 4.9), there is an nearly immediate reflection from the opposite side of the bar.

Simulation using the maps shown in fig. 4.12 will be discussed in chapter 6, after we have reviewed friction models.

4.5 Summary

In this chapter, we showed the consistency of the convolution algorithm presented in the last chapter in the case of a linear penalty model. The relationship of the memory length of the contact response map to the loss of energy to internal modes was exhibited for a set of artificially chosen step response curves. The non-local nature of the contact algorithm makes an analytic prediction of the energy loss difficult in all but the simplest maps. It was also shown how to use this algorithm as an off-line analysis tool to calculate the coefficients of impulse and force response rigid models. This technique offers to increase the accuracy of such algorithms with minimal extra computation during a simulation. Finally we generated some actual

contact maps for the rigid bar geometry.

Chapter 5

Friction

5.1 Introduction

Up to this point in the thesis, we have assumed collisions did not involve any velocity tangential to the point of contact. However, in nearly all contact situations there is some tangential motion, which can have a significant effect on the collision resolution. Forces which occur during tangential motion are largely the result of friction, however there has been some discussion in recent years of *tangential restitution* [8].

We begin by discussing the physics of friction and the commonly used models. Following this, we describe some impulse rigid models which try to capture frictional effects and some of their problems. Finally, we describe models which try to predict frictional forces.

5.2 Historical Friction

The most common friction law, still widely used, was proposed by Coulomb nearly 2 centuries ago. This law is taught in most first year physics courses and serves as

a benchmark against which new models must be measured.

Coulomb's most remarkable discovery was that friction is independent of the area of contact and only slightly dependent on velocity. The largest component of friction is due to the load or normal force of contact. Also if two objects are pressed together with zero relative velocity, they tend to *stick* even for moderate non-zero tangential force. Coulomb's law of friction is stated mathematically as,

$$|f_T| \leq \mu |f_N|. \tag{5.1}$$

Where $\mu \in [0, 1]$ is known as the coefficient of friction. The coefficient μ is sometimes given by two values, one for static and one for kinetic motion. Strict equality of (5.1) is required for non-zero tangential velocity (kinetic). For zero tangential velocity, the tangential friction force is constrained only to lie within the force cone defined by the inequality.

The indeterminacy of (5.1) during sticking (zero tangential velocity) leads to non-uniqueness of solutions. This is discussed by several authors [45], [6], [7]. We also discuss this dynamic effect below.

5.2.1 Rotational Friction

One immediate extension of the simple Coulomb model is for rotational friction. When two bodies in contact have nonzero relative angular motion, friction will occur over the contact area A. We can write,

$$M = \int_A x f_y(x, y) - y f_x(x, y) dA. \tag{5.2}$$

Where $f_j(x, y)$ is the coulomb friction at point (x, y) in direction j as given by (5.1). Further extensions of this are discussed in [28], [29].

5.3 Tribology

Modern theories of surface contact forces and effects are the subject of the field of tribology. From an engineering perspective the Coulomb model is usually sufficient, however it provides little insight into the physics of friction nor do we have a good understanding of the limits of the Coulomb model.

5.3.1 Physics of Contact

Nearly all surfaces, no matter how smooth they appear to the eye contain a rough terrain at micro (10^{-6}m)to nano (10^{-9}m) scales. The peaks of this terrain are called *asperities*. Asperities can be observed readily with a profilometer and are usually characterized by their statistical properties [30]. Asperities are the arbiters of contact force between two bodies.

In addition to microscale roughness, all solids have chemical properties and some combinations can significantly affect contact forces. For example, coefficient of friction is known depend strongly on the level of oxidation in the contact area. Tabor [61] describes placing two indium metal spheres into contact, when highly cleaned they adhere, but when exposed to air before contact, they slide easily.

More complex models of friction can be constructed, based on modern knowledge of tribology. One example is a proposal by Pollak [50] which includes adhesive forces,

$$|f_T| = \mu(P + 2\pi Rw + \sqrt{6\pi RwP + 9\pi^2 R^2 w^2}). \tag{5.3}$$

Where P is the surface pressure, R is the radius of curvature near the contact zone, w is an adhesion coefficient. The first term of (5.3) represents the load, the second is the adhesion and the third expresses the variance of adhesion with load.

One of the chief problems with models such as (5.3) is that for real bodies, it is difficult to obtain the correct coefficients and once obtained, they may easily change due to environmental effects. In light of this, we are forced to agree with Chatterjee [14] who states that *collision models are currently so poor that we can hardly err by using the Coulomb model.*

5.4 Impulsive Collisions with Friction

When using an impulse response rigid contact model, the addition of friction dramatically effects model consistency. The three definitions of restitution given above are all equivalent in non-frictional collisions but predict dramatically different outcomes when friction is included. Following an insightful paper by Routh [36] on the two dimensional case, almost no research was done in this area until the work of Keller [40], who considered the full 3D contact problem. Since [40], a large number of papers have been published on this topic, see for example [8], [57], [58], [65]. An excellent review can be found in [14].

Stronge noted in [57] that the standard definitions of restitution, either (2.8) or (2.9) fail to conserve energy when the tangential velocity is zero at some point during the collision (i.e. contact sticking). This problem is resolved by using Stronge's work related restitution definition.

$$e^2 = \frac{W_c}{W_r} = \frac{\int_0^{t_{mc}} \mathbf{p} \cdot \mathbf{n} dt}{\int_{t_{mc}}^{t_f} \mathbf{p} \cdot \mathbf{n} dt} \tag{5.4}$$

Where t_{mc} is the time of maximum compression, and t_f is the collision termination.

Chatterjee [14] analyzes in detail the problem of energy conservation for

three dimensional collisions. The chief tool in his analysis is the energy ellipse, an ellipsoidal region in impulse space that bounds all energically consistent collision impulses. The one parameter impulse models in chapter 2 can be thought of as defining curves within this ellipsoid. The addition of tangential forces (and tangential coefficients) transform this line into a surface. The difficulty encountered by many authors constructing impulse response rigid models is to constrain the parameters of their models to the interior of the energy ellipsoid.

In addition to the energetic consistency problem, the so called stick-slip phenomenon when the tangential velocity vanishes during collision, introduces non-uniqueness. The direction of the tangential force when objects are sticking is not predicted by the Coulomb model. We discuss a possible solution to this problem in the next section.

Finally, we note that a complete simulation environment has been written which uses the Stronge definition of restitution along with Coulomb friction. The system, called *Impulse* is the subject of Mirtich's PhD thesis [45].

5.5 Solution Existence and Uniqueness

In this section, we begin by formulating the impulsive contact problem as a *linear complementarity problem* or LCP. LCP's are widely known in the mathematical community and this standard formulation will expose the core issues for consistent solution of frictional collisions.

We consider a situation in which several bodies are colliding at n points. By consideration of the nonpenetration constraints at each contact point we can write Newton's law,

$$\mathbf{a}_n = \mathbf{A}\mathbf{f}_n - \mathbf{b}. \tag{5.5}$$

Where $\mathbf{A}$ describes the relative inverse inertias of the bodies, $\mathbf{f}_n$ are the contact forces, $\mathbf{b}$ are the external accelerations and $\mathbf{a}_n$ are the resulting outward normal accelerations. For impulsive collisions, we use the same equation but define $\mathbf{a}_n$ as the resulting velocity, $\mathbf{b}$ as the initial velocity and $\mathbf{f}_n$ as the impulse. For non-frictional collisions, it can be shown that $\mathbf{A}$ is positive semi-definite (see [5], [45]).

Kinematic consistency is imposed on the problem by requiring nonnegative normal accelerations and forces,

$$\mathbf{a}_n \geq \mathbf{0} \quad \mathbf{f}_n \geq \mathbf{0}. \tag{5.6}$$

In addition we require the complementarity condition,

$$\mathbf{a}_n^T \mathbf{f}_n = 0, \tag{5.7}$$

which states that the force is zero in non-constrained configurations.

The equations 5.6 and 5.7 together define a LCP. Solutions to LCP's can be obtained in expected polynomial time if the matrix $\mathbf{A}$ is positive semi-definite (PSD) by reformulating the problem into a quadratic program (QP), see [5], [49].

The existence of solutions to the LCP is guaranteed in the case when $\mathbf{A}$ is PSD and the forces $\mathbf{b}$ lie in its column space, since the corresponding QP is convex. Uniqueness of solutions can be shown only in the restricted case when $\mathbf{A}$ is positive definite (see [18], and [11]). A simple example of this situation is a four legged chair falling onto a flat surface. The distribution of constraint forces among the four legs is non-unique (see [47]).

The addition of Coulomb forces further reduces the exact solubility of this problem. Since frictional forces occur in the tangent plane to the point of contact, we must add a tangential term $\mathbf{Q}\mathbf{f}_t$ in eqn. (5.5. We then construct an LCP,

$$\mathbf{a} = \mathbf{B}\mathbf{f} + \mathbf{b}. \tag{5.8}$$

Where **a** is the resulting accelerations, **f** are the applied forces (normal and tangential), **b** are the body forces and **B** contains the columns of both **A** and **Q**. This resulting matrix **B** is no longer PSD. The consequence of this fact is the non-convexity of the corresponding QP. The solution of non-convex QP's is well known to be NP-hard, *if* a solution even exists.

Baraff recommends using the QP algorithm of Lemke [42] with modifications to ensure termination in the non-convex case. We note that the non-convex problem expressed in QP form is a problem of constrained global optimization. This is currently an active area of research and new techniques and methods which perform better than the modified Lemke algorithm may soon exist.

5.6 Contact Sticking

In this section we consider the case in which sticking occurs during collision integration. This introduces additional non-uniqueness into the problem because static Coulomb friction does not define a direction for the resulting forces. Thus when sticking occurs during a collision and the Coulomb model indicates a resumption of sliding, it is unclear how this transition can be accomplished consistently.

In a series of papers, Bhatt and Koechling [6], [7] consider this problem in detail. It is also discussed at length in [45]. Their starting point is the impulsive version of (5.5),

$$\mathbf{v} = \mathbf{A}\mathbf{p} + \mathbf{v}_0. \tag{5.9}$$

Because they are concerned primarily with impulsive collisions, the authors express the collision dynamics in terms of the normal impulse p_z instead of the collision time t. This change of variables gives additional insight into the meaning of the coefficient

of restitution e. Since collision integration termination is given directly by e it can be thought of as defining the *flow length* in state space of the corresponding dynamical system.

In the tangent plane to the collision point, the Coulomb model can be written,

$$\frac{dP_x}{dP_z} = \frac{dF_x}{dF_z} = -\frac{\mu v_x}{\sqrt{v_x^2 + v_y^2}} \tag{5.10}$$

$$\frac{dP_y}{dP_z} = \frac{dF_y}{dF_z} = -\frac{\mu v_y}{\sqrt{v_x^2 + v_y^2}}. \tag{5.11}$$

Where μ is the coefficient of friction, v_x and v_y are the tangential velocity components.

