A Visual Model For Blast Waves and Fracture

Michael Neff Eugene Fiume

Department of Computer Science
University of Toronto

Abstract

The expense, danger, planning and precision required to create explosions suggests that the computational visual modelling of explosions is worthwhile. However, the short time scale at which explosions occur, and their sheer complexity, poses a difficult modelling challenge. After describing the basic phenomenology of explosion events, we present an efficient computational model of isotropic blast wave transport and an algorithm for fracturing objects in their wake. Our model is based on the notion of a *blast curve* that gives the force-loading profile of an explosive material on an object as a function of distance from the explosion's centre. We also describe a technique for fracturing materials loaded by a blast. Our approach is based on the notion of *rapid fracture*: that microfractures in a material together with loading forces seed a fracturing process that quickly spreads across the material and causes it to fragment.

Key words: Fracture, blast waves, explosions, physically based modelling, animation.

1 Introduction

Few phenomena are at once so short lived, so powerful and so awe inspiring as explosions. They are a source of wonder, delight, and destruction. Used widely in many industries, the special effects industry among them, real explosions are costly, dangerous, difficult to control, and impossible to undo. That makes them excellent candidates for visual and physical simulation, but the time scales at which they operate, and the sheer number of physical interactions involved makes this a difficult problem. The benefits are significant, since having a consistent mathematical formulation of a phenomenon can give reproducible user-controlled simulations. Our work is primarily concerned with deriving a mathematical model that gives rise to convincing visual depictions. It is unlikely our work will be of significant relevance to the accurate physical simulation of explosion mechanics.

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

Most previous computer graphics models have concentrated on the explosive cloud portion of the explosion event. One of the earliest related attempts was made by Reeves using particle systems[18]. This work was used in the "Genesis sequence" of the movie *Star Trek: The Wrath of Khan* to generate an explosive burst resulting from the impact of a missile on a moon[16]. More recent research efforts at modelling these effects have been made using both physically based [24, 20] and fractal noise approaches[13].

The focus of this paper is on a physically based, visual model for the blast wave portion of an explosion. A blast wave transports an explosion's damage-causing energy. The wave is a quickly advancing pressure discontinuity that generates large forces on the objects in its path. Our approach provides an accurate basis for calculating these forces. After briefly surveying the scientific literature, we present a blast wave model in the second section of this paper.

Much of the visual excitement of a blast wave relates to its ability to shatter objects. The third section of this paper develops a fracture model based upon the idea of *rapid fracture*, where microcracks in a material are propagated and forked, leading to the shattering of the material. This model is significantly different from previous computer graphics fracture models. We subsequently show how the blast wave and fracture models can be combined to produce an animation of a shattering window. The paper concludes with a discussion of areas for future work.

2 Blast Wave Modelling

2.1 Blast Phenomenology

This work is concerned with the detonation of condensed high explosives, such as TNT, which are capable of generating powerful blast waves. The nature of such an explosion can be illustrated by considering the detonation of a spherical charge of TNT, initiated at the centre of the explosive. For an ideal, uniform explosive, a detonation wave will propagate outward from the centre at great speeds: 6800 m/s in TNT [9]. Detonation propagation speeds are essentially constant and depend on the density of the explosive involved. As the detonation wave passes through the explosive, it generates immense pressure and high temperatures. Pressures are normally in the range of a few thousand atmospheres and the temperatures range between 2000 and 4000K for solid and liquid explosives[6]. These high temperatures and pressures are a result of the extremely rapid chemical reaction that takes place just behind the wavefront. Typically, the reaction is ninety percent complete in between 10^{-6} and 10^{-9} seconds[6].

Following the discussion in [19], 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 in front of the gases is thus formed, which expands outward, containing most of the energy of the explosion. This is the *blast wave*. As the gases expand, their 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 further from the explosive. The blast wave, moving through air, will have a much slower velocity than the detonation wave which is travelling in the explosive.

