A Visual Model for Blast Waves and Fracture

by

Michael Neff

A thesis submitted in conformity with the requirements
for the degree of Master of Science
Department of Computer Science
University of Toronto

A Visual Model for Blast Waves and Fracture

Michael Neff
Department of Computer Science
University of Toronto
1998

Abstract

Explosions generate extreme forces and pressures. They cause massive displacement, deformation and breakage of nearby objects. The chief damage mechanism of high explosives is the blast wave which they generate. A practical theory of explosives, blast waves and blast loading of structures is presented. This is used to develop a simplified visual model of explosions for use in computer graphics. A heuristic propagation model is developed which takes object occlusions into account when determining loading. The study of fracture mechanics is summarized. This is used to develop a fracture model that is significantly different from those previously used in computer graphics. This model propagates cracks within planar surfaces to generate fragmentation patterns. Visual results are presented, including the explosive destruction of a brick wall and the shattering of a window.

Acknowledgements

It is a pleasure to write acknowledgements at the end of a body of work. First of all, it gives you a chance to thank the people that have made the work possible. Second, it indicates that the work is finally done!

To begin, I would like to thank my supervisor, Eugene Fiume. His insight contributed greatly to this work and his passionate nature was an important source of inspiration. My second reader, Michiel van de Panne, provided thoughtful and thorough feedback that significantly improved this thesis. More than this, he made himself available throughout this work to answer my numerous questions. I am also grateful to James Stewart and Demetri Terzopolous for useful discussions.

My fellow students in DGP provided important help during this work and also made DGP a fun place to be.

I'd like to thank some wonderful friends for providing an important source of distraction. Whether it was late night stories of Taiwanese pop stardom, driving across the continent with me to jump in the ocean or just getting me out of the lab for a few hours, thanks for being there!

Finally, to my family – Mom (Diane), Grandma (Eunice), Grampa (Ken) and Christine (sister)– for 25 years of love and support: how can I begin to thank you?

This work was financially supported through a Post Graduate Scholarhsip from the Natural Science and Engineering Research Council.

Contents

List of Figures

Chapter 1

Introduction

1.1 Motivation

Explosions inspire both fear and awe. Few, if any, phenomena can match their power and generate such excitement. They capture the human imagination with both their destructive force and their beauty. The goal of this work is to take the initial steps towards a physically based visual model of explosions.

Animations of explosions have many uses. They can be applied for artistic ends and are very important in the special effects industry. Virtual environments could make use of them. Furthermore, if computed with sufficient accuracy, they can also be used in safety training and to study the explosive phenomenon itself.

Throughout its development, one of the major goals of the computer graphics field has been the creation of realistic images of the natural world. Over the last ten years, this effort has increasingly made use of physics based models in developing increasingly accurate modelling techniques. These efforts include the use of fluid dynamics models for animating fire and hot gases [37, 16], dynamic models for animating creatures [42] and physically inspired material models [38], among many others.

Explosions are an excellent candidate for a physically based computer graphics model. It is expensive and dangerous to blow up objects in the real world. Furthermore, it is only practical and feasible to blow up a very limited range of objects. Even when a real world explosion is possible, it is difficult to film these events at the extremely high speeds necessary for generating slow motion films. Furthermore, traditional cinematic pyrotechnics have involved a labour intensive process where small models must be built, scored by hand along arbitrarily determined fracture lines and filled with miniature debris that can be flung out during the explosion [6]. Anything that can be modelled using a computer graphics model can potentially be combined with an explosion model to generate animations at arbitrary frame rates. These animations should also require less work than is involved in the creation of detailed, hand built models. The case for computer modelling is strong.

There are two distinct visual aspects to an explosion: the explosive cloud and the blast wave. The explosive cloud can be a bursting fire ball, or a collection of small particles which are propelled outwards during the explosion. The *blast wave* is a shock wave which expands outwards from the explosion's centre. It is generated by the rapidly expanding gases created by the chemical reaction. The blast wave causes objects to accelerate outwards, deform and shatter.

One of the earliest attempts to model the explosive cloud was made by Reeves using particle systems[33]. This work was used to show a planet exploding in the opening sequence of Wrath of Khan[30]. More recent research efforts at modelling these effects have been made using both physically based [41, 35] and fractal noise approaches [27]. Despite these efforts, a mish mash of techniques are still employed by practicing graphic artists [6]. They also divide the explosion event into the explosive cloud and the physical effects of the explosion. The cloud portion is generated using various techniques, often in combination, including the use of lighting, volumetric effects, particle systems and pre-rendered sequences. Pre-rendered sequences are digitized versions of filmed explosions traditionally used in the special effects industry. They can be combined with three dimensional models by playing them back on planar or conical surfaces which are inserted into the 3D environment.

To model the physical aspect of the explosion, computer animators create two models. One is a clean model of the object, the other is a model of the object as a series of chunks which will exist after the explosions. At the moment of the explosion, a switch is made between the two models. "You then hand animate these [chunks] flying willy-nilly through your scene."([6], p.75) If the animation package affords it, physics based collision detection can be used to reduce the need for keyframing. Neither of these techniques, however, are tied in to an accurate model of the forces being generated by the explosion. The focus of this work is to create a physically based visual model of the effects of blast waves.

1.2 Comparison to Conventional Animation

Numerous techniques are available for generating animations. The model developed here relies on dynamic simulation. Dynamic simulation is a physics based approach for generating animations which moves objects by calculating the forces acting on them.

Traditional hand drawn animation is done using a very different approach. In this work, a scenario is first developed and it is then laid out by story boarding. During story boarding, key moments in an animation are drawn to give an overview of the entire work. A larger number of *keyframes* are then drawn, which show the entities being animated in extreme or characteristic positions. Intermediate frames are filled in by a process known as *in-betweening*[15]. Keyframing is also used in computer animation. Here the in-betweening can be done automatically by specifying an appropriate interpolation function[15].

Key framing offers the animator maximum flexibility and control, but it requires a great deal of work. It would be very time consuming to key frame a complicated explosive event with a large number of moving objects. A key framed animation has no physical basis, relying solely on the animator's skill. Generating a physically accurate animation would be challenging. Objects will go through different phases during an animation. Following an initial rest period, they will be accelerated forward. After this phase completes, objects will be at a high velocity, but will have resistance forces slowing them and gravity pulling them down. Linear interpolation will not give pleasing results when used to model acceleratory motion. Special interpolation functions would be needed. Finally, there will often be a large number of spinning objects that will be rotating at widely varying speeds. To interpolate these motions correctly, a very large number of keyframes would be needed, rendering the process inefficient. Keyframing may be a useful technique when control and flexibility are paramount, but it is inefficient and lacks physical accuracy.

Scripting languages are another technique for generating animations. A script can be written that specifies an object's exact motion as well as the time line for this motion. Such techniques have potential, but will likely be inefficient for blast wave events due to the large number of objects often present. There is also no clear method for determining the motions that are needed. This would likely mean that a large number of iterations would be required, experimenting with different motions until a physically plausible result was achieved. A dynamic simulation provides a principled way of determining object motion.

The technique of procedural animation develops animations through writing procedures, but these procedures normally do not have a basis in physics. This technique suffers from one of the same weaknesses as scripting: there is no clear method for determining what are "correct" object motions. This, again, will likely require a trial and error process. Procedural techniques normally involve a large number of parameters, and these may not have a clear intuitive link to the underlying phenomenon. This is especially true if the procedures are used by someone other than their author. In a dynamics simulation, the parameters all relate to physical quantities, so their meaning is well defined. Procedural techniques do, however, allow more direct control of the final product. If a very specific effect is desired, it may be easier to code for this effect, rather than using a general model.

The use of dynamic rather than kinematic models is also well justified. An explosive event is very complicated. The acceleration phase is of key interest in a slow motion animation. It would be difficult to accurately model this kinematically. Dynamic models can calculate object motions based on accurate models of the forces involved and they can easily handle large numbers of objects. They will provide an accurate and consistent framework in which to generate animations of explosions because they are simulating the actual explosive event, and enough parameters can be made available to give animators the control they need.

The dynamics approach also has an advantage that is lacking in the previous approaches. Our dynamics approach is easily scalable and expandable. The examples in this work are relatively simple, but the same techniques could be used to generate considerably more complex simulations with a minimum of additional work. The blast wave model can also be directly combined with other physically based techniques. For example, the model could be used with a deformable object model to generate animations of deforming structures. The two models can be combined in a straightforward manner because they are both based on the same underlying physical principles.

A dynamics approach provides a coherent approach for blast wave simulation. It will provide consistent, physically based motion for objects throughout a scene. It also offers a solid foundation on which to build additional features.

1.3 Contributions of This Work

The first contribution of this thesis, contained in Chapter 2, is a research summary of the nature of explosions and efforts made to model them. A great deal of both mathematical and empirical research on explosions has taken place. This work is summarized and some insight into different approaches for modelling explosive targets is also introduced.

Based upon this research, a physically based model of blast waves is developed in Chapter 3. This model uses an approach developed in structural engineering research for calculating the load generated by an explosive. The approach is based upon the use of pre-computed blast tables and explosive scaling. It is less computationally expensive than a Navier-Stokes based model and still provides reasonably accurate, physically based results. It can generate near real time animations for scenes with a limited numbers of objects. The model's effectiveness is demonstrated by employing it to blow up a brick wall.

When a blast wave expands outwards, it diffracts around large objects in its path. These objects can provide partial protection for the objects behind them. This is the principle behind the construction of blast walls and many other forms of explosive protection. Modelling the effect of obstacles on a blast wave is a very complicated task that has not been fully solved. In order to obtain reasonable visual results, a heuristic model is developed. This model treats the centre of the bomb as the view point and projects the world onto an icosahedron surrounding it. Calculations determine the degree to which blocking obstacles reduce the impact of the blast wave on the objects behind them. Model details are described in Chapter 4.

One of the most visually interesting aspects of blast waves is the fracturing and deformation of objects that they cause. Chapter 5 presents a background summary on fracture mechanics along with a fracture model which will grow cracks within planar surfaces. This model is based upon placing initial micro-cracks in a panel. This crack propagates by extending and forking, forming a tree-like structure. Results include figures of various generated crack patterns and an animation of a shattering window presented in Chapter 6.

The final chapter presents conclusions and future directions for this research.

Chapter 2

A Practical Theory of Explosions

2.1 Introduction

2.1.1 Defining an Explosion

The exact nature of an explosion can be defined in numerous ways. Baker et al. suggest that an explosion involves an energy release which is rapid enough and takes place in a small enough volume to produce a pressure wave that you can hear[4]. In other words, an explosion must generate a blast wave. *Deflagrations*, defined as rapid burning with flames, are not included as part of the explosive regime in the above definition. The definition does include physical explosions, such as those caused by meteor impact, lightning, or the mixing of certain liquids. It captures the fundamental aspect of an explosion: the very rapid, concentrated release of energy.

A chemical explosive is a material which is normally in a state of metastable equilibrium, but which is capable of violent exothermic reaction[12]. Chemical explosions involve the rapid oxidization of fuel elements. The oxygen needed for this process is contained within the explosive compound[34]. The reaction wave within an explosive can be of two types: a deflagration or a detonation[12].

A deflagration wave is slow, being far subsonic. Deflagrations are propagated by the liberated heat of their reaction[34]. For these waves, transport processes – viscosity, heat conduction and matter diffusion – dominate. Changes in momentum and kinetic energy are small. To a good approximation, pressure changes through a deflagration wave can be ignored[12].

The behaviour of a *detonation* wave is in many ways opposite to that of a deflagration wave. Detonation waves are supersonic, moving at speeds of six to eight thousand m/s in liquids and solids[12]. In gas filled tubes, waves can propagate at velocities between 1 - 3.5 km/s[25]. Momentum and kinetic energy changes are dominant for detonations whereas transport processes are relatively unimportant. Compressibility and inertia are also important, unlike for deflagrations[12]. The great pressure and temperature generated by a detonation wave maintain the conditions necessary for the fast chemical reaction rates that cause the detonation to propagate. It is hence self sustaining and once initialized, will react to completion[11].

For chemical explosions, almost one hundred percent of the energy liberated is converted into blast energy. This figure is only fifty percent for nuclear explosions, with most of the rest going into heat and thermal radiation[34].

Low explosives, such as propellants and pyrotechnics, burn, tending not to detonate, whereas detonation will always occur in high explosives[34]. This work will concentrate on high explosives, where detonation occurs and the blast wave is the dominant damage mechanism.

2.1.2 A Picture of an Explosion

To better understand the explosive process, consider the explosion of a spherical charge of a condensed high explosive such as TNT, as shown in Figure 2.1. A detonation is initiated at the centre of the explosive. For an ideal, uniform explosive, a detonation wave will propagate outwards from the centre at great speeds: 6800 m/s in TNT [18]. Detonation propagation speeds are essentially constant and depend on the density of the explosive involved. Denser explosives sustain higher propagation speeds. As the detonation wave passes through the explosive, it generates immense pressure and high temperatures. Pressure is normally in the range of a few thousand atmospheres and the temperature ranges between 2000 and 4000 K for solid and liquid explosives[11]. These high temperatures and pressures are a result of the extremely rapid chemical reaction that takes place just behind the wavefront. The chemical reaction is typically ninety percent complete in between 10^{-6} and 10^{-9} seconds[11].

Following the discussion in [34], the chemical reaction releases large quantities of gas in a very short period of time. These gases expand violently, forcing out the surrounding air. A layer of compressed air forms in front of the gases which expands outwards containing most of the energy of the explosion. This is the *blast wave*. As the gases move outwards, the pressure drops to atmospheric levels. Thus, the pressure of the compressed air at the blast wavefront reduces with distance from the explosive. As cooling and expansion continue, pressure falls a little below ambient atmospheric levels. This occurs because the velocity of the gas particles causes them to over-expand slightly before their momentum is lost. The small difference in pressure between the atmosphere and the wavefront causes a reversal of flow. Eventually equilibrium will be reached. As with pressure, the velocity of the blast wave decreases as the wave moves farther from the explosive. The blast wave will have a much slower velocity than the detonation wave.

The above discussion ignores the role of wave reflections, as shown in Figure 2.1. The detonation wave propagate outwards through the explosive and hits the boundary between the explosive and its surroundings. At this point, some of the wave transmits outwards as the primary blast wave and part of the wave is reflected back towards the centre. At the centre, it "bounces" and reflects outwards again. This process can repeat several times and there is an intensity decline in the wave each time[18]. These secondary waves appear to be significantly less important with regards to their damage causing potential than the primary wave.

2.1.3 Damage Mechanisms of Explosives

There are several mechanisms by which explosions cause damage, as shown in Figure 2.2. The first is damage caused by flying missiles, also known as primary fragments. Missiles are pieces of the explosive casing or objects located close to the explosive which are accelerated outwards by the explosion. Missiles are initially accelerated more slowly than the associated gases so will lag behind the wavefront. Due to their greater momentum, however, they may outstrip the blast wave and arrive at the target before it. Missile impact energy is the kinetic energy of the missile, $\frac{1}{2}mu^2$. Flying missiles are the primary

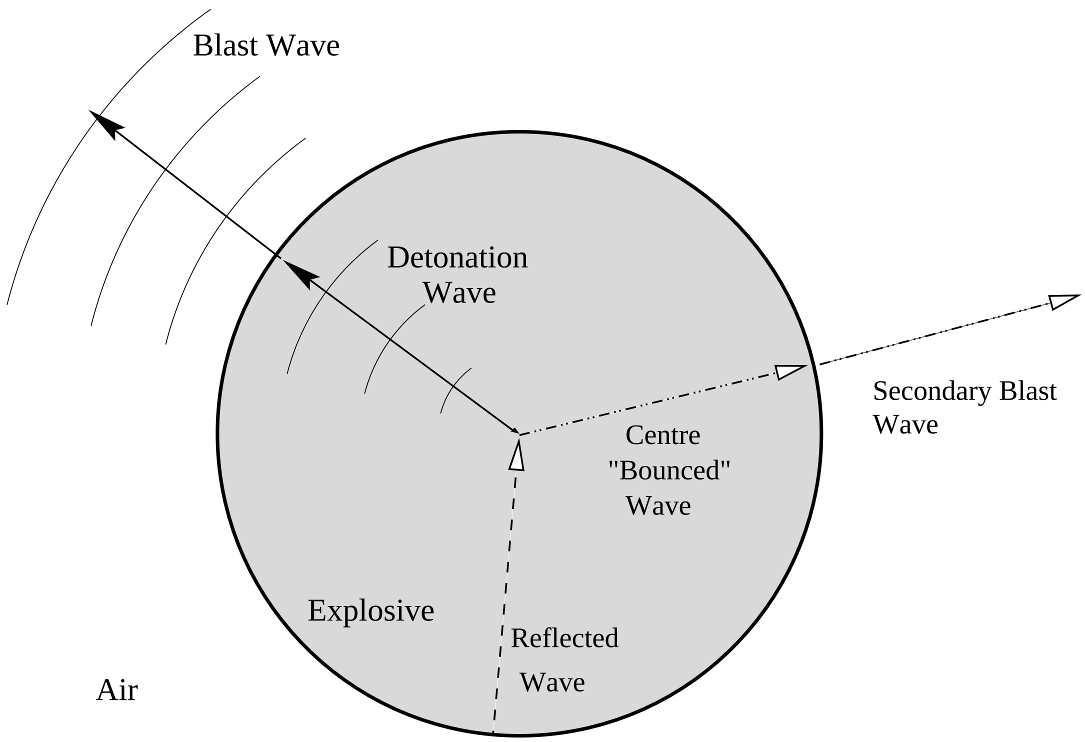


Figure 2.1: The explosion of a spherical charge of TNT. Detonation is initiated at the centre of the charge. The detonation wave spreads outwards until it reaches the air/explosive boundary. Most of the energy and released gas is forced outwards as the primary blast wave. Some of the wave's energy is reflected back towards the middle. This wave "bounces" off the centre of detonation and travels outwards again, eventually forming the secondary blast wave. The secondary wave is signifanctly less powerful than the primary wave.

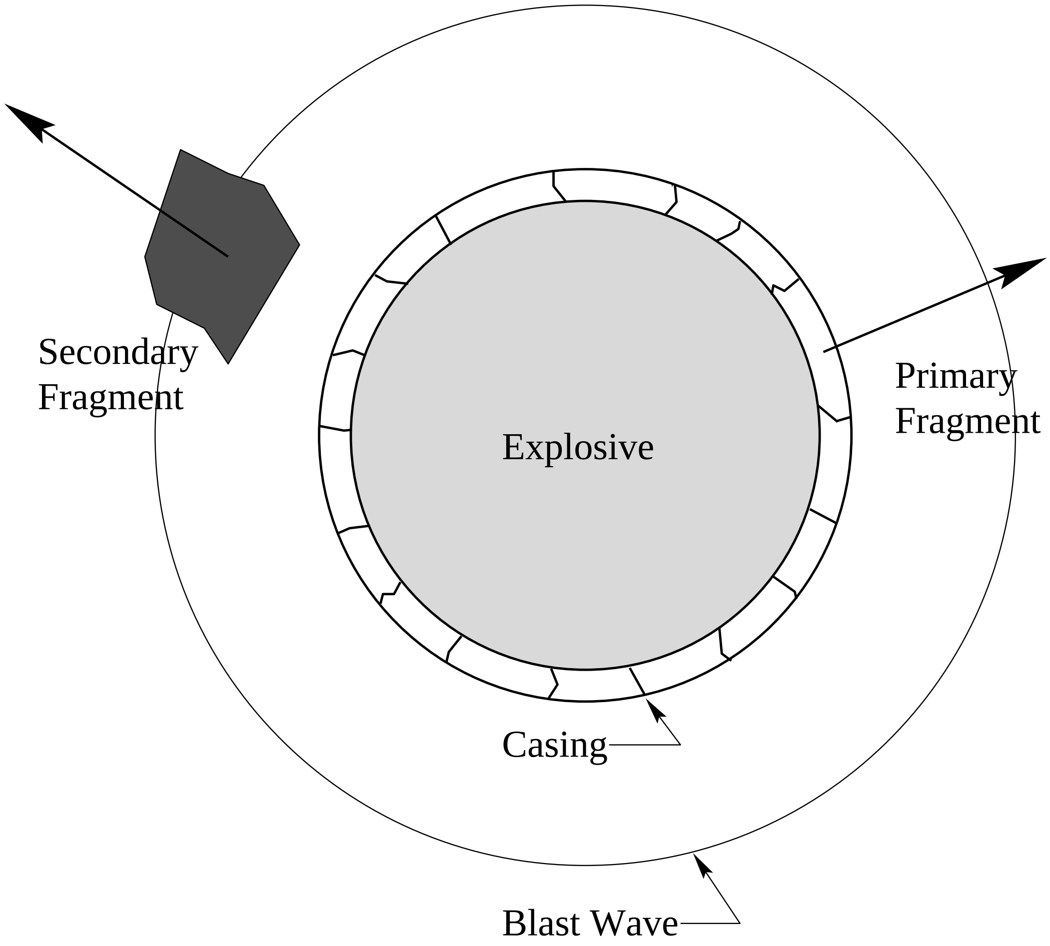


Figure 2.2: Primary fragments consist of pieces of the explosive's casing which are accelerated outwards. Secondary fragments are nearby objects which the blast wave accelerates outwards. Note that secondary fragment is placed abnormally close to the explosive in this diagram for illustrative purposes.

damage agent for small explosives such as grenades[21].

A second cause of damage is the blast wave. The first effect of a blast wave is a "...forcible and violent frontal blow..." followed by a "...tremendous enveloping crushing action while the target is simultaneously subjected to a blast wind of super hurricane velocity." [21, p.3] Blast waves are the dominant damage mechanisms for large explosives. They are the prime focus of this work. Thermal radiation is another damage mechanism, but it is only important for large nuclear weapons.

Blast waves can cause objects to shatter. An explosive's *brisance* is a measure of its shattering power. Brisance is proportional to the speed at which an explosive decomposition occurs (the speed with which it releases its energy). High explosives have high brisance, deflagrations have very low brisance[21].

A third damage mechanism is impact by secondary fragments. These are objects that are near to the explosive which are accelerated outwards by the blast wave.

2.2 Detonations

The first phase of an explosion is the detonation. This phase involves the rapid chemical reaction and the high pressures and temperatures associated with the detonation wave, as discussed above. A detonation can be initiated by a deflagration. When the burning liberates sufficient energy, a transformation from deflagration to detonation occurs. A detonation can also be started by a shock wave. This wave provides sufficient energy to initiate the detonation. Due to the damage that they can cause, high explosives are normally designed so that they require a shock wave for initiation. A smaller, more sensitive explosive, such as found in a blasting cap, is attached to the high explosive. The more sensitive explosive can be detonated by a deflagration or even electrically. When it detonates, it will release a shock wave with sufficient energy to cause detonation in the high explosive. By storing blasting caps separately from high explosives, the chance of an accidental detonation is reduced. Accidental detonation of explosives by shock waves generated by nearby explosives is an area that has received some modelling attention (see [21], for example). This is not worth modelling in computer graphics work, where the desired image is known ahead of time, but might be useful in simulation of virtual environments.

The study of detonations represents a combination of fluid dynamics and chemistry. If the chemical reaction at the detonation front is ignored, a detonation wave is equivalent to a shock wave – a supersonic wave travelling in inert material. A shock wave can be approximated as a jump discontinuity. The pressure, and particle velocity are viewed as changing instantaneously at this discontinuity. The simplest models for detonation waves play little attention to chemistry, assuming the reaction completes instantly at

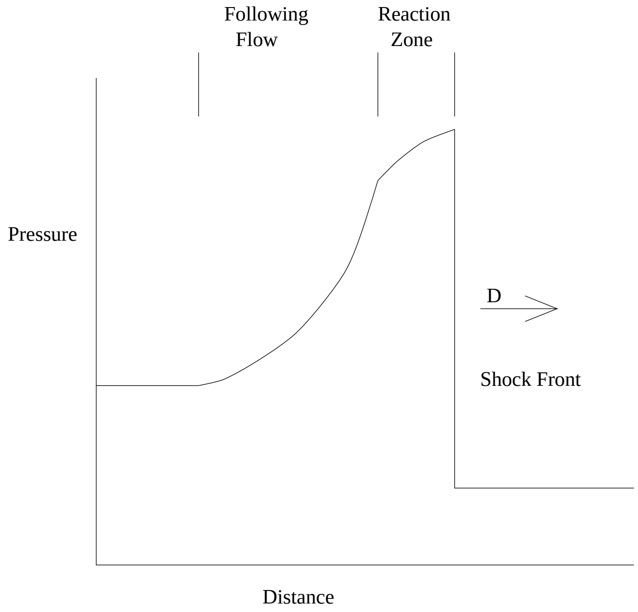


Figure 2.3: ZND Model of a detonation front moving through a section of explosive.

the jump discontinuity. Following is a brief discussion of the ZND model of detonations and an introduction to the Rankine-Hugoniot equations which are used for modelling the transition which occurs at the discontinuity. The ZND model is a more complicated model which includes a finite reaction zone. The model is used, in combination with an equation of state, to determine the conditions immediately behind the wavefront.

2.2.1 The ZND Model of Explosions

The ZND model is a one dimensional, fluid dynamic model of detonation waves developed independently by Zeldovich in the U.S.S.R., von Neuman in the U.S. and Doming in Germany in the 1940s[11]. The model bases its analysis around a semi-infinite tube filled with gas that contains a piston at one end. It captures the structure of a detonation wave as it travels through the explosive. Figure 2.3 shows the ZND model in terms of pressure vs. distance. The wave front is represented by a sudden pressure jump, moving at velocity D. As the front passes over a particle, the particle is suddenly placed in a highly compressed, high energy state. This is where the chemical reaction takes place. The chemical reaction completes very rapidly and is confined to the thin reaction zone immediately behind the wavefront[12]. The *simple model*, assumes the reaction completes instantaneously and has a reaction zone of zero thickness[13]. The reaction zone is a subsonic flow region. Energy liberated here can flow forward and drive the shock wave[11]. After the reaction zone is the *following flow* [12] or *rarefaction* zone[11]. This zone sees a rapid loss of pressure and corresponding expansion of the material. It provides a bridge between the high pressure state that exists at the end of the chemical reaction and the much lower pressure state which exists after the passing of the detonation wave. The rarefaction zone is a supersonic flow region. Waves in, or energy released in, this region cannot effect either the wavefront or the reaction zone. [11]

2.2.2 The Rankine-Hugoniot Relations

The Rankine-Hugoniot relations can be derived in many different ways, but all derivations share a few common characteristics. They are defined on a coordinate system which moves with the wavefront, they treat the wavefront as a mathematical discontinuity and they enforce the conservation of mass, momentum and energy across this discontinuity. The presentation of the equations here is based on Davis [11] and to a lesser extent on Meyers [26] and Hetherington and Smith [34]. Several other formulations are possible depending upon the quantities that are to be tracked. In the following, ρ refers to density, p pressure, e specific internal energy, D the velocity of the detonation wavefront and u_d is the particle velocity immediately behind the wavefront. The specific volume, v, is equivalent to $1/\rho$. The subscript 0 refers to the area ahead of the detonation front and the subscript 1 refers to the region immediately behind the wavefront. The conservation equations for mass, momentum and energy must hold across the wavefront.

Consider a shock wave propagating in a tube of area A for a period t. The wave front is moving at speed D and will hence move over a mass $\rho_0 ADt$ during time t. The material at the beginning of the wavefront at $t = 0$ moves a distance $u_d t$ during time t. Therefore, the material which is passed over by the wavefront during time t will be contained within the volume $A(D - u_d)$ and will have mass $\rho_1 A(D - u_d)$. Since mass must be conserved across the wavefront, these two equations can be set equal. Dividing out At yields the mass conservation equation:

$$\rho_0 D = \rho_1 (D - u_d) \ . \tag{2.1}$$

The material which is passed over is initially at rest and is accelerated to u_d. Recalling that momentum is the product of mass and velocity, the material's momentum change is $\rho_0 ADtu_d$. The change in momentum is equal to the impulse acting on the material. The force acting on the material is $(p_1 - p_0)A$, so the impulse is $(p_1 - p_0)At$ [1]. Equating the impulse and momentum change and dividing by At yields the equation for momentum conservation:

$$p_1 - p_0 = \rho_0 D u_d \ . \tag{2.2}$$

The change in energy in the system is equal to the amount of work done. The total energy is the sum of the internal and kinetic energy. Internal energy changes by $\rho_o ADt(e_1 - e_0)$. Kinetic energy is given by $\frac{1}{2}mu^2$ and hence changes by $\frac{1}{2}\rho_0 ADtu_d{}^2$. Work, the product of force and distance, is given by $p_1 Au_d t$. Equating these yields:

$$\rho_0 ADt(e_1 - e_0) + \frac{1}{2}\rho_0 ADtu_d{}^2 = p_1 Au_d t \ . \tag{2.3}$$

Dividing by At and noting that $\rho_0 D = \frac{p_1}{u_d}$, which is a rearrangement of the equation for momentum conservation (note that p_0 is negligible compared to p_1), yields the energy conservation equation:

$$e_1 - e_0 + \frac{u_d^2}{2} = \frac{p_0}{\rho_0} - \frac{p_1}{\rho_1} + u_d D \ . \tag{2.4}$$

These are the *Hugoniot relations* for a jump discontinuity. Through rearrangement and substitution, they can be used to obtain the single Hugoniot relation, in one of its common forms:

$$e_1 - e_0 = \frac{1}{2}(p_1 + p_0)(v_0 - v_1) \ . \tag{2.5}$$

This equation represents an hyperbola in $p - v$ space. It defines all states which the material can reach during the passing of a detonation wave. From the equations of

[1] recall that $I = Ft$. refer to page 27

conservation of momentum and mass, the equation for the *Rayleigh line* can be derived as follows. The Rayleigh line is used to determine the final state and its slope is can be used to determine the speed of the wavefront. Rearranging equation 2.1 to solve for u_d yields:

$$\rho_0 D - \rho_1 D = -\rho_1 u_d \ , \tag{2.6}$$

$$u_d = D(1 - \frac{\rho_0}{\rho_1}) \ . \tag{2.7}$$

This can be substituted into equation 2.2 to yield:

$$\begin{aligned} p_1 - p_0 &= D^2(\rho_0 - \frac{\rho_0^2}{\rho_1}) \\ &= D^2 \rho_0^2 (\frac{1}{\rho_0} - \frac{1}{\rho_1}) \\ &= (v_0 - v_1)\rho_0^2 D^2 \ . \end{aligned} \tag{2.8}$$

Because these equations are derived from the conservation equations, for any given starting state, the final state must also lie on these lines. That is, the final state must be at the intersection of the Hugoniot curve and the Rayleigh line.