We insert (5.10) into the derivative of (5.9) to generate a first order dynamical system in the tangent plane,

$$\frac{dv_x}{dP_z} = -a_{11}\frac{\mu v_x}{\sqrt{v_x^2 + v_y^2}} - a_{12}\frac{\mu v_y}{\sqrt{v_x^2 + v_y^2}} + a_{13} \tag{5.12}$$

$$\frac{dv_y}{dP_z} = -a_{21}\frac{\mu v_x}{\sqrt{v_x^2 + v_y^2}} - a_{22}\frac{\mu v_y}{\sqrt{v_x^2 + v_y^2}} + a_{23}. \tag{5.13}$$

By carefully considering the possible values of the parameters in (5.12), the authors show that the parameter space is a compact subset of $\Re^3$,

$$\delta = \frac{a_{23}}{a_{13}} \quad \lambda = \frac{\mu a_{11}}{a_{13}} \quad \phi = \frac{\mu a_{22}}{a_{13}}. \tag{5.14}$$

We can express the stick-slip condition in terms of the flow parameters as

$$\lambda < \frac{\phi}{\sqrt{\phi^2 - \delta^2}} \quad or \quad \delta > \phi. \tag{5.15}$$

Note that in (5.12), the flow is undefined in the case of sticking $v_x = v_y = 0$. The singularity can be moved to infinity with the singular transformation,

$$d\tau = \frac{\mu}{\sqrt{v_x^2 + v_y^2}} dP_z. \tag{5.16}$$

Finally Bhatt and Koechling transform into polar co-ordinates,

$$\frac{dv_r}{d\tau} = v_r(-\lambda \cos^2 v_\theta - \phi \sin^2 v_\theta + \cos v_\theta + \delta \sin v_\theta) \tag{5.17}$$

$$\frac{dv_\theta}{d\tau} = (\lambda - \phi)\cos v_\theta \sin v_\theta + \delta \cos v_\theta - \sin v_\theta. \tag{5.18}$$

Obviously this system has a fixed point at $v_r = 0$. Invariant flow directions are obtained from the real roots of $\frac{dv_\theta}{d\tau} = 0$.

The authors show that the conditions in (5.15) exactly correspond to the situation in which there is a unique invariant flow direction on which $\frac{dv_r}{d\tau} > 0$ (i.e. outflow). This effectively resolves the stick-slip indeterminacy problem for single contact configurations.

We note that although this approach is concerned with impulsive collision models, a similar analysis can be carried out for force response rigid models, using eqn. (5.5). In this way, the results of this section could be incorporated into a simulator using either type of collision resolution model.

5.7 Force Response Rigid Collisions with Friction

Although much of the preceding discussion applies in principle to force response rigid models, it is insightful to consider the simplification provided by assuming a particular model. In this section, we consider using the 3D harmonic penalty model to resolve a frictional collision. The algorithm developed here is presented as part of a simulation system in [26], [27].

We begin with the penalty equation in 3D,

$$\mathbf{f}_i(t) = \mathbf{K}_i(\mathbf{q}_i - \mathbf{p}_i) + \mathbf{C}_i(\dot{\mathbf{q}}_i - \dot{\mathbf{p}}_i). \tag{5.19}$$

Where $\mathbf{p}$ represents the rigid body motion, $\mathbf{q}$ is the deformed motion so that their difference can be thought of as the surface strain. Subscripts refer to the two bodies

in collision. The vectors $\mathbf{p}$ and $\mathbf{q}$ are in $\Re^3$ and we assume that the third component points in the direction normal to the contact point. The penalty parameters are given by two positive definite rank 2 tensors $\mathbf{K}$ and $\mathbf{C}$ representing spring force and viscous damping respectively. We assume that the third component of $\mathbf{f}$ is normal to the collision point.

Coulomb friction is incorporated with the quadratic form,

$$\mathbf{f}^T\mathbf{Q}\mathbf{f} \leq 0. \tag{5.20}$$

Where $\mathbf{Q} = diag(1, 1, -\mu^2)$. The Coulomb model also gives a direction to $\mathbf{f}$ in the tangent plane,

$$\dot{\mathbf{q}}_2 - \dot{\mathbf{q}}_1 = \lambda\mathbf{P}\mathbf{f}. \tag{5.21}$$

For some positive real value of λ. Note that $\mathbf{P} = diag(1, 1, 0)$.

Goyal uses a first order Taylor expansion to generate the next value of $\mathbf{q}_i$,

$$\mathbf{q}_i(t + \Delta t) = \mathbf{q}_i + \Delta t\dot{\mathbf{q}}_i(t + \Delta t). \tag{5.22}$$

Note that this formula corresponds to a backward Euler numerical integration scheme. Inserting (5.19) into (5.22) and doing a little algebra, we arrive at the familiar equation

$$(\mathbf{A} + \lambda\mathbf{P})\mathbf{f}(t + \Delta t) = \mathbf{a}. \tag{5.23}$$

Where $\mathbf{A}$ and $\mathbf{a}$ are functions of known quantities. Further, $\mathbf{A}$ is positive definite because of the positive definiteness of $\mathbf{K}$ and $\mathbf{C}$. At this point the only remaining unknowns are λ and $\mathbf{f}$.

We proceed by considering the case $\lambda = 0$ (sticking), giving the force as $\mathbf{f}^* = \mathbf{A}^{-1}\mathbf{a}$. If the Coulomb condition $\mathbf{f}^{*T}\mathbf{Q}\mathbf{f}^* \leq 0$ is satisfied, we set

$$\mathbf{f}(t + \Delta t) = \mathbf{f}^*. \tag{5.24}$$

If the Coulomb condition is not satisfied then the objects must be slipping. In this case, we can use the quadratic form (5.20) to determine λ,

$$((\mathbf{A} + \lambda\mathbf{P})^{-1}\mathbf{a})^T\mathbf{Q}((\mathbf{A} + \lambda\mathbf{P})^{-1}\mathbf{a}) = 0. \tag{5.25}$$

Which we solve to get the positive root λ^*, that gives the next force value as,

$$\mathbf{f}(t + \Delta t) = (\mathbf{A} + \lambda^*\mathbf{P})^{-1}\mathbf{a}. \tag{5.26}$$

This algorithm produces unique contact forces while incorporating the Coulomb friction model. This is achieved primarily because of the positive definite assumption on the contact parameter tensors $\mathbf{K}$ and $\mathbf{C}$ which forces the problem to be convex. As mentioned in chapter 2, the authors of this algorithm give no straightforward method for determining the values of these tensors for real bodies. Finally we note that because of the tensor nature of the contact parameters, it is possible to incorporate tangential spring and viscous forces in addition to friction.

5.8 Discussion

In this chapter, we have considered the complexity added to contact models by the addition of frictional forces. Because of the nature of the Coulomb model, both existence and uniqueness can be lost. Impulsive models must be especially careful when adding friction since conservation of energy is not guaranteed in this case. Force response rigid models, in particular the harmonic model can be used to generate closed form algorithms with the proper conditions on the contact model.

Chapter 6

Contact Map Simulations with Friction

6.1 Introduction

In this chapter, we hope to establish the physical consistency of the contact map algorithm by comparison with experimental data. We will focus primarily on a two dimensional collision event for which we have good observational data. Unfortunately, it is very difficult to measure directly the physical effects which occur during a contact event. In addition, as discussed in chapter 2, contact mechanics is affected by a large number of variables many of which are difficult to isolate and quantify. For these reasons, there is a shortage of applicable experimental data with which to compare our algorithm.

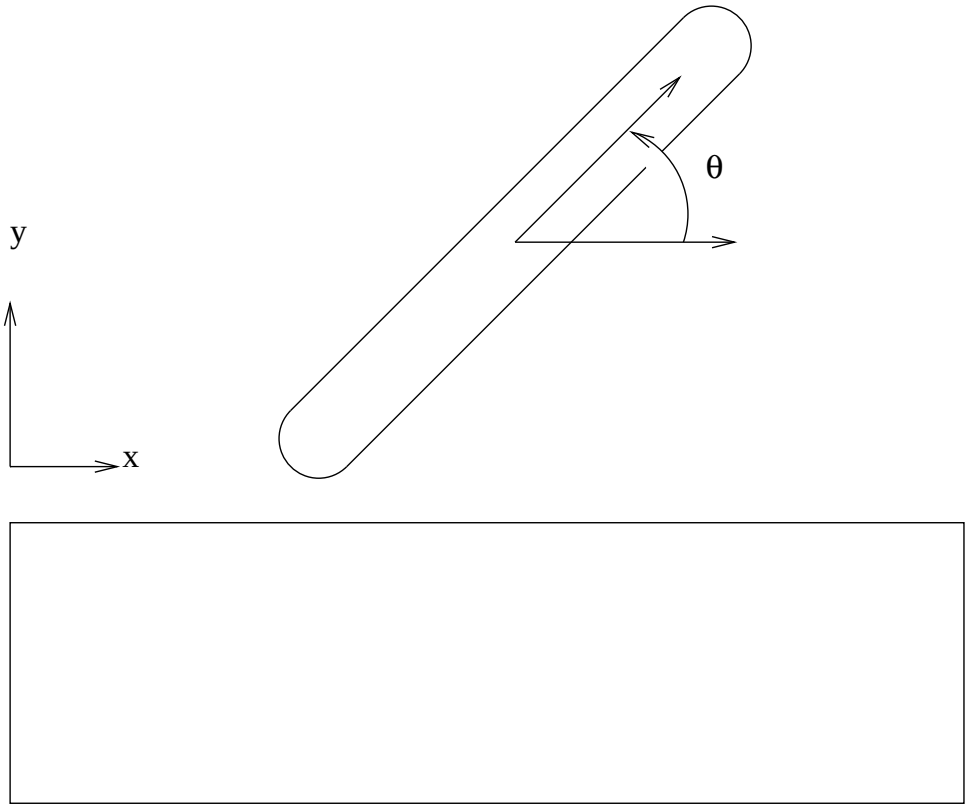


Figure 6.1: Bar experiment configuration.

6.2 Previous Work

Our primary source of data will be a paper by Stoianovici and Hurmuzlu [56]. The authors considered the impact of a rigid steel rod onto a massive foundation. The rod was dropped at various orientations and with differing impact velocities.

Data collected during the experiments consisted of pre and post impact tip velocities in the normal and tangential direction. From this data, the authors inferred the friction model using the impulse ratio μ,

$$\mu = \frac{\dot{\bar{x}}^+ - \dot{\bar{x}}^-}{\dot{\bar{y}}^+ - \dot{\bar{y}}^-}. \tag{6.1}$$

Where $\dot{\bar{x}}$ is the tangential velocity, $\dot{\bar{y}}$ is the normal velocity and superscripts indicate pre and post collision. The experimental data indicated that the impulse ratio was zero at normal impact ($90^{\cdot}$), increased linearly with decreasing angle and then became constant below some slip angle (roughly $81^{\cdot}$). The linear portion of the impulse ratio is consistent with static friction and the constant portion with kinetic friction.

In all experimental setups below the slip angle, sticking did not occur at any point during the collision. We know that in this situation, the three definitions of the coefficient of restitution are equivalent, hence the authors calculate restitution using the Newtonian definition (2.8).

The coefficient of restitution was observed to decrease with decreasing angle to a critical angle (roughly $63^{\cdot}$. for a 200mm bar). Below the critical angle the coefficient became constant at a higher value.

The authors proceeded to construct a numerical model of the experiment by decomposing the rigid bar into n rigid segments each connected by two sets of springs. The rigid foundation was modeled using a Hunt and Crossley type penalty model. We present a similar model below. Model parameters were calculated by fitting the solution curves with a single set of experimental data.

6.3 Rigid Body Simulations

Before we directly compare simulation with contact maps to the results reported in [56], it was considered important to first model the collision as faithfully as possible without including elastic effects. This was done for two reasons, first we wanted to get a good idea of how sensitive the physics was to elastic waves and secondly we wanted a check for the portions of the data in [56] which were indicated to have

minimal dependence on the presence of elastic waves.

Our model is based on the setup depicted in figure 6.1.