The above discussion is somewhat simplified, as it ignores the role of wave reflections at the explosive-air boundary. The reflection causes a portion of the wave energy to be sent back toward the explosive centre, where it will "bounce" and be sent outward again. This leads to secondary blast waves, but their strength is significantly lower[9], and will therefore be ignored in this work.

An explosion can cause damage in several ways. The explosive casing is sent outward as small, high speed pieces known as primary fragments. Objects near the explosive are also propelled outward, although at slower speeds, and are called secondary fragments. For large explosives, however, the blast wave is the dominant damage mechanism [10] and will be the focus here. The profile of the pressure pulse is shown in Figure 1. Notice that it represents a jump discontinuity where the pressure increases by P_s. This is the *peak static overpressure* and represents the pressure that is felt by a particle moving with the wave. With time, this pressure decays to below the ambient pressure P_0 due to the overexpansion of gases as described above. P_{min} is the minimum pressure reached, t_a is the arrival time of the pressure pulse and T is the period of the positive phase of the pulse. Often it is sufficient to represent the positive pulse as a triangular wave. There are two ways to do this: one that preserves the period of the pulse, and another that preserves the *impulse* of the pulse. The impulse is defined as the change in momentum[4], and can be calculated as the product of force and time, where time is the duration during which the force was acting 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 $t_a + T$. The pulse's peak overpressure decreases with distance from the centre of the explosion. This affects the shape of the pressure pulse. Pressure pulses for fronts at four different radii are shown in Figure 2.

When a blast wave encounters an object, it will both re-

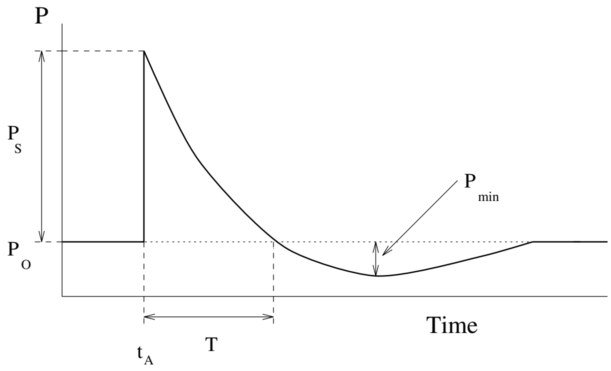

Figure 1: Pressure pulse as a function of time at a point in space.

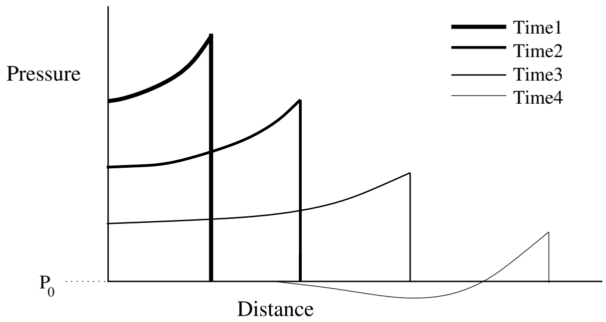

Figure 2: Pressure pulses at four different times show how the shock wave changes as it propagates away from the explosive source.

flect off the object and diffract around it. It will also generate stress waves within the object. The exact behaviour depends upon the angle of incidence and power of the wave and the geometry of the object. Wave propagation is an important topic, but space prevents us from describing our approach here. Some good published research exists on sound-wave propagation and on the impact of occluding and reflecting objects on wave propagation; a good start would be [23]. We develop a heuristic model to account for object occlusions in[14]. Reflection will only be dealt with as it relates to the strength of the initial impact of the wave.

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, in other words, at zero degrees incidence. *Oblique* reflection occurs when the angle of incidence is small, less than about forty degrees in air[19]. *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[10]. The reflecting wave catches up with and fuses with the incident wave to form a third wavefront called the Mach stem.