The Rankine-Hugoniot relations involve five variables: pressure, particle velocity, wave front velocity, energy and density or specific volume. Along with the three conservation equations, another equation is needed if all parameters are to be determined as a function of one of them [26]. The fourth equation is the equation of state for the material. It is a polynomial equation which relates wave front velocity and particle velocity of the form:

$$D = c_0 + s_1 u_d + s_2 u_d + \dots \ . \tag{2.9}$$

c_0 is the sound speed in the material at zero pressure. The s coefficients are determined experimentally. There are a large number of equations of state available for various materials[26].

Depending on their derivations, the Hugoniot curve and Rayleigh lines may be drawn in either $p-v$ or $p-u$ space. Figures 2.4 and 2.5 below show them drawn in $p-u$ space. Figure 2.4 shows the result for a shock wave (wave in inert material) and Figure 2.5 applies to a detonation wave.

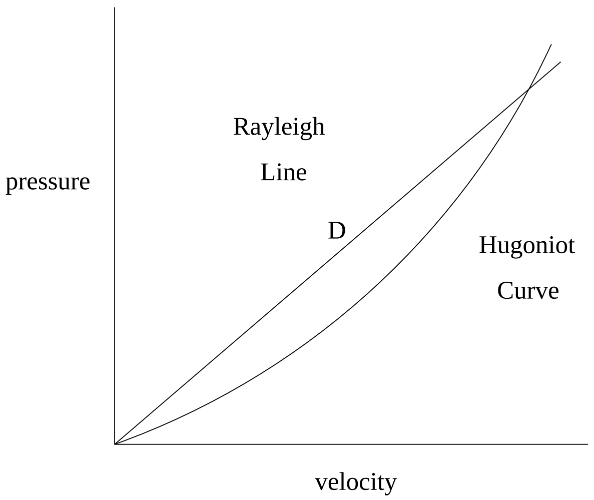


Figure 2.4: Shock front Hugoniot curve and Rayleigh line.

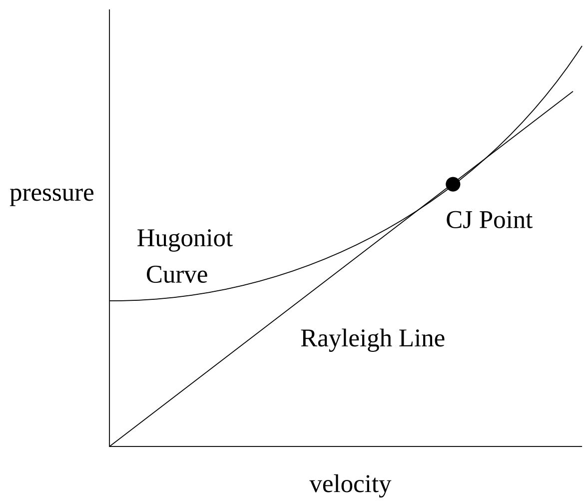


Figure 2.5: Detonation front hugoniot curve and Rayleigh line.

The Hugoniot curve for an explosive in $p - u$ does not pass through the $p = 0$ point (see Figure 2.5). The slope of the Rayleigh line in the $p - u$ plane is proportional to the velocity of the wavefront D. When the Rayleigh line is just tangent to the Hugoniot curve, the wave velocity is a minimum. For any velocity greater than this, the Rayleigh line will intersect the Hugoniot curve at two points. The final state for the shocked explosive is indeterminate.[11]

This ambiguity was resolved by Chapman in England and Jouget in France in the late 1890's [11]. They argued that an unsupported detonation wave (i.e., one that is not overdriven, where overdriven implies that an external force is also acting to advance the wavefront) proceeds at the minimum detonation velocity. This observation is in keeping with the observation that detonations have a well defined velocity that depend upon the composition and density of the bomb, not on external conditions[11]. The details of the Chapman and Jouget argument are available in [25]. The minimum velocity is the point where the Rayleigh line is tangent to the Hugoniot curve. It is referred to as the *Chapman-Jouget* or CJ point and reactions which have this point as their final state are referred to as *CJ reactions*. The CJ point is the starting point for determining the inert flow behind the detonation front[11].

In the previous discussion, there was an implicit assumption that the detonation wave can be represented by a single Hugoniot curve. This assumption implies that the reaction zone is infinitesimally thin: the chemical reaction completes at the discontinuity. This is not physically accurate. In fact, the reaction zone has some thickness, as was seen in the ZND model. This leads to the introduction of a reaction progress variable λ. When the reaction starts, λ is zero and it is one when the reaction completes. Each value of λ defines a Hugoniot curve. The $\lambda = 0$ curve is to the right of the $\lambda = 1$ curve as can be seen in Figure 2.6. Intermediate values of λ generate a family of curves in between these two curves. Figure 2.6 includes two possible Rayleigh lines, one just tangent to the Hugoniot curve and one that intersects it twice. In a CJ reaction, as a particle passes through the detonation wave, it first jumps to the lower point N and then follows the lower Rayleigh line down to point CJ. Some interesting behaviour can be observed in the figure. Namely, there is a pressure and velocity spike at the beginning of the reaction.

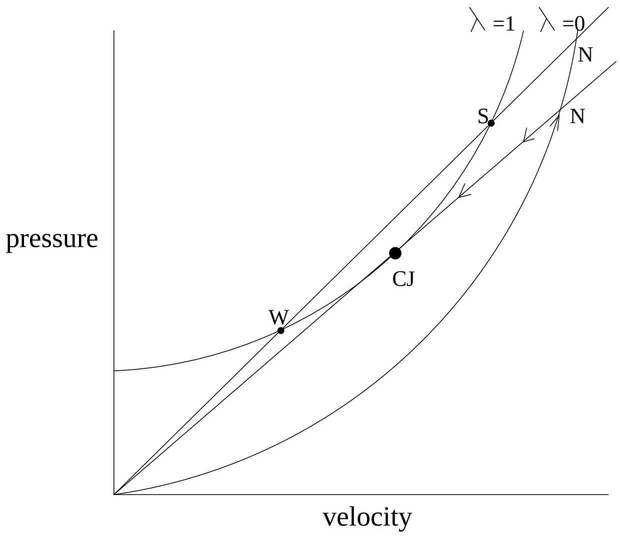


Figure 2.6: Hugoniot curves and Rayleigh lines including a reaction progress variable.

This value drops as the reaction proceeds. This counterintuitive occurrence was predicted by von Neuman and is known as the von Neuman spike [11].

In a real explosion, it appears likely that the final state will not be a CJ point. Davies [11] postulates that the reason for this is the small scale phenomena that take place during an explosion. Local inhomogeneities, hot spots and transverse waves all play a role in the explosion reaching a different final state than the ideal. It has also been suggested that the steady flow that is necessary for a CJ reaction would not be stable in a real system[12].

The upper Rayleigh line in Figure 2.6 intersects the $\lambda = 1$ curve at two points, S for strong and W for weak. Given the single rate model, only the S state is reachable. This is because the state must always be an intersection of the Hugoniot curve and the Rayleigh line. As long as λ is always increasing, as in the single rate model, the final state must lie at or above the CJ point. There is no path that will connect the N and W points. The single process variable model is not very accurate, however. In a real explosion, there could be two or more progress variables and reactions could proceed backwards and forwards. With two progress variables, it is possible to have Hugoniot curves joining S and W, so W is also a possible final state. Indeed, many real explosions end in a W state.

2.3 Modelling Explosions: The Mathematical Approach

The Rankine-Hugoniot relations discussed above give some insight into the mathematical modelling of explosions in the plane. The analysis in this section is complementary to that discussion. The Rankine-Hugoniot equations can provide analytic results for one dimensional configurations: planar shocks where the particle and shock velocities are parallel [26]. Many real world situations are considerably more complicated and will not yield to a simple analytic solutions. This motivates a differential form of the conservation equations, which can be of higher dimension and, through the use of finite-difference or finite-element techniques, solved for complicated simulations via a computer [26].

Following [12], the governing equations for detonations and explosions are the Euler equations of inviscid compressible flow, with chemical reaction added. These are obtained from the compressible Navier-Stokes equations by dropping the transport terms. The Euler equations are presented below in their normal one dimensional linear coordinate form. Vanderstraeten et al. present a one dimensional spherical form of their equations which they use in their study of the failure of spherical vessels containing pressurized gas[43]. Using spherical coordinates allows a three dimensional phenomenon to be approximated using one dimensional equations. The equations can also be derived in higher dimensions.

Before presenting the Euler equations, it is worth defining the *material derivative*, also called the *total time derivative*, as

$$\dot{f} = f_t + u f_x \tag{2.10}$$

for one dimension, where the material derivative is denoted by the dot and the t and x denote the derivatives in time and distance respectively [12]. u is the particle velocity. The total time derivative is used to derive the equation in a Lagrangian referrential – a referential that follows the movement of the particles[26].

Normally only a single chemical reaction is included which can be represented by $A \to B$. A progress variable λ represents how complete the reaction is and is defined as a mass proportion of B.

The Euler equations are

$$\dot{\rho} + \rho u_x = 0 \ , \tag{2.11}$$

$$\rho \dot{u} + p_x = 0 \ , \tag{2.12}$$

$$\dot{e} + p\dot{v} = 0 \ , \tag{2.13}$$

$$\dot{\lambda} = r \ , \tag{2.14}$$

where ρ is the density, p the pressure, e the internal energy and $v = 1/\rho$ the specific volume. The dependent variables are ρ, e, λ and u. The pressure p and reaction rate r are given by state functions which describe the explosive material[12].

A nice derivation of a similar set of equations for spherical coordinates is offered by Henrych [18] and followed in Smith and Hetherington[34]. As a basic unit of analysis, they take a cone with area A at one side and area $A + dA$ at the other. The conservation equations are then applied to this section. For example, the total mass in the system must be conserved. This implies that the mass entering at one side minus the mass leaving at the other side is equal to the mass retained in the volume due to changes in density. Applying this analysis and the Chapman-Jouget relation, they develop a system of six differential equations with six unknowns. The same set of equations is used to model the detonation wave and the blast wave. Boundary conditions are enforced at the transition. The details will not be repeated here, but are available in both works.

Both of the numerical techniques described here rely on the specification of initial conditions and state equations. To use them, either the explosion must be tracked through the detonation phase using initial conditions and state equations for the explosive, or initial conditions and state equations must be determined at the explosive-air border and the explosion can be tracked from there.

2.3.1 Hydrocodes

At research facilities such as Los Alamos National Laboratory, detailed computer simulations of detonation, blast and shock waves have been developed by scientists studying explosives. “These codes are commonly referred to as hydrocodes. They are very complex and involved, and their study and utilization constitutes a specialized field of knowledge.” [26, p.152] Generally speaking, hydrocodes are grid based, using difference equation forms of the differential equations presented above. Unfortunately, direct application of these equations does not work. Spurious high frequency signals dominate the shock region. Large errors are due to the discontinuity in the shock front. The most successful method to overcome this deficiency is the introduction of an artificial viscosity term. This technique was proposed by von Neuman and Richtmyer. It involves adding a term to the pressure that will cause the pressure change to be spread out over several cells[26]. In effect, this introduces a low-pass filter to attenuate the high frequency artifacts.

Hydrocodes normally consist of three components: conservation equations, constitutive equations and failure models. The conservation equations are the mass, momentum and energy equations discussed previously. Constitutive equations describe material behaviour in the elastic, plastic and shock regimes. These equations range from the purely phenomenological to those based on material microstructure. Failure models describe fracture, spalling and shear band formation. [26]

2.4 Modelling Explosions: The Blast Curve Approach

An explosion in a built up area, whether accidental, an act of terrorism or an act of war, can cause a great deal of damage. Due to this danger, there is substantial interest in understanding the impact of explosions, both on the part of weapons designers and on the part of those who wish to design buildings which are better able to withstand explosive attack. A body of literature extending at least as far back as the 1950s has attempted to understand the impact of explosives using a combination of empirical data and mathematical models. References such as [21, 34, 4, 18] document this work. The general approach has been to use a combination of empirical data and mathematical models to formulate both a set of curves and a set of formulas that can be used to determine important quantities such as peak over pressure and impulse at a given distance from an explosive. The work by Vanderstraeten et al., discussed above is a very recent example of using costly mathematical modelling to develop a simple model for predicting overpressures[43].

Several wavefront parameters are of particular importance. These were first identified by Rankine and Hugoniot in 1870 [34] and include p_s, peak static overpressure; ρ_s, static density; q_s, maximum dynamic pressure; and U_s, the blast wavefront velocity. The following equations can be used to determine the last three parameters for normal reflections given p_s. The subscript 0 denotes ambient conditions ahead of the blast wave. The speed of sound in the medium is denoted by a.

$$U_s = \sqrt{\frac{6p_s + 7p_0}{7p_0}} a_0 \tag{2.15}$$

$$\rho_s = \frac{6p_s + 7p_0}{p_s + 7p_0} \rho_0 \tag{2.16}$$

$$q_s = \frac{5p_s^2}{2(p_s + 7p_0)} \tag{2.17}$$

The remaining problem is to determine p_s. There are three possible approaches to this: the use of a fluid dynamics simulation as discussed in the previous section, the use of a set of equations as discussed below or the use of a set of pre-computed *blast curves* as described in Chapter 3. Notice that the blast curves contain values for many parameters besides peak overpressure. Both these charts and the equations use a dimensionless scaling parameter Z, which will be explained below.

2.4.1 Scaling

Two different weight TNT explosives will generate the same overpressure, but they will do so at a different distance from the explosive centre. For a target to experience the same overpressure with a smaller bomb, the target will need to be much closer to the bomb than with a more massive explosive. This is the basic idea behind explosive scaling. Since the same overpressures will be generated by different weight explosives, the weight of the bomb can be combined with distance from the explosive to create a scaled distance

Explosive	TNT Equivalence Factor
Amatol 80/20 (80% ammonium nitrate 20% TNT)	0.586
Compound B (60% RDX, 40% TNT)	1.148
RDX (Cyclonite)	1.185
HMX	1.256
Mercury fulminate	0.395
Nitroglycerin (liquid)	1.481
PETN	1.282
Pentolite 50/50 (50% PETN, 50% TNT)	1.129
TNT	1.000
Torpex (42% RDX, 40% TNT, 18% Aluminum)	1.667

Table 2.1: A given explosive can be converted to an equivalent amount of TNT by multiplying its mass by the TNT equivalence factor.

parameter. This will allow equations or charts giving the peak overpressure to be defined once using this scaled parameter and then applied for an explosive of any mass. The modelling task is thus greatly simplified. The scaled distance Z is used for this purpose and is defined as follows:

$$Z = \frac{R}{W^{\frac{1}{3}}} \tag{2.18}$$

where R is the radius from the centre of the explosion given in metres and W is the equivalent weight of TNT given in kilograms.

The mass of the actual explosive is multiplied by a TNT conversion factor to determine the equivalent mass of TNT. This figure is then used in the above formula. Conversion factors can be based either on the impulse delivered by the explosive or the energy per unit mass of the explosive. The two factors will be slightly different[34]. The table below gives conversion factors for a few explosives based on the energy per unit mass (a partial list from [34]).

As well as the above scaling, it is possible to scale between different explosives to see where an equivalent impact will be delivered. For example, if a given overpressure is felt at radius R_1 for an explosive with TNT equivalent mass W_1, a second explosive with equivalent mass W_2 will generate the same overpressure at radius R_2 as given by the following relation:

$$\frac{R_1}{R_2} = \left(\frac{W_1}{W_2}\right)^{\frac{1}{3}} \tag{2.19}$$

2.4.2 Equations for Predicting Peak Static Overpressure

There are several sets of overpressure equations developed using both numerical and experimental techniques. Two are given below, the first was developed by Brode in the 1950s, the second follows Henrych[34]. Note that with the Brode equations, there is one equation for the near field, where pressure is over 10 bar (1 bar = 10^5 Pa) and one is given for the medium and far field where pressure is between 0.1 and 10 bar.

The Brode equations are

$$p_s = \frac{6.7}{Z^3} + 1[\text{bar}]\ (\text{for } p_s > 10 \text{ bar})\ , \tag{2.20}$$

$$p_s = \frac{0.975}{Z} + \frac{1.455}{Z^2} + \frac{5.85}{Z^3} - 0.019[\text{bar}]\ (\text{for } 0.1 < p_s < 10 \text{ bar})\ . \tag{2.21}$$

The equations presented by Henrych divide the analysis into three fields, a near middle and far field. They are presented below.

$$p_s = \frac{14.072}{Z} + \frac{5.540}{Z^2} - \frac{0.357}{Z^3} + \frac{0.00625}{Z^4}[\text{ bar}]\ (0.05 < \text{Z} < 0.3)\ , \tag{2.22}$$

$$p_s = \frac{6.194}{Z} - \frac{0.326}{Z^2} - \frac{2.132}{Z^3}[\text{bar}]\ (0.3 < \text{Z} < 1)\ , \tag{2.23}$$

$$p_s = \frac{0.662}{Z} + \frac{4.05}{Z^2} + \frac{3.288}{Z^3}[\text{bar}]\ (1 < \text{Z} < 10)\ . \tag{2.24}$$

The Brode equations give good correspondence to experimental peak overpressure results in the middle and far field, but not in the near field. The Henrych equations give good correspondence in the near and far fields, but not in the middle field[34]. This is shown graphically in Figure 2.7.

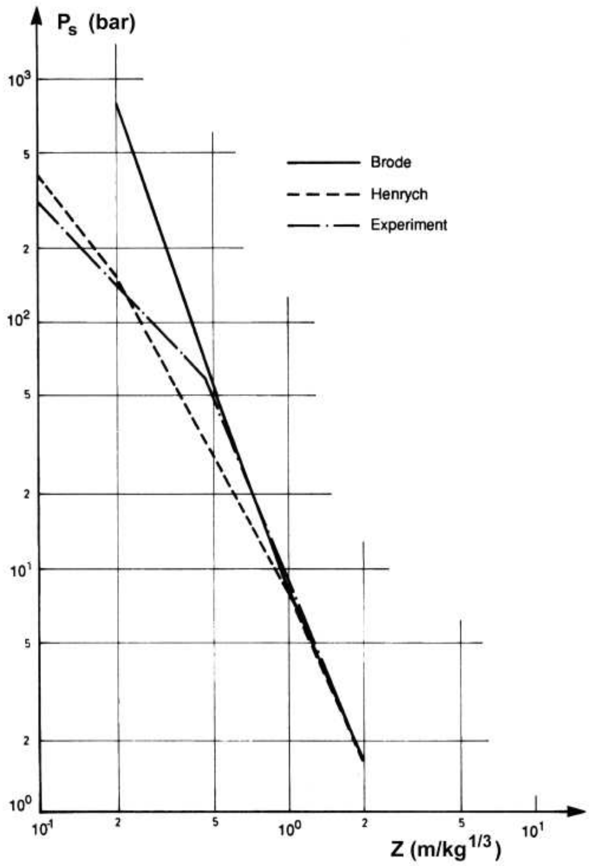


Figure 2.7: A comparison of the Brode and Henrych equations to experimental results.(from [4])

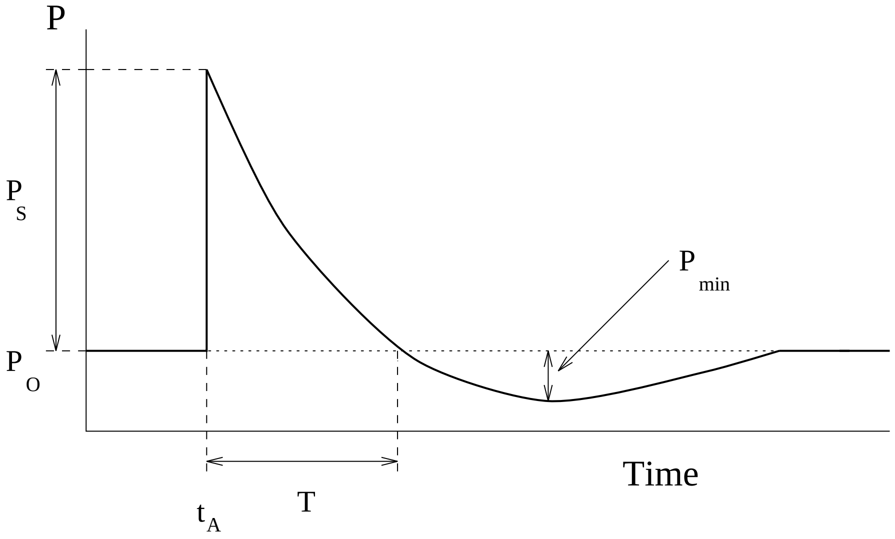


Figure 2.8: Pressure pulse as a function of time.

A different set of equations for overpressure is given by Kinney[21]. He takes a different approach and derives the overpressure from the ambient pressure and the current Mach number of the explosion. A Mach number is a dimensionless way of representing the speed of the wave front that incorporates the speed of sound.

The blast wave curves are included in Chapter 3. They contain curves describing the variation in dynamic pressure, static pressure, wavefront velocity and other important factors all as a function of Z. They provide more comprehensive and more accurate information than the above equations.

2.5 The Pressure Pulse

The profile of the pressure pulse is shown in Figure 2.8. Notice that it represents a jump discontinuity where the pressure increases by p_s. This is the peak static overpressure. With time, this pressure decays to below the ambient pressure p_0. This is due to the overexpansion of gases as described below. P_{min} is the minimum pressure reached. t_a is the arrival time of the pressure pulse. T is the period or time length of the positive phase of the pulse.

An impulse is defined as a change in momentum[9]. It can be calculated as the product of force and time, where time is the duration during which the force was acting

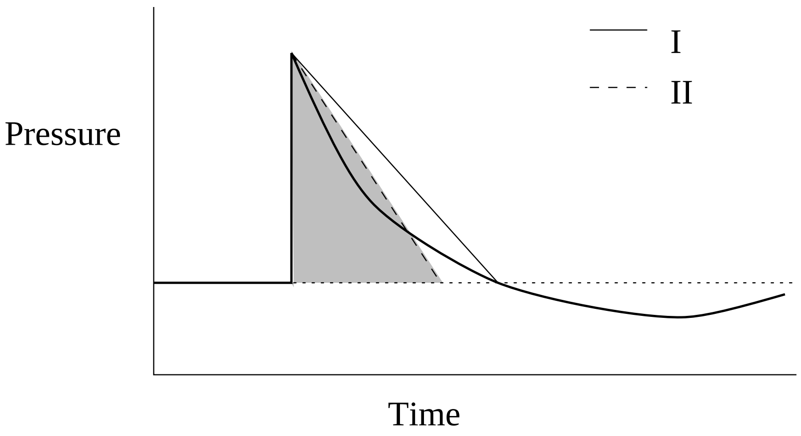


Figure 2.9: Impulse as a function of time approximated by two different triangle pulses.

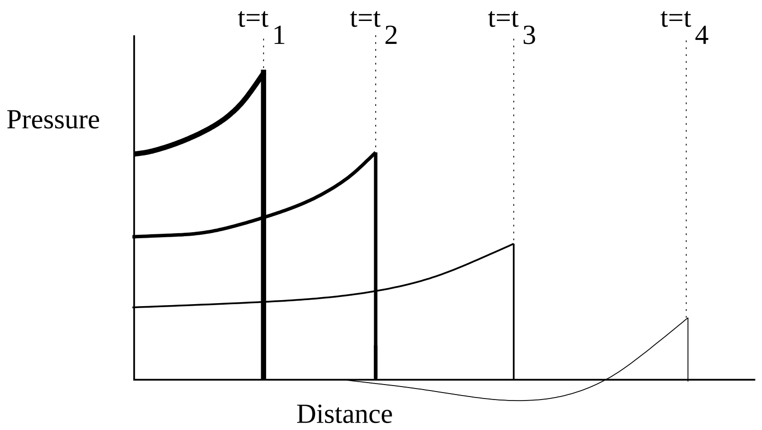


Figure 2.10: Pressure pulses at four different radii.

on an object. The impulse generated by the pulse wave can be computed by integrating the positive phase of the pulse wave; from t_a to $t_a + T$,

Often it is sufficient to represent the positive pulse as a triangular wave. There are two ways to do this which are shown in Figure 2.9. The line labelled I will preserve the period T because it intersects the P_0 line at the same point. The line labelled II creates a triangle with the same area as the original pulse. This maintains the same impulse. The peak overpressure decreases with distance from the centre of the explosion. This effects the shape of the pressure pulse. Pressure pulses for 4 different radii are shown in Figure 2.10.

2.5.1 Surface Effects

The above treatment applies to explosions in air. Another common case is surface explosions, where the explosive is located on (say) the earth's surface. A surface explosion can be modelled by considering it to be an air explosion with 1.8 times the total energy[34, 4]. If the ground was a perfect reflector, this factor would be 2. Since the ground will absorb some of the energy of the explosion, the heuristic factor 1.8 gives a good correspondence with observed results.

2.5.2 Transport Medium

For this work, air is the transmission medium of interest for the blast waves. It might be interesting in the future to model underwater blast waves. Over the range of interest for explosions, air will generally behave as an ideal gas. This allows the ideal gas equation to be used as an equation of state. The ideal gas equation is

$$PV = \frac{m}{M} RT \ , \tag{2.25}$$

where P denotes pressure, V volume, T temperature, R is the gas constant, m is the mass of the gas particles in the volume V and M is the molecular mass of the gas[9].

The sound speed, a, is needed for some calculations. It is given by the following relation for an ideal gas:

$$a = \sqrt{kRT} \ , \tag{2.26}$$

where k, also called γ, is the ratio of specific heat at constant pressure to specific heat at constant volume. It equals 1.4 for air[21]. R is the universal gas constant and T is temperature. A good approximation gives the speed of sound in air around room temperature as 350 m/s [34, 21].

2.5.3 Other Damage Mechanisms

Aside from the primary blast wave, there are several other significant damage mechanisms. These include stress waves and ground shocks.

When a blast wave strikes an object, a stress wave is generated in the object. As the stress wave propagates forward into the object, it acts as a compression wave. Striking the rear boundary of the object, it will be reflected back as a tensile wave. Most materials are stronger in compression than tension, so the tension wave will often cause more damage. In concrete structures, the compression wave can lead to *spalling*, also known as scabbing, on the rear surface of the object. Spalling occurs when part of the surface layer of concrete "flakes" off[34].

When an explosion is very powerful or if the explosive is on the ground, the explosion will cause waves to propagate through the soil. These waves are of three types: P or compression waves, S or shearing waves and R waves which are circular surface waves[34]. R waves are surface waves whereas P and S waves are body waves. The seismic wave velocity varies from 200 m/s in dry sand to 1500 m/s in saturated clay[34]. These waves can cause damage to structures in or on the ground.

2.6 The Interaction of Blast Waves and Structures

2.6.1 General Picture

When a blast wave encounters an object, it will both reflect off the object and diffract around it. It will also generate stress waves within the object. The exact behaviour depends upon the geometry of the object, the angle of incidence and the power of the wave. An example of a wave striking a rectangular object is shown in Figure 2.11. Following [34], three examples will be given, one for a large scale blast wave and a large object, one for a large blast wave and small object and one for a small blast wave and large object.