Our equations of motion, written in first order form are,

$$\dot{x}_1(t) = x_2(t) \tag{6.2}$$

$$\dot{x}_2(t) = \frac{1}{m} f_x(t) \tag{6.3}$$

$$\dot{y}_1(t) = y_2(t) \tag{6.4}$$

$$\dot{y}_2(t) = \frac{1}{m} f_y(t) \tag{6.5}$$

$$\dot{\theta}_1(t) = \theta_2(t) \tag{6.6}$$

$$\dot{\theta}_2(t) = \frac{L}{2I_{xx}} (f_y(t)\cos(\theta_1(t)) - f_x(t)\sin(\theta_1(t)). \tag{6.7}$$

Where m is the bar mass, I_{xx} is the moment of inertia of a uniform bar of length L rotating about it's geometric center. We also define the tip velocities as

$$\dot{x}_{tip} = \dot{x}_2 - \frac{L}{2}\sin(\theta_1)\theta_2 \tag{6.8}$$

$$\dot{y}_{tip} = \dot{y}_2 + \frac{L}{2}\cos(\theta_1)\theta_2. \tag{6.9}$$

The normal contact force is given by

$$f_y(t) = (K - B\dot{y}_{tip})y_{tip}. \tag{6.10}$$

Note that this is a Hunt and Crossley type penalty force and as such incorporates a velocity dependence. In one dimension, this dependence is expressed in [32] with relation to the kinematic coefficient of restitution as

$$e = 1 + \alpha v_i. \tag{6.11}$$

Where $\alpha = \frac{2B}{3K}$ and the initial normal velocity v_i is assumed negative.

Finally, the tangential contact force f_x is given by standard Coulomb friction with static coefficient μ_s and kinetic coefficient μ_k.

L	.2 m
K	5.5e7 N/m
B	2.2e7 N s/m
μ_s	0.1
μ_k	0.075

Table 6.1: Experimentally determined collision parameters.

6.4 Model Parameters and Observations

The parameters K, B, μ_s and μ_k were determined experimentally in the paper. Values are listed in table 6.1.

From the given values of K and B we compute α as $.24s/m$. In [32], the range of validity of their model is estimated as $.08 - .32$. We may thus feel secure in using this contact model in this situation.

We present the simulation results in figures 6.2 to 6.5.

We first observe that there appears to be negligible angular dependence for the post impact tip velocity. This is due to the local nature of the collision in this model. We expect this because there is no global coupling in the absence of elastic waves.

In this experimental setup, we define the *critical angle* as the impact configuration in which the final center of mass normal velocity remains negative. In the context of [56], this is the angle beyond which multiple bounces will be needed to resolve the collision. From figure 6.5, we can clearly see this is in the $60^{\cdot}$ range. This corresponds to the figure of $64.3^{\cdot}$ given in [56], so we conclude that this effect is mainly a function of the impact geometry and not of elastic waves.

In addition we note that as expected from the penalty model, the energy loss and normal tip restitution are functions of impact velocity. This is not what

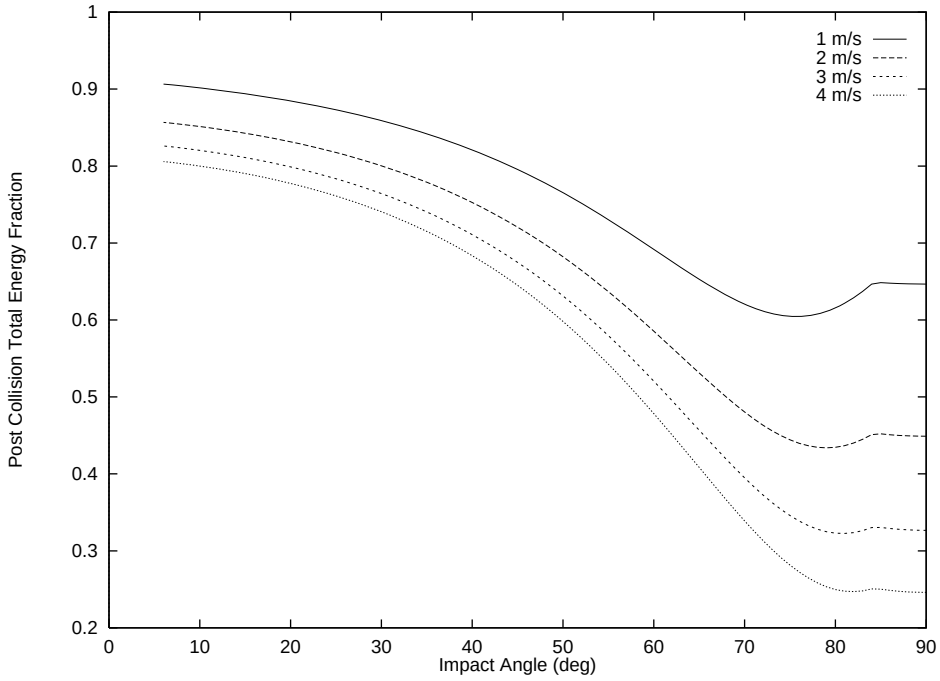


Figure 6.2: Post impact total energy.

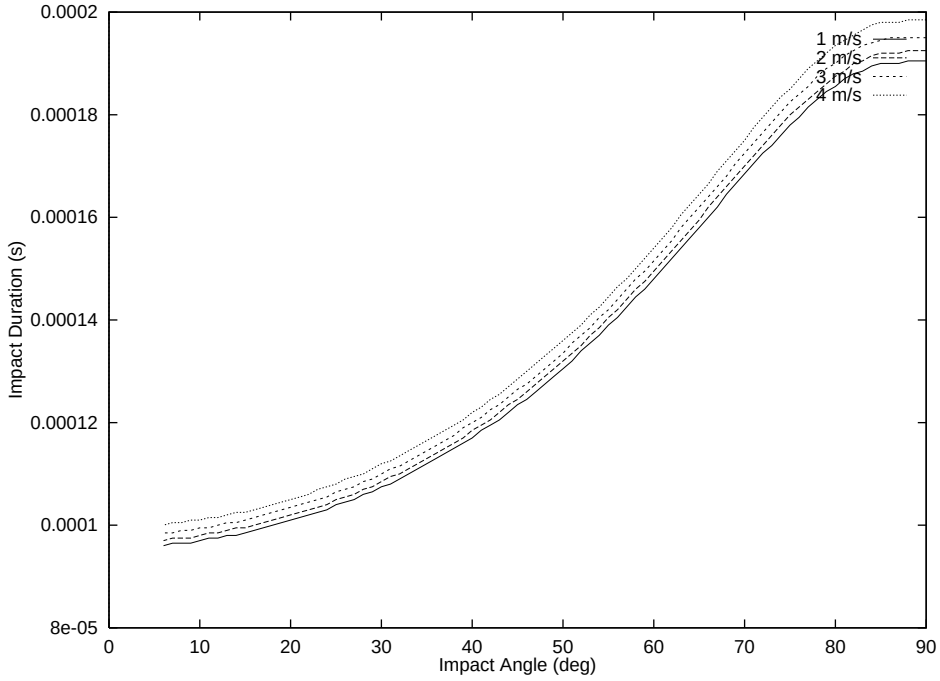


Figure 6.3: Impact duration.

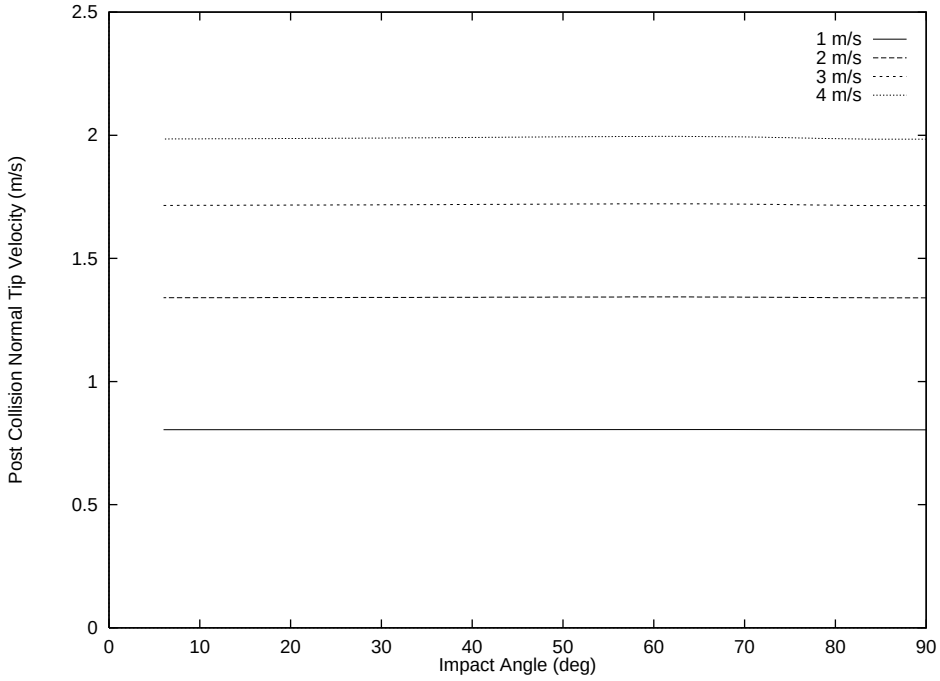


Figure 6.4: Post impact normal tip velocity.

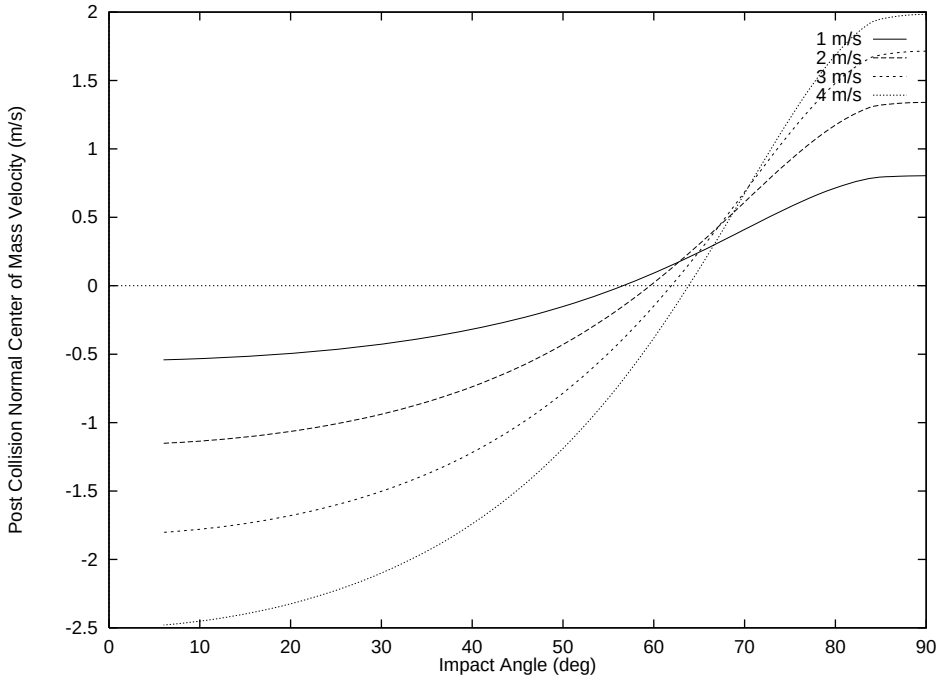


Figure 6.5: Post impact normal center of mass velocity.