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 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

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

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. A reflection coefficient, C_r, can be defined as the ratio of P_r to P_s. As the angle of incidence for the blast wave increases (a head-on collision corresponds to zero degrees), the reflection coefficient 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. 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, and hence the amount of loading an object will experience, is related both to the angle of incidence of the blast wave and the magnitude of P_s.

2.2 Techniques for Blast Wave Modelling

There are two main approaches for modelling blast waves. One involves a full mathematical simulation of the explosive event based upon the Navier-Stokes equations of fluid dynamics. The other, coming out of structural engineering research, uses either a simplified set of equations or a set of blast curves to compute values for key quantities related to a blast wave simulation.

Following [7], 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 front of the blast wave was seen to be a jump discontinuity between the ambient conditions in the atmosphere and the high pressure state within the blast wave. The Euler equations enforce the conservation of mass, momentum and energy across this discontinuity. They can be combined with an equation of state for the transmission material to solve for the necessary parameters. These equations have been applied at research facilities such as Los Alamos National Laboratory by scientists studying explosive phenomena to create computer simulations known as hydrocodes[12].

A body of literature in the structural engineering field, 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 [10, 19, 2] 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 explosion. Sometimes these curves are developed using the more expensive Navier-Stokes based simulations, as was done by Vanderstraeten et al.[25].

Two explosives with different masses of TNT will generate the same overpressure, but they would do so at different distances from the explosive centre. Thus, 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 (at some distance) by different amounts of explosives, the mass of the bomb can be combined with distance from the explosive to create a scaled distance parameter. This will allow equations or charts

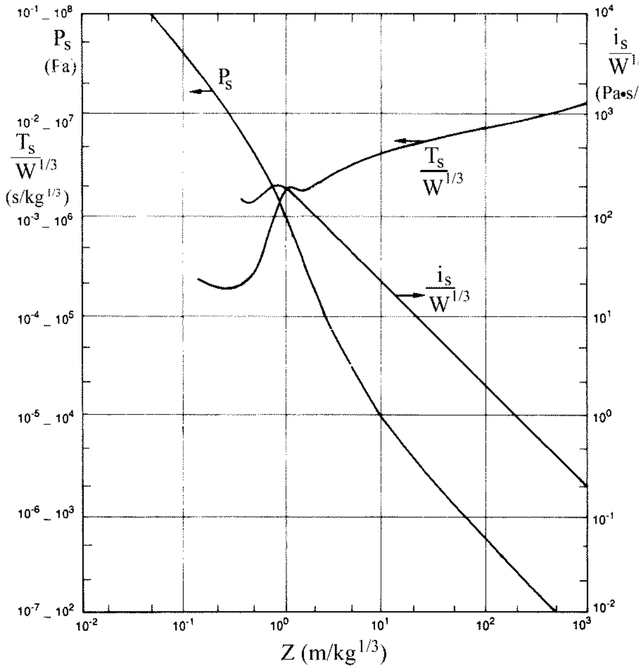

Figure 3: The blast curves for static overpressure(P_S) and scaled pulse period ($\frac{T_S}{W^{\frac{1}{3}}}$) as a function of scaled distance Z (from [19]). These curves are for spherical TNT charges, exploded in air, at ambient conditions.