The magnitude of the reflected pressure is related both to the angle of incidence of the blast wave

Large Blast Wave, Large Object

- wave will completely engulf the large object, generating very strong crushing forces
- some translational force will also be generated, but the object is unlikely to move

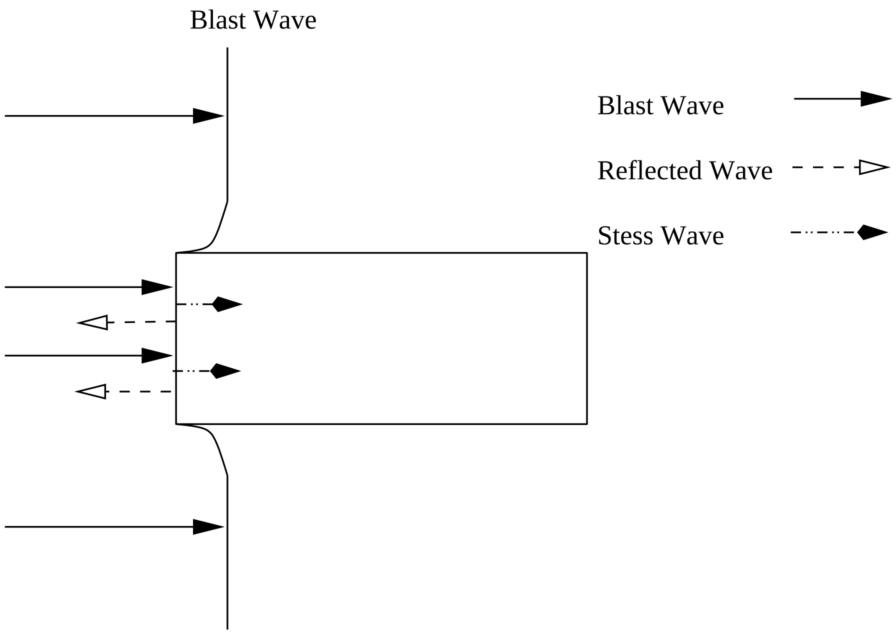


Figure 2.11: When a blast wave strikes an object, it reflects off it, diffracts around it, and generate stress waves in it. These three waves, with their correct directions, are shown here for a blast wave striking a rectangular object.

due to its large size

Large Blast Wave, Small Object

- object will also be engulfed and crushed by blast wave
- squashing overpressure will be more or less equal over the entire object and the translational force will only last for a brief time
- translatory force due to dynamic or drag loading will also be generated which will have a longer duration and can lead to significant movement and damage

2.6.2 Small Blast Wave, Large Object

- analysis must normally be performed on the object elements separately because the small wave will not load the structure evenly
- by the time the end of the structure is being loaded, the beginning will be experiencing very different loads

2.6.3 Reflections

Reflections are divided into three categories: normal reflections, oblique reflections and Mach stem formation. *Normal* reflections occur when the blast wave hits the object head on, or at zero degrees incidence. *Oblique* reflection occurs when the angle of incidence is small, less than about forty degrees in air. [34] The angle of reflection is not normally the same as the angle of incidence for oblique reflections [21] and the discrepancy depends upon the pressure of the wave. *Mach stem* formation occurs for larger angles of incidence (> 40 degrees). A "spurt-type effect" occurs when the shock front impinges on the surface at near grazing incidence[21]. The phenomenon occurs when the reflecting wave catches up with and fuses with the incident wave to form a third wavefront called the Mach stem. The place where the three come together is called the triple point[34].

A blast wave striking an object will generate a pressure on the face of the object which is greater than the peak static pressure of the wave. This occurs because the forward moving air molecules are brought to rest and further compressed by the collision. The peak static overpressure is the pressure that would be felt by a particle moving with the wavefront. When a stationary object is struck by the blast wave, however, the object will face this pressure and will also be hit by the particles being carried with the stream – the blast wind. This leads to the concept of *dynamic pressure*, q, which is defined as

$$q = \frac{1}{2}\rho u^2 \ . \tag{2.27}$$

Here u is the particle velocity and ρ is the air density immediately behind the wavefront.

The total pressure experienced by the object face is the *peak reflected pressure*, p_r, a combination of static and dynamic pressure. For normal reflections it is given by the following relation:

$$p_r = 2p_s + (\gamma + 1)q_s \ . \tag{2.28}$$

For air, $\gamma = 1.4$. The following relation can be derived for air where p_0 is the ambient pressure:

$$p_r = 2p_s \frac{7p_0 + 4p_s}{7p_0 + p_s} \ . \tag{2.29}$$

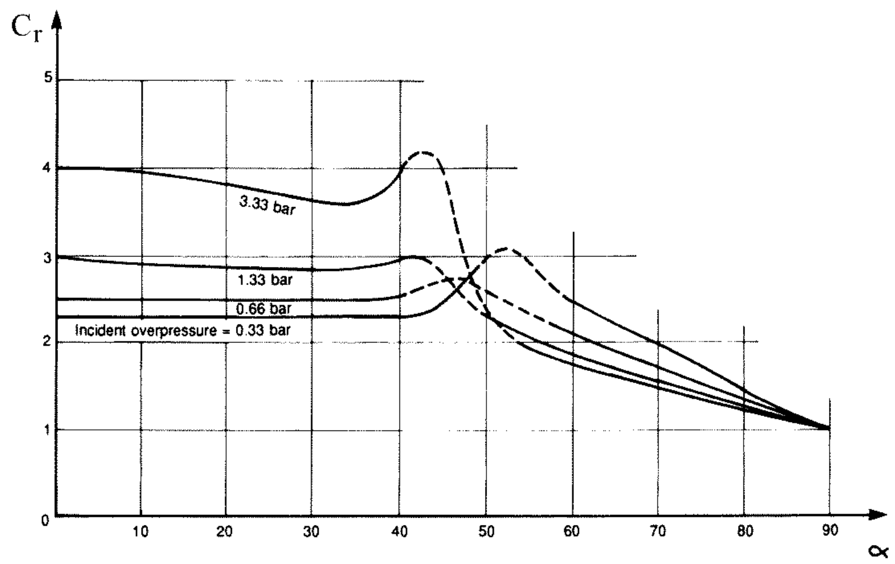


Figure 2.12: Co-efficient of reflection versus angle for three different pressures. Notice that C_R increases at the mach stem transition. (from [4])

These relations are derived from the Rankine-Hugoniot relations [34].

A reflection coefficient, C_r, can be defined which is the ratio of p_r to p_s. From the above relation, taking the limit for small p_s, much smaller than p_0, C_r will equal 2. In the limiting case where p_s is very large and p_0 can be taken as zero, C_r will equal 8. This is not completely physically accurate as for very large overpressures, C_r can actually exceed 8.

As the angle of incidence for the blast wave increases, the reflected pressure gradually decreases. This continues until the Mach stem transition is reached. At this point there is a jump in reflected pressure which can actually exceed that of normal reflections for low peak overpressures. From this point, the reflected pressure will again decline with angle of incidence. This is shown in Figure 2.12. At ninety degrees, there is no reflection and the peak reflected pressure is equal to the peak static or side on overpressure, p_s.

The magnitude of the reflected pressure is related both to the angle of incidence of the blast wave and the magnitude of p_s. For normal reflections with very high static overpressures, C_r can exceed 12[34]. A chart is available that will give C_r as a factor of both angle of incidence and magnitude of pressure. It is included in Chapter 3.

2.6.4 Dynamic Loading

Front Face

Due to the reflection pressure that develops, the front face will experience a pressure much higher than exists in the surrounding medium. This generates a flow from the high pressure area to the lower pressure air surrouding the object and a rarefaction wave develops to dissipate the excess pressure. This pressure relief will start in the air at the edge of the object and spread in to the centre of the front face[21]. The pressure drops to the stagnation overpressure, which is defined as [34]

$$p_{stag}(t) = p_s(t) + q_s(t) \ . \tag{2.30}$$

The time it takes to reach stagnation pressure from the beginning of loading is given approximately by:

$$t' = \frac{3S}{U_s} \ , \tag{2.31}$$

where S is one half of the smaller of the object front's height or base width. S represents the minimum distance the dissipation wave must travel. Once stagnation pressure is reached, the loading will follow the stagnation pressure curve during the remainder of its loading. The situation is shown at the top graph of Figure 2.13. As was done with the original pulse wave, it is possible to treat this pulse as a single triangle which preserves the impulse of the wave. Again, this can be done in such away as to either preserve the maximum pressure or the period of the positive wave. [21]

Side Faces

Side faces will receive loading equal to the peak side overpressure. The sides will be loaded as the wave passes over them. Therefore, the time for loading can be calculated from the blast wave velocity. This loading is shown in the middle graph of Figure 2.13.

Rear Face

The loading of the rear face is similar to the pressure relief on the front face. Loading will begin after the blast wave has travelled the length of the structure. Loading begins

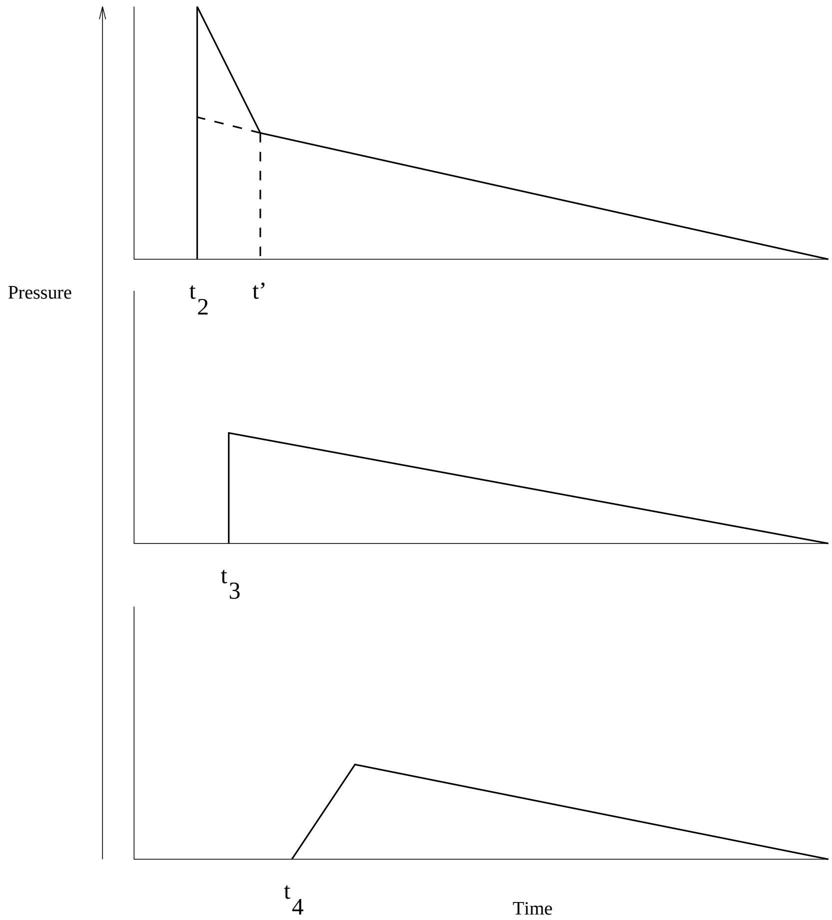


Figure 2.13: Pressure profile versus time for the front, side and rear faces, respectively, of a rectangular prism shaped object.[34]

at the rear face's edges and moves in towards the middle. This is the reason the pressure curve for the rear face does not feature a jump discontinuity at the left side (see lower graph of Figure 2.13). The average time to reach peak load for the entire face is $\frac{3S}{a}$ [21] .

Drag Loading

Drag forces are generated by the dynamic loading of the object. They are the forces that will tend to cause translatory movement. Drag force is defined as:

$$F_D = C_D \times q_s(t) \times A \ , \tag{2.32}$$

where A is the area loaded and C_D is the drag coefficient of the object. C_D is based upon the shape of the object. Tables of C_D values for different shapes are available (see [21] for example). Typically, C_D is near one.

The net transverse force on an object is a combination of the force caused by the reflected pressure and the drag force[4]. This is at least true for objects that are loaded by waves with long positive periods compared to object length. According to Smith and Hetherington [34], if the pulse length is long, the object can be considered a drag target. If the positive phase is short or the target is long, it is more of a diffraction target.

2.7 Lumped Mass Object Models

The modelling of objects' response to blast loading is complex, so it is useful to make simplifying assumptions. One such assumption is to consider the object to be a single lumped mass. A common technique is to treat the object as a single degree of freedom (SDOF) mass-spring system. Other techniques are possible, but in the interest of brevity, only SDOF models will be discussed here. The equation of such a model is

$$M\ddot{x} + c\dot{x} + kx = F(t) \ . \tag{2.33}$$

where M is the mass, c is the structure's damping resistance, k is the structural resistance, x is the displacement and F is the force acting on the object. The frequency of the object is $\omega = \sqrt{\frac{k}{m}}$ and the period of the vibrations is $T = \frac{2\pi}{\omega}$.

Most of the literature studying blast loading of structures comes out of engineering research [21, 34, 4, 18] which tries to establish the maximum displacement, and hence the maximum damage, that a structure experiences subject to a given loading. For this work, it is customary to take the structure's damping resistance, c, to be zero. This has little effect on the maximum damage, the long term behaviour of the building is not of interest, and accurate values of c are difficult to determine [4, 34]. Such an assumption may also be justified for computer graphics work as the magnitude of the object vibrations appears to be small enough as to not be visually interesting. An example calculation by Baker et al.[4] shows a 2800kg object experiences a maximum deflection of just over an inch under a significant blast load. Of much greater visual interest will be the movement of the structure after its breaking point has been exceeded.

The mass-spring equation can be solved using an average acceleration technique which uses small time steps and updates values of displacement, velocity, and acceleration considering acceleration to be constant within a given time step [4]. An explicit alternative exists for the idealized triangular blast force with maximum force F and period t_d and an object with zero initial displacement and velocity. For $t < t_d$,

$$x(t) = \frac{F}{k}(1 - \cos \omega t) + \frac{F}{kT_d}(\frac{\sin \omega t}{\omega} - t) \ . \tag{2.34}$$

For $t > t_d$,

$$x(t) = \frac{F}{k\omega t_d}(\sin \omega t_d - \sin \omega (t - t_d)) - \frac{F}{k} \cos \omega t \ . \tag{2.35}$$

The spring-mass equation with a constant structural resistance k represents an object undergoing *elastic deformation*. Elastic deformation is deformation that the object can recover from. When an object is displaced beyond its maximum elastic displacement, it undergoes *plastic deformation*. Plastic deformation represents permanent damage to the structure. An object can experience a certain amount of plastic deformation before it breaks. An idealized resistance curve showing the elastic and plastic phases is shown in Figure 2.14. x_{max} represents the yield point. The elastic phase goes from $x = 0$ to x_{el} and the plastic phase from x_{el} to x_{max}.

The spring-mass equation can be reformulated to take into account resistance as given

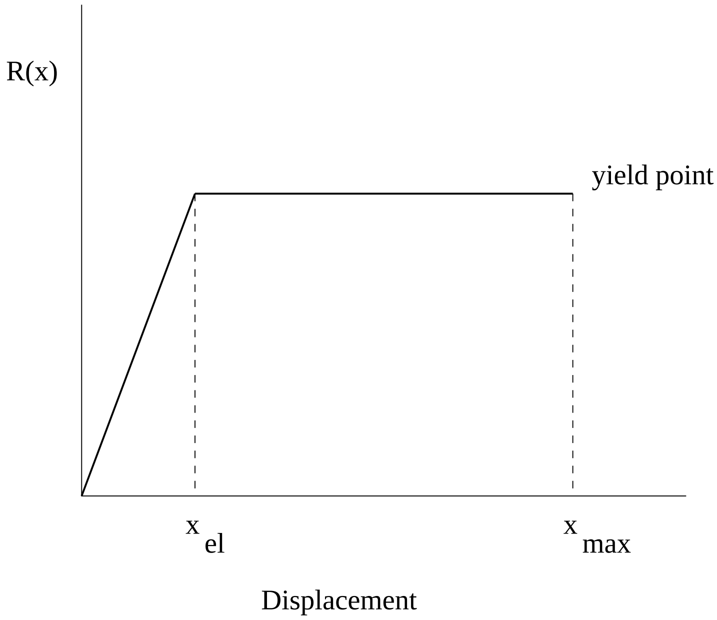


Figure 2.14: Resistance versus Displacement.

in Figure 2.14 as:

$$F(t) - R(x) - c\dot{x} = m\ddot{x} \ . \tag{2.36}$$

This can be solved using similar numerical techniques. Kinney [21] uses this form exclusively, but does not consider a damping force and expresses force as pressure times frontal area $(p(t)A)$.

2.8 Pressure-Impulse Diagrams

Another way to analyze a structure's response is in terms of work. The work done on the object is equal to the sum of the strain energy and the kinetic energy generated in the structure. $W = E_{strain} + E_{kinetic}$ which can be stated as

$$fx_{max} = \frac{1}{2}x_{el}R_{max} + R_{max}(x_{max} - x_{el}) \ , \tag{2.37}$$

where x_{el} is the maximum elastic displacement as in Figure 2.14 [34].

If the ratio of the positive pulse length to the structure's period, $\frac{t_d}{T}$ is large, the structure receives quasi-static loading and kinetic energy changes dominate. If the ratio of t_d to T is small, the structure experiences impulsive loading and the strain energy

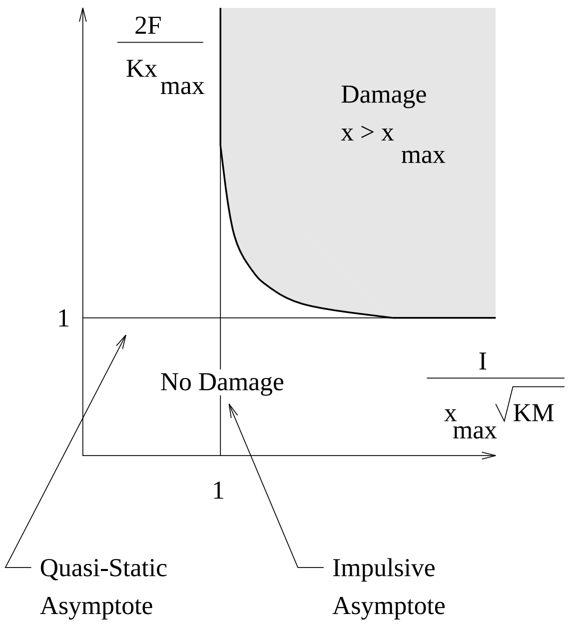


Figure 2.15: Pressure-Impulse diagram to determine damage threshold [34].

changes dominate. These limits define impulsive and quasi-static asymptotes. When graphed as in Figure 2.15, a given load can be plotted on the graph to see if x_{max} is exceeded and damage occurs.

2.9 Primary and Secondary Missiles

Primary missiles are generated from material enclosing the explosive (see Figure 2.2). For high explosives, the casing ruptures into a very large number of very small pieces. These pieces normally weigh a gram or less and have initial velocities of several thousand m/s. They have a chunky geometry, meaning that all their linear dimensions are approximately the same[4].

Vessels containing high pressure gas may burst into larger chunks. Often these larger chunks have flat, elongated geometry. They can weigh several kilograms and have speeds in the hundreds of m/s[4].

Secondary missiles are objects which are near a powerful explosive and are accelerated powerfully outwards by it.

The forces which act on these objects after the initial acceleration are gravity and fluid dynamic forces. The force of gravity is given by the product of the mass and the acceleration due to gravity, Mg. Fluid dynamic forces commonly considered are the drag and lift forces. The drag force acts along the trajectory of the object and the lift force is normal to the trajectory. They are given by the following equation where L refers to lift, D drag, C is the lift/drag coefficient and A is the lift/drag area. For objects of arbitrary shape, both C and A will vary as the object's orientation changes.

$$F_L = \frac{1}{2} C_L \rho v^2 A_L \tag{2.38}$$

$$F_D = \frac{1}{2} C_D \rho v^2 A_D \tag{2.39}$$

Notice that these definitions are comparable to the earlier definition of drag force. For a chunky fragment, $C_D \gg C_L$ and the object is a drag target (lift can be ignored). If $C_L \geq C_D$ the fragment is a lift type fragment. Lifting fragments have a diameter several times greater than their thickness. Missiles often have irregular shape and may be tumbling, making precise fluid dynamic calculations difficult[4]. In a visual model, the drag resistance can be used as a damping term, to reduce the speed of the outward flying objects.

2.10 Solving for Object Motion

The drag loading of an object is shown by the pressure profile in Figure 2.16 [4]. Notice that this ignores the peak side on overpressure, p_s, which will act everywhere on the object and can hence be ignored. The pressure remaining is the pressure that will act only on the front of the object and will not be balanced by any other pressure. This is the pressure which can lead to the object accelerating.

In a similar style to the mass-spring model above, a general equation of motion is given by:

$$M\ddot{x} + R(x) = F(t) \ . \tag{2.40}$$

Here, $R(x)$ represents any combination of forces which are acting to resist the movement of the object. If surface friction is the major form of resistance, $R(x)$ can be replaced with μMg where μ is the coefficient of friction and g the acceleration due to gravity.

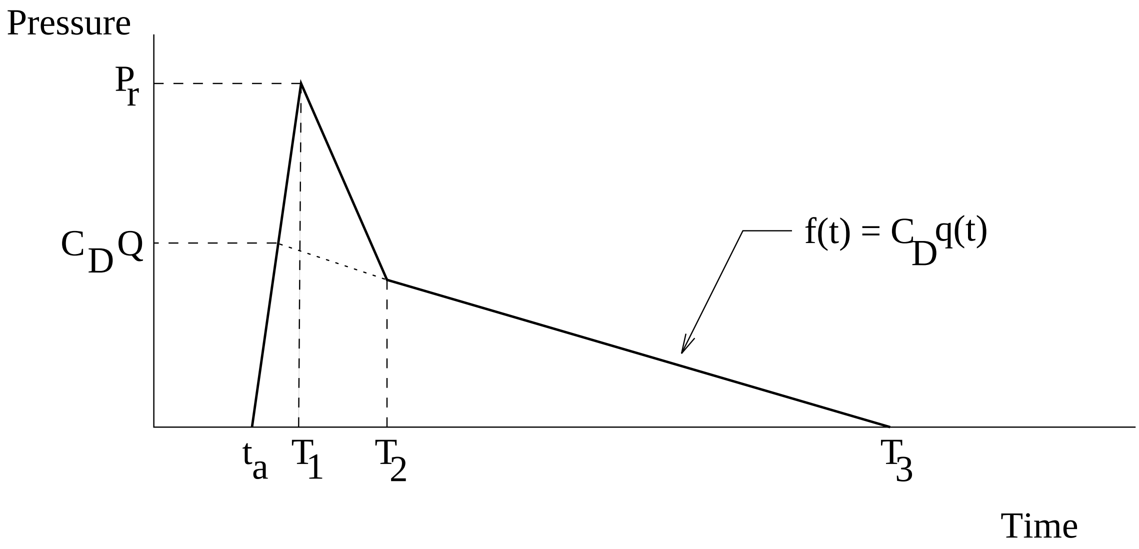


Figure 2.16: Translatory Pressure versus Time.

Following [4] and ignoring the effects of gravity during acceleration yields the simpler equation $Ap(t) = M\ddot{x}$, recalling that $F = pA$. Rearranging and integrating gives the object velocity as

$$\dot{x}(T_3) = \frac{A}{M}\int_0^{(T_3-t_a)} p(t)dt = \frac{A}{M}i_d \ , \tag{2.41}$$

where i_d is the total impulse, p(t) is given in Figure 2.16. The above relation assumes zero initial velocity and displacement. An alternative is to solve the full equation iteratively using the following relations in the first step and then proceeding with the same pattern [34]. Acceleration:

$$\ddot{x}_0 = \frac{F(0) - R(x_0)}{M} \ . \tag{2.42}$$

Velocity:

$$\dot{x}_1 = \dot{x}_0 + \ddot{x}_0 \Delta t \ . \tag{2.43}$$

Displacement:

$$x_1 = x_0 + \dot{x}_0 \Delta t + \frac{1}{2}\ddot{x}_0 \Delta t^2 \ . \tag{2.44}$$

2.11 Modelling Glass

Glass is very brittle and will tend to shatter under blast loading. This makes the modelling of glass a visually interesting task for computer graphicists. Since glass particles are a major source of penetrating fragments, empirical studies have been done on glass fragments generated by the breaking of window panes (cited in [4]). These studies found the mean frontal area of glass fragments to be

$$A' = 6.4516 \times 10^{-4} e^{[2.4 - \sqrt{12.5 + (5.8566 \times 10^{-5} P_e)^2}]} \ . \tag{2.45}$$

P_e is the peak pressure experienced by the windows. It is P_R for front facing windows and P_S for side facing windows.

The mean velocity is

$$v = ((0.2539) + (1.826 \times 10^{-4})(h - 7.62 \times 10^{-4})^{-0.928}) \times (0.3443 P_e^{0.547}) \ , \tag{2.46}$$

where h is the thickness of the glass in metres, $h \geq 7.62 \times 10^{-4}$ and P_e must be in the range of 690 Pa to 689 kPa.

Chapter 3

Blast Wave Model

3.1 Determining a Modelling Approach

A blast wave model must consist of at least two parts: a model of the wave and a model or models of the objects which the wave interacts with. The wave model is responsible for developing, in a time based way, the forces associated with a blast wave. It generates all forces acting on the objects. The object model determines both how the objects are represented and how they can react to forces. A wave model will be described first.

3.1.1 Selection of a Wave Model

Before deciding upon a model, it must first be determined what quantities need to tracked for a visual simulation. The model sought is a dynamic model. All movement, breakage and deformation of objects occur as a result of the forces acting upon them. Force can be determined by the product of pressure and the area over which the pressure is applied or

$$F = PA \; . \tag{3.1}$$

In generating an animation, the timing of events must also be tracked. It is necessary to know when the blast wave strikes each object and for how long, as well as with what pressure. For this, the velocity of the blast wave is needed. This will allow its position to be tracked with time. Also, the period of the blast wave is required to determine the duration of the loading. This is the minimum set of quantities that must be tracked to calculate the loading on an object as a function of time.

In the previous chapter, two basic modelling paradigms were presented: the first involved mathematical simulation based upon equations derived from the Navier-Stokes relations; the second relied upon either a simpler set of formulas or pre-computed blast curves. A richer set of information is available from blast curves than from the simplified formulas. Both approaches are one dimensional and hence have certain inherent shortcomings.[1] Chiefly, one dimensional models cannot properly handle reflections or diffractions. Both of these are higher dimensional phenomena. Non-normal reflections will reflect at an angle other than the angle of incidence and hence will require more dimensions to be tracked. Diffraction will generate vortices that require three dimensional modelling. Once a wave has passed over an object, there is no clear method for determining how much the wave has been affected by the object. This draws into question how the wave will effect secondary objects which are located behind and partially or fully obscured by an object. This is the problem examined in Chapter 4.

The numerical simulation method is more costly than the blast curve approach. A complicated system of equations needs to be solved and quantities other than pressure, wave velocity and pulse period need to be computed. These quantities can include temperature, density, entropy and particle velocity[34]. For the blast curve approach, properties such as the blast wave speed, peak overpressure and positive phase period can be read directly from a pre-computed curve. This can be implemented efficiently using simple linear interpolation. The blast equations are also easy to solve, but they provide less accurate information and also do not provide all the information needed.

Numerical simulation allows an analysis of the entire explosive event, from the detonation regime to the blast wave. This can allow for changes in explosive composition to be modelled and can lead to more accurate results. For computer graphics, however, this is overkill. A computer graphics simulation is not concerned with the small differences associated with different explosives as long as an acceptable range of visual phenomena can be modelled. The blast curve approach outlined here uses TNT blast curves and is based on TNT equivalence. This will not model different explosives as accurately, but

[1]Strictly speaking, this is not true. Recent efforts in computational fluid dynamics extend the numerical simulations to higher dimensions. See, for example, [31, 20] This further increases the cost of these models.

modelling TNT alone should yield a sufficiently broad range of visual phenomena. Blast curves are available for different explosives [3], if this becomes a desirable modelling goal. Note that it is also possible to use numerical simulations to only model the blast wave portion of the explosive by specifying appropriate initial conditions at the border of the explosive. This will still involve relatively expensive calculations, however, and a method for determining initial conditions is necessary.

Due to both its cheaper cost and the relatively rich data it provides, the blast curve model will be used in this work. The two major weaknesses of the blast curve approach are shared by the numerical simulation and related to these being one dimensional approaches: it can not handle reflections well and it is difficult to determine the impact of blocking objects on the future strength of the wave.

The model is a one dimensional spherical or isotropic model where the blast wave is considered to be expanding evenly in all directions from the bomb located at its centre. This is not completely physically accurate, but is justifiable. "...[C]lose to most real blast sources, behaviour is usually non-spherical. Fortunately, asymmetries smooth out as the blast wave progresses, and 'far enough' from most sources, the wave will become a spherical wave."[4, p.141] Determining 'far enough' relies on analysis or experiment. Since this work does not need to guarantee physically accurate results, but only visually accurate ones, it will be assumed that objects are always 'far enough' and the spherical approximation will be used. The spherical approximation is generally used in structural engineering works[18, 4, 21, 34]. When the bomb is located on the ground, the ground reflection factor of 1.8 is used, as discussed previously.

This work is aiming at a certain sweet spot. Highly detailed computational models have been developed and used by explosive researchers[26]. These are very complicated and will necessarily be slower than a blast curve approach. Procedural kinematic approaches could be used for generating explosive effects, but without a physical basis, it will be difficult to develop a good method for consistently determining object velocities. Furthermore, the forces generated through a dynamic approach can be used in a more general way, for example to create material deformations. This work aims at the sweet spot between the kinematic approach and the rigorous Navier-Stokes approach. The goal is to obtain reasonable physical results at rates that are close enough to real-time to give animators good feedback. If such response times are obtained, the model can also be used in virtual environments, something that could not be done with either of the other two modelling approaches.

Dealing with Reflections

The blast wave is, as the name suggests, a wave. As such, it will reflect off objects when it strikes them. As discussed above, this is a difficult phenomenon to track in a one dimensional model. The reflection problem will be dealt with in a manner analogous to a single scattering event in light tranport. The initial impact of the blast wave will be modelled, including the peak reflected overpressure, but future bounces of the wave will be ignored. This is analogous to approaches often taken to modelling light propagation in gaseous phenomenon [36]. This approach is justifiable for explosion modelling. The primary blast wave will have by far the largest impact. If this maximum loading does not damage the structure, subsequent smaller loadings will also have no impact. If damage does occur, subsequent loading could also cause damage. Ignoring this secondary damage will lead to an underestimate in total damage, but this will likely be small for the following reasons. First, a reflected blast wave can be approximated as having half the peak pressure of the incident blast wave [34]. Second, the energy of blast waves decreases quite rapidly with distance. Third, a certain threshold of elastic resistance or friction must be exceeded before damage can occur. This energy is reduced from the total "damaging" energy of both reflected and primary waves, so a wave that is half as strong will have less than half the "damaging" energy. It should also be noted that no object can be struck by a reflected wave before previously being struck by the primary wave because a reflected wave cannot overtake the primary blast front. For scenes with reasonably simple geometry, the low albedo approach should produce damage estimates that are perhaps low, but still quite realistic. For tightly enclosed scene geometries, such as an explosive contained within a box, it would be necessary to track reflected waves. In a sealed environment, the pressure will also increase to a significant degree [34].