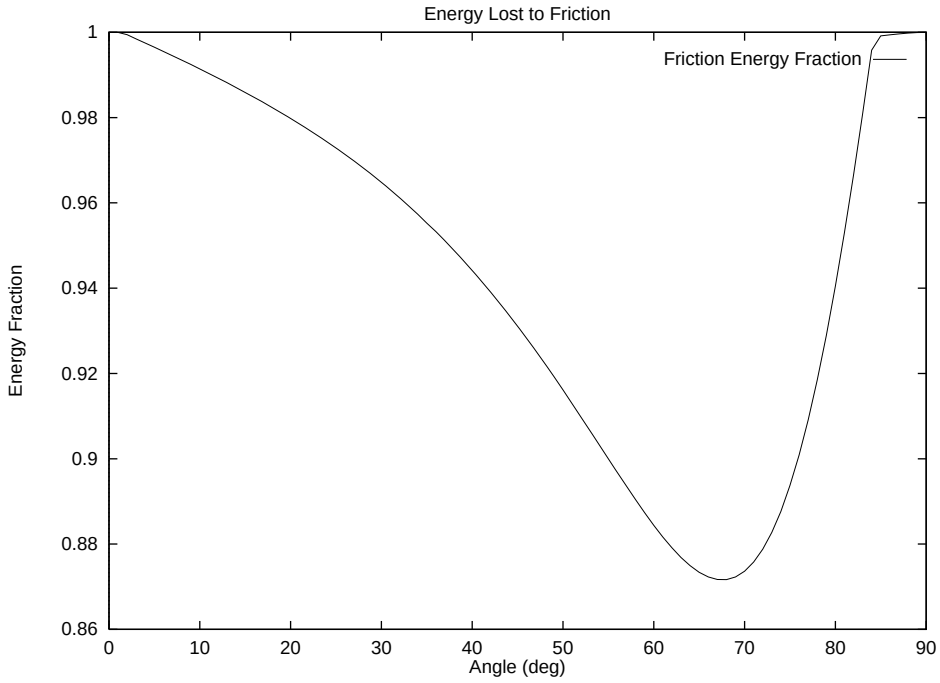


Figure 6.6: Post impact energy fraction due to friction only.

[56] state nor appear to observe experimentally. It seems unlikely that this velocity dependence is affected by the presence or absence of elastic waves.

By removing the damping from the foundation and since the only other energy loss is via friction, it is possible to use this model to estimate the energy lost solely to friction. This was done for the bar for a variety of angles, the results are plotted in figure 6.6.

The fractional energy loss to friction also exhibits strong angular dependence, with the distinct peak at the critical angle near 65˚. We discuss the importance of this type of energy loss further in the next section.

6.5 Contact Map

For contact map simulations, we implemented the algorithm discussed in chapter 3. The contact maps used for the bar were generated using the 2D BEM software discussed in [22]. For the rigid foundation, we used the correspondence between the penalty spring constant K above and the step response discussed in chapter 4. Unfortunately this approach makes it difficult to incorporate damping into the foundation so we simply used a flat step response.

We only calculated collisions at the nodal points used in the BEM calculation. This eliminated the need for collision detection and element following during contact. As a result the following graphs have data points at impact angles of 90, 78, 66, 54, 42, 30 and 18 degrees.

In figure 6.7, we present a plot of the kinetic energy during the collision as a function of time for each impact angle. The final states are summarized in figure 6.8 where we see the kinematic coefficient of restitution, presented as an energy fraction. Clearly there is a peak in the energy loss near the critical angle of $\theta_c = 66^{\cdot}$. This peak is clearly absent from figure 6.2 so we may conclude that this effect is related to the presence of elastic waves.

Impact durations are shown in 6.9. Comparing these to the curves shown in 6.3 we see very similar results. It appears that as with the linear harmonic penalty model, the impact duration is affected most strongly by the stiffness parameter and very little by the damping. This effect is also noted by [56].

The energy lost only to elastic waves was obtained by subtracting the energy lost to friction from fig. 6.6. This data is shown in fig. 6.10. We observe a monotonic decrease in the elastic energy with decreasing impact angle. Intuitively, a normal impact should retain more wave energy because the wave front and the contact

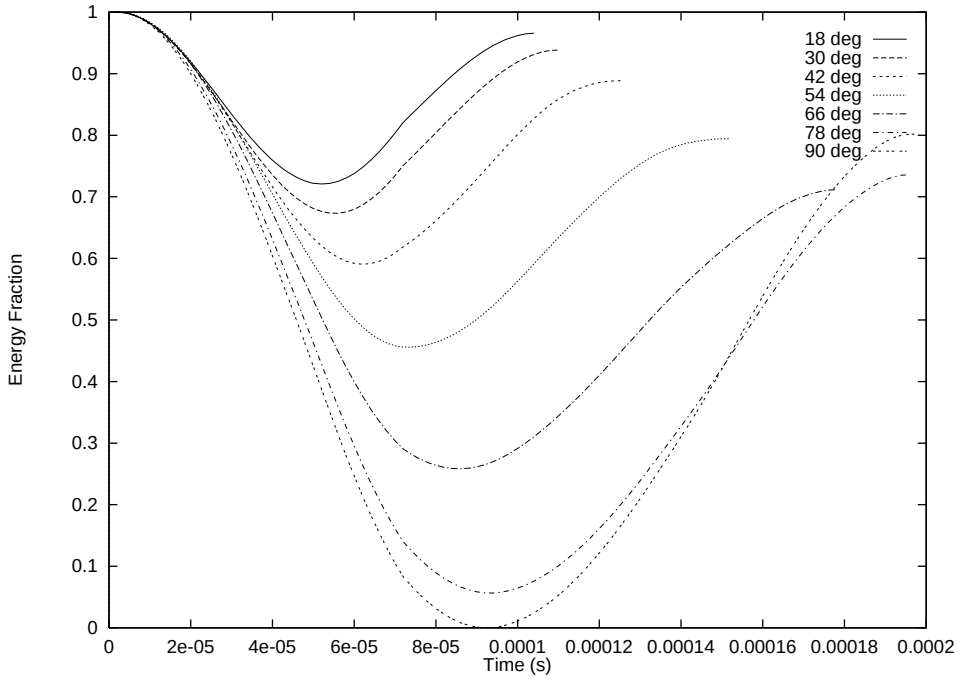


Figure 6.7: Total kinetic energy during collision of bar for various impact angles.

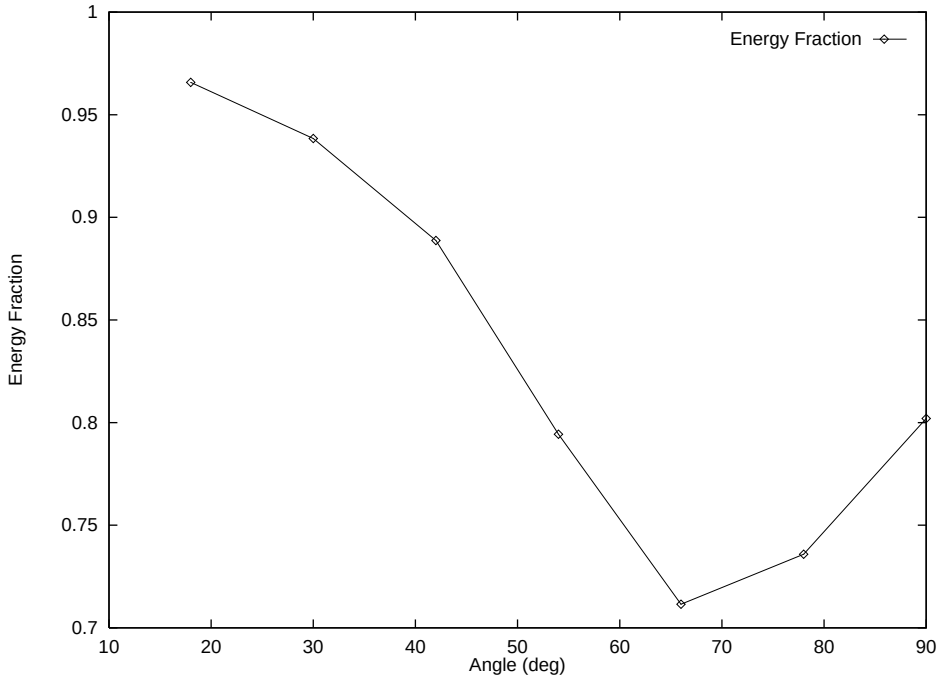


Figure 6.8: Total kinetic energy of bar for various impact angles.

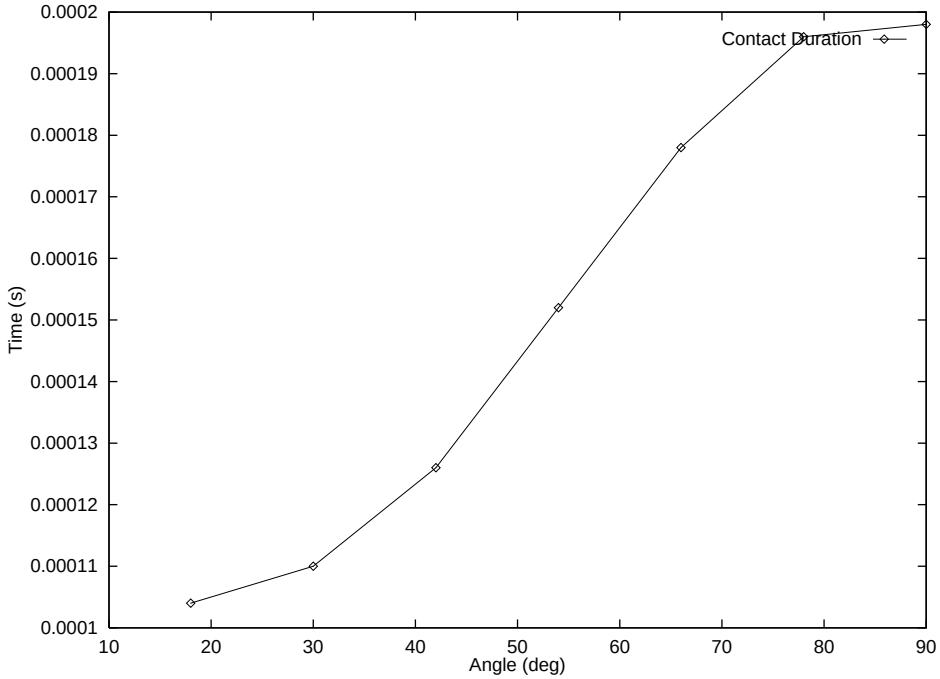


Figure 6.9: Impact duration for various impact angles.

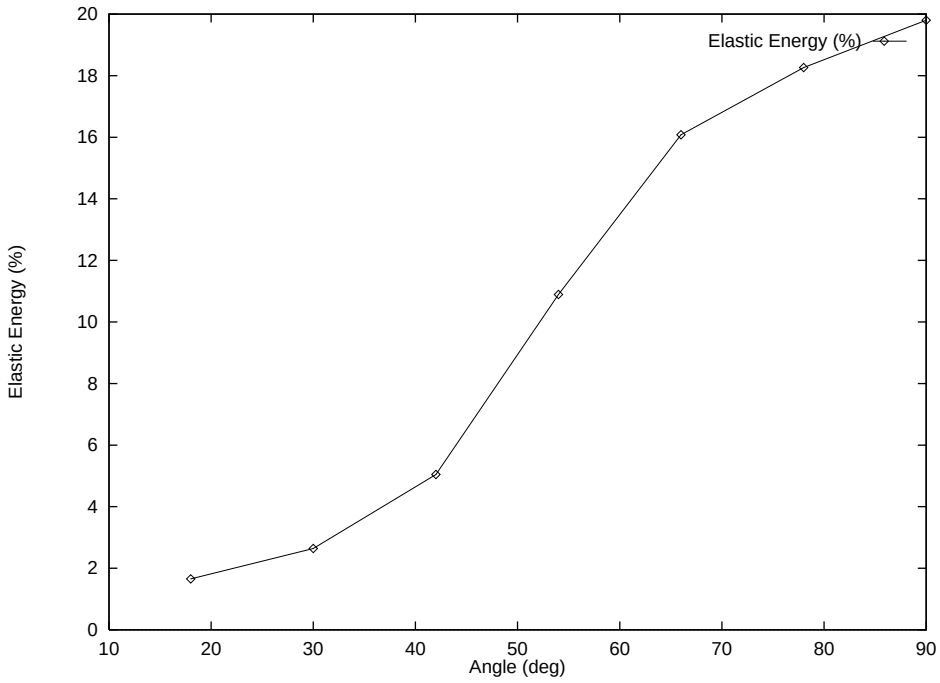


Figure 6.10: Percentage of energy lost to internal vibrations.

region are coincident less often than for a lesser angle. For small angles, the wave energy is more uniformly distributed in the bar and thus has more opportunity to affect and be affected by the contact region. In addition, the collision duration curve is very similar to the elastic wave energy loss, indicating that the total energy transfer is related to the time spent in contact.