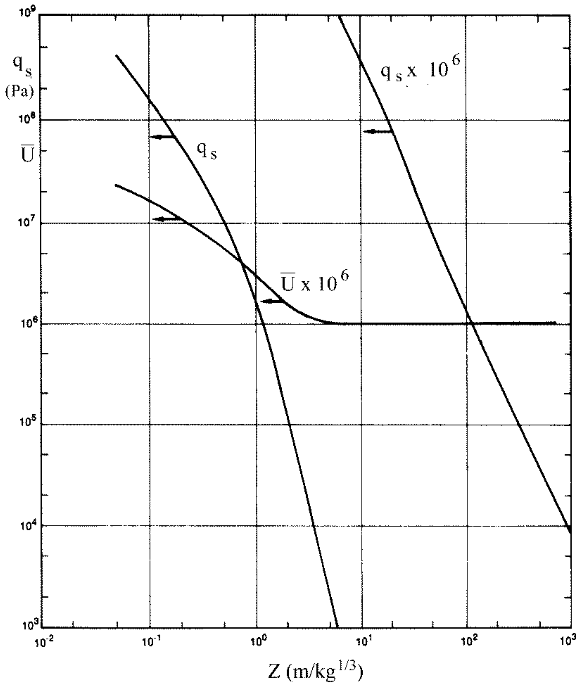

Figure 4: The blast curves for normalized wave velocity ($\overline{U}$) and dynamic pressure q_s as a function of scaled distance Z(from [19]). These curves are for spherical TNT charges, exploded in air, at ambient conditions.

giving the peak overpressure to be defined once using this scaled parameter and then applied for explosives of any mass, greatly simplifying the modelling task. The scaled distance Z is used for this purpose and is defined as follows:

$$Z = \frac{R}{W^{\frac{1}{3}}} \tag{2}$$

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

2.3 Blast Wave Model

The blast curve approach is significantly less computationally expensive than a simulation based upon the Euler Equations. Furthermore, a more complete set of data is available from the blast curves than from the blast equations also used in structural engineering. For these reasons, blast curves are used in our model to provide the physical data needed. The basic idea of the model is to approximate the blast wave as a triangular pressure pulse and propagate this pulse through space, appropriately modifying it as its distance from the explosion centre increases.

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 centre. This is justifiable as blast waves propagate spherically once at a reasonable distance from the explosion centre[2].

Quasi-static loading occurs when the blast 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 faces of the object. Our work will assume quasi-static loading, which allows loading calculations to be restricted to the drag force acting on the front facing panels. This assumption is not necessary in our model, but it simplifies the calculations.

All the blast curves are indexed in terms of the scaled distance parameter Z. Figures 3 and 4 show blast curves for the main blast parameters: static pressure, P_s; dynamic pressure, q; scaled pulse period, $\frac{T}{W^{\frac{1}{3}}}$; and normalized wave velocity, $\overline{U}$. These charts are used to define functions in the model that will return the value of a given

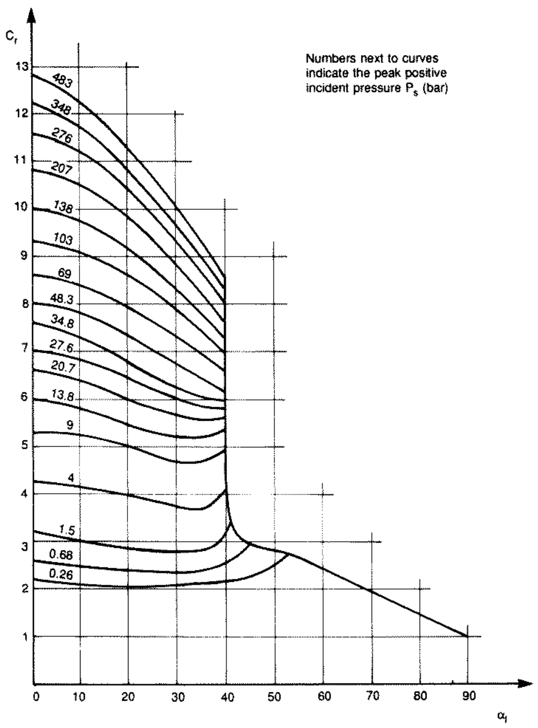

Figure 5: The reflection coefficient (C_r) as a function of both angle of incidence (α) and static overpressure (from [19]). Each curve corresponds to a specific static overpressure (bar) as labelled.

parameter for a given Z.