3.1.2 Geometry Model

Now that a wave model has been chosen, it must be decided how to represent the scene geometry. It is necessary to determine a model for the objects themselves and also to locate them within an environment. An object model will determine how the objects are rendered, how forces are calculated on the object and how the object reacts to forces. It is necessary to locate the objects within the environment in order to determine their location relative to the bomb. This will be used to calculate when they are loaded and with what pressure.

There are two main candidates for representing the geometry: volumetric representations or polygonal meshes. Volumes are discretely modelled as sampled voxel grids. Voxels are useful for determining spatial occupancy information efficiently and provide an easy to work with grid based model. Foster and Metaxas use a voxel model to define the environment in their work on modelling hot gases[16]. There are significant disadvantages to voxels, however. Being cubic, they do not model the true geometry of the objects. Reflected overpressure depends very strongly on the angle of incidence, so it is important to have an accurate representation of the geometry. Polygonal objects can be easily translated, rotated and rendered. They also do not require a grid structure to be maintained. For these reasons, a polygonal mesh approach will be developed.

In a polygonal mesh model, both the objects and the bomb are located in Euclidean 3-space. For simulation purposes, each object is divided into a set of panels and only the front panels need to be loaded. Panels are discussed below. The panels can be related to the bomb by calculating their radial distance from the bomb and the angle of incidence for a line travelling from the bomb to the panel centre. This provides all the information necessary for determining which panels in simple objects will be loaded and when the loading will occur. Chapter 4 discusses recovering a scene map from the geometry that can be used for modelling diffraction.

3.2 General Model Properties

3.2.1 Quasi-Static Modelling

Recall from Chapter 2 that quasi-static loading occurs when the wave period is long compared to the length of the object being loaded. This will always be the case for a strong explosion loading a small object. When an object experiences quasi-static loading, the object is completely engulfed in the blast wave. The static pressure acts everywhere on the object and hence can be cancelled out, leaving a net drag loading which acts on the front face of the object. This loading is a combination of static pressure and drag pressure as shown in Figure 2.16.

For our work, most simulations focus on the acceleration of small objects. Quasi-static loading is assumed. This allows loading calculations to be restricted to the front facing panels. The number of calculations that are needed is hence reduced.

The quasi-static assumption is well founded, but the wave model is not restricted to the quasi-static case. The wave model tracks reflected pressure, dynamic pressure and static pressure. It can be used to calculate impulsive loading, if for instance, the loading of a large, long building is to be modelled. It can also calculate the static pressure experienced over the entire model if object deformations are of interest. These modelling efforts would require a more complex geometrical model, such as a winged edge structure, that would allow the wave to be tracked over the surface of the object. The main use of such a model would be in determining object deformations. Since deformations were not within the scope of this work, such a model was not implemented.

3.3 User Control

The user has initial value control over the simulation. This control relates to both the bomb and the objects in the scene. The bomb can be located anywhere and its strength can be specified as a mass of TNT. More parameters are controllable for specifying object properties. These include the object's mass, its frictional resistance and whether it is anchored or free to move. The details of object specification are below in section 3.5.2.

3.4 Blast Curves

All the blast curves are indexed in terms of the scaled distance parameter Z. This allows them to be automatically scaled for different strength explosives. A scene distance must be divided by the cube root of the bomb's mass before it can be used to index a blast curve.

Figures 3.1 and 3.2 show the main blast parameters: static pressure, P_s; dynamic pressure q, scaled pulse period, $\frac{T}{W^{\frac{1}{3}}}$ and wave velocity U. These charts are used to define functions in the model that will return the value of a given parameter for a given Z. (The charts used to build these functions are of higher resolution than the ones shown here. They are plotted on full log paper with each decibel shown here subdivided into tenths and further subdivided into hundredths or fiftieths. See [4].) The charts are built by taking at least ten samples per decibel. More samples were taken for area of a curve that changed rapidly. The original plots are log-log, so values are computed by taking a linear interpolation of the points stored in the charts.

The reflection coefficient C_R is used to determine the initial impact of the wave. It models the effect of the increased particle density caused by a collision. (It does not relate to subsequent reflections of the wave.) The reflection coefficient is multiplied by the static pressure to determine the peak pressure felt by a frontal face. It is highly sensitive to both the static pressure and the angle of incidence.

Figure 3.3 shows the reflection coefficient as a function of both static pressure and angle of incidence. An angle of incidence of zero indicates a wave striking the face head on. The transition point in the middle of the chart corresponds to the mach stem transition. There is a small increase in reflected pressure for low static pressure near the mach stem transition. As the static pressure becomes higher, the mach stem transition marks a steep decrease in reflected pressure.

The blast curves are used to calculate the pressure pulse shown in Figure 2.16, which determines the loading an object experiences.

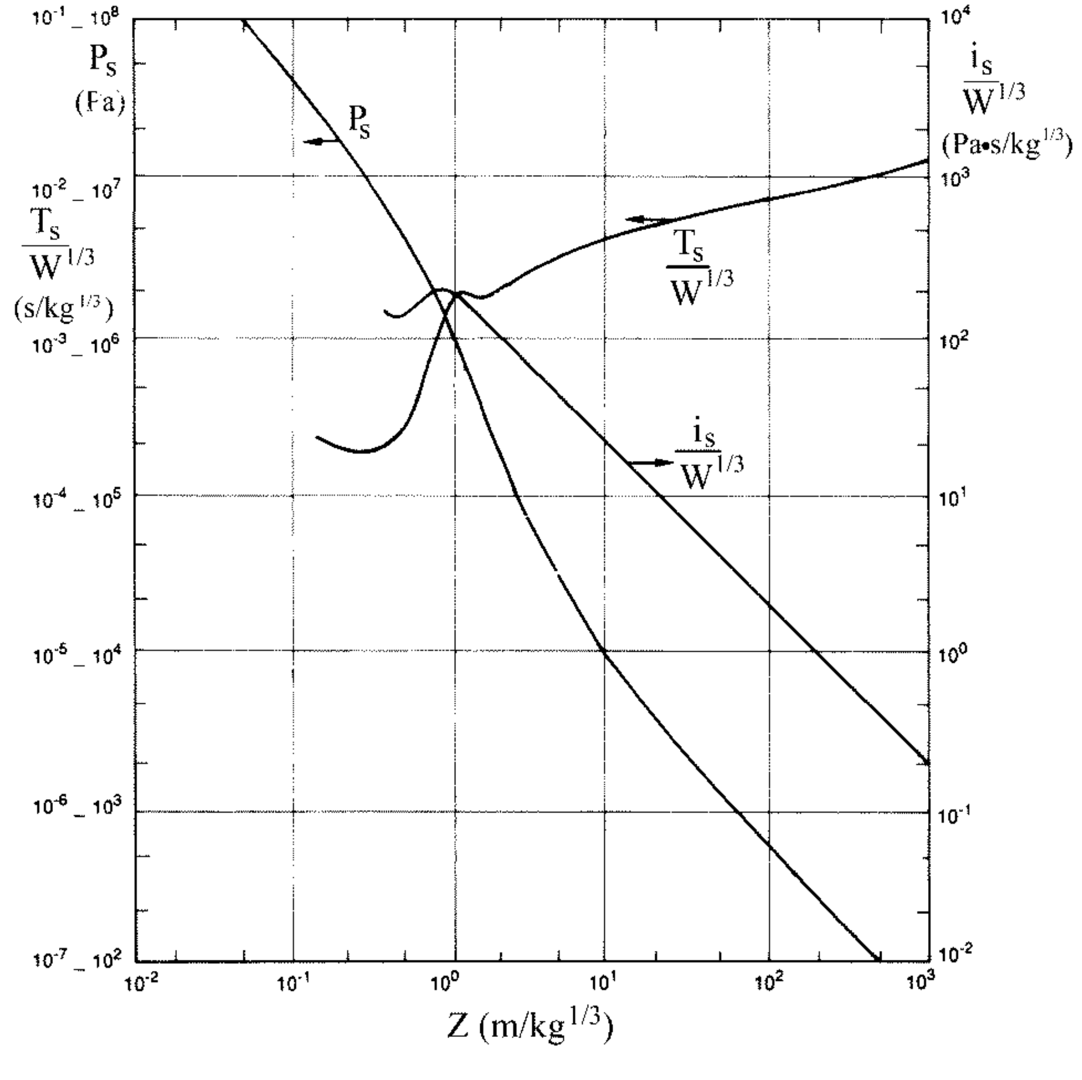


Figure 3.1: The blast curves for static overpressure(P_S) and scaled pulse period ($\frac{T_S}{W^{\frac{1}{3}}}$) as a function of scaled distance Z (from [34]). These curves are for shperical TNT charges, exploded in air at ambient conditions.

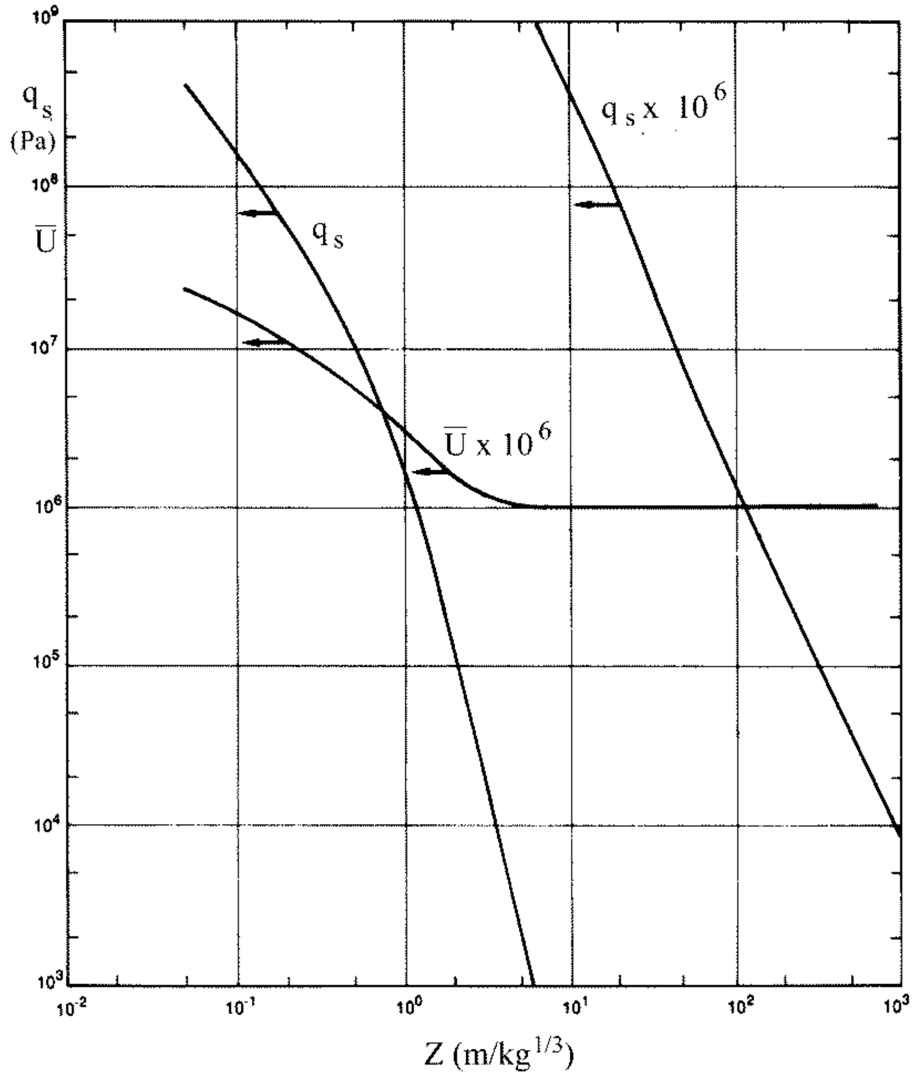


Figure 3.2: The blast curves for scaled wave velocity (U) and dynamic pressure q_s as a function of scaled distance Z(from [34]). These curves are for shperical TNT charges, exploded in air at ambient conditions.

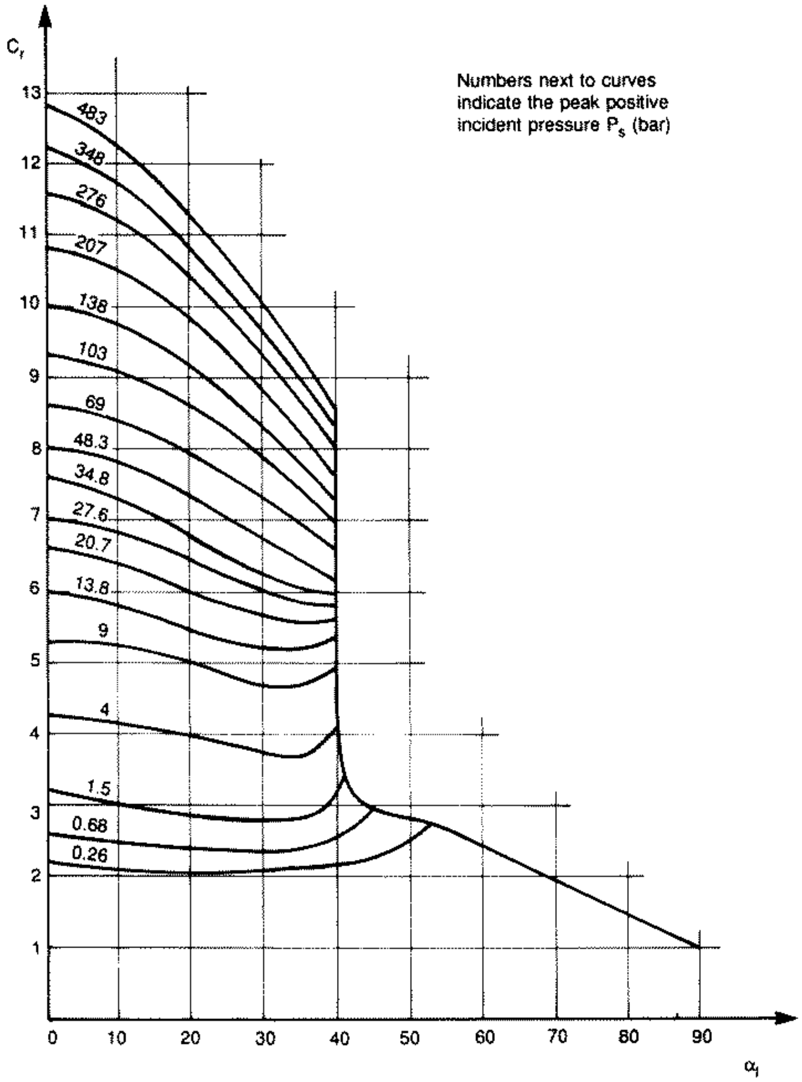


Figure 3.3: The reflection coefficient (C_r) as a function of both angle of incidence (α) and static overpressure (from [34]). Each curve corresponds to a specific static overpressure (bar) as labelled.

3.5 Elements of the Model

The model was implemented using an object oriented paradigm. The elements discussed in this section are all individual objects, implemented as C++ classes.

3.5.1 Panels

Panels are our basic modelling unit. The minimum set of data a panel must contain is: a centre, a normal, an area and a pointer to the object which contains it. Forces and torques are calculated as acting at the centre of each panel. The panel is considered to be planar and the centre is taken as being the centroid of the specified area. The normal is necessary for calculating the angle of incidence of the blast wave. In the base implementation, panels maintain more information about themselves, including their coordinates and radial distance from the bomb. Panels also must record the triangular pulse with which they are being loaded.

Forces and torques are calculated for the panel at each time step during its loading. These are then passed on to its object parent where they can be summed to determine the total loading experienced by the object during that time step.

In the base implementation, every panel corresponds to a polygon in the object. This is neither necessary nor even desirable. A panel is a simulation unit, not a geometric unit. For complex polygonal models, a panel may consist of several polygons. This allows the simulation to be carried out at a coarser resolution than that of the geometric representation. Due to the high sensitivity of loading to angle of incidence, it is an interesting research problem to see how coarse a resolution still gives accurate results. In cases where the bomb is close to a large, flat polygon, it may be desirable to divide the polygon into several panels to more accurately model the load distribution. When this model was developed, it was designed to work with multi-scale meshes as described in [44]. The panels could correspond to a coarser or finer level in the mesh hierarchy than is used to model the geometry. This will be discussed in the future work section.

3.5.2 Objects

Objects are individual items in a scene such as a car or a brick. As with panels, objects maintain a great deal of information about themselves. They have data on their velocity, angular velocity, mass, centre of mass, dimensions and frictional resistance. Objects also know whether they are anchored or free to move; whether or not they can act as blocking objects (Chapter 4); and whether or not they can rotate.

The object is also responsible for integrating the forces and torques acting on it. It does this by integrating Newton's equations of motion using Euler's method integration. This is a simple integration scheme which assumes constant acceleration over each time step, solving for velocity, and then using the average velocity to solve for displacement. The formulas are the basic Newtonian physical relations following from $F = ma$. The shortcoming of this approach is that it requires small time steps to handle the high accelerations.

Objects also calculate a damping term. This damping term is equal to the air resistance experienced by the object as it moves, given by

$$F = \frac{1}{2} C_D \rho v^2 \ . \tag{3.2}$$

This force acts in the direction opposite to the movement of the object.

The rotation of the object is determined by calculating the torque acting on it. Torque can be related to angular acceleration and velocity by the following formula:

$$\sum \tau = I\dot{\omega} + \omega \times I\omega \ , \tag{3.3}$$

where ω is the angular velocity, τ is the torque and I is the inertial tensor. The inertial tensor characterizes how the object spins. It is normally defined in a base reference frame for a given orientation of an object. The current inertial tensor (I) can be calculated from the base (I_b) using the following relation:

$$I = R I_b R^T \ , \tag{3.4}$$

where R is the rotation matrix corresponding to the current orientation of the object. Equation 3.3 can be solved for the angular acceleration and this can be integrated to determine the new angular velocity and rotation. Note that these are all vector quantities.

3.5.3 The Simulator

The simulator controls the running of the simulation. It calculates the current state of the blast wave. For each time step, the blast radius is increased. This is done using a prediction-correction solution scheme. The velocity of the blast wave at its current radius is determined. This is multiplied by the time step to determine a new blast radius. The slower wave velocity at this new radius is calculated. The new blast radius is then recalculated using the average of the first velocity and the new velocity. This new estimate is accurate in practice, but if desired, the method could be repeated until the estimate was within a given error tolerance [4].

During a preprocessing stage, all the panels which are loading targets (those which are front facing) are loaded into an event queue. The event queue is sorted in order of the panel's radial distance. As the blast radius increases, it will pass over the panels in the event queue. When a panel is first struck, its pressure pulse is calculated. With each future time step, this pulse is integrated until the panel has completed loading. At this point the panel is removed from the queue. This is the acceleration phase for the object. During each time step, the panel passes its load to the owning object where the torques and forces contributed by all the panels are summed and integrated.

When an object has no panels in the event queue, it is free flying. Free flying objects are acted upon by the downward force of gravity and the opposing force of wind resistance. Free flying objects are advanced at each time step and the forces acting upon them are integrated to determine a new velocity.

In essence, an impulse is being applied to each panel. This impulse is integrated in a step wise fashion in order to capture the acceleration phase of an object's motion. The acceleration phase is important in creating convincing slow motion animations.

The pulse being applied is a pressure vs. time pulse, not a spatial pulse. It is set when the wave first strikes a given panel, and does not change after this. This approach is completely accurate for objects that are stationary or move very little with respect to the wave front. For fast moving objects, it still produces visually reasonable results, although it is likely less accurate. This is because the frame of reference for the pulse is a stationary bomb centre, not a moving object. If the object is moving, it will take longer for the pressure pulse to pass over the object and the shape of the pulse may change.

Not enough information is available to propagate a spatial pulse. The location of the pulse's front border is known because the wave front velocity is known. The only measure of the length of the pulse, however, is its period. This does not yield a spatial location for the rear border because the rear border's velocity is slower than the front's velocity and is unknown. The velocity could be considered to vary according to some function over the length of the pulse. Unfortunately, I have not found a technique for determining this function.

One possible technique for trying to account for high speed object movement is to scale the time step during each integration stage. An object which is moving at high speeds would be passed by less of the wave during a given time step. To account for this, a shorter time step could be used which is based upon the difference in velocity of the object and the wave. Some work was done in this direction, but this too relies on unavailable information about the time velocity gradient of the pulse. A function determining the wave velocity over the length of the period is again needed.

Applying an impulse rather than a spatially based pulse appears to be the most reasonable approach. Good results were obtained using this method, such as the shattering window described in Chapter 6. Furthermore, it is common to use the impulse directly to determine object velocities in the explosives literature [4]. The potential shortcoming of this approach is that it may lead to an overestimate of the force received for small, fast moving objects. It is reassuring to note that this concern was not raised in any of the texts consulted.

The simulation and rendering time steps can be set independently. This allows the user to control the frame rate and produce images every few simulation steps as desired.

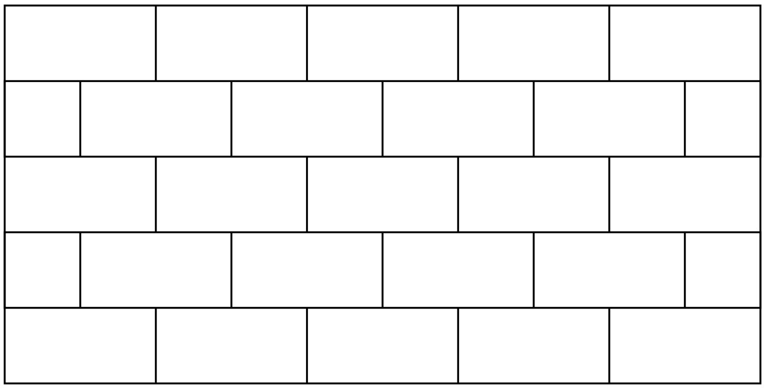


Figure 3.4: The brick pattern used for the brick wall.

3.6 Blowing Up a Brick Wall

The model is demonstrated by using it to blow apart a brick wall. A bomb is located behind a wall of 27 large bricks. Frictional forces acting between the bricks are the only resistance forces modelled. Each side of the brick is divided into four equal triangles along the two diagonals of the face. These triangles serve as the panels used in the simulation.

The brick pattern is shown in Figure 3.4. The large bricks each have a mass of 10kg. The half size bricks weigh 5kg. The bomb has a mass of 100 kg and is located 15 metres behind the wall. It is horizontally centred and slightly higher than the base of the wall. A front view of the animation is shown in Figure 3.5 and a view from below in Figure 3.6. The animation generates 40 frames in about 3 seconds running on a dual PentiumII 300 MHz Linux workstation.

The frictional forces acting between the bricks act to resist the damage of the blast wave. When a strong bomb is used, as in the above example, friction has a negligible impact on the simulation. This remains an avenue of animator control, however. By specifying unrealistically high friction coefficients, a wall could be generated that could partially resist the blast wave. Depending on the application, an animator could freely choose bomb strengths and friction coefficients to yield a desired visual effect without needing to worry about the physical accuracy of the parameters. Such an effect is shown in Figure 3.7. This is a single frame from part way through an animation. A smaller bomb is used with the same brick wall and the intensity of friction is increased. With the bomb located behind the middle of the wall, the central bricks receive the most loading.

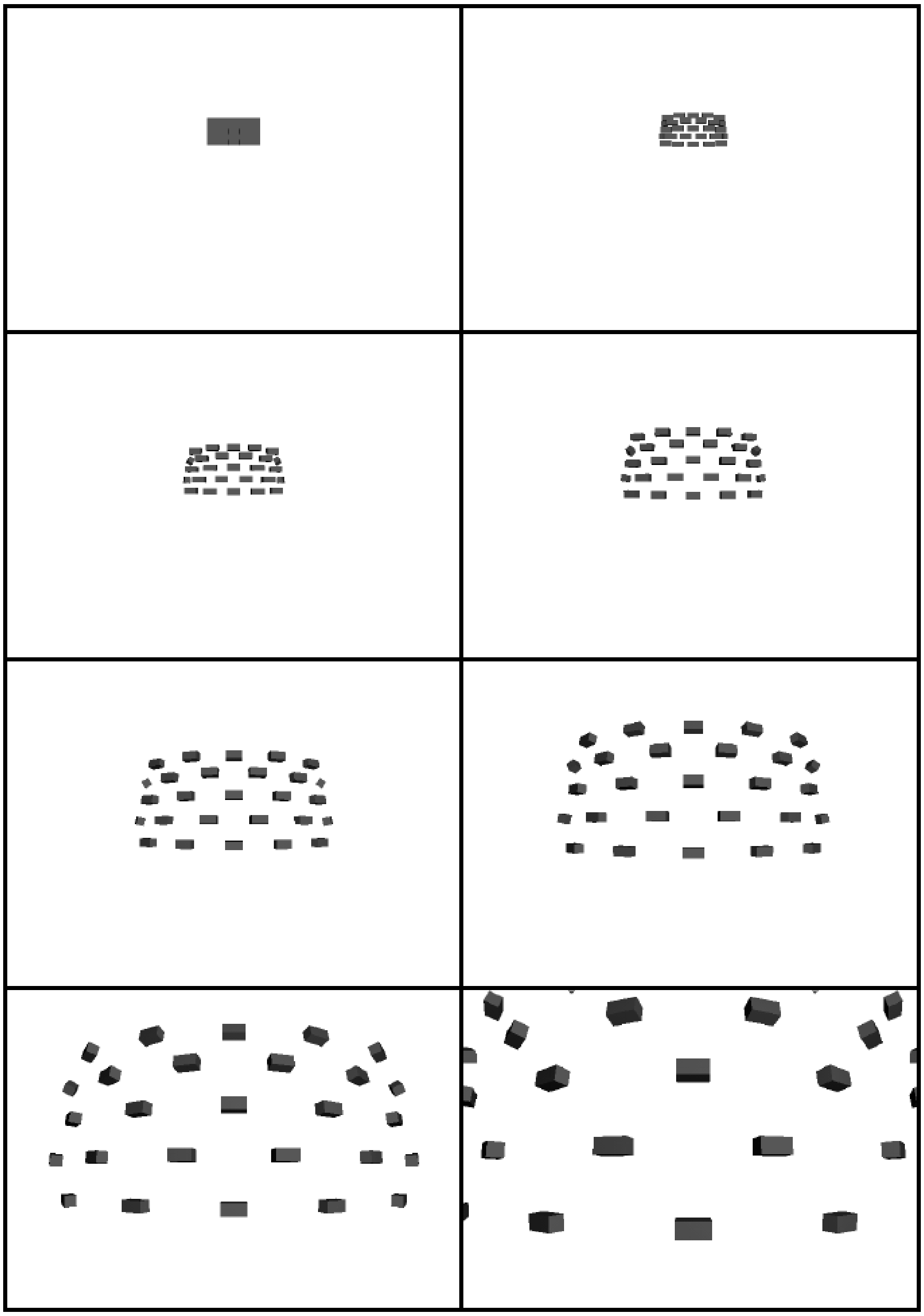


Figure 3.5: Eight equally time spaced frames from an animation of a brick wall blowing apart. Front view. Images are ordered from left to right.

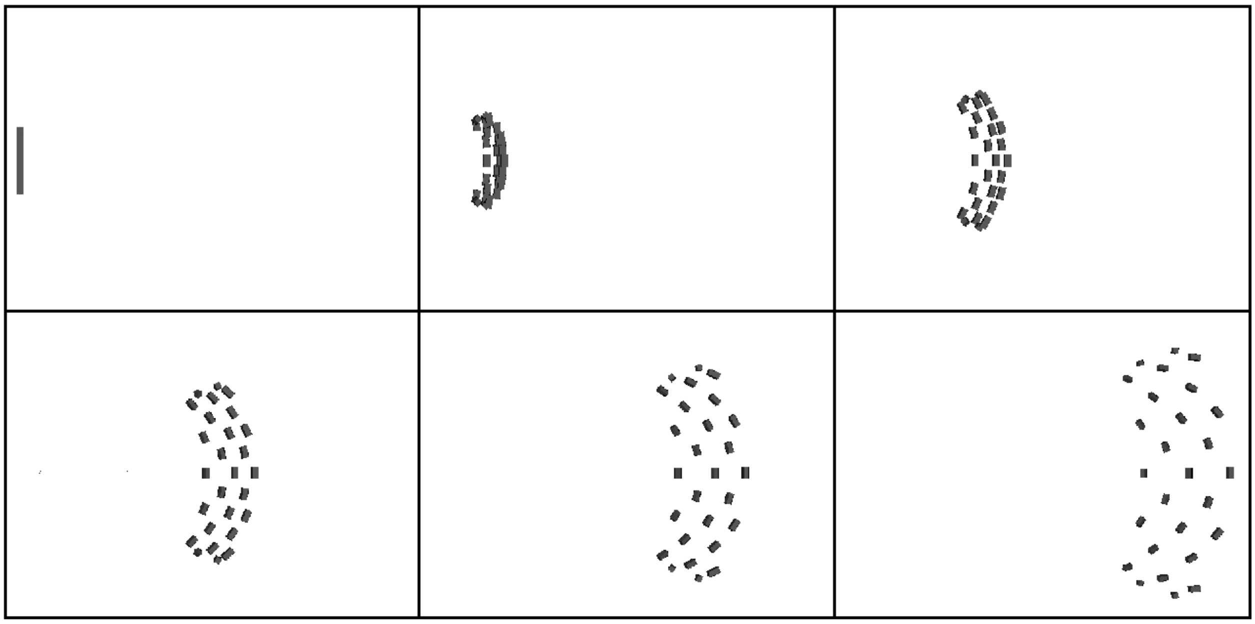


Figure 3.6: Six equally time spaced frames from an animation of a brick wall blowing apart. View point is below the wall, looking up. Images are ordered from left to right.