In [56], the authors further discussed the effect of the multiple collisions which will occur for impact angles below θ_c (due to the still negative center of mass velocity). This effect requires that the bar is already vibrating on impact for the second and subsequent collisions. Unfortunately a straightforward implementation of the Green's function model requires that the initial conditions of impact be assumed zero. This makes it difficult to compare the contact map approach to the remaining data in [56].

6.6 Conclusion

In this chapter, we have shown the effectiveness of the contact map approach to the simulation of rigid bodies. The predicted energy loss due to elastic waves qualitatively matches the observations of [56] with values which are also in the same range. In addition, the data generated using the contact map approach took only a fraction of a second to compute on a Pentium microprocessor using an unoptimized implementation of the algorithm in chapter 3.

We believe that with a proper implementation, the contact map algorithm could be fast enough for use in real time simulation. This would allow more accurate simulation with a very minor penalty in computational overhead.

Chapter 7

Conclusions

In this thesis, we have reviewed the state of the art in rigid body contact models. Both local and global models were presented, but it was shown that local models suffer from a very narrow range of applicability. However global contact models present the greater difficulty of large computational complexity.

This thesis proposed a global contact model for point contacts which are of finite duration. The contact map model is this approach, which achieves accuracy by solving the wave equation subject to the actual geometry, and achieves speed by caching the solutions. Comparison to a similar model and experimental data showed that this approach behaves correctly, although it predicts different fractions for energy lost to elastic waves.

In addition, we considered the problems of existence and uniqueness presented by the inclusion of friction in any contact model. This is still an active area of research.

The contact map algorithm presented in this paper can already be used in simple applications requiring only normal contact. For example in the design of

industrial part feeding machines, simulations requiring hundreds or thousands of contacts per second are necessary. While this algorithm may not give the minimum 30 frames/sec. it will certainly operate at interactive speeds. Other examples include collisions of objects which are decomposable into spherical sub-objects. In this case simple point friction could be used as in chapter 6.

We note finally that a full implementation of contact response maps for three dimensions, including friction would be of great value both to researchers and to applied engineering industries. Extremely rapid and accurate contact calculations along with fast collision detection would allow system designers to use automatic optimization tools and have significant feedback starting from the earliest design steps. The result of all this would be improved efficiency and reliability in many CAD/CAE designed machines.

Bibliography

[1] Brebbia C. A. and Dominguez J. *Boundary elements : an introductory course.* Computational Mechanics, New York, 1989.

[2] H. Antes and P. D. Panagiotopoulos. *The Boundary Integral Approach to Static and Dynamic Contact Problems.* Birkhauser, Basel - Boston - Berlin., 1992.

[3] U. Ascher, D. K. Pai, and B. Cloutier. Forward dynamics, elimination methods, and formulation stiffness in robot simulation. International Journal of Robotics Research, 16:6, December 1997.

[4] Owe Axelson. *Iterative Solution Methods.* Oxford University Press, Oxford UK, 1994.

[5] D. Baraff. *Dynamic Simulation of Non-Penetrating Rigid Bodies.* PhD thesis, Cornell University, 1992.

[6] V. Bhatt and J. Koechling. Partitioning the parameter space according to different behaviors during 3d impacts. *ASME Journal of Applied Mechanics*, 62:740–746, 1995.

[7] V. Bhatt and J. Koechling. Three-dimensional frictional rigid-body impact. *ASME Journal of Applied Mechanics*, 62:893–898, 1995.

[8] R. M. Brach. Rigid body collisions. *ASME Journal of Appied Mechanics*, 56:133–138, 1989.

[9] G. Burdea. *Force and touch feedback for virtual reality.* John Wiley, New York, 1996.

[10] R. L. Burden and J. D. Faires. *Numerical Analysis.* PWS, Boston, 1993.

[11] Ferris M. C. and Pang J. S. Engineering and economic applications of complementarity problems. *SIAM Review*, 39(4):669–714, 1997.

[12] CADSI. Dads - dynamic analysis design system. Technical report, 1998. http://www.cadsi.com/dads.html.

[13] S. Cameron. Enhancing gjk: Computing minimum and penetration distances between convex polyhedra. In *IEEE International Conference on Robotics and Automation*, pages 3112–3117, 1997.

[14] A. Chatterjee. *Rigid Body Collisions: Some General Considerations, New Collision Laws, and Some Experimental Data.* PhD thesis, Cornell University, 1997.

[15] R. M. Chritensen. *Theory of Viscoelasticity An Introduction.* Academic Press, New York, 1971.

[16] VRML Consortium. The virtual reality modeling language. Technical Report 14772-1:1997, ISO/IEC, 1997. http://www.vrml.org/Specifications/VRML97/.

[17] Sense8 Corp. World toolkit. Technical report, Engineering Automation Inc., 1998. http://www.sense8.com.

[18] R. W. Cottle, J. S. Pang, and R. E. Stone. *The Linear Complementarity Problem.* Academic Press, Boston, 1992.

[19] J. F. Cremer. *An Architecture for General Purpose Physical System Simulation - Integrating Geometry, Dynamics and Control.* PhD thesis, Cornell University, 1989.

[20] Lewis A. D. and Rogers R. J. Experimental and numerical study of forces during oblique impact. *Journal of Sound and Vibration*, 125:403–412, 1988.

[21] D. Delchamps. *State Space and Input-Output Linear Systems.* Springer-Verlag, New York, 1988.

[22] J. Dominguez. *Boundary Elements in Dynamics.* Elsevier Science, Essex, 1993.

[23] P. DuChateau and Zachmann D. *Theory and Problems of Partial Differential Equations.* McGraw Hill, New York, 1986.

[24] D. Flanagan. *Java in a Nutshell.* O'Reilly, Sebastapol, 1997.

[25] W. Goldsmith. *Impact: The theory and physical behavior of colliding solids.* Edward Arnold, London, 1960.

[26] S. Goyal, E. Pinson, and F. Sinden. Simulation of dynamics of interacting rigid bodies including friction i: General problem and contact model. *Engineering with Computers*, 10:162–174, 1994.

[27] S. Goyal, E. Pinson, and F. Sinden. Simulation of dynamics of interacting rigid bodies including friction ii: Software system design and implementation. *Engineering with Computers*, 10:175–195, 1994.

[28] S. Goyal, A. Ruina, and J. Papadopoulos. Planar sliding with dry friction part 1. limit surface and moment function. *Wear*, 143:307–330, 1991.

[29] S. Goyal, A. Ruina, and J. Papadopoulos. Planar sliding with dry friction part 2. dynamics of motion. *Wear*, 143:331–352, 1991.

[30] J. A. Greenwood. Problems with surface roughness. In *Fundamentals of Friction: Macroscopic and Microscopic Processes*, pages 57–76, 1991.

[31] W. Hackbush. *Integral Equations: theory and numerical treatment.* Birkhauser Verlag, Basel, 1995.

[32] K. H. Hunt and Crossley F.R.E. Coefficient of restitution interpreted as damping in vibroimpact. *ASME Journal of Applied Mechanics*, 42:440–445, 1975.

[33] Mechanical Dynamics Inc. Adams simulation package. Technical report, 1998. http://www.adams.com.

[34] Canny J. Collision detection between moving polyhedra. *IEEE Trans. PAMI*, 8:200–209, 1986.

[35] Canny J. 3d direct interfaces: An interactive pipeline. Technical report, The University of California, Berkeley, 1997. http://www.cs.berkeley.edu/ healey/MURI/.

[36] Routh E. J. *Dynamics of a System of Rigid Bodies.* Dover, New York, 1960.

[37] K. L. Johnson. *Contact Mechanics.* Cambridge University Press, New York, 1985.

[38] T. Kailath. *Linear Systems.* Prentice-Hall, New Jersey, 1980.

[39] H. Keller, H. Stolz, and T. Brunl. Aero - a physically based simulation and animation system. Technical report, The University of Western Australia, 1997. http://www.ee.uwa.edu.au/ braunl/aero/.

[40] J. B. Keller. Impact with friction. *ASME Journal of Applied Mechanics*, 53:1–4, 1986.

[41] N. Kikuchi and J. T. Oden. *Contact Problems in Elasticity: A Study of Variational Inequalities and Finite Element Methods.* SIAM, Philadelphia, 1988.

[42] C. E. Lemke. Bimatrix equilibrium points and mathematical programming. *Management Science*, 11:681–689, 1965.

[43] M. C. Lin. *Efficient collision detection for animation and robotics.* PhD thesis, UC Berkeley, 1993.

[44] Ch. Lubich, Nowak U., Pohle U., and Engstler Ch. Mexx - numerical software for the integration of constrained mechanical multibody systems. *ZIB Preprint SC92-12*, 1992.

[45] B. Mirtich. *Impulse-based Dynamic Simulation of Rigid Body Systems.* PhD thesis, UC Berkeley, 1996.

[46] R. L. Mullen and J. J. Rencis. Iterative methods for solving boundary element equations. *Computers and Structures*, 25(5):713–723, 1988.

[47] Ivanov A. P. On multiple impact. *J. Appl. Maths. Mechs.*, 59(6):887–902, 1995.

[48] D. Pai. ???acme testbed??? *NSERC Research Grant Proposal*, 1997.

[49] F Pfeiffer and C Glocker. *Multibody Dynamics with Unilateral Contacts.* Wiley, New York, 1996.

[50] H. M. Pollak. Surface forces and adhesion. In *Fundamentals of Friction: Macroscopic and Microscopic Processes*, pages 77–94, 1991.

[51] W. Press, Teukolsky S., and Vetterling W. *Numerical Recipies in C.* Cambridge University Press, New York, 1993.

[52] J. G. Reid. *Linear System Fundamentals.* McGraw Hill, New York, 1983.

[53] A. Signorini. Sopra akune questioni di elastostatica. *Atti della Societa Italiana per is Progresso delle Scienze*, 1933.

[54] I. L. Singer and H. M. Pollock. Epilogue. In *Fundamentals of Friction: Macroscopic and Microscopic Processes*, pages 569–588, 1991.

[55] H Sowizral, K Rushforth, and M Deering. Java 3d api specification. Technical report, Sun Microsystems, 1998. http://java.sun.com.

[56] D. Stoianovici and Y. Hurmuzlu. A critical study of the applicability of rigid-body collision theory. *ASME Journal of Applied Mechanics*, 63:307–316, 1996.

[57] W. J. Stronge. Rigid body collisions with friction. *Proc. R. Soc. Lond. A*, 431:169–181, 1990.

[58] W. J. Stronge. Unraveling paradoxical theories for rigid body collisions. *ASME Journal of Applied Mechanics*, 48:1049–1055, 1991.

[59] W. J. Stronge. Planar impact of rough compliant bodies. *Int. J. Impact Engng*, 15(4):435–450, 1994.

[60] W. J. Stronge. Theoretical coefficient of restitution forl planar impact of rough elasto-plastic bodies. *Impact, Waves, and Fracture*, 205:351–362, 1995.

[61] D. Tabor. Friction as a dissipative process. In *Fundamentals of Friction: Macroscopic and Microscopic Processes*, pages 3–24, 1991.