Bricks high on the wall experience less friction and are blown off. Lower bricks have more resistance and are not moved. Some of the bricks at the side also manage to stay in place because they are further from the explosion.

Friction is used here to build a very simplistic model of a brick wall. This is done to illustrate how material models can be used in conjunction with the explosion model to generate different effects. It is not meant to represent an accurate model of a brick wall. Within civil engineering, the field of building science has undertaken substantial research on modelling structural failure. An introduction to fracture mechanics is given in Chapter 5, but more detailed structural modelling is beyond the scope of this work. The important point is that this model can be combined with constitutive models in order to simulate the effects of structural failure.

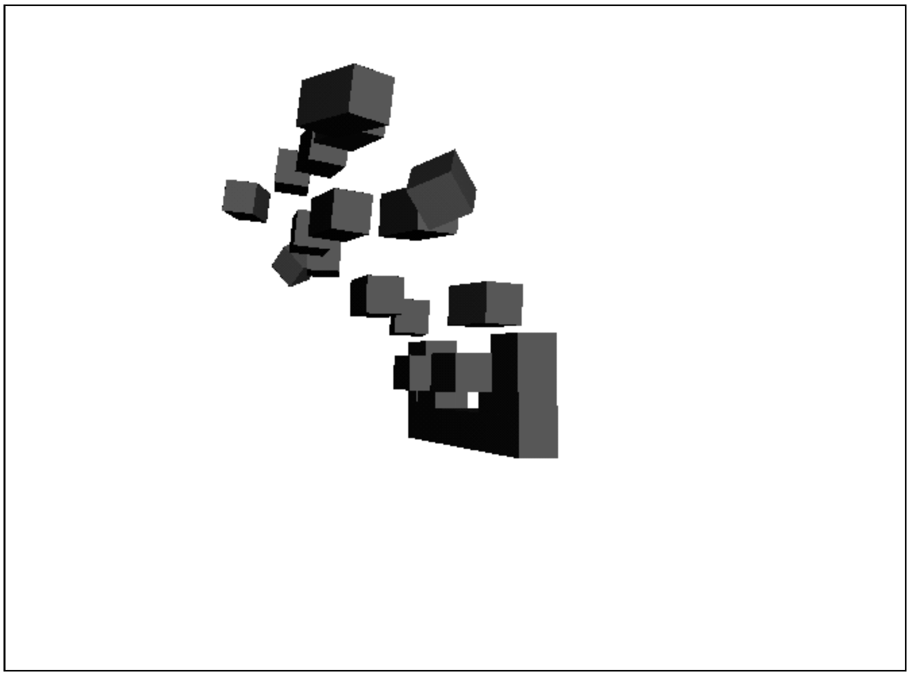


Figure 3.7: A single frame from an animation of a brick wall blowing apart. The friction has been increased in the wall and a weaker bomb is used, so the lower portion of the wall is able to withstand the explosion.

Chapter 4

A Heuristic Propagation Model

4.1 Introduction

The explosion model developed thus far will predict pressures on a given object independently of all other objects in the scene. That is, the forces on each object are calculated as if that object is the only object in the scene; objects cannot offer protection for objects which lie behind them. The model this work is based upon comes from structural engineering where the main goal is determining whether a given structure will withstand a given explosion. In this work, there is normally only one object of interest, so the approach is appropriate. Our visual model will often be concerned with scenes which combine many objects. This requires us to consider the effect objects have on wave propagation. A propagation model determines the pressures resulting on a given object as a result of both the object's location and the scene geometry that lies between the object and the bomb.

A blast wave can interact with an object in some combination of three ways: it can reflect off an object, it can propel or deform an object or it can diffract around an object. As discussed in Chapter 3, wave reflections will not be tracked in this model. Object propulsion and fracture are the subjects of chapters 3 and 5 respectively. The diffraction of a blast wave around objects will have a significant visual impact and will be the major subject of this chapter.

Whereas reflection can be ignored to a reasonable approximation, in even moderately complicated environments, diffraction cannot. Imagine a bomb exploding in an empty underground parking garage. Your chances of avoiding injury will be dramatically increased if you are behind a large pillar than if you are standing in the open. Blocking objects reduce the impact of the blast wave on the objects they obscure. This is not a strict binary relationship where exposed objects get full loading and obscured objects remain unloaded. Smith and Hetherington write in discussing the effects of blast walls: "If the blast wall is too far from the structure, the blast wave from the attack will reform behind the wall and could produce a significant load on the structure. If the attack is located at too great a distance from the wall, there will be little energy absorption by the wall itself through wall deformation."[34, p.273] The blast wave will diffract around objects, reforming behind them.

Modelling diffraction phenomena is very complicated. It is an area of active research in the shock wave community (for example, see [1, 22]). Accurate modelling requires very complicated numerical simulations and is still not a solved problem. When diffraction occurs, vortices are created near corners in the objects. Vortices are three dimensional phenomena and hence a three dimensional model is required to track them. Often, experimental results are used in modelling because the theory is incomplete. The diffraction around a given object can be determined through the use of blast tunnels[5]. These are similar devices to wind tunnels, except an explosive blast wave is propagated down the tunnel instead of a strong wind. Such results are unfortunately of little use for determining diffractions in a general visual model, but they might be useful in testing a general model.

It should be clear that the accurate modelling of diffraction phenomenon is well beyond the scope of this work. What is needed is a rapid way of estimating the impact of blocking objects on the propagation of the wave. A heuristic model for accomplishing this is developed next.

4.2 Propagation Model

As was illustrated by the quotation from Smith and Hetherington, it is important to determine not just whether or not an object is obscured by another object, but how obscured the object is. This is shown in Figures 4.1 and 4.2 below. Figure 4.1 shows two scene geometries. In both, block A is the same distance from the explosive and block

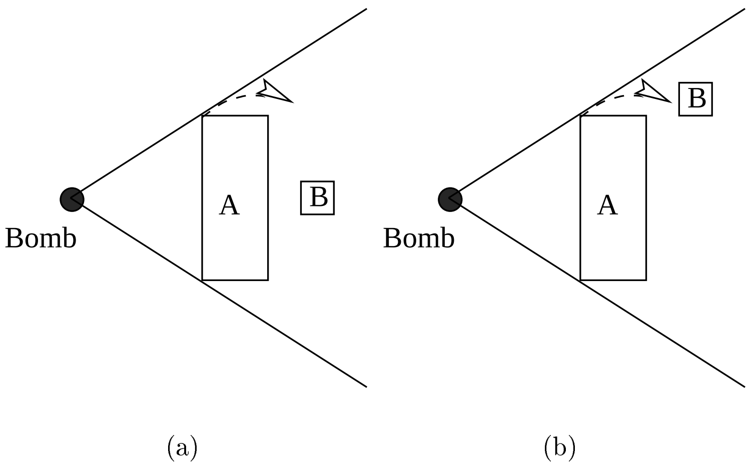
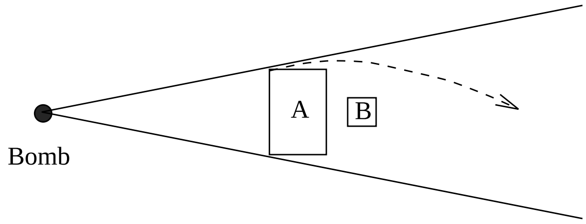


Figure 4.1: Loading of a well obscured (a) and less obscured (b) target.

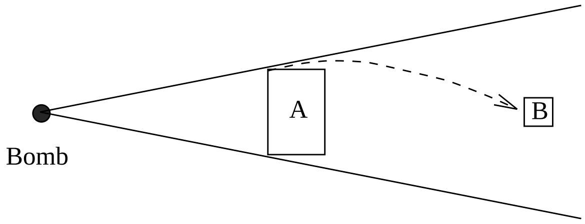
B has approximately the same seperation from block A. Block B is completely obscured in both instances. In the left figure, block B is well obscured and is unlikely to receive any loading. In the right figure, however, the object is almost exposed. The wave has time to diffract around A and create some loading on B, as shown with the dashed line. This is due to the short distance the wave must diffract. Despite both object B's being the same distance behind object A, only the second object B will receive loading because it is closer to the edge of object A.

Figure 4.2 shows another example where differential loading is an issue. Again, the location of A and the bomb are the same for both sides of the figure. B is located behind the middle of A in both diagrams, but is further from A on the right. For Figure 4.2 a), one would expect block B to receive no loading from the blast wave. For Figure 4.2 b), however, the blast wave has time to diffract around block A and reform behind it, loading block B. In this case, B is centred behind A on both sides of the diagram. The difference is how close B is to A.

Three parameters are crucial in determining the loading received by block B. First, the solid angle blocked by A (blockage ratio) relative to the bomb. This takes into account the size of A and how close it is to the bomb. A larger blockage ratio will produce more shielding. The second parameter is the separation between A and B (the pitch). This is illustrated in Figure 4.2. The greater the separation, the less effective A will be at



(a)



(b)

Figure 4.2: Loading of a well obscured (a) and a less obscured (b) target.

reducing B's load. The third consideration is how completely A blocks B. This is the case shown in Figure 4.1. It refers to how well an object is hidden behind another object. The closer the hidden object is to the obscuring object's edge, the more likely it is to experience loading.

The model developed attempts to determine these parameters and use them to calculate the loading on blocked objects such as B. There is some precedence for this kind of model in the explosives literature. For instance, Baker et. al. describe a model based on blockage ratio and pitch that is used to predict flame speeds in vapour cloud explosions[3].

It should be noted that the algorithm developed here determines the amount of loading an object experiences as a percentage of the amount of loading it would receive if it was completely unoccluded. It does not model differential loading–the fact that an object may receive more loading on one side than another. This would require more detailed knowledge of how the wave reforms after diffraction.

4.2.1 Basic Idea

A bomb can be viewed as being analogous to an ideal light source with one very important exception discussed below. A bomb propagates equally in all directions from a central point. The problem of discovering which objects are lit by the light source is equivalent to the problem of deciding which objects can be seen from a viewpoint located at the light source. This is the very familiar computer graphics problem of depth culling. It is normally solved by using a depth buffer or z-buffer [1] This idea is the basis of the heuristic

[1] The details of the z-buffer are available in standard graphics texts such as [15].

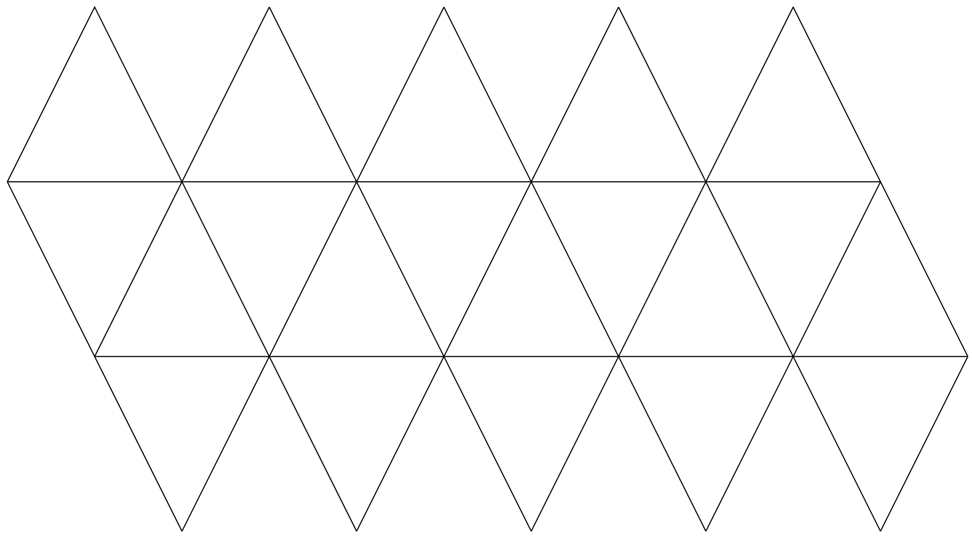


Figure 4.3: A planar fold out of an icosahedron.

algorithm. The important exception is that blast waves can reform behind an object, causing loading on an object that is completely obscured. A propogation algorithm must go beyond the basic occlusion testing of lighting propogation to account for wave deformation.

Normally, projections are used to calculate which objects appear in a single view plane–a plane perpendicular to the direction the camera is currently pointing in. Due to the fact that a blast wave expands equally in all directions, a projection map of the scene onto a sphere surrounding the bomb is needed. The projection needs to be onto a sphere because calculations will be based upon the area objects occupy in the projection. Projecting onto the view plane is a standard part of the graphics pipeline, so it can often be computed very quickly in hardware. In order to take advantage of this, the sphere must be discretized into planar facets. This is done by surrounding the bomb with an icosahedron.

An icosahedron is a 20 sided regular polyhedron with triangular faces. It was chosen because it gives a reasonable approximation of a sphere and it is easy to work with. Triangle faces can be dealt with efficiently and the icosahedron has a nice mapping into the plane as shown in Figure 4.3. In our implementation, each equilateral triangle has a base of 32 pixels and a height of 28 pixels.

Each object is given a distinct colour before projecting it. This allows the object to be identified in the projection. A direct application of z-buffering, however, yields a very impoverished set of information. If an object is completely blocked by other objects, there is no record of it after projection. If there is an ordering of several objects which are occluded by a front object, only the front object will be recorded in the buffer. For the propagation algorithm, the relative spacing of all objects must be known to properly determine each objects loading. Furthermore, if an object is partially occluded, the z buffer gives no information about how much of it may be hidden behind another object. This information is also necessary for determining its loading. For these reasons, each object must be projected separately and the relevant information recorded.

4.2.2 Building the Data

The first stage in the algorithm is to gather information on how the objects are located in the scene relative to the bomb. This forms a type of environment map. An object needs to know which objects block it, and how much the objects block it. An occlusion list contains all the objects which occlude a given object. The amount of occlusion can be calculated by storing their projections and comparing them.

In the pseudo code and discussion of the next two sections, *triangle* refers to a triangle in the icosahedron and *object* refers to an object in the scene. The basic algorithm for building the scene map is as follows:

```
Information Gathering Stage:
For each triangle do :
   For each object do :
      PROJECT object onto triangle's plane
      IF object projection intersects triangle
          STORE information
   END (for each object)

   BUILD object occlusion lists using stored information
END (for each triangle)
```

Three pieces of information are stored during the *store information* phase: a bit mask of the object projection's occupancy within the triangle; a temporary copy of the depth buffer for the triangular region; and finally, the triangle in which the object's projection falls is added to the object's data.

Building the object occlusion list is slightly more complicated. Each object has a list of all objects which obscure it. This list is built here. The *store information* procedure built a list of all the objects which project to this triangle. The bit masks of all of these objects are compared to see if they overlap. If two bit maps overlap, one object is occluding the other. By comparing depth buffers, it can be determined which object is behind and which in front. The blocked object adds the occluding object to its occlusion list. The comparison is performed in a bubble sort like manner. The first object is compared to every subsequent object to see if there are any occlusions. The second object is then compared to every object with higher index and so on. If an object is occluding another object in more than one triangle, it will only be recorded in the occlusion list once. The depth buffer information can be destroyed after the comparisons have been completed.

Once the information for the scene map has been built, it can be used to determine the attenuation in the second phase of the algorithm. This is shown in the following pseudocode and explanation.

```
Determine Attenuation:
For each object DO:
   For each occluding object DO:
      DRAW obscuring object bit mask onto the icosahedron map
   END (for each occluding object)
   ADD DISTANCE values to the map
   READ the now formed occlusion map
   CALCULATE the attenuation factor
   STORE attenuation factor with object
END (for each object)
```

The occluding objects are available from each object's occlusion list. The algorithm uses an icosahedral map which is a planar foldout as shown in Figure 4.3. Again, each triangle in the icosahedron map is 32 pixels wide by 28 pixels high. The map is initialized with zeroes. The stored bit mask for each object which is occluding the current object is drawn into the icosahedron map. This is the *DRAW* operation. When the mask is initially transferred, pixels are marked with minus ones.

Once all the masks have been drawn on the map, distances are added. Recall that it is necessary to know how obscured an object is, not just whether or not it is obscured. To do this, we must determine how far a blocked object is from the edges of the blocking

```
00000000000000000000
00000000110000000000
00000001221000000000
00000012332110000000
00000012343221000000
00000123454332110000
00000123455443221000
00001234566554332100
00001234567654321000
00012345666543210000
00123344556543210000
00112233445432100000
00001122334432100000
00000011223321000000
00000000112221000000
00000000001110000000
00000000000000000000
```

Figure 4.4: A projection of an object shown with distance values added to give an approximate measure of how far a pixel is from the object's edge.

objects. This is facilitated by numbering the pixels of the obscuring objects with values that increase with their proximity to the object centre. The pixels on the perimeter are numbered one, the pixels immediately inside these are numbered two and so on. This is shown in Figure 4.4. This is the *ADD DISTANCE* phase of the algorithm.

When the distances have been added, the occlusion map can be read. For each location in the occluded object's bit mask, the corresponding location in the icosahedron map is read and added to a sum. This is the *READ* operation. After reading is complete, the sum is divided by the number of bits in the occluded object bit map. This gives a measure, termed the blocking factor, of how well obscured the object is. Objects that have a large blocking ratio (span a large solid angle) will contain larger values at their centres. This gives them more blocking power. Furthermore, objects which are near the edge of an obscuring object will have lower blocking averages than if they were at its centre. Therefore, the blocking factor takes into account both the blocking ratio of the obscuring objects and how well the obscured object is hidden behind them.

The blocking factor can be used in conjunction with the radial separation of the objects to determine an attenuation factor. Radial seperation is expressed as the distance betweend the blocked and blocking object divided by total distance of the blocked object from the bomb. This is the *CALCULATE* operation. The basic explosive model requires the calculation of the radial distance of all object panels. From this an average front and back distance is calculated. The separation distance is taken as the distance from the rear of the closest obscuring object to the front of the obscured object. This will not be completely accurate for some geometries, but should perform well for generally chunky geometry. Other approaches considered included computing an average blocking distance based on how much each object contributes to the obscuring. This would significantly increase the complexity of the algorithm and did not seem warranted.

The blocking factor and radial separation need to be combined in some meaningful way to produce an attenuation factor. To do this, the following function of the two parameters was developed:

$$f(b,k) = \left(\frac{b}{1+7k}\right)^3 - 2\left(\frac{b}{1+7k}\right)^{1.5} + 1 \text{ for } b < 1+7k \tag{4.1}$$

f equals zero for larger b. The radial distance is k and b is the blocking factor. This function is shown if Figure 4.5. It is loosely based upon one of the basis functions of the Hermite curve, but is modified so that one edge, $b = 0$, is set to one and the other edge, is tied to the line $b = 1 + 7k$ and set to zero. (The second term is modified as well to give a more reasonable shape.) The constants 1 and 7 in the equation are somewhat arbitrary, but have given good results in practice. For different simulations, these parameters can be adjusted to give different results. In general, as b and k increase, the amount of attenuation decreases.

Once the attenuation factors have been determined, they are set for each object. The factor, between zero and one, determines what proportion of normal loading the object receives. These factors only need to be calculated once in a preprocessing stage because the blast wave should move faster than obscuring objects in the scene.

4.3 Testing

The effectiveness of the algorithm can be demonstrated with the test cases shown in Figures 4.6 and 4.7. These tests demonstrate the basic properties of the algorithm. In the future, more complex, "real world" tests should be conducted.

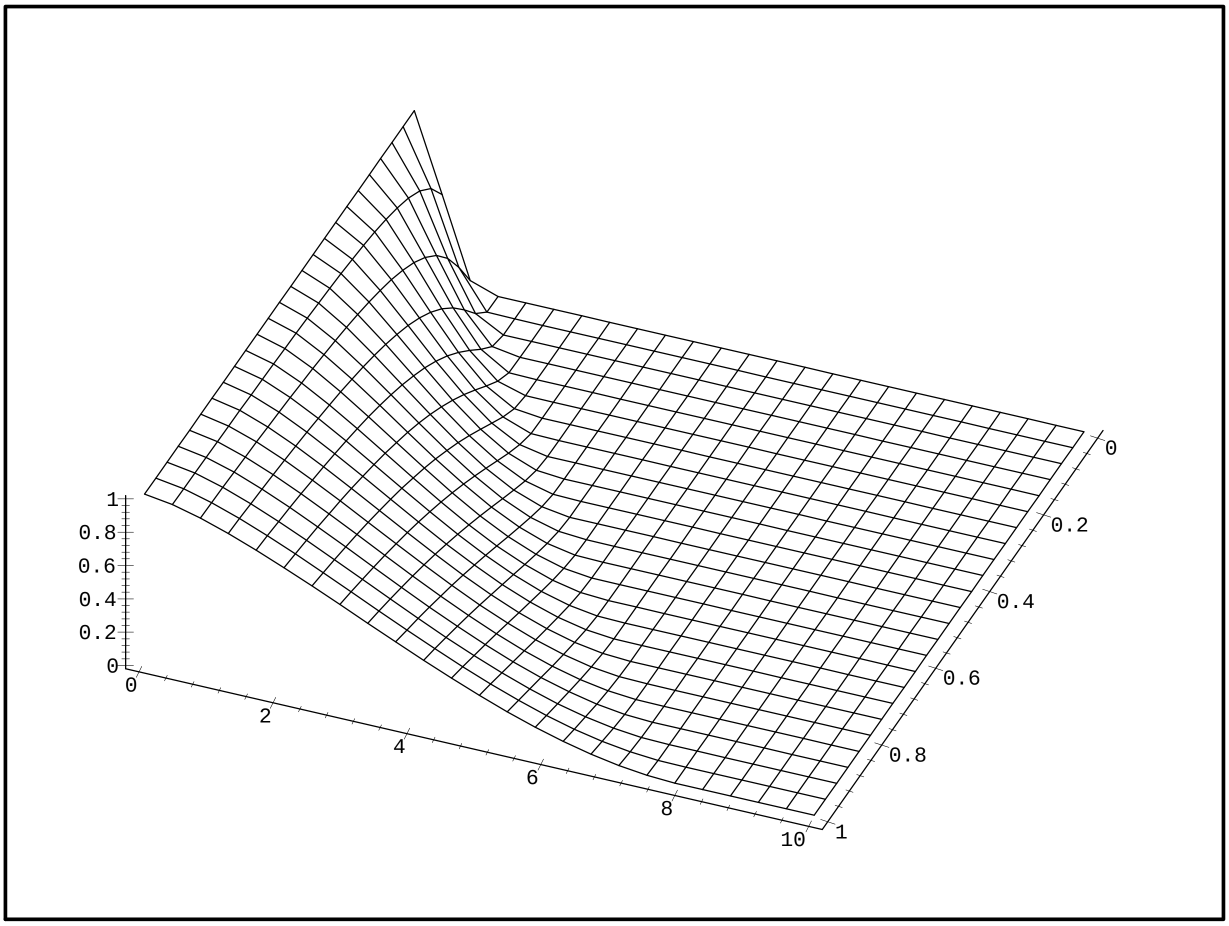


Figure 4.5: Attenuation as given by blocking factor (y) and radial separation (x).

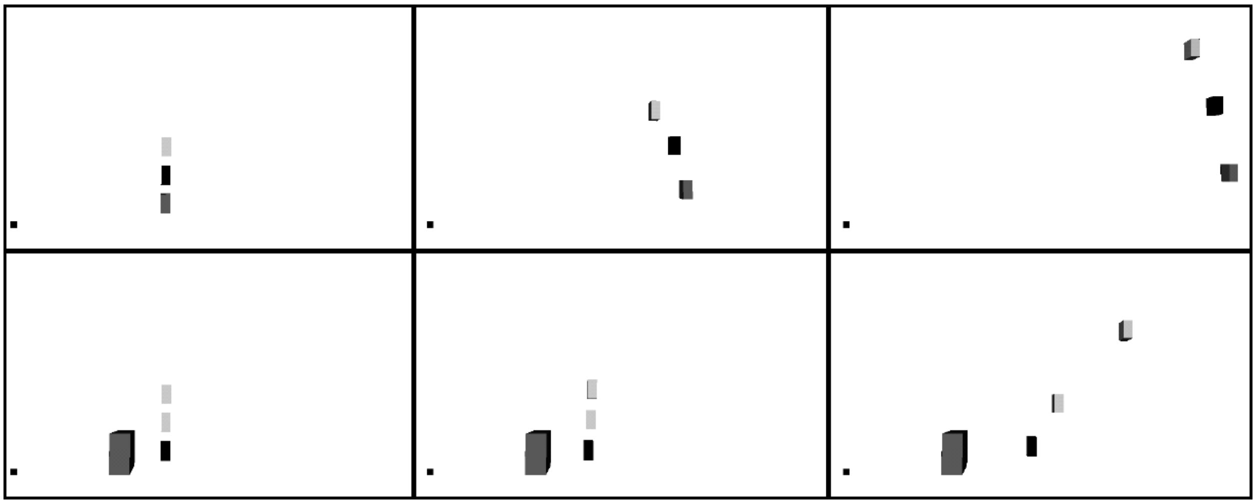


Figure 4.6: Propagation testing for blocks with similar radial distances from the bomb. The bomb is the small black square in the lower left-hand corner. The top row shows three equally spaced frames from an animation where the blocks are unobscured. The closest block receives the most loading. The bottom row shows three frames taken at the same times for an animation which features a large occluding object. The block which is closest to being exposed receives the most loading.

In both figures, the bomb is represented by the small black square in the lower left corner of the frame. Figure 4.6 shows three frames from two different animation sequences. The top row shows the acceleration of three blocks with no blocking object and the bottom shows the acceleration of the same three blocks with a large occluding object. The frames are evenly time spaced and the spacing is the same for each series. Without an occluding block, the resulting object pattern is the same as was seen with the brick wall. The objects which are closest to the bomb receive the most acceleration. When an occluding object is introduced, the more exposed objects are accelerated much more rapidly than those which are behind cover. Indeed, the object closest to the bomb travels the shortest distance because it receives the most cover. This is the desired result.

Figure 4.7 shows the effect distance from the blocking object has on the attenuation of the blast wave. The animations are organized in the same manner as Figure 4.6; the top row shows three frames from an animation without blocking and the bottom row shows three equally spaced frames from an animation with blocking. Note that for illustrative purposes, blocking has been turned off for the small blocks, so only the large block is behaving like an occluder.

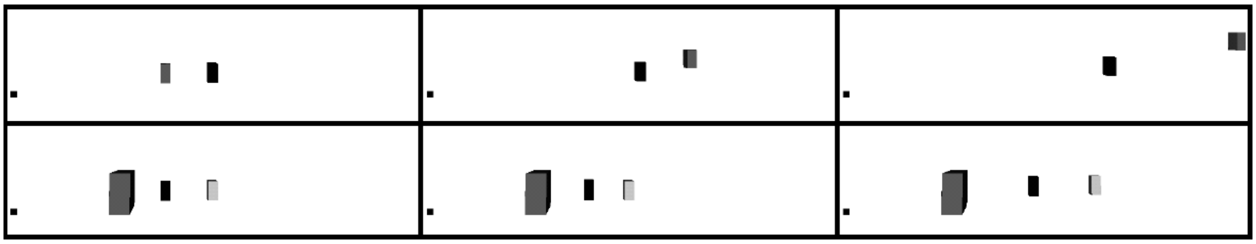


Figure 4.7: Propagation testing for blocks with varied radial distances from the bomb. The bomb is the small black square in the lower left-hand corner. The top row shows three equally spaced frames from an animation where the blocks are unobscured. The closest block receives the most loading, and actually passes the more distant block. The bottom row shows three frames taken at the same times for an animation which features a large occluding object. The near block receives more protection, and hence recieves less loading than the far block.

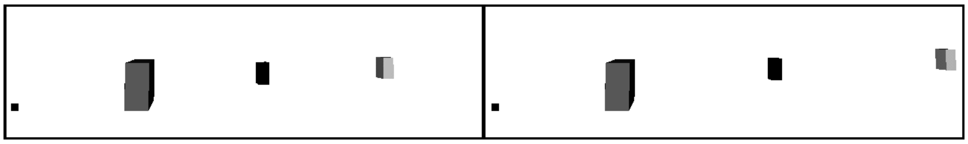


Figure 4.8: These are two extra frames from the bottom row of Figure 4.7. They are the next two frames in the sequence and show the further progress of the two blocks.

The black block is much closer to the bomb than the grey block. It therefore receives much greater loading than the grey block and is accelerated to a much higher speed. This is shown in the top row. In the second row, the black object is closer to the occluding object. This gives it greater protection from the blast wave. Due to this, the grey block receives more loading and is accelerated more rapidly. Two additional frames are shown from the second animation in Figure 4.8 to further illustrate this point. Again, this is the desired effect of the algorithm.

Chapter 5

Modelling Fracture

5.1 Introduction

Much of the excitement caused by blast waves is due to their ability to shatter objects. Modelling the fracturing process is a complicated task and will be explored in this chapter. First, the physical causes of fracture will be explained. Second, previous computer graphics attempts at representing fracture are discussed. Finally, a new physically inspired fracture model is described.

5.2 The Physical Basis of Fracture

5.2.1 Background Definitions

Stress

Imagine a block sitting on a rigid surface. A force acts directly down on the block, forcing it against the surface, which does not move, generating a balancing force. This is shown in Figure 5.1. The squeezed block is said to be in a state of *stress*[2]. Quantitatively, stress is defined as

$$\sigma = \frac{F}{A} \ . \tag{5.1}$$

Notice the equivalence of pressure and stress (recall that $F = PA$). By convention, stresses are positive when they pull and pressures are positive when they push.

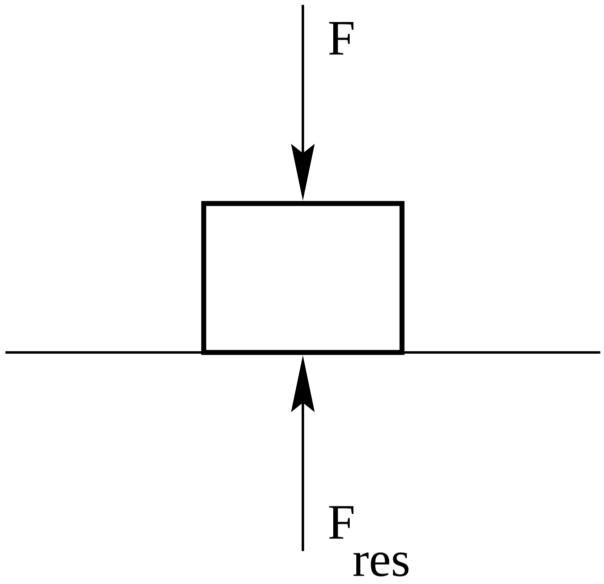


Figure 5.1: A block sitting on a rigid surface has a force acting down on it which is resisted by the surface. The block is experiencing *stress*.

Strain

When an object is stressed, it reacts to that stress by straining. *Strain* is a measure of how much the object bends in response to the applied stress. For a cube of side length one which extends a total length u under tensile stress, tensile strain can be defined as:

$$\epsilon_n = \frac{u}{l} \tag{5.2}$$

By a similar analysis, shear and lateral strains can be defined.

Elastic Behaviour, Plastic Behaviour and Breakage

For many materials, when strains are small, they are very nearly proportional to the stress. For simple tension, this relationship is given by:

$$\sigma = E\epsilon_n \tag{5.3}$$

where E is Young's modulus, a property of the material. Young's modulus is 1000 GNm^{-2} for diamond (G denotes the modifier giga) and 2-4 GNm^{-2} for nylon. This indicates the expected result that nylon will strain much more for a given stress than diamond. The stress-strain behaviour discussed here is equivalent to an ideal Hookean

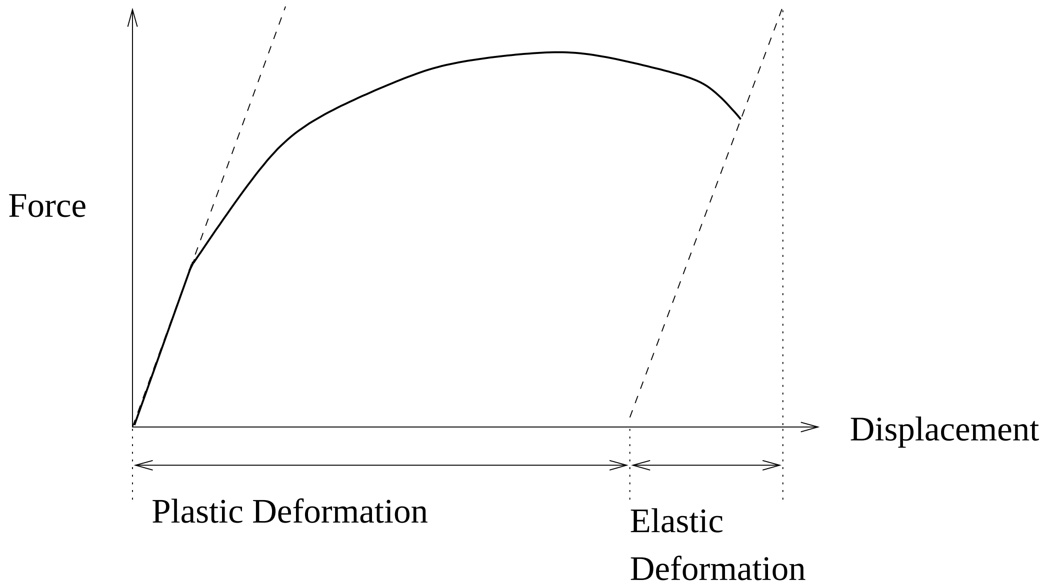


Figure 5.2: A force displacement curve for a material just below its yield point.

spring, where displacement is proportional to applied force (Hooke's Law: $F = kx$). This kind of behaviour characterizes the elastic behaviour of the material and is only valid for small strains.

All materials have a plastic limit and if the material's deformation exceeds this, its behaviour changes. Highly brittle solids will fracture. Glass will fracture rapidly and concrete progressively. Most engineering materials will yield plasticly. *Elastic* deformations are temporary. The material will spring back from elastic deformations, returning to its original shape. *Plastic* deformations, on the other hand, represent permanent changes to the shape of the material.

The elastic/plastic material can be characterized by a stress-strain curve or resistance-displacement curve as shown in Figure 5.2. To the right hand side of the curve is a yield point. Displacements which exceed this will cause the material to break. Notice that the elastic deformation is characterized by the slope of the initial part of the curve. The material will recover its elastic deformation when loading is removed, but the plastic deformation will remain. This situation is shown in Figure 5.2 for a material which is loaded to just below its yield point. The resistance/displacement curve can be approximated as two linear components as in Figure 5.3.

When a material is elasticly deformed, it stores elastic energy. This energy is what

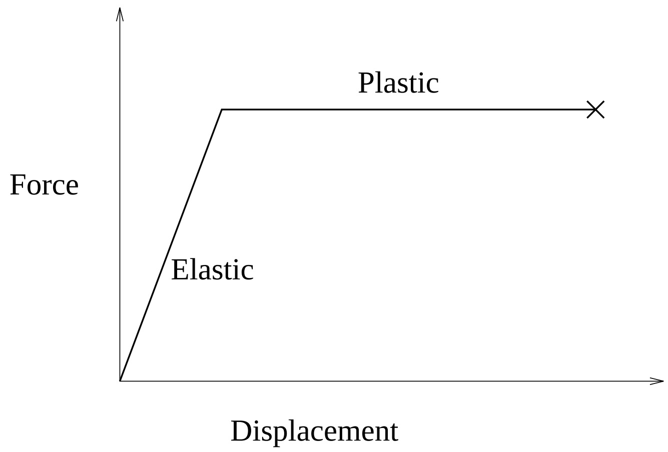


Figure 5.3: An approximated force displacement curve including the material's yield point.

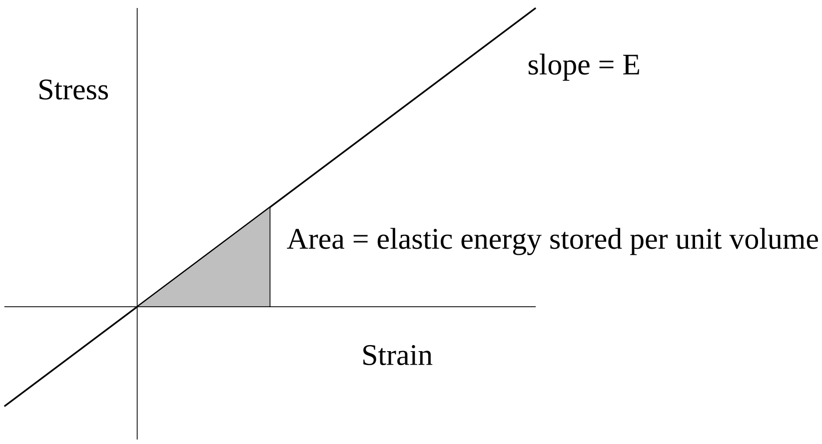


Figure 5.4: Generic stress-strain curve.

allows the material to restore its shape. The elastic energy stored per unit volume is equal to the area under the stress/strain curve for a given deformation. This is the shaded area shown in Figure 5.4. Notice that the slope of this curve is Young's modulus, E. This curve is a Hooke's law stress/strain curve and reflects ideal elastic behaviour.

5.2.2 Fracture Mechanics

The modern study of fracture mechanics began largely in the 1950's [8], but owes a debt to the classic work presented in A. A. Griffith's 1920 paper[24]. He looked at the issue of crack propagation in terms of an energy balance. Cracks involve mechanical energy and surface energy. Mechanical energy, U_M, is calculated as the sum of strain potential energy in the elastic medium (U_E) plus the potential energy of the outer applied loading

(U_A), expressible as the negative of work associated with any displacement of the loading points: $U_M = U_E + U_A$. The surface energy, U_S, is the free energy expended in expanding the crack. The total energy is the sum of the surface and mechanical energy. When the mechanical energy exceeds the surface energy, the crack will expand. When the two energies are equal, the material is at a critical point.

Inglis published an influential 1913 paper in which he examined the stress concentration at the tips of ellipsoidal cracks (cited in [24]). He showed that stresses concentrate at the tips and this concentration is dependent on the radius of curvature of the hole. Griffith used Inglis's results in his numerical simulations. He was able to show that at the critical point, the system energy is at a maximum and hence, the system is unstable[24]. Cracks will continue to propagate once the critical point is exceeded. Note that this is not true of all materials: cleavage cracks in mica, for instance, have a critical point that is at a minimum energy and hence stable. The unstable behaviour does characterize brittle materials such as glass[24].

There are flaws in Griffith's analysis, chiefly in the use of the surface energy term to represent the energy used to fracture the material[8]. This term is accurate for brittle materials such as glass, but will be far too small for materials that experience plastic deformation at the crack tip. This kind of fracture will be explained in more detail below. Within this context, it is sufficient to note that materials which yield plasticly at the crack tip require far greater energy for crack propagation.

A modern analysis taking this into account gives the following condition for a crack to advance: The work done by loads must be greater than the change in elastic energy created by crack propagation plus the energy absorbed at the crack tip. Symbolically:

$$\partial W \geq \partial U_{el} + G_c T \partial a \ , \tag{5.4}$$

where W is work, U_{el} is elastic energy, G_c is the energy absorbed per unit area of the crack. It is a measure of toughness and is called the "critical strain energy release rate". Note that it is a property of the material and is used in connection with the crack area, not the new surface area created by the crack. T is the thickness of the material and a is the length of the crack. G_c is material dependent and varies widely. It is approximately

10 Jm^{-2} for glass, a brittle material, and $10^6 Jm^{-2}$ for copper, a ductile material[2]. G_c is the critical value of G beyond which crack propagation will occur.

To gain a better understanding of G consider the following example. A plate is being pulled apart by forces acting on opposite sides of it. These forces are causing a small crack in the block to grow. The condition at which crack growth occurs is $\frac{\partial}{\partial a}(U - F + W) = 0$ or $\frac{\partial}{\partial a}(F - U) = \frac{\partial W}{\partial a}$ where U is the elastic energy in the plate, F is the work performed by the external force and W is the energy of crack formation[8]. G, the energy release rate, is defined by $G = \frac{\partial(F-U)}{\partial a}$. The crack resistance is given by $R = \frac{\partial W}{\partial a}$. When the energy release rate equals the crack resistance, the crack will grow. Fracture instability occurs when upon crack extension, G remains greater than R. At this point, more energy is released by extending the crack than is consumed in the process. The difference $G - R$ can be used to calculate the kinetic energy associated with the material moving away from the crack tip.

As well as the energy balance method, there is another method for determining failure that is based on the size of the crack and the stress applied. This involves the "stress intensity factor" K. For flat planes where $t \ll a \ll$ other plate dimensions[2]:

$$K = \sigma\sqrt{\pi a}\ . \tag{5.5}$$

This formula will vary by an often small constant for other geometries[2]. Fast fracture will occur when $K = K_c$, a critical value representing the fracture toughness. This is a property of the material and can be found by consulting the appropriate table (see for instance [2, 26]). For a given value of a, the stress needed for failure can be found and vice versa. Note that K_c and G_c are related. For the above geometry, $K = \sigma\sqrt{\pi a} = \sqrt{EG_c}$ where E is Young's modulus.

5.2.3 Fracture Types

Ductile vs. Brittle Fracture

A ductile material is one that will tend to stretch rather than breaking when a load is applied. As indicated above, brittle materials will fracture much more easily than ductile materials. This is because the fracture mechanism is different for the two classes.

Cracks in glasses, ceramics and other brittle materials propagate without any plastic flow taking place. The only energy required is that used to break interatomic bonds and this is much less energy than that used during ductile tearing[2]. Brittle propagation often has a definite crystallographic orientation and in these cases is called cleavage. During ductile tearing, the material will stretch around the crack tip. Metals and other ductile materials exhibit this plastic deformation at the crack tips. This consumes a large amount of energy. Ductile cracks propagate by voids forming ahead of the crack tip, enlarging and eventually joining up to move the crack forward[26].

Brittle crack tips are very sharp. Ductile crack tips tend to have a much larger radius of curvature. The stress concentration at a crack tip is proportional to $\sqrt{\frac{1}{\rho}}$ where ρ is the radius of curvature of the tip[19]. The sharper the crack tip, the higher its stress concentration will be and hence the more likely the crack will be to propagate. Indeed, the damaging effect of an existing stress concentration depends strongly on a material's ability to deform plasticly and thereby blunt the crack tip[19].

5.2.4 Fast Fracture

The fracture of interest for this work is *rapid* or *fast fracture*. It occurs over small time scales and often leads to the fragmentation of the host material. Two different mechanisms for material breakage have been described. First we considered a material's stress-strain curve and showed that if the material is displaced past its yield point, it will break. The second mechanism depends on an initial crack in the material and a critical stress that is sufficient to cause this crack to propagate. It is the second mechanism that is normally associated with rapid fracture.

Take the example of a balloon which is partially inflated. If it is pricked with a pin, it will not fail at this low pressure. For the flaw to expand, the rubber must be torn, causing additional crack surface to be created. This requires energy. If the pressure inside the balloon plus the release of elastic energy is less than the energy required for tearing, tearing will not occur. As the balloon is inflated, the pressure increases. At a certain point, the balloon will have stored enough energy that if the crack advances, it will release more energy than it will absorb. At this point, the balloon will "burst": Fractures propagate rapidly through the balloon and it breaks apart. This is rapid fracture. It is very important to note that rapid fracture occurs below the balloons material yield point (following [2]).

5.2.5 Material Flaws

Most materials yield at about two orders of magnitude below their theoretical yield point[24]. This suggests the presence of micro-cracks in materials which are causing the lower yield points. This phenomenon was first discovered by Leonardo daVinci while he was attempting to determine the strength of iron wires[19]. He showed that on average, the longer a wire was, the less weight it could hold. A longer wire has a higher probability of containing a micro-crack of sufficient length to yield for a given load. This suggests that to at least some degree, the strength of a material is a problem of statistics. Indeed, there are probabilistic formulas that predict the strength of materials based upon the odds of a crack of a certain length being present in the specimen.

5.2.6 Wave Effects

Most of our analysis has been for static loading. When dealing with blast waves, loading will be highly dynamic. The blast wave from the explosive will generate a shock wave in the object. This creates highly uneven loading. The front of the object will be compressively loaded first, with subsequent sections being loaded as the wave passes through the object. Upon striking the back of the object, the wave will reflect as a tensile wave and travel back through the object. This wave will then bounce back and forth in the object as it decays. The waves move at the elastic wave velocity of the material. Microfractures will experience tensile loading as the wave passes over them. If a crack starts to propagate, it may complete or proceed in steps each time a tensile wave passes over it[24].

In the general case, wave loading is complicated because the wave shape is not simple and it will propagate in three dimensions, containing shear and dilational components. Furthermore, boundary reflections for all but the simplest geometry create intricate wave patterns. The micro-cracks in the object will all be loaded independently.

When the wave first reflects off the rear of an object, it creates strong tensile forces. These tend to cause chunks to break off the back of the object. This effect is called *spalling* and is a significant damage mechanism in blast loading[26, 24].

Fragmentation is much more likely under dynamic loading for several reasons. Dynamic loading provides more kinetic energy. Independent crack nucleation occurs profusely. Crack branching is prevalent and leads to smaller fragments. Finally, as the loading becomes more rapid, the material will fragment into smaller pieces. As the duration of the compressive wave increases, fragmentation will also increase[26]. While not a cause of fragmentation, it is important to observe that the final fragments produced contain many internal cracks. In rock fragmentation for mining, it was found that microcracks within the rock fragments have greater surface area than the fragments' surface area[26].

5.2.7 Crack Propagation

Crack Velocity

Cracks accelerate towards their terminal velocity. Calculations based upon the kinetic energy of the system suggest that the terminal crack velocity is approximately $0.38v_1$ where v_1 is the elastic velocity within the material. The terminal velocity is typically between 1 and 5 $km \cdot s^{-1}$. In general, the velocity is given by:

$$v \approx 0.38v_1(1 - \frac{a_c}{a}) \tag{5.6}$$

where a_c is the critical length for fracture initiation and a is the current crack length. This formula tends to overestimate the velocity when compared to experimental results. Typically the ratio of v/v_1 is 0.20 to 0.37. For glass, $v = 1500m/s$ and for steel $v = 1000m/s$ [26, 24, 8].

Crack Branching

One of the most significant causes of fragmentation is crack branching. Possible causes of branching include dynamic crack tip distortion, where dynamic behaviour changes the nature of the stress field. Also, secondary fractures may occur ahead of the primary

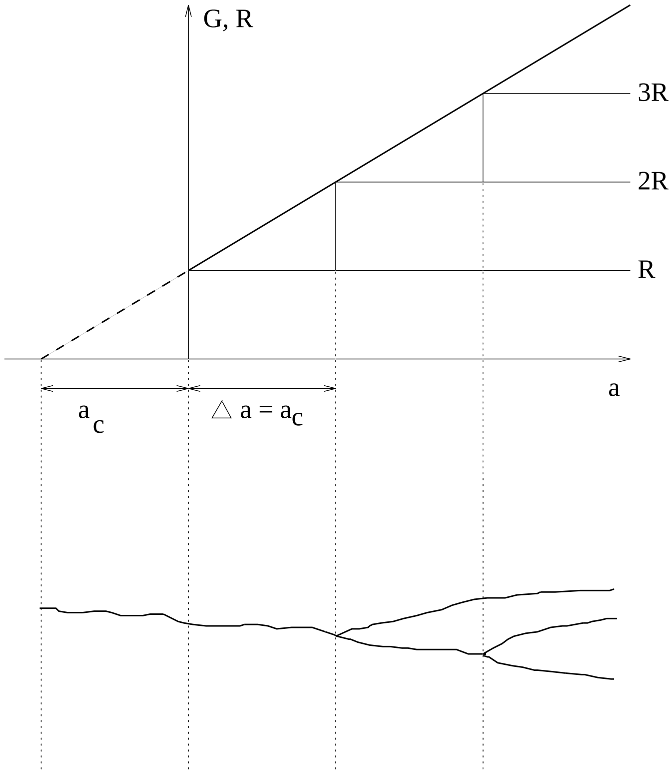


Figure 5.5: The crack will branch whenever G exceeds a multiple of R..

fracture system and link back to the primary system. Finally, stress waves which reflect off the boundaries can create branching when they intersect with the crack tips. [24]

Figure 5.5 shows the branching pattern that occurs as the crack propagates. This assumes that the crack resistance R does not change during dynamic loading. Strictly speaking it can go up or down in the dynamic case, but assuming a constant R simplifies the analysis. a_c is the critical crack length at which propagation begins. At this point, $G = R$ (recall that G is the elastic energy release rate.). As a increases, G continues to grow. When $a = 2a_c$, $G = 2R$. This indicates that there is enough energy to propagate two cracks. Similarly, when $G = 3R$, there is enough energy to propagate a third crack.[8]

Crack branching may reduce the crack's velocity[8, 26]. The angle between branches can also be predicted. When a a crack deviates from the plane perpendicular to the tensile stress, it is also subject to shear stress. Analysis under these conditions suggests that the angle between the branches must be on the order of fifteen degrees. This corresponds well to observed behaviour[8].

5.3 Previous Models of Fracture

Previous attempts at modelling fracture for computer graphics have been based on spring mass models. These models represent materials as a grid of distributed masses connected together by springs. Demetri Terzopoulos with several different collaborators did a great deal of pioneering work on deformable material models for computer graphics and computer vision[38]. He developed elastic models which correspond to masses joined by ideal Hooke's Law springs[40]. Terzopoulos and Fleischer also developed models which had plastic, elastoplastic and viscoelastic behaviour[39]. The elastic models will return to their initial shape when loading is removed, whereas the models with plastic behaviour can be permanently deformed. These modelling efforts have generally focussed on relatively thin surfaces, although Terzopoulos's work is extensible to full three dimensional solids.

Fracturing occurs in these models when the "spring" connecting two masses is stretched beyond its yield point. This is equivalent to exceeding the yield point on the stress/strain curve as discussed above. These models have been used for tearing paper, breaking a net over a sphere and breaking a solid over a sphere. The results are quite good, but the materials do not fragment (i.e. shatter into a large number of pieces) and tend to have a fairly elastic appearance.

Norton et al. concentrated on modelling fracture. They used a damped spring mass system similar to Terzopoulos's[29]. Again, breakage occurs when the springs are stretched beyond their maximum yield point.

Norton et al. used their model to create the animation "Tipsy Turvy" for Siggraph89[28]. This video shows a teapot falling to a table and breaking. It illustrates several shortcomings of the spring mass approach. First of all, the teapot is very rubbery in appearance. It is difficult to create the appearance of more rigid materials such as glass and ceramics using a spring mass model. This is because stiff materials require very large spring constants in order to hold the mass points close together. With large spring constants, the differential equations governing the system become stiff and require the use of implicit solvers for their solution. This can be very costly, often prohibitive, for grids with a large number of mass points. In order to efficiently solve spring mass systems, smaller spring constants are normally used, yielding materials with an elastic appearance.

The teapot breakage in "Tipsy Turvy" has a noticeable staircase appearance. This is an artifact of the underlying model. The mass spacing on the teapot is approximately 1 cm. Breakage occurs along the lines of the grid. This yields noticeable staircase artifacts. Hart and Norton published another paper in which they fit curves to the broken edges in order to remedy the staircasing problem[17], but this still does not address the resolution problem.

Finally, the breakage pattern is not what one would expect from dropping a teapot. The teapot breaks into a small number (three or four) of quite large fragments and a large number of very small fragments. The small fragments are all single cells and are rendered as tetrahedrons. In shattering a teapot, one would expect a more continuous distribution of fragment sizes.

There is no explicit concept of crack propagation or bifurcation in spring mass models. Cracks propagate simply when adjacent springs are stretched past their yield point. As one spring yields, the force on the next spring will increase. This kind of crack propagation is perhaps why the fragmentation pattern of these models is not consistent with what one would expect of a shattering solid. They tend to appear to tear rather than shatter.

Spring mass models work well for creating deformable materials, but do not shatter well. It is a focus of this work to create more realistic fracture patterns for rapid fracture and dynamic loading of brittle solids.

5.4 A New Fracture Model

Our model focuses on the sub-problem of generating fracture patterns in a plane. This work attempts to generate realistic fracture patterns that can then be used in generating animations of things such as a shattering window. Such patterns should have a reasonably chaotic appearance and feature a variety of fragment sizes and shapes.

5.4.1 Basic Algorithm

The crack propagation algorithm is based upon the relationship between G and R shown in Figure 5.5. An initial, very short micro-crack is specified as a line segment. The crack is propagated in each direction. Every time the crack structure grows by length a_c (i.e. every propagating edge grows by that amount), one propagating edge of the crack structure is forked. This process generates a crack tree. Edges are allowed to propagate until they either hit another edge, or the border of the geometry, being terminated at these points. Note that only the colliding edge is terminated. The hit edge proceeds unaffected. In the base algorithm, cracks fork at a set angle specified by the user. All crack lines are straight. Stochastic variations on the base algorithm will be described below.

The crack tree is maintained as a logical tree structure which can be searched. A leafList list is maintained of all edges which are currently propagating. These are the leaves in the tree. Conceptually, at each time step, every leaf is extended a length a_c[1]. After each a_c extesnsion, an edge is chosen to fork. Edges stop prpogating when they hit another edge or the border of the geometry. New edges are added to a bifurcation queue and elements of the queue are popped off when a new forking candidate is needed. This is a FIFO queue. Forking candidates could also be picked randomly from the list.

Since the crack structure is a connected tree, whenever there is an intersection of two edges, a new face must have been formed. These faces correspond to fragments. At the time of intersection, the fragment outline is traced and its set of coordinates is stored. This process is outlined below. Fragments which have an edge on the border are only traced once the crack structure has finished propagating.

[1] In order to perform the collision detection described below, the steps are kept to a maximum of one square in the bit bucket per time step. Every edge is propagated a length a_c, however, before an edge is forked.

The algorithm can be summarized as follows:

```
while !done
    leaf = getFirstLeaf
    while leaf != NULL
        result = extend leaf->edge by distance
                                            //(distance = timestep * crackVelocity)
        if result == hitOtherEdge
            traceFragment
            if leaf->edge was forking candidate
                setNewForkingCandidate
            removeLeaf
        else if result == endOfEdge
            if edgeIsForkCandidate
                create new node, leftChildEdge and rightChildEdge
                add right and left edges to the leaf list
                add right and left edge to bifurcation queue in random order
                setNewForkingCandidate
                remove old leaf
            else
                add node
                extend edge in same direction
                update leaf
        leaf = getNextLeaf
   end //while leaf != NULL
   if all edges have reached the object edge
        done = TRUE
end //while !done
traceBorderFragments
```

The algorithm can be directly coupled to the blast wave model. The model calculates the pressure profile over the panel. This pressure can be used as the critical stress in Eq. 5.5, thereby yielding an initial crack length. It can alternately be used to determine whether or not fracture occurs for a set initial crack size. Notice that if the pressure is used to determine the initial crack size, the general relationship between pressure and fragment size shown by Equation 2.45 will be maintained. That is, for higher pressure, the average fragment size will be smaller. This follows because higher pressure gives a smaller value for a_c, implying that the tree forks more frequently, which yields smaller fragments.

The initial crack can be placed anywhere on the panel, and indeed the algorithm could be modified to accommodate multiple crack trees if crack nucleation from many locations is desired. Most of the images in this section were created to model the shattering of a window. Windows are held in a frame and have the least resistance at their mid-point, so cracking will normally begin here. Therefore, initial cracks were placed near the middle of the panel.

5.4.2 Bit Bucket

The previous section describes the logical structure of the algorithm. Physically, the edges are propagated on a bit bucket structure. A bit bucket is an occupancy grid or coverage mask (cf. [14, 10]) where a grid location, or bucket, is marked when an edge propagates into it. The advantage of a bit bucket is very fast detection of edge collisions.

The results shown here were computed on a 2k x 2k bit bucket. This allowed for the specification of very small initial cracks, leading to very small fragments. A coarser grid could be used if very small fragments were not desired.

When a collision occurs, it is necessary to know the identity of the two edges involved in order to trace the chunk. For this reason, the bit bucket was used to store edge pointers rather than just on or off bits. This is very time efficient, but represents a significant increase in space cost–from 1 bit per bucket to 4 bytes. If memory is limited, a coarser grid structure of say 128 x 128 could be introduced. Each location in this grid would correspond to a 16 x 16 section of the original bit bucket and would hold a list of pointers to all the edges present in this area. This list would be short, likely containing no more than twenty entries and normally far fewer. This represents an order of magnitude reduction over the 256 pointers maintained in the original section of the bit bucket. A search could be made of the edges in the list to determine which edge was intersected. This would be time costly, especially when fragments are small, because many collision tests would have to be performed. It would, however, save on space.

For every time step, every leaf edge is propagated a distance equal to the crack velocity times the time step. The time step is chosen so that the edges grow the width of one grid location with every iteration. This makes it easy to determine which buckets are touched during each step and allows for animations of the crack propagation to be generated. During each time step, the edge can touch at most one horizontal or vertically adjacent

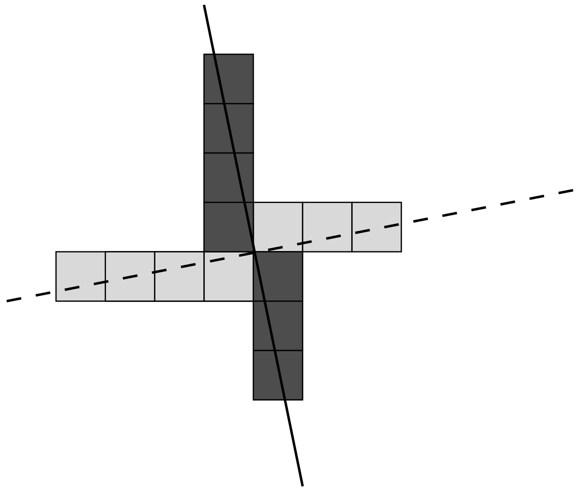


Figure 5.6: Failed collision detection with scan-line type conversion.

bucket and one diagonally adjacent bucket. Every bucket an edge passes through in the grid must be filled. If a normal scan line approach was to be used, it would be possible for edge collisions to be missed, as shown in Figure 5.6.

5.4.3 Fragment Tracing

A collision occurs between a hitEdge and a haltEdge. The haltEdge strikes the hitEdge and stops propagating. After the collision is detected, the fragment is traced and its coordinates are stored in a fragment structure. The haltEdge is being propagated at the time of the collision and the identity of the hitEdge can be obtained from the bit bucket.