[62] S. Takahashi and Brebbia C. A. Elastic contact analysis with friction using boundary elements flexibility approach. In M. H. Aliabadi and Brebbia C. A., editors, *Computational Methods in Contact Mechanics*. Elsevier Science, Essex, 1993.

[63] S. P. Timoshenko and Goodier J. N. *Theory of Elasticity*. McGraw-Hill, New York, 1970.

[64] C. Ullrich and D. K. Pai. Contact response maps for real time dynamic simulation. In *IEEE International Conference on Robotics and Automation*, 1998.

[65] Y. Wang and M. Mason. Modeling impact dynamics for robotic operations. In *IEEE International Conference on Robotics and Automation*, pages 678–685, 1987.

[66] J Wernecke. *The Inventor Mentor*. Addison-Wesley, 1995.

Appendix A

Boundary Element Review

This appendix discusses the boundary element method and it's application to the solution of partial differential equations. For simplicity, we use the Laplace equation $\Delta u = 0$ on some fixed domain Ω as the reference example for the technique.

A.1 Boundary Elements and Finite Elements

Before we discuss the details of the boundary element method (BEM), we compare some of it's properties with the popular finite element method (FEM) solution technique.

Both BEM and FEM are integral equation solution procedures for partial differential equations (PDE's). Both techniques rely on Green's integral theorem and both produce *weak* solutions.

The FEM method solves a PDE by discretizing the domain over which the solution is defined whereas the BEM approach discretizes only the boundary of the domain. In this sense, the BEM technique always has a lower dimensionality than the FEM procedure for the same problem. In the solution of an FEM problem, a

great deal of symmetry is always present, and all equations are of a local nature. The BEM solution procedure presents no special symmetry and involves global equations.

A.2 Boundary Integral Equations

A.2.1 Fundamental Solutions

To construct a BEM solution for a given PDE, we first find the so called fundamental solution (or Green's function). For the sample equation, $\nabla u = 0$, the corresponding fundamental equation is,

$$\Delta w(\mathbf{x}) - \delta(\mathbf{x}^i - \mathbf{x}) = 0. \tag{A.1}$$

For a given fixed point $\mathbf{x}^i$. In this case, the fundamental solutions are just the harmonic functions,

$$w(\mathbf{x}) = \frac{1}{2\pi} log(r) \quad in \Re^2 \tag{A.2}$$

$$w(\mathbf{x}) = \frac{1}{4\pi r} \quad in \Re^3. \tag{A.3}$$

Where $r = |\mathbf{x}^i - \mathbf{x}|$.

A.2.2 Residual

Now consider the weighted residual, $w\Delta u = 0$ over the problem domain Ω,

$$\int_\Omega w \Delta u d\Omega = 0. \tag{A.4}$$

Given an approximate solution $u = u'$, (A.4) gives a measure of the approximation error. The choice of w as a weight allows us to make direct use of Green's integral theorem,

$$\int_\Omega w\Delta u + \Delta w \cdot \nabla u d\Omega = \int_\Gamma w \frac{\partial u}{\partial n} d\Gamma. \tag{A.5}$$

Substituting (A.4) into (A.5) gives,

$$0 = \int_\Gamma w \frac{\partial u}{\partial n} d\Gamma - \int_\Omega \nabla w \cdot \nabla u d\Omega. \tag{A.6}$$

And applying Green's integral theorem again to the second integral leads to,

$$0 = \int_\Gamma w \frac{\partial u}{\partial n} - u \frac{\partial w}{\partial n} d\Gamma + \int_\Omega u \Delta w d\Omega. \tag{A.7}$$

Now by (A.1) for points in the interior of Ω, this reduces to,

$$u = \int_\Gamma w \frac{\partial u}{\partial n} - u \frac{\partial w}{\partial n} d\Gamma. \tag{A.8}$$

Which contains only boundary integrals. For points on the boundary, slightly more analysis is required (refer to [1] for details). The net result is

$$c(\mathbf{x}) u(\mathbf{x}) = \int_\Gamma w \frac{\partial u}{\partial n} - u \frac{\partial w}{\partial n} d\Gamma, \tag{A.9}$$

where in $\Re^2$,

$$c(\mathbf{x}) = \begin{cases} 1 & \text{for } \mathbf{x} \in \Omega \\ \frac{1}{2} & \text{for } \mathbf{x} \in \Gamma \\ \frac{\alpha}{2\pi} & \text{for corner of angle } \alpha. \end{cases} \tag{A.10}$$

Equation (A.9) represents the solution to the original Laplace equation in terms of integrals over the boundary of the region Ω. At this point, given a set of boundary conditions the problem can in theory be solved by integration.

A.3 Numerical Considerations

In practice, the integrals in (A.9) are not soluble by any analytic method. This occurs for even moderately complex boundary geometries or for non-trivial boundary conditions. Also, in engineering, most of the boundary information will be available only numerically. Thus in this section, we summarize the numerical solution of the integral equation (A.9).

A.3.1 Boundary Elements

We begin by dividing the boundary Γ into disjoint subsets Γ_i, where,

$$\Gamma = \bigcup_{j=1}^{N} \Gamma_j. \tag{A.11}$$

This reduces (A.9) to,

$$c(\mathbf{x})u(\mathbf{x}) = \sum_{j=1}^{N} \int_{\Gamma_j} w\frac{\partial u}{\partial n} - u\frac{\partial w}{\partial n} d\Gamma. \tag{A.12}$$

Now we choose a basis set ϕ_i (eg. the standard linear elements of FEM) such that,

$$u_j = \sum_i \phi_i u_{ji} \tag{A.13}$$

$$q_j = \frac{\partial u_j}{\partial n} = \sum_i \phi_i q_{ji}. \tag{A.14}$$

The choice of basis functions is an arbitrary component of this method. Typically the same bases are used for the geometry, u_j and q_j. However this choice can be optimized for specific problems (see [2] or [1]).

With a basis, the BIE becomes,

$$c(\mathbf{x})u(\mathbf{x}) = \sum_{j=1}^{N} \sum_i q_{ji} \int_{\Gamma_j} \phi_i w d\Gamma - \sum_{j=1}^{N} \sum_i u_{ji} \int_{\Gamma_j} \phi_i \frac{\partial w}{\partial n} d\Gamma. \tag{A.15}$$

There are several ways to generate a set of linear equations from (A.15), but we discuss only the collocation method. For a detailed discussion of the other approaches, see [31].

In the collocation method, we choose a set of L boundary nodes at which we require (A.15) to be satisfied exactly. This gives,

$$c_l u_l + \sum_{j=1}^{N} \sum_i u_{ji} \int_{\Gamma_j} \phi_i \frac{\partial w_l}{\partial n} d\Gamma = \sum_{j=1}^{N} \sum_i q_{ji} \int_{\Gamma_j} \phi_i w_l d\Gamma. \tag{A.16}$$

Where the subscript l indicates evaluation at $\mathbf{x}_l$. The result of this is a system of L linear equations,

$$\mathbf{Au} = \mathbf{Bq}. \tag{A.17}$$

Given a well posed problem, the system (A.17) can be written in standard form as $\mathbf{Ax} = \mathbf{b}$. Where $\mathbf{A}$ is a rank L *dense* matrix with no special symmetry.

The primary difficulty in arriving at the matrix A is the quadrature over the basis functions ϕ_i. Since these integrals involve the fundamental solution, they are nearly always singular and require special numerical treatment. See [1] and [10] for a detailed discussion of this problem.

A.4 Discussion

The boundary element method is a useful and efficient way to solve complex PDE problems by reducing them to boundary integrals. The BEM approach is particularly well suited to problems in which the boundary is the primary location of interest (ie. contact problems) and when the domain Ω is large compared with the boundary. In other situations, it is often easier to use FEM since a great deal of good software already exists. We note that both integral techniques are specific to a particular PDE and must be re-derived for each new differential operator.

For the solution of time dependent problems such as the Navier equations, or the wave equation more theory must be constructed. However the basic approach is the same as presented here. Details of the mathematics can be found in [2] and of the numerics in [22].

Appendix B

Simulation Framework

B.1 Introduction

In this appendix, we outline the design for an interactive simulation framework. The framework is known as the *IS Simulator* and represents the combined effort of a small research group over an 18 month period. The IS Simulator is a software framework in which interactive physical simulations can be constructed and experienced. The motivations of this design are platform independence, extensibility and physical accuracy. One of the primary goals of this simulation environment is to allow *plug and play* experimentation for new algorithms, such as the contact maps discussed previously.

This simulation framework differs significantly from previous general purpose simulation engines like *Newton* [19] and *Impulse* [45] in that most user interaction does not require the simulator to be stopped. By constructing the simulator with multiple execution threads, it is straightforward to implement this feature. In this way, user interaction becomes another component of the simulation which in turn

permits a much wider class of applications.

B.2 Design Goals

Before the simulator project could begin, a number of goals were selected to drive the design. These goals provide the design constraints to which all other aspects of the simulator must conform to. The goals can be summarized as:

- extensibility
- powerful numerical engine/core
- responsive/real time performance
- ease of use

Extensibility implies the ability to easily construct and add extensions to the basic functionality. Many previous simulators provide no explicit method for improving or extending them. This goal is accomplished by choosing an object oriented design paradigm and implementing it with an object paradigm, in our case Java BEANS [24].

The numerics of the simulator are critical for both accuracy of results and user interactivity. This design goal was accommodated by the MEXX numerical integration package. MEXX provides robust integration of non-stiff DAE (differential algebraic equations) systems.

The two main bottlenecks in an interactive simulation are the numerics and collision detection. Most naive collision detection algorithms are computationally very expensive. As this is still a very active area of research, we chose to encapsulate this functionality to allow for upgraded algorithms when they become available.

Previous simulation environments often suffered from difficult or obfuscated interfaces. In this simulator, the user is presented with a number of mouse driven views of the simulation in progress. These views facilitate interaction, design and control of the virtual space.

B.3 Design Overview

We begin by defining an overview of the functional levels of simulator framework. Then we discuss the key components and how they interact.

The simulator system is broken down into six levels:

1. Simulator
2. Experimental Design
3. Universe Design
4. Model Editing
5. Model Design
6. Numerical Editor

We briefly outline each of these levels below, followed by a detailed discussion of the objects that comprise them.

B.3.1 Simulator

The top level component of the IS simulator is the simulator itself. The simulator provides the user(s) with a real time view of the current simulation state, including audio, haptic and any other output mechanisms. In addition, the interface allows

the user to directly manipulate the simulation while it is running, using methods defined by the experiment design module (see below). The display should ideally be a stand-alone client which can be run from within an internet browser such as $Netscape^{TM}$.

From a design perspective, the simulator encapsulates the majority of the functions associated with traditional virtual reality applications. The simulator itself is simply a framework in which a number of components interact. In the most basic situation this will include only a numerical engine and a display. These two elements interact in a variety of ways, but primarily through a third entity called a state registry which acts as a kind of central clearing house for simulation data.

B.3.2 Experiment Design

The initial physical setup of the simulated reality is done with the tools of the experiment design level. `Things` can be added to a scene, and their initial values defined.

The experiment designer will be presented with palettes of physical components. These palettes consist of collections of constructed `things`, such as metal spheres, robot arm assemblies, microphones, or cameras. Amalgamation of `things` (i.e., building a robot arm from parts) is not done at this level but rather in the model editing level.

It should be possible for the experiment designer to co-exist with the experiment that is currently running. In this case, it would be possible for a designer to add or remove `things` from the simulation while it runs, or to manipulate object properties. Systematic design manipulations (i.e., design macros) are to be supported with a scripting mechanism.

B.3.3 Universe Design

In any simulation, there are various constants, such as the gravitational constant, that are unlikely to change during the run. The universe design level will be used for setting these properties, which we refer to as the laws of physics.

Properties set in this level cannot be changed while the simulator is executing. Although there are certainly times and places where such restriction is important, we achieve a similar result by adding a `thing` property type which is immutable during simulation execution.

B.3.4 Model Editing

The fundamental components of simulations are `things`, which are created and manipulated using a collection of model designers that form the model editing level. An incomplete list of `thing` properties that can be edited includes:

- meta-`thing` generator
- geometry
- acoustic properties
- haptic properties
- contact force models
- control properties

There are two fundamentally different states that each of these editors will operate in. First they will be used to create new properties for `things` (including other `things`). In this mode, the user generates a `thing` which is saved to the

appropriate `thing` palette for later use. The second mode of operation is to manipulate the properties of a `thing` which has been instantiated in a simulation. In this mode, the editors are *hot* in that they are part of the simulation loop and can change `things` that are being simulated.

For example, a user could create a robot arm assembly, using the various editors to make a fully functional robot arm which is then saved to the robot-arm palette for later use. Once this arm has been incorporated into a simulation, the control editor could be instantiated as part of the robot arm and the control algorithm could then be tuned, etc. to determine it's behaviour.

In addition to the standard editors, the `thing` class will specialize to external data sources, and for this purpose utilities such as parameter estimation and live data filtering are also included in this level.

B.3.5 Model Design

Fundamentally, all the properties of a `thing` must reduce to a set of equations (i.e. a system of DAE's) and algorithms, to be integrated numerically. The model design level takes care of this detailed work. Various interfaces to the equations should be provided, such as a bond graph editor, block diagram (such as Simulink) editor, etc. Digital filters and control algorithms should also be laid out using the editors of this level.

Given a rich set of such equations, it is unlikely the average user would have much use for the editors at this level. However, many users have needs that cannot be foreseen or may want to extend the simulator to novel situations, such as fluid dynamics, or novel types of materials, etc.

B.3.6 Numerical Editor

As stated in the last section, underneath it all is a (possibly large) set of equations. The equations are accessed though the numerical editor. The intent is to provide equation classes that may be specialized for particular problems. Various classes of DAE's (differential algebraic equations), PDE's (partial differential equations) and ODE's (ordinary differential equations) will be represented as well as more esoteric objects, such as DDE's (delay differential equations). The interface to the user at this level will be through the Java language.

B.3.7 Summary

The six layers of the simulator are much like an onion, they form a nested hierarchy. It is possible to use the simulator at any level, but each level requires knowledge of the levels above it. Casual users will most likely restrict their interaction to the simulator and perhaps the experiment design level. More regular users will use all the levels down to the model editors whereas the serious user and researcher will use all levels.

B.4 Object Components

The simulator levels are realized in the following objects. Functional relationships among these objects are discussed in the next section.

B.4.1 Display

The display object encapsulates all the input/output streams for the user. Each display will run as a separate thread, perhaps remotely and render state informa-

tion received from the state registry. The display uses information from the `thing` database to generate appropriate output, such as a 3D display, haptic force, etc. Inputs to the display object are set by default for a standard 'user' `thing`, but displays may be constructed for other `things`, such as scene components. Outputs from the display include pick events, mouse motion, speech, etc. The mapping of these values into state variables again will have defaults for the user `thing`, but are mutable.

The concept of the current time is local to each display object, though multiple displays may be configured to remain consistent. The display time is separate from the current state time (discussed below).

B.4.2 State Registry

Underlying any simulation are a set of equations which relate variables to one another. While a simulation is running, the variables evolve according to these equations but at a given fixed moment in time, the numeric value of all the variables is collectively known as the state. The purpose of the state registry is to maintain a record of the system state in time.

In our simulator, we extend the state concept slightly to include the user state, which could consist of such things as mouse location, velocity, pick location, etc. and also arbitrary values obtained from external sources of state.

The state registry is a passive object which accepts state variable updates, and generates state variable lists in response to queries. The state registry also keeps track of the state time, which is defined as the greatest time at which all state variables are known.

The registry may be queried to provide all or some defined subset of the

current state. In addition, trajectories are available for all state variables, stored to a pre-defined accuracy. The registry may also do internal curve fitting or state history compression in an arbitrary manner within the accuracy constraints. Future states are added for all or some of the variables from a select set of classes. Note that future states are not limited to the next time step, and may be arbitrarily ahead of the state time.

B.4.3 Relationship Managers

Relationship managers are auxiliary state registry entities which store second order state information. One example of this is inter-object distance which is derived from the current state. Relationship managers may generate state information or serve as repositories for other higher order objects. Relationship managers encapsulate algorithms that operate on the state but are not equations or external inputs.

At this point, the primary relationship manager is the collision manager. This object will monitor the state registry along with the geometries defined in the `thing` database. Using this information, it calculates inter-object minimum translation distances (MTD) and generates collision reports to the integration engine.

The advantage of a collision manager as a separate object is that it encapsulates collision detection functionality. Collision detection can be divided into two steps, wide mode (determining which objects to check) and narrow mode (checking distances of a list of object pairs). There are very efficient algorithms for both modes [34], [43], [13]. However a clearly optimal approach has yet to be demonstrated. For this reason the current simulator design only defines an interface to the collision manager.

B.4.4 Engine

The engine is a numerical solver which collects together all the equations governing the simulation variables from the `thing` database and the managers, and generates the next time ordered set of states. The simulator will contain a DAE solver by default, which should be sufficient for most users requirements.

The current prototype simulator uses the DAE solver MEXX [44] to perform numerical integrations. This solver integrates equations which can be formulated as,

$$\dot{p} = T(T,p)v \tag{B.1}$$

$$M(t,p)\dot{v} = f(t,p,v,\lambda,u) - G(t,p)^T\lambda \tag{B.2}$$

$$0 = G(t,p)\cdot v + g^I(t,p) \tag{B.3}$$

$$\dot{u} = d(t,p,v,\lambda,u). \tag{B.4}$$

$$\tag{B.5}$$

A great many dynamical mechanical systems can be formulated in this way. In the solution of such systems, matrix equations of the form,

$$\begin{pmatrix} M & G^T \\ G & 0 \end{pmatrix} \begin{pmatrix} w \\ \lambda \end{pmatrix} = \begin{pmatrix} f \\ g \end{pmatrix} \tag{B.6}$$

must be solved at each time step. By using recursive elimination algorithms [44], [3] the computational work to solve such systems grows only linearly with the number of bodies being simulated. One drawback of the MEXX integrator is it's inability to deal with stiff equations, for example when there is significant friction or collision damping. This is a result of the extrapolation time stepping technique used.

In practice, the engine should run far ahead of the users, predicting the states for a significant future time. If the users do nothing, this simply continues,

but when users perform some action or there is an unforeseen event, the engine must reset to that event and continue. There is deliberate latitude for tricks to optimize the engine, for example speculative execution using multiple solvers, or partial re-integration.

B.4.5 Thing Database

The `thing` database is the collection of entities in the simulation. This database is similar to a scene graph, as found in Java3D [55], OpenInventor [66], VRML [16], etc. In those systems, the database consists of objects and their properties in a hierarchy. In the IS system however, the database will be a collection of `things`, which contain their own properties.

The `thing` database members are constructed using the experiment design editor and the root node is constructed using the universe design editor.

B.4.6 Thing

A `thing` is a simulation entity. Examples of `things` are spheres, robot arms and user interfaces. `Things` may be hierarchical in that they may contain `things` as children. All `things` have a set of properties or parameters which are determined by using a property editor, such as the geometry editor. User related properties might consist of focal length, hearing sensitivity and handedness.

`Things` contain methods which the simulation engine uses to advance the state. For example, a robot arm would contain a control law dictating how it behaves and how it would respond to an external force.

`Things` are passive in that they do not directly generate events or calls to other threads. `Things` may contain clocks or refer to the simulation or user clock

however. In addition, `things` should be implemented using the bean component model and may contain arbitrary code segments (even running in their own threads).

B.4.7 Clocks

Each separate execution thread must contain a clock object to drive it. The executable code in a clock can be the display update, a simulation engine, an editor, etc. Clock threads usually contain some notion of time, such as user time, state time or other.

B.4.8 Data Streams

Data streams are sources or sinks of state modifying information. Data streams may have arbitrary filters and may be connected to external sources, even over the internet. They form the socket abstraction within the simulation environment.

B.5 Object Interaction

Figure B.1 shows the operational relationship among the various components.

B.6 IS Simulator Implementation

The implementation of this simulator is currently in the prototype phase. We seek to characterize the key problems with the realization of this design.

By virtue of it's java implementation, the simulator runs on a number of platforms including IRIX, Linux, HP-UX and Solaris. On the Sun UltraSparc-2 computer, the simulator runs in sub-realtime for small scenes with a few constraints.

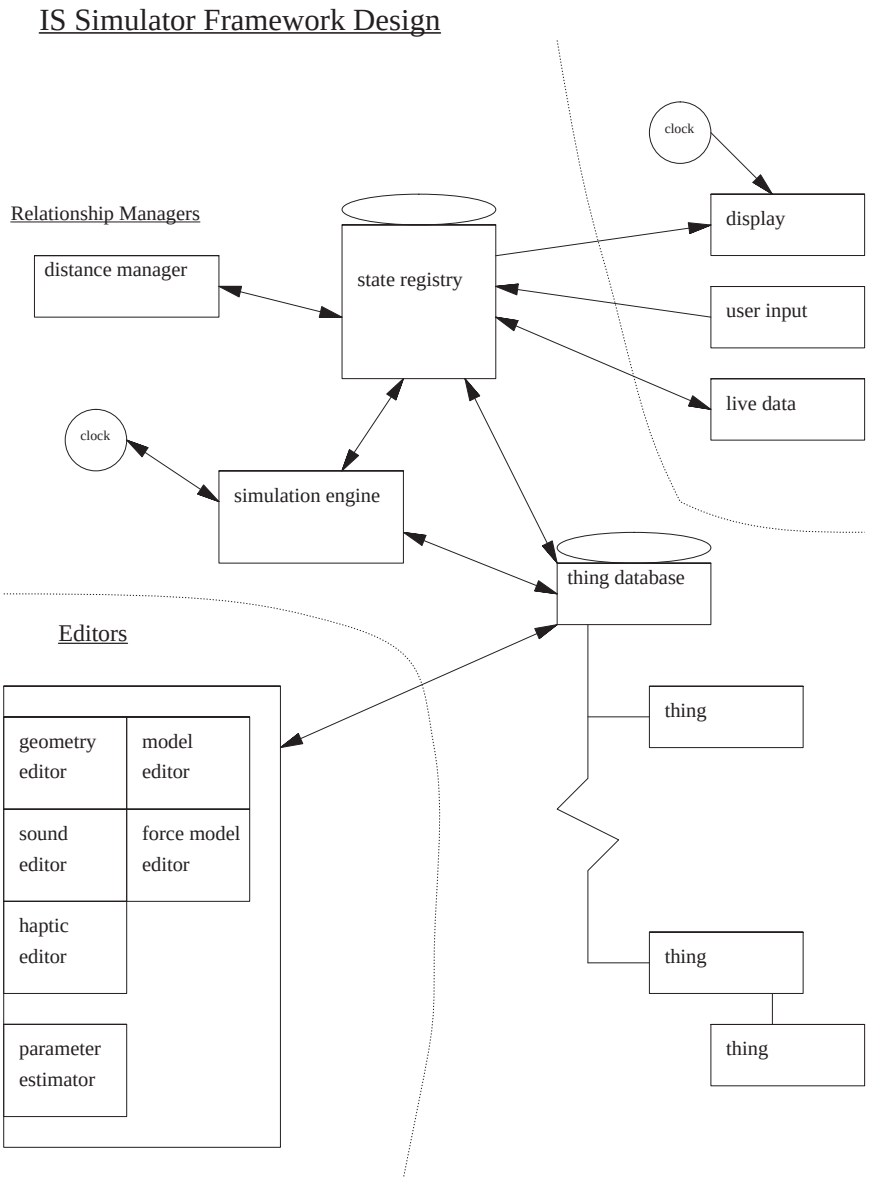
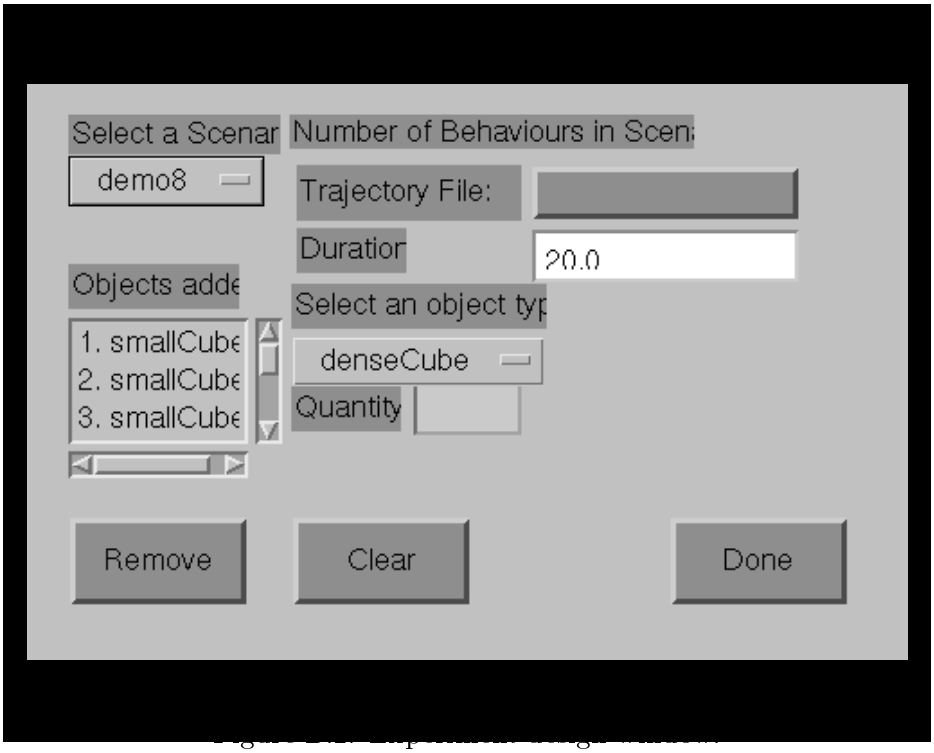