To trace the fragment, the haltEdge is first traced back to the root node by following each parent link. Along the way, each node is coloured with the current marker. Figure 5.7 shows the fragment after the collision, but before it has been traced back. Notice the hit and halt edges. Figure 5.8 shows the halt edge side being traced back to the root. It is theoretically possible for a crack to bend around and intersect its ancestor. This was not observed in practice, but can be trivially detected if a node already marked with the current marker is detected during the trace back. The next step is to trace the hit edge back towards the root. When it finds a node coloured with the current marker, that node is the shared ancestor of both the hitEdge and haltEdge. This is shown in Figure 5.9. Note that the markings are different on the hit and halt edges in the diagram simply to preserve clarity. The same marker must be used in the actual algorithm. The chunk can

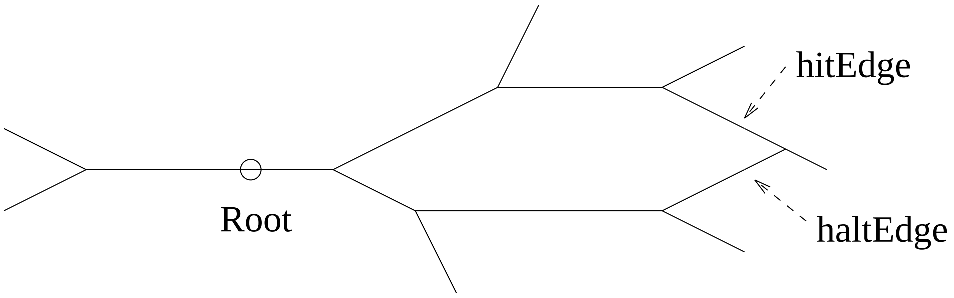


Figure 5.7: The fragment as generated by the algorithm.

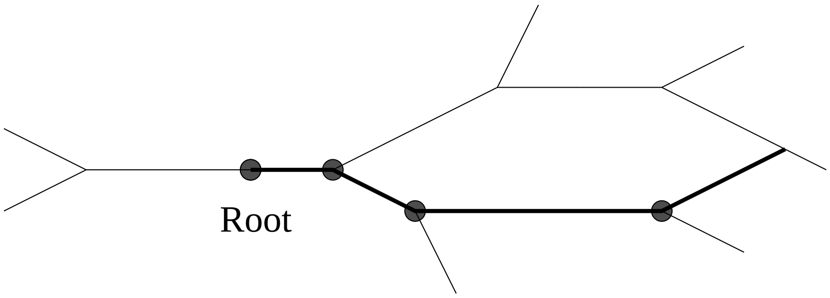


Figure 5.8: The fragment traced up the halt edge side to the root node.

now be traced by following the haltEdge up to the common ancestor and then following the hitEdge's ancestors down to the intersection point. The geometry is retraced in order to check for internal edges, as described below. After each trace, the marker is incremented so that future traces can be completed without having to re-initialize the node colours.

In rare cases, when a fragment is "completed" there may be an edge inside that fragment that is still propagating. As it does so, it will spiral inwards, dividing the chunk into many small fragments. This behaviour is not physically reasonable and leads to visual artifacts. To correct this problem, edges internal to a crack are killed. When

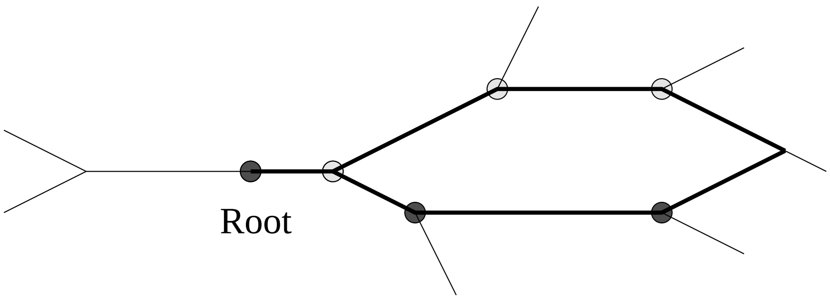


Figure 5.9: The fully traced fragment.

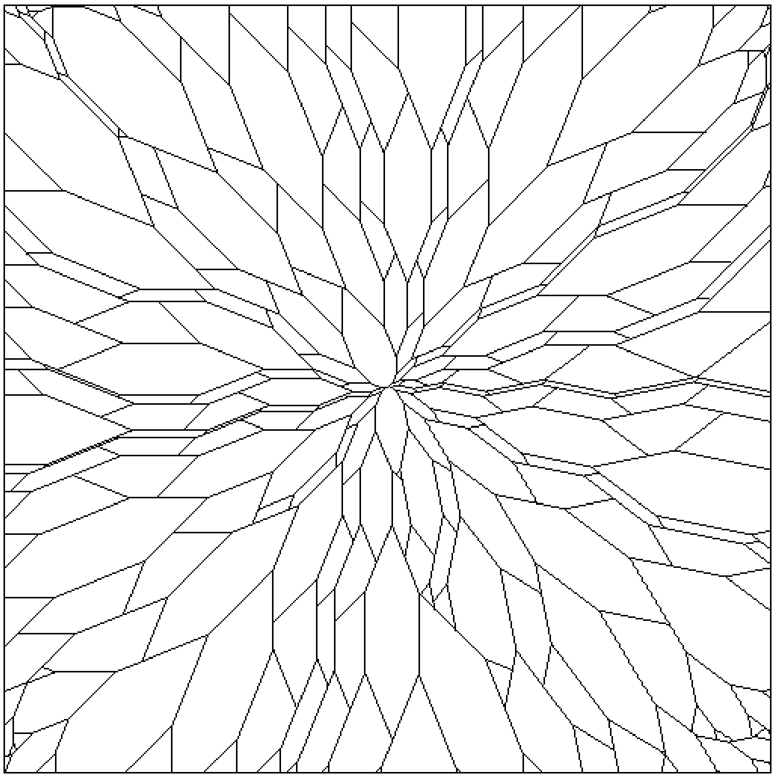


Figure 5.10: A crack pattern using the base algorithm and a bifurcation angle of 25 degrees.

the trace algorithm begins, a test is performed to determine if the face is to the right or left of the path which will be followed to outline it. When the fragment is traced, the algorithm will always pick the path that corresponds with this direction (e.g. always veering right). If it hits an incomplete edge while tracing down the hitEdge's ancestors, this edge is internal to the chunk. It is killed and the algorithm backtracks to the last fork and proceeds down the other path. A trace is also performed down the haltEdge side of the fragment.

5.4.4 Results and Algorithm Enhancements

A very large number of patterns can be generated by varying the branch angle and initial crack size. Two of these are shown in Figures 5.10 and 5.11 where the first has a propagation angle of 25 degrees and the second a propagation angle of 35 degrees. Notice that fragments are more shard like, being long and narrow, with a smaller propagation angle. As the propagation angle increases, they become more round.

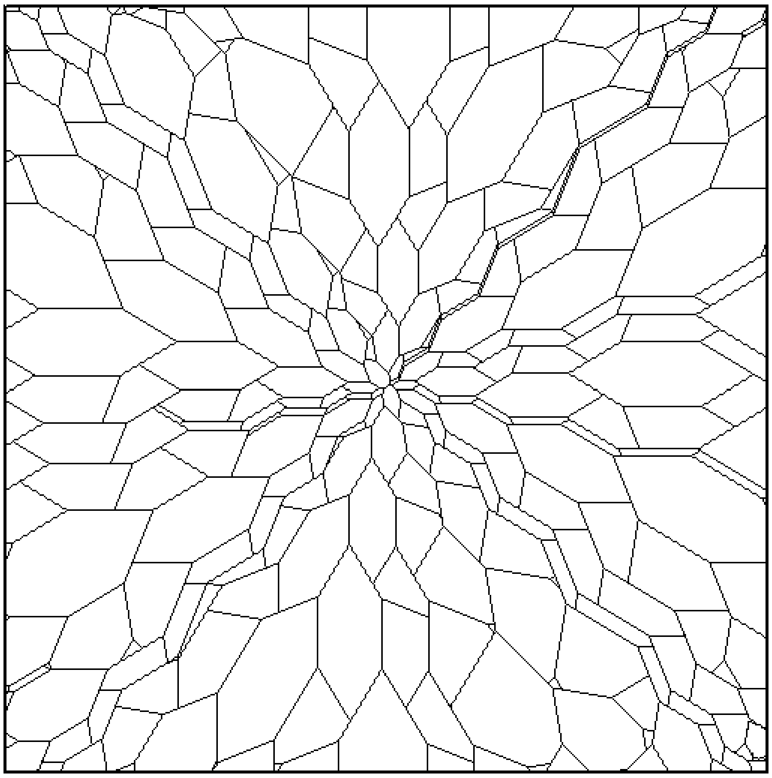


Figure 5.11: A crack pattern using the base algorithm and a bifurcation angle of 35 degrees.

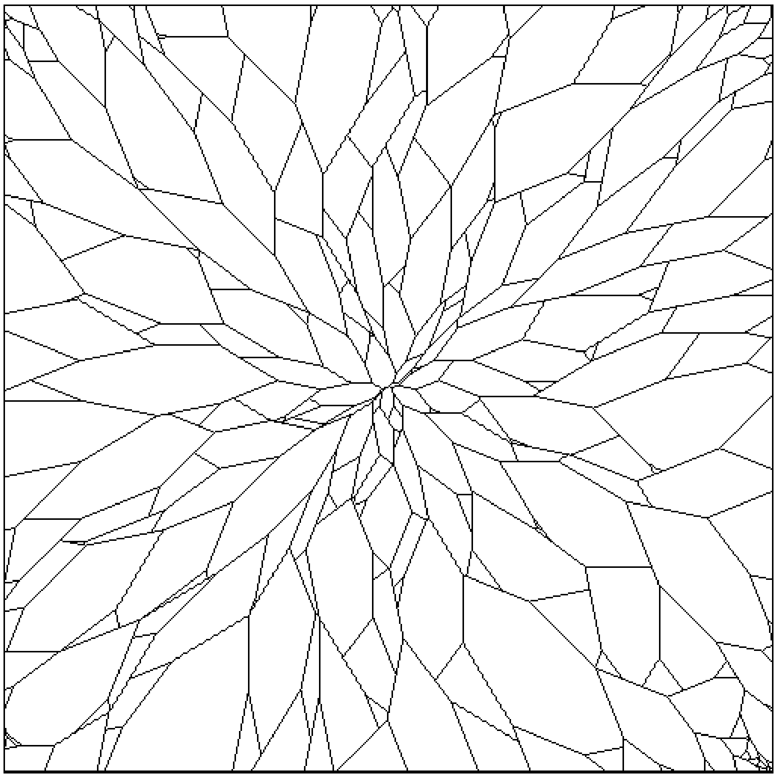


Figure 5.12: A crack pattern featuring random variation of the bifurcation angle.

Although these patterns are visually interesting, they are a long way from appearing like shattered glass. They have too much correlated structure. To compensate for this problem, two methods of random variation were included as discussed below.

Varying The Propagation Angle

The patterns look unrealistic in part because there are a large number of parallel lines. This is due to the use of a constant propagation angle. An edge forks two children. When these children both fork their own children, two of the new edges will have the same orientation as the grandparent yielding parallel lines. This crystalline structure is not observed in amorphous materials such as glass. The regularity of the fork angles is also visibly noticeable in itself. To rectify this problem, a variance of the crack angle can be defined. The propagation angle will be randomly computed as the base propagation angle +/- half the maximum variance. The base fracture pattern is shown in Figure 5.10. The result of varying the propagation angle is shown in Figure 5.12. The maximum angle variation used here is 34.5 degrees.

Line Wiggling

The other major distracting feature of the patterns is their long straight lines. Cracks in crystalline materials may propagate along straight cleavage lines, but this will not occur for most materials. Line wiggling was introduced to create more natural shapes. This corresponds to a propagating crack hitting some imperfection in the material which causes its direction to change or to changes in the stress field. The probability of a wiggle and the maximum variance can be defined. Each time a new edge is generated that should be a straight extension of the previous one, a random decision is made to determine whether or not its propagation angle should vary. If it does wiggle, the edge's angle is calculated randomly in the range defined by its parents orientation +/- half the maximum variation. Figure 5.13 shows the effects of line wiggling and Figure 5.14 shows the combined effect of both line wiggling and varying the propagation angle. Note the vastly improved pattern as compared with the base algorithm. The new fragments have a much more realistic, natural appearance.

To provide some basis of comparison to real world crack patterns, Figure 5.15 shows a series of cracks in a section of pavement outside outside of the computer science building. The general overall structure of the pavement cracks appears to correspond quite well with the generated fractures. The branching tree structure is clearly evident in the pavement cracks. The general fragment shape also shows good agreement.

Long Time Scale Cracks

It is also possible to generate cracks that would normally be associated with creep type phenomena. Such cracks do not correspond to the complete destruction of a panel, but show directed fracture patterns over a part of the surface. Such an effect is shown in Figure 5.16. It is generated by specifying a much longer initial crack, a bifurcation angle of 15 degrees and using both forms of stochastic variation.

A Comparison to Related Work

There are some similarities between this work and the Reed and Wyvill work on modelling lighting[32]. Lightning features a main channel that connects the source and the

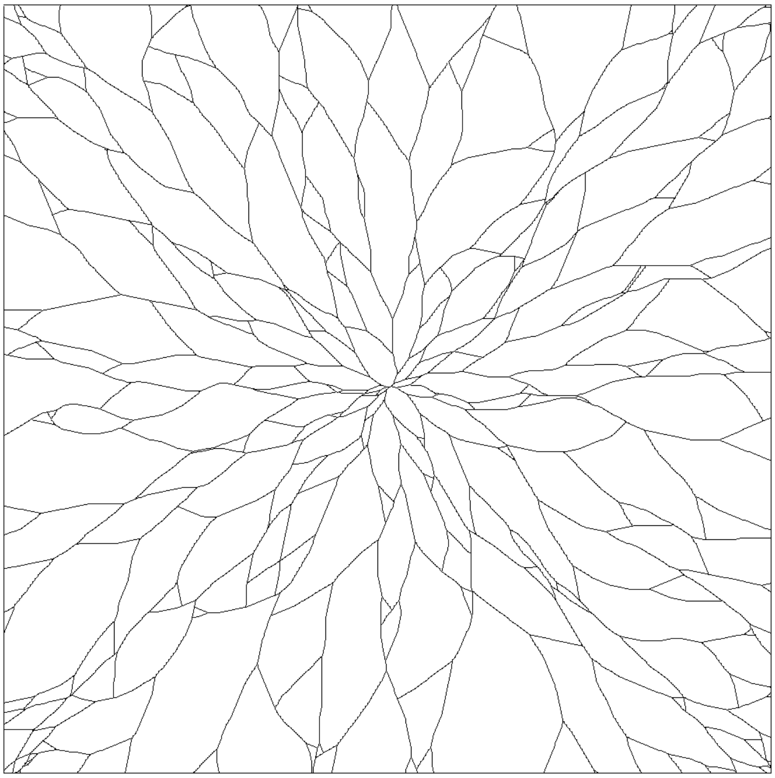


Figure 5.13: A crack pattern featuring random line wiggling.

target (e.g. a cloud and the ground), and often a number of branches that fork off this main trunk. Reed and Wyvill propagate the main channel as a series of connected line segments. They stochasticlly vary the propagation angle for each segment from a uniform distribution. This is comparable to the line wiggling employed here, although their algorithm appears to be designed to produce decidedly more jagged results. The length of the segments in the lightning are also randomly varied. The major difference between the two algorithms is that Reed and Wyvill use a probability function to control branching. This made branching difficult to control as some random seeds would produce no branches off the main trunk and some would produce a very large number. In our algorithm, branching is controlled by the length of the initial crack. This gives consistency across random number seeds, as the frequency of branching will always be the same. This makes it easier to control the algorithm.

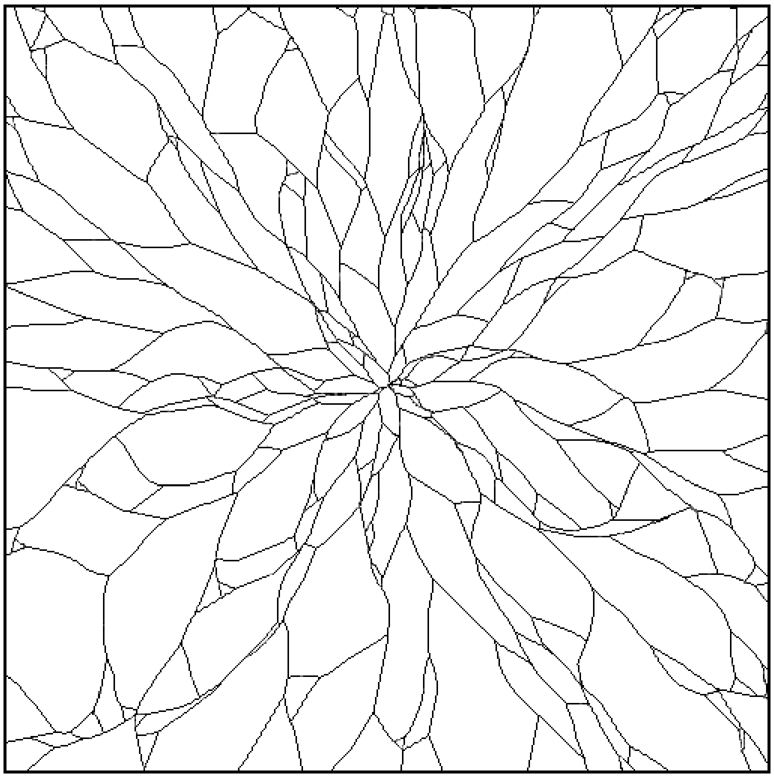


Figure 5.14: A crack pattern featuring random variation of the bifurcation angle and line wiggling.

Figure 5.15: A photograph of cracks in a section of pavement. (The contrast of the photo has been increased and the cracks have been outlined in black to make them easier to see.) To trace the crack tree pattern, start at the arrow and trace the cracks downwards. Notice that the pattern is consistent with a crack propogating forwards and forking every so often along the way. The one exception to this occurs the upper left hand edge, where an extra branch comes off the main structure.

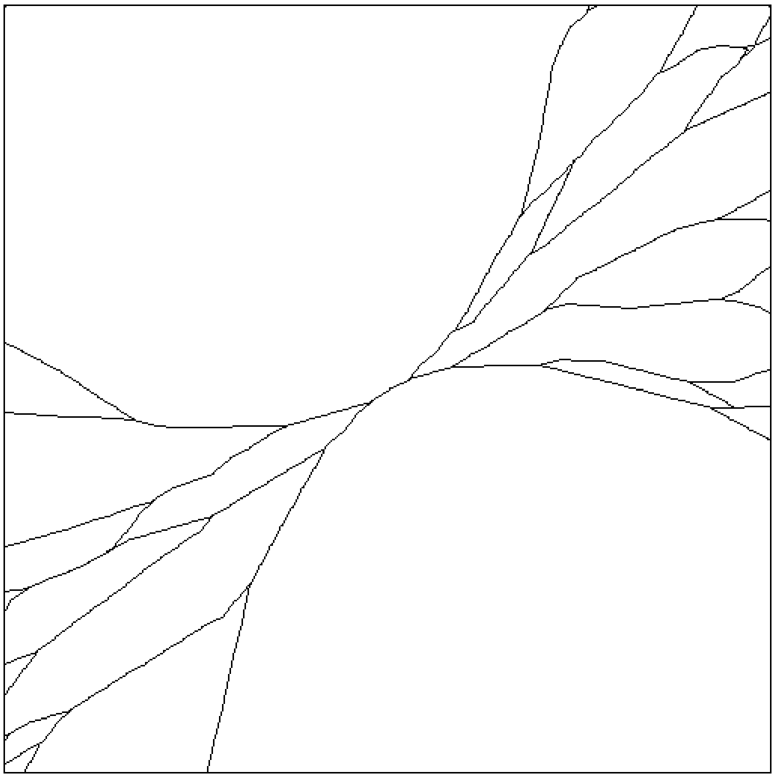


Figure 5.16: A creep style crack.

Improving the Algorithm

There is no precise physical basis for the stochastic variation used in this algorithm. Finding a physically accurate statistical basis for generating fracture patterns would be a significant addition to this work. Further to this, a more detailed material model could track pressure profiles over the surface of an object. These could possibly be fed back into the propagation algorithm to weight the stochastic variation so that cracks tended to respond to stresses in the material.

Variations can be added to the algorithm to produce different effects. For instance, instead of starting with a line segment, the initial crack could be an X. This would change the crack pattern, especially around the origin. It is also possible to cause the algorithm to change with the depth of the tree. This could be used, for instance, to force fragments further from the crack origin to be smaller than they are now.

Chapter 6

Blowing Out a Window

In order to demonstrate the effectiveness of the fracture and explosion models, an animation must be generated that makes use of them. An excellent test candidate is the blowing out of a window. Due to the brittle nature of glass, windows often shatter during explosions. Indeed, shattering windows are both one of the most significant safety hazards during an explosion and one of the most visually interesting events. Clearly, visualizing the shattering of a window is a desirable modelling goal.

Shattering a window is also a desirable goal for its inherent aesthetic interest. It involves hundreds of glass fragments scintillating in the light as they are blown apart by a blast wave. Computer graphics provide probably the best mechanism for viewing such an event, and it can be simulated and viewed at any desired speed.

In order to generate this animation, the fracture pattern shown in Figure 5.14 was coupled with the explosion model. As was mentioned earlier, the explosion model can be used to determine the peak stress experienced by the window and this can be used to generate the base crack used in growing the fracture pattern. It was decided for this animation to not couple the two models in this way. This gives the animator more freedom in controlling the final product. A desirable fracture pattern can first be generated and then the bomb parameters can be varied independently to generate the desired movements. Coupling remains an option for cases where physical accuracy is more important than animator control.

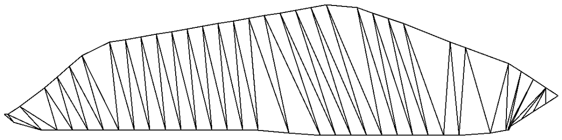


Figure 6.1: A tesselated fragment.

6.1 Tessellation

If the fracture algorithm is used without stochastic variation, the fragment geometry is very simple. Most fragments are defined by no more than six vertices and the fragments are convex. Allowing stochastic variation in the crack lines, however, increases the complexity of the geometry. For the fracture pattern shown here, fragments have a range of 3 to 250 vertices. Altogether, the fragments contain 21 459 vertices, of these, 1447 are branching vertices. This counts a vertex once for every fragment it is a part of. Every vertex is a part of either two or three fragments. Many of these fragments are also concave.

The fragments have a high degree of structure that can be taken advantage of when they need to be tessellated. Generally speaking, a fragment is long and slim and bends along its major axis. Fragments also tend to have roughly the same number of control points on both sides of their major axis. By tessellating a fragment with triangles whose major axes are roughly perpendicular to the fragment's major axis, a tessellation can be formed which deals with the concavity of the fragments (ensures all triangles are interior to the fragment).

The tessellation scheme used is very simple. It starts a triangle strip at one apex and traces it up to the other apex, adding a triangular fan at the far end to account for any extra vertices on the one side of the fragment. A tesselated fragment is shown in Figure 6.1. This technique worked well in practice. Its main advantages are ease of computation and dependability. Rendering speed is improved by the use of triangle strips and fans. The front side of the fracture pattern was tessellated into 20 917 triangles. The same number of triangles is needed for the back faces and the sides of each of the fragments are wrapped with a rectangle strip.

6.2 Sampling and Panel Definition

Since effort has already been exerted in generating a tessellation of the fragments, it would be desirable if this information could be used in determining sample points for the loading calculations. The easiest technique would be to make each triangle a panel and use every triangle as a sample point. Twenty thousand sample points is an unnecessary quantity, however, and would have too high a computational cost.

Multiple sample points are needed in order to calculate the torques acting on a fragment. A minimum of three non-collinear points is required. If the tessellation triangles are used, some fragments will have over 200 sample points. In the interest of being conservative and reducing the impact of any poorly chosen sample point, it was decided to attempt to sample each fragment with at least five points. This was done by selecting the centroids of triangles separated by odd increments (e.g. triangle 2, 5, 8 etc.). This ensures sampling that alternates between the two sides of the major axis. The sampling points are shown in Figure 6.2.

Each sample point is used as a panel in the blast wave simulation. It has an area associated with it that is the sum of the area of the triangles surrounding it.

6.3 Inertial Tensor

In order to accurately calculate the rotation of the fragments, an inertial tensor must be calculated for each fragment. This was accomplished by translating each fragment so that its centre of mass was at the origin and calculating the integration specified below, which defines the inertial tensor. A closed form of the integration was determined for generic triangular polyhedra with equal front and back faces and fixed thickness. It was applied individually to each triangular polyhedron in the fragment and the results were summed to yield the inertia tensor for the fragment.

The generic formula for the inertia tensor is

$$I = \iiint \rho(x,y,z) \begin{bmatrix} y^2+z^2 & -xy & -xz \\ -yx & x^2+x^2 & -yz \\ -zx & -zy & x^2+y^2 \end{bmatrix} dy\,dx\,dz \ , \tag{6.1}$$

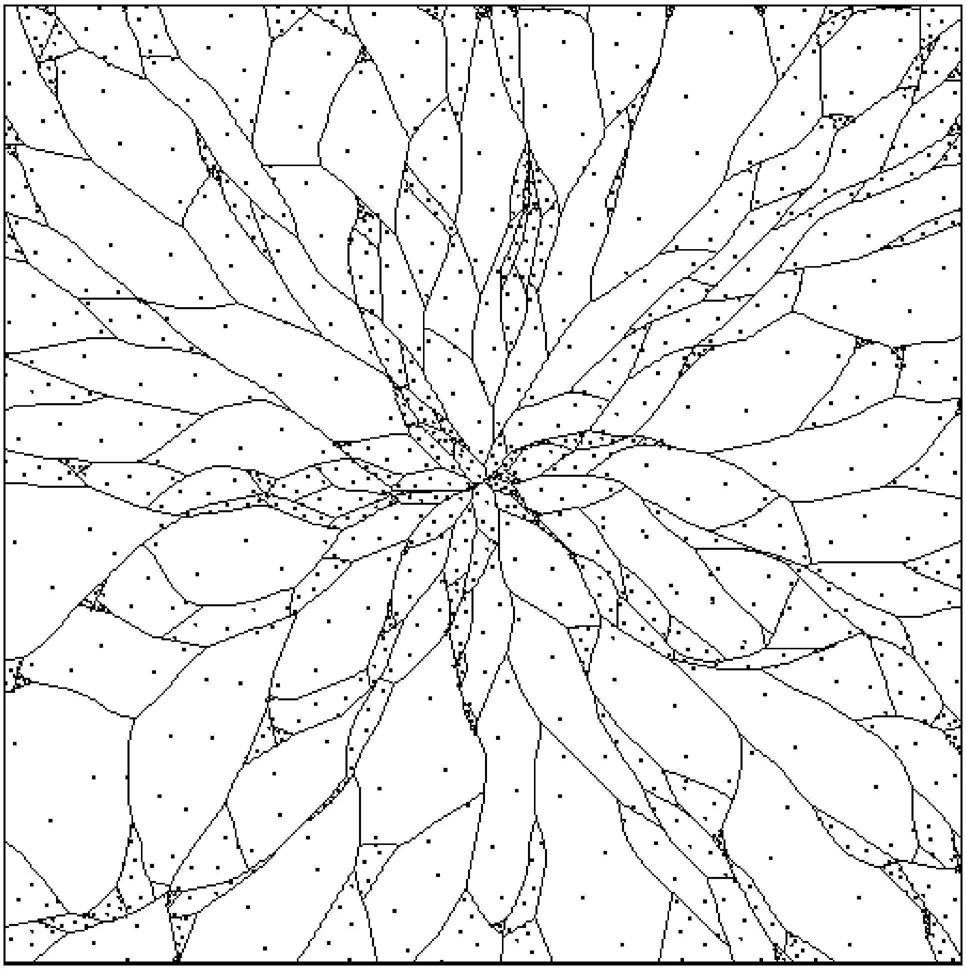


Figure 6.2: The sample points used to calculate loading are indicated with dots.

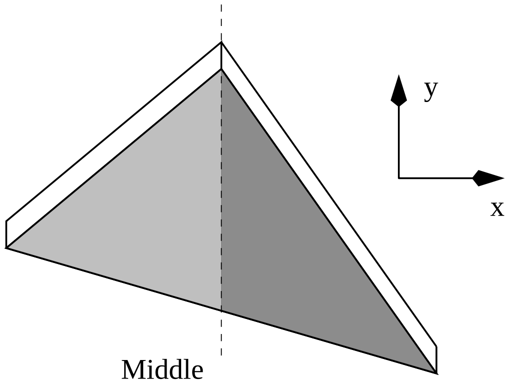


Figure 6.3: The triangular polyhedron is the basic unit for integrating the inertial tensor. It is divided in the middle to give two triangles such that each triangle is bounded by a vertical line and two sloped lines.

where ρ is the density of the fragment.

A generic triangular section is shown in Figure 6.3. It is divided into two sections by drawing a vertical line from the middle point to the opposite edge, as shown. Each section is then integrated separately. The equations of the lines which create the border of the triangle will be represented in the familiar explicit form $y = mx + b$. The upper line will be denoted with the subscript u and the lower line with the subscript l. x_r will denote the right x limit for integration and x_l the left. The polyhedron has a depth t. For a given section, the bounds of integration are therefore $m_u x + b_u$ and $m_l x + b_l$ in y, r_x and l_x in x and $\frac{t}{2}$ and $-\frac{t}{2}$ in z.

The inertia tensor is symmetric, so only six closed forms need to be defined. The equations for these are presented below. These equations must be applied to each half of the triangle and the results summed in order to calculate the correct result for a generic triangular polyhedron.

The closed form solution for entry $[1, 1]$ is

$$\left[(m_u^3 - m_l^3)\frac{x_r^4 - x_l^4}{4} + (m_u^2 b_u - m_l^2 b_l)\frac{x_r^3 - x_l^3}{3} + \frac{3}{2}(m_u b_u^2 - m_l b_l^2)(x_r^2 - x_l^2) + (b_u^3 - b_l^3)(x_r - x_l)\right]\frac{t}{3}$$
$$+ \left[(m_u - m_l)\frac{x_r^2 - x_l^2}{2} + (u_b - l_b)(x_r - x_l)\right]\frac{t^3}{12} \; . \quad (6.2)$$

The closed form solution for entry [2, 2] is

$$\left[(m_u - m_l)\frac{x_r^4 - x_l^4}{4} + (u_b - l_b)\frac{x_r^3 - x_l^3}{3}\right] t + \left[(m_u - l_m)\frac{x_r^2 - x_l^2}{2} + (b_u - b_l)(x_r - x_l)\right]\frac{t^3}{12} \ . \tag{6.3}$$

The closed form solution for entry [3, 3] is

$$\left[(\frac{1}{3}m_u^3 + m_u - \frac{1}{3}m_l^3 - m_l)\frac{x_r^4 - x_l^4}{4} + (m_u^2 b_u + 3b_u - m_l^2 b_l - 3b_l)\frac{x_r^3 - x_l^3}{3}\right] t \\ + \left[(m_u b_u^2 - m_l b_l^2)\frac{x_r^2 - x_l^2}{2} + (b_u^3 - b_l^3)(x_r - x_l)\right] t \ . \tag{6.4}$$

The closed form solution for entries [2, 1] and [1, 2] is

$$-\left[(m_u^2 - m_l^2)\frac{x_r^4 - x_l^4}{4} + 2(m_u b_u - m_l b_l)\frac{x_r^3 - x_l^3}{3} + (b_u^2 - b_l^2)\frac{x_r^2 - x_l^2}{2}\right]\frac{t}{2} \ . \tag{6.5}$$

The closed form solution for entries [3, 1] and [1, 3] is zero. To see this, notice that this entry contains a single z, which will integrate to $\frac{z^2}{2}$. When the bounds on the definite integral, $\frac{t}{2}$ and $-\frac{t}{2}$ the result will be:

$$(\frac{t}{2})^2 - ((-\frac{t}{2})^2) = \frac{t^2}{4} - \frac{t^2}{4} = 0 \ . \tag{6.6}$$

Using the same reasoning, it can be shown that entries [3, 2] and [2, 3] must also be zero.

6.4 Centre of Mass

Calculating the inertia tensor required the fragment to be shifted so that its centre of mass was at the origin because Euler's equations are designed to work with I calculated in this manner. The centre of a given surface has coordinates $\overline{x}$ and $\overline{y}$ as given by[23]

$$\overline{x} = \frac{1}{A}\iint_R x \, dx \, dy \tag{6.7}$$

and

$$\overline{y} = \frac{1}{A}\iint_R y \, dx \, dy \tag{6.8}$$

where R is the surface the integral is calculated over and A is the area of this surface. The fragments have uniform density, so their centre of mass has x, y coordinates corresponding to the centre of the surface and a z coordinate given by half the thickness of the fragment.

The integrals 6.7 and 6.8 can be solved using the same approach as was applied for the inertia tensor. The surface integral is calculated for each triangular section where a triangular section has a point at its one x extrema and a vertical line at the other as shown in Figure 6.3. The closed form for the integrals using the same notation as in the previous section is

$$(m_u - m_l)\frac{x_r^3 - x_l^3}{3} + (b_u - b_l)\frac{x_r^2 - x_l^2}{2} \tag{6.9}$$

for $\overline{x}$ and

$$\frac{1}{2}\left[(m_u^2 - m_l^2)\frac{x_r^3 - x_l^3}{3} + (m_u b_b - m_l b_l)(x_r^2 - x_l^2) + (b_u^2 - b_l^2)(x_r - x_l)\right] \tag{6.10}$$

for $\overline{y}$. The results from these integrals are summed for all the triangles in the fragment and divided by the total area to obtain $\overline{x}$ and $\overline{y}$.

6.5 Results

Animations were generated for an exploding window viewed from the front and from the side. Several frames from these animations are shown in Figures 6.4 and 6.5 respectively. It is clear from the side view that the larger particles move faster than the smaller ones. This occurs in the blast wave model because the time for the reflected pressure to dissipate is longer for larger particles (cf. Section 2.6.4). This means that larger particles experience the much higher reflected pressure for longer than the smaller particles do.

Damping is also proportional to area. This indicates that although the larger particles initially move more quickly, they will also be damped more, allowing the smaller particles to eventually overtake them. To illustrate this, an animation was generated using a damping factor 1000 times stronger than normal. This compresses the effect into a very short spatial distance. Several frames from the animation are shown in Figure 6.6. Notice that the larger particles undergo more rapid initial acceleration, but are overtaken by the smaller particles.

To achieve different effects, an animator might want to control the damping or amount of rotation which occurs in a simulation. Global damping and rotation factors are defined to give them this control. These factors serve to multiply the damping or the amount

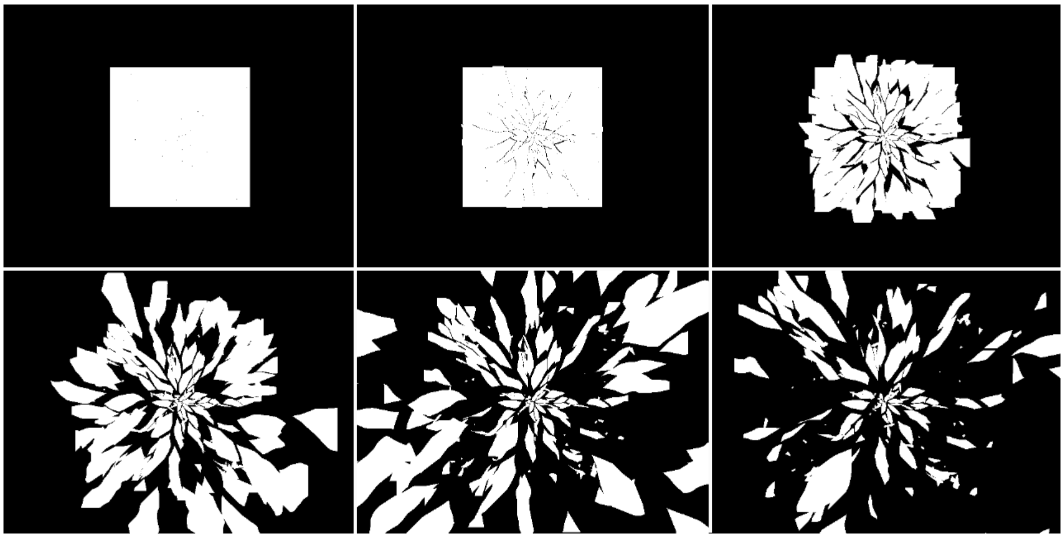


Figure 6.4: Six equally spaced frames from an animation of a shattering window. Front view.

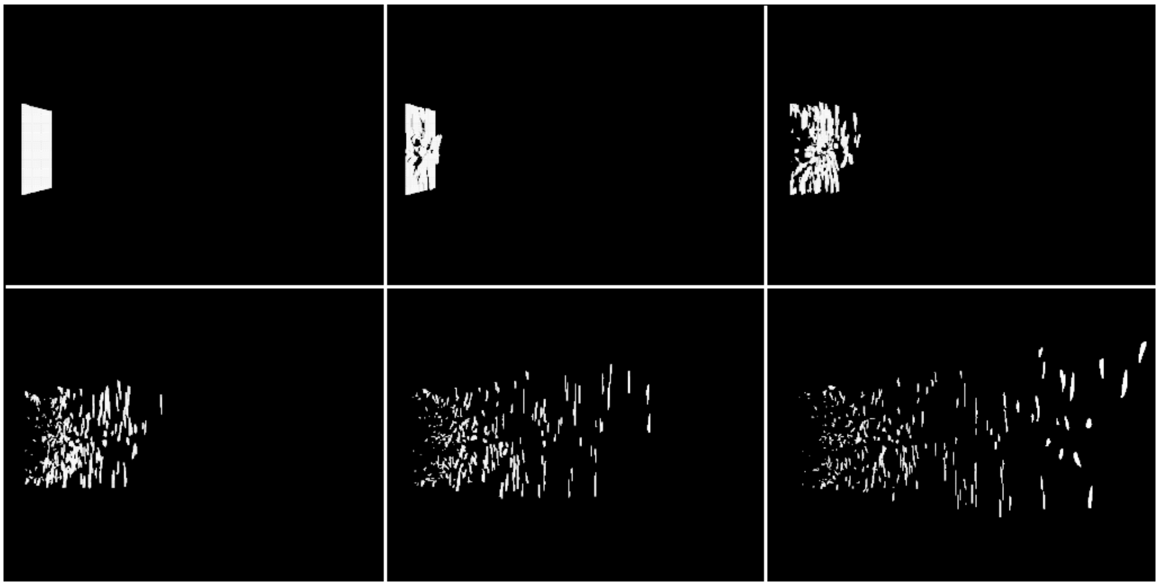


Figure 6.5: Six equally spaced frames from an animation of a shattering window. Side view.

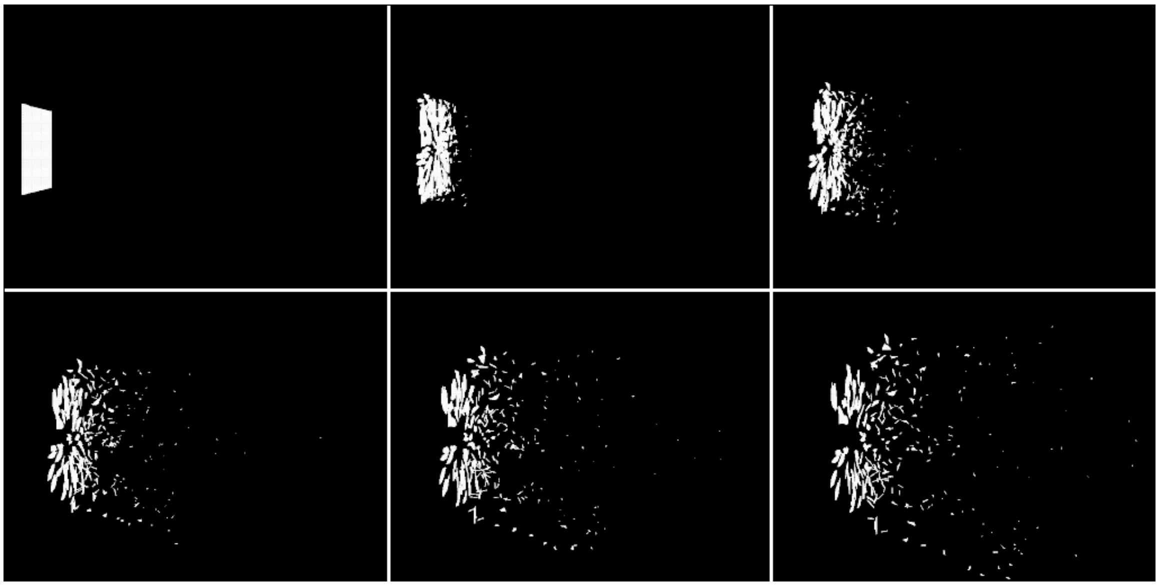


Figure 6.6: Six equally spaced frames from an animation of an over damped shattering window. Damping is 1000 times normal. Side view.

of rotation naturally present in the system. Except for the overdamped example, these factors are not used for the figures shown here.

6.6 A Comparison to Experimental Results

Figure 6.7 and 6.8 show two frames from videos of windows being destroyed by explosions [7]. Unfortunately, the quality of these images is low. The first figure shows a front on view of an exploding window and the second a side view of a different window. It should be noted that the shape of glass fragments formed is dependent on the kind of glass used in the window.

The general appearance of the generated results is in good agreement with the actual results. Both feature a wide variety of fragment shapes and the general appearance of the blast in Figure 6.7 is comparable. The plume is narrower in Figure 6.8 than in the calculated result. This is because the real window is located in a frame, which was not modelled in this work. The frame causes the central fragments to be blown out first and accelerated quite a bit before the fragments near the frame have been broken free. This causes the particles to spread out, forming the long central plume.

There appears to be more spin in the acutal results than the generated ones. This

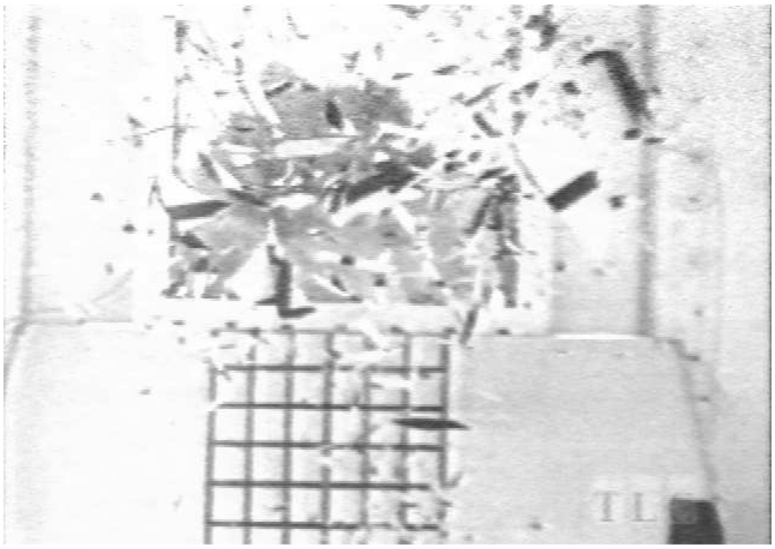


Figure 6.7: This is a frame captured from a video of a blast wave destroying a window. The camera is looking directly at the square window located near the middle and in the upper half of the frame and the glass fragments are being blown towards the camera. The window originally had a black and white grid drawn on it, which is still visible on some of the fragments. The quality of the image is poor, but it is still possible to see the large range of fragment sizes. (from [7])

again is likely caused by the model not tracking the effects of the window frame and not modelling the fracture process over time. If the explosion and fracture model were more tightly coupled, fragments would be broken free

6.7 Future Work

As was mentioned in the opening of this chapter, the fragment pattern was generated separately and then applied to the window. As suggested above, an interesting avenue for future work would be to couple the two models more directly, so that the pressure from the blast wave is used to drive the fracturing process. This would allow fragments to be broken free at different times as the blast wave passed through the window, rather than all the fragments existing a priori. Including the effect of the window frame will also greatly improve the accuracy of the model. No effort was made to model the window frame. Even with this simplified modelling approach, however, reasonable results were achieved, so the potential of the technique appears strong.

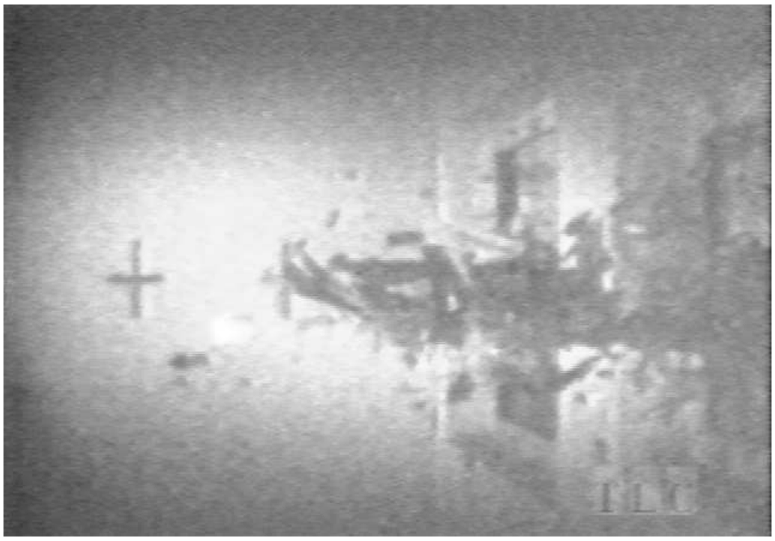


Figure 6.8: This is another captured frame from a video of a blast wave blowing out a window. This is a side few. Two windows are located to right side of the image, one behind the other. The outward plume in the middle of the frame shows the glass shards being blown out from the frint window. The shards are spread over a considerable distance.(from [7])

Chapter 7

Conclusions and Future Directions

A model has been presented for generating animations based upon the effects of blast waves. This model calculates the pressure profiles experienced by objects as a result of the detonation of high explosives. It relies on TNT equivalence and the use of precomputed blast curves to calculate all the parameters necessary. A heuristic model for blast propagation has also been presented along with a model for growing fracture patterns in the plane.

The model was used successfully to generate an animation of an explosion blowing out a window. Such an animation would be difficult to generate using any other technique.

This work aims at a sweet spot between ad hoc approaches to modelling explosions and the rigorous, computationally intensive approaches taken by explosion researchers. The model appears to fit well in this sweet spot. It can generate visually accurate, physically based animations at a reasonable computational cost.

The modelling of explosions is a very complex task and this work only offers a first step on what needs to be a long journey. There are many areas of future investigation that can improve and inform this work. When taking these future steps, however, a researcher must bear in mind how this effort fits in the broader spectrum of explosion modelling. At a certain point, if the desired result is a more accurate model, especially in the area of blast wave propagation, the basis of this work needs to be carefully reconsidered. Significant improvements in accuracy may require replacing this framework in favour of a full three dimensional fluid dynamics model.

That being said, there are several areas that could prove fertile ground for future work on this model. They include improvements and extensions of the base model, the incorporation of new geometry models and improvements to the fracture model.

7.1 Improvements and Extensions to the Base Model

The blast curve approach and the model generated based upon it is more general than may be apparent in this work. It can be extended to generate pressure profiles over the surface of objects rather than just differential loading, as was done here. By combining this with a deformable object model, effects such as the crushing of objects could be modelled.

The model is similarly well suited for calculating loads on large objects that may be anchored and do not translate, but still deform under loading. This would include the loading of large buildings. This is the task that the blast wave approach was first developed for, so the model would be ideal in this work. A possible way of exploring this would be to apply the model to the loading of structures represented by single degree of freedom spring-mass chains.

The current model propagates the wave front spatially, but computes the loading in the time domain. This is due to a lack of information about the spatial structure of a blast wave. Spatially based models for a blast wave could be explored. This might better account for the simultaneous movement of the blast wave and the objects it is loading.

This work has concentrated on the explosive event, rather than the aftermath of the explosion. An interesting area for investigation would be determining what a scene looks like after an explosion has taken place. Objects would need to be tracked until they come to a stop and suitable collision detection would need to be added. Expanding the model in this way would also provide a new set of data with which to check the accuracy of the model. Simulation results could be compared with dispersal patterns from real world explosions.

Another interesting visual effect is the dust cloud associated with the explosion. The modelling work here concentrated on secondary fragments. The model is not considered accurate at the surface of the explosive and, indeed, most of the curves are not defined here. Nonetheless, due to their visual interest, it remains a worthwhile objective to explore techniques for extending the model to primary fragments and dust.

7.2 Geometry

The most significant opportunity for improvement is in a new geometry model. The ideal geometry model would be multi-resolution, queriable and deformable.

The explosion model was developed with an eye towards a multi-resolution geometry such as that developed by Zorin and Schröder [44]. The advantage of such a representation is that it could be used to integrate simulation panels and geometry "panels" into a single framework. The rendered geometry could correspond to a fine level in the hierarchy. The forces could be calculated at a coarser level of the hierarchy. Due to the model's strong dependence on angle of incidence, it could prove an interesting research question to see how coarse a mesh could be used in the simulation and still yield accurate results.

A useful addition would be a set of tools combined with the geometry that could be used to answer questions such as "What is the area of my current silhouette as seen from the bomb?" and "How far is any given point on the surface from the edge of the silhouette?". This would allow the dissipation of reflected pressure due to diffraction waves to be calculated more accurately. It also could be used in damping calculations and for tracking the progress of a wave along an object's surface.

Deformable models would allow objects to be crushed and to stretch before shattering. This would have been useful in the window animation, where it would have been desirable to have the window buckle and then fly into pieces. Spring-mass models are the likely candidate here.

7.3 Improvements to the Fracture Model

One of the simplest extensions to the fracture model would be to use it to grow multiple crack trees. There is some mention of multiple crack nucleation in the literature, so this might be worth exploring. One obvious application of such an extension is in the modelling of cracks in pavement. Normally, there are a large number of independent crack structures in a stretch of pavement.

The model was greatly improved by the use of stochastic variation. It would be ideal if a physical basis could be determined for specifying this stochastic variation. It might be possible to model a pressure gradient field and make use of this. A spring-mass model might also be used to track force distributions which could be used to steer cracks.

Finally, as interesting as blowing out a window is, eventually people will want to shatter arbitrary curved surfaces. Techniques must be developed to propagate cracks on general curved surfaces.

7.4 Conclusion

New discoveries always bring with them new challenges. This thesis attempts to lay an early stone in the foundation for a computer graphics model of explosions. It has achieved nice progress, but there are many more challenges ahead.

Bibliography

[1] S. B. Bazarov and. V. Bazhenova, O.V. Bulat, V.V. Golub, and A.M. Shulmeister. Three-dimensional diffraction of shock wave. In K. Takayama, editor, *Shock Waves: Proceedings, Sendai, Japan*, volume 1 of *Shock Waves*, pages 251–254. Springer-Verlag, 1991.

[2] Michael F. Ashby and David R. H. Jones. *Engineering Materials 1: An Introduction to their Properties and Applications.* Pergamon Press, Oxford, 1980.

[3] Quentin A. Baker, Ming Jun Tang, Ephraim A. Scheier, and Gustavo J. Silva. Vapor cloud explosion analysis. *Process Safety Progress*, 15(2), 1996.

[4] W. E. Baker, P. A. Cox, P. S. Westine, J. J. Kulesz, and R. A. Stehlow. *Explosion Hazards and Evaluation.* Elsevier Scientific Publishing Company, New York, 1983.

[5] William E. Baker. *Explosions in Air.* University of Texas Press, Austin, 1973.

[6] Jon A. Bell and Scot Tumlin. Extremely explosive:: Explosive effects. *3D Design*, 4(8):74–80, August 1998.

[7] Blast masters: The science of explosion, 1997.

[8] David Broek. *Elementary Engineering Fracture Mechanics.* Kluwer Academic Publishers, Dordbrecht, fourth edition, 1986.

[9] Frederick J. Bueche. *Introduction to Physics for Scientists and Engineers.* McGraw-Hill Book Company, New York, fourth edition, 1986.

[10] Loren Carpenter. The A-buffer, an antialiased hidden surface method. In Hank Christiansen, editor, *Computer Graphics (SIGGRAPH '84 Proceedings)*, volume 18, pages 103–108, July 1984.

[11] W. C. Davies. High explosives: the intercation of chemistry and mechanics. *Los Alamos Science*, 2:48–75, 1981.

[12] Wildon Fickett. Detonation in miniature. In John D. Buckmaster, editor, *The Mathematics of Combustion*, pages 133–181. SIAM, Philadelphia, 1985.

[13] Wildon Fickett and William C. Davis. *Detonation*. University of California Press, Berkeley, 1979.

[14] E. Fiume, A. Fournier, and L. Rudolph. A parallel scan conversion algorithm with anti-aliasing for a general purpose ultracomputer. In *Computer Graphics (SIGGRAPH '83 Proceedings)*, volume 17, pages 141–150, July 1983.

[15] James D. Foley, Andries van Dam, Steven K. Feiner, and John F. Hughes. *Computer Graphics: Principles and Practice*. Addison-Wesley Publishing Company, Reading, Massachusetts, second edition, 1993.

[16] Nick Foster and Dimitris Metaxas. Modeling the motion of a hot, turbulent gas. In Turner Whitted, editor, *SIGGRAPH 97 Conference Proceedings*, Annual Conference Series, pages 181–188. ACM SIGGRAPH, Addison Wesley, August 1997.

[17] John C. Hart and Alan Norton. Use of curves in rendering fractures. In Leonard A. Ferrari and Rui J. P. de Figueiredo, editors, *SPIE–The International Society for Optical Engineering: PROCEEDINGS: Curves and Surfaces in Computer Vision and Graphics*, volume 1251, pages 322–328, 1990.

[18] Josef Henrych. *The Dynamics of Explosion and Its Use*. Elsevier Scientific Publishing Company., Amsterdam, 1979.

[19] Richard W. Hertzberg. *Deformation and Fracture Mechanics of Enginneering Materials*. John Wiley and Sons, New York, fourth edition, 1996.

[20] T. Huld, G. Peter, and H. Stadke. Numerical simulation of explosion phenomena in industrial environments. *Juronal of Hazardous Materials*, 46:185–195, 1996.

[21] Gilbert Ford Kinney. *Explosive Shocks in Air.* The MacMillan Company., New York, 1962.

[22] H. Kleine, E. Ritzerfield, and H. Gronig. Shock wave diffraction - new aspects of an old problem. In R. Brun and L. Z. Dumitrescu, editors, *Shock Structure and Kinematics, Balst Waves and Detonations*, volume 4 of *Shock Waves @ Marseille IV*, pages 117 – 122. Springer, 1993.

[23] Erwin Kreysig. *Advanced Engineering Mathematics.* John Wiley and Sons, New York, sixth edition, 1988.

[24] Brian Lawn. *Fracture of Brittle Solids.* Cambridge University Press, Cambridge, second edition, 1993.

[25] Bernard Lewis and Guenther von Elbe. *Combustion Flames and Explosions of Gasses.* Academic Press, New York, second edition, 1961.

[26] Marc Andre Meyers. *Dynamic Behaviour of Materials.* John Wiley and Sons, New York, 1994.

[27] F. Kenton Musgrave. Great balls of fire. In Turner Whitted, editor, *SIGGRAPH 97 Animation Sketches*, Visual Procedings, pages 259 – 268. ACM SIGGRAPH, Addison Wesley, August 1997.

[28] Alan Norton. Tipsy turvy. Siggraph 89 Video Proceedings.

[29] Alan Norton, Greg Turk, Bob Bacon, John Gerth, and Paula Sweeney. Animation of fracture by physical modeling. *The Visual Computer*, 7:210–219, 1991.

[30] Paramount Pictures, 1982. Star Trek II: The Wrath of Khan.

[31] N. R. Popat, C. A Catlin, B. J. Arnzten, R. P. Lindsted, B. H. Hjertager, T. Solberg, O. Saeter, and A. C. Van den Berg. Investigations to improve and assess the accuracy of computational fluid dynamic based explosion models. *Journal of Hazardous Materials*, pages 1–25, 1996.

[32] Todd Reed and Brian Wyvill. Visual simulation of lightning. In Andrew Glassner, editor, *Proceedings of SIGGRAPH '94 (Orlando, Florida, July 24–29, 1994)*, Computer Graphics Proceedings, Annual Conference Series, pages 359–364. ACM SIGGRAPH, ACM Press, July 1994.

[33] W. T. Reeves. Particle systems – a technique for modeling a class of fuzzy objects. *ACM Trans. Graphics*, 2:91–108, April 1983.

[34] P. D. Smith and J. G. Hetherington. *Blast and Ballistic Loading of Structures.* Butterworth and Heinemann Ltd., Oxford, 1994.

[35] Jos Stam. A general animation framework for gaseous phenomena. Technical Report R047, European Research Consortium for Informatics and Mathematics, January 1997.

[36] Jos Stam and Eugene Fiume. Turbulent wind fields for gaseous phenomena. In James T. Kajiya, editor, *Computer Graphics (SIGGRAPH '93 Proceedings)*, volume 27, pages 369–376, August 1993.

[37] Jos Stam and Eugene Fiume. Depicting fire and other gaseous phenomena using diffusion processes. In Robert Cook, editor, *SIGGRAPH 95 Conference Proceedings*, Annual Conference Series, pages 129–136. ACM SIGGRAPH, Addison Wesley, August 1995. held in Los Angeles, California, 06-11 August 1995.

[38] Demetri Terzopoulos and Kurt Fleischer. Deformable models. *The Visual Computer*, 4(6):306–331, December 1988.

[39] Demetri Terzopoulos and Kurt Fleischer. Modeling inelastic deformation: Viscoelasticity, plasticity, fracture. In John Dill, editor, *Computer Graphics (SIGGRAPH '88 Proceedings)*, volume 22, pages 269–278, August 1988.

[40] Demetri Terzopoulos, John Platt, Alan Barr, and Kurt Fleischer. Elastically deformable models. In Maureen C. Stone, editor, *Computer Graphics (SIGGRAPH '87 Proceedings)*, volume 21, pages 205–214, July 1987.

[41] Fabrice Uhl and Jacque Blanc-Talon. Rendering explosions. In *SCS, Military, Government, and Aerospace Simulation*, volume 29, 4, pages 121 – 126, 1997.

[42] Michiel van de Panne and Eugene Fiume. Sensor-actuator networks. In James T. Kajiya, editor, *Computer Graphics (SIGGRAPH '93 Proceedings)*, volume 27, pages 335–342, August 1993.

[43] B. Vanderstraeten, M. Lefebvre, and J. Berghams. A simple blast wave model for bursting spheres based on numerical simulation. *Journal of Hazardous Materials*, 46:145–157, 1996.

[44] Denis Zorin, Peter Schroder, and Wim Sweldens. Interactive multiresolution mesh editing. In Turner Whitted, editor, *SIGGRAPH 97 Conference Proceedings*, Annual Conference Series, pages 259 – 268. ACM SIGGRAPH, Addison Wesley, August 1997.