Figure B.1: IS simulator design overview.



Currently the user interface to the simulator consists of three basic windows. The experiment design window (see fig. B.2 is where the simulation is setup and all initial values are set. The running simulation is viewed through the current attached user view. An example view is shown in fig. B.3. The user view can be manipulated either directly with a pointing device such as a mouse or with the navigation panel (fig. B.4). Finally, while a simulation is running, `things` may be selected and interacted with dynamically or through a property sheet (see fig. B.5).

The contact map algorithm discussed in this thesis has not yet been implemented in the simulator. Further work to consider the cases of tangential force and multiple contact regions, as well as the extension to 3D need to be completed before contact maps can be used.

However prototype simulator has demonstrated that a major bottleneck in dynamic simulation is due to collision detection. Although this was implemented

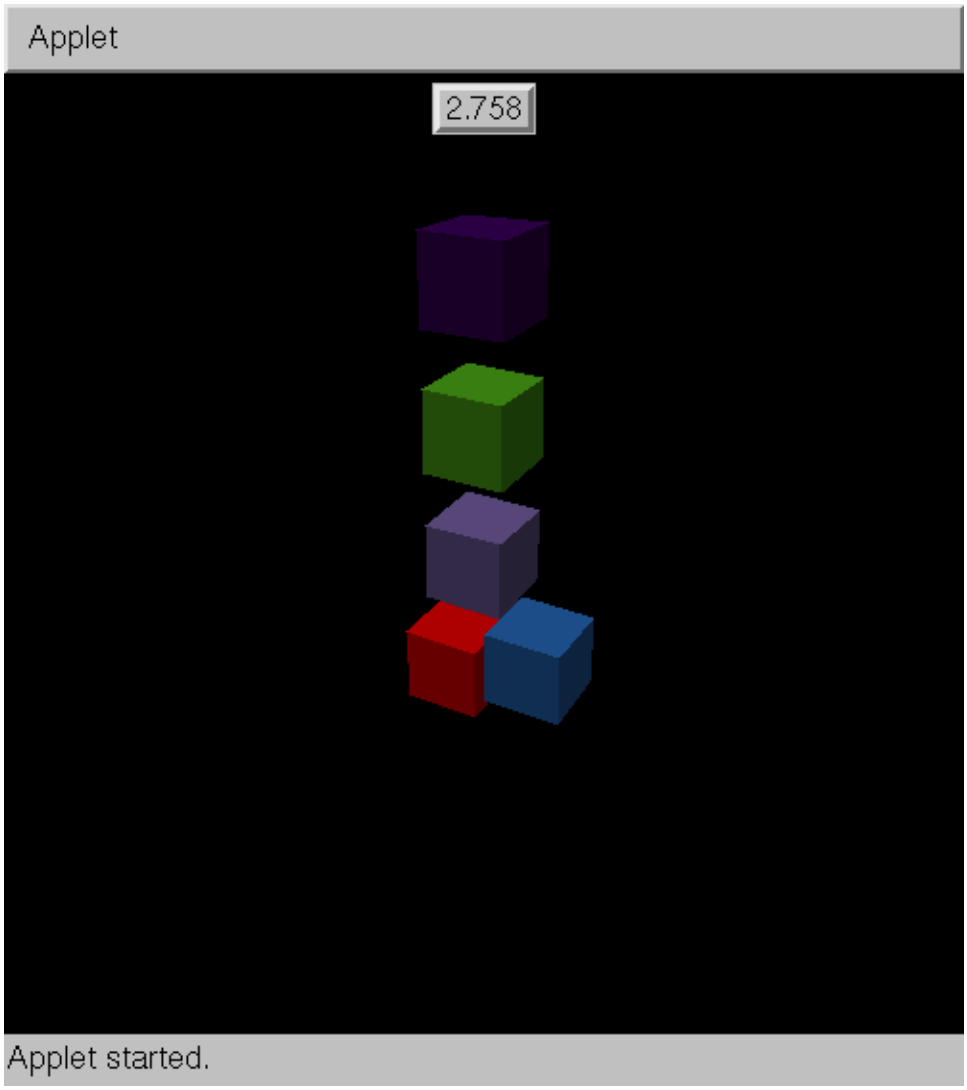


Figure B.3: A running simulation shown from the user view.

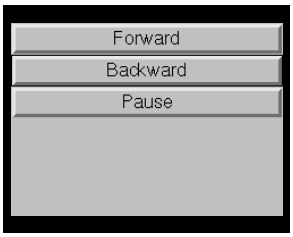


Figure B.4: The scene navigation bar.

as a native code block, it is clear that collision detection must be optimized in any way possible.

B.7 IS Simulator Improvements/Future Work

A great deal of work still needs to be done on the IS Simulator project before any firm conclusions can be drawn with regard to it's usefulness. Certainly the ability to interact and simulate in real time has been shown. However the numerical core is still a native component of the simulator. There are no existing numerical integrators written in Java which provide the power of the MEXX engine. One critical future improvement is thus to make the simulator pure java.

The current prototype presents only limited user interaction. Upgrades to the prototype could include support for haptic (force feedback) rendering, stereo display and 6 degree of freedom pointer devices. The last two are supported in the current specification of Java3D [55]. Haptic rendering presents a special challenge because it must run at a high rate (300-1000 Hz) for objects to feel *firm* [9].

Finally, a critically important feature is the support of multiple views and/or users. As an example, consider using the IS simulator to teach an introductory physics course. The professor could setup an *experiment* demonstrating some important dynamic principle. The simulator would run on a server and students would

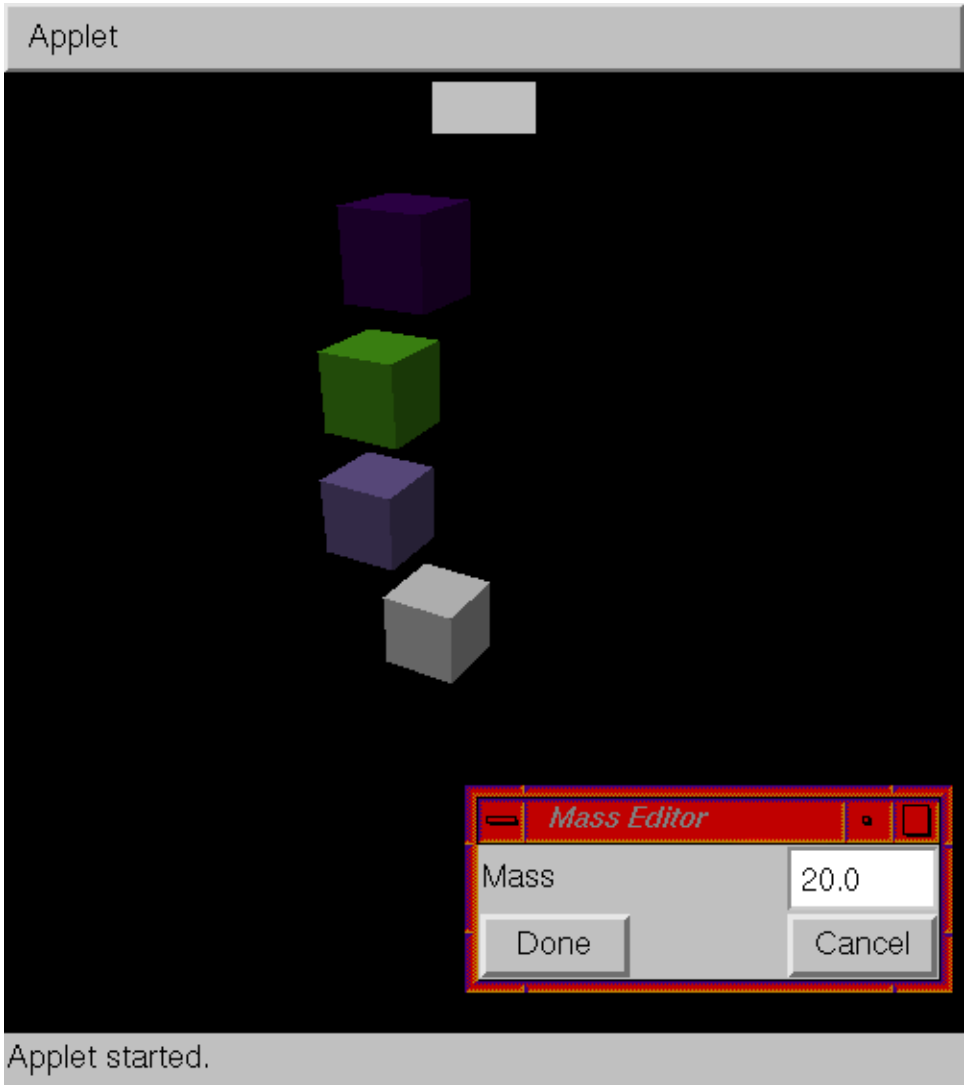


Figure B.5: An example of a live `thing` property editor. While the simulation is running, the selected `things` properties may be edited.

connect and be permitted to interact with and observe the apparatus. This functionality is not discussed in the design explicitly, but it will be an important feature to provide an implementation for.