To model the impact of the blast wave on an object, the reflection pressure must be calculated. The reflection coefficient C_R is used for this purpose. It is multiplied by the static pressure to determine the peak pressure felt by a frontal face. The reflection coefficient is highly sensitive to both the static pressure and the angle of incidence. Figure 5 shows the reflection coefficient plotted as a function of both parameters. 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.

The blast curves are used to calculate the pressure pulse shown in Figure 6, which determines the loading an object experiences. This figure is a superposition of the peak reflected pressure and the drag pressure, both being represented as triangular pulses. The time it takes for the reflected pressure to fall to drag pressure is given approximately by

$$T_2 - T_1 = \frac{3S}{U_s}\,. \tag{3}$$

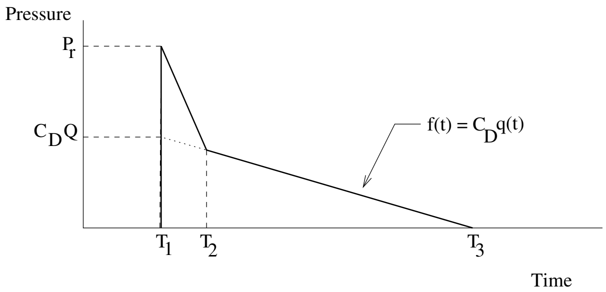

Figure 6: Translatory Pressure versus Time.

S represents the minimum distance the dissipation wave must travel and is defined for rectangular objects as one-half the smaller of the object front's height or base width. U_s is the wave's speed at the front of the object.

Objects in a scene are represented using polygonal meshes. The basic modelling unit for calculating forces in the simulation is a *panel*. The object's geometry is mapped to a series of panels. The panels can be at a coarser, finer or equivalent granularity to the polygons in the mesh, depending upon the resolution required. The minimum set of data a panel must contain is: a centre point, 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. Panels can thus be seen as a point sampling of the blast wave on the surface of the object being struck. Panels record the triangular pulse (impulse) with which they are being loaded.

Forces and torques are calculated for each panel at each time step during its loading. These are then passed on to the panel's object parent where they can be summed to determine the total loading experienced by the object during that time step. Objects store their velocity, angular momentum, inertial tensor and frictional resistance for use in calculating their translatory and rotational movement.

During each time step of the simulation, the blast radius is increased 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 second blast radius is used.

During a preprocessing stage, all front facing panels are loaded into an event queue. The event queue is sorted in order of the panel's radial distance. As the blast's

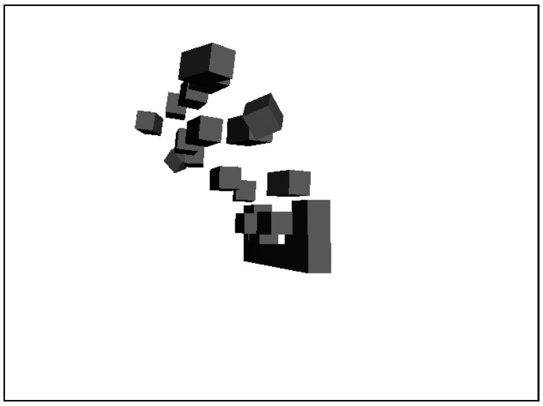

Figure 7: A single frame from an animation of a brick wall blowing apart.

radius increases, it passes over the panels in the event queue. When a panel is first struck, its pressure pulse is calculated, defining the impulse the panel will receive. 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 end of the acceleration phase for the object. The object is now ballistic and is acted upon by the downward force of gravity and the opposing force of wind resistance.

During the loading phase, an impulse is applied to each panel. This impulse is integrated in a stepwise fashion in order to capture the acceleration phase of an object's motion. The acceleration phase is important in creating convincing slow motion animations. Figure 7 shows a single frame from an animation of a brick wall being blown apart.

3 Fracture Modelling

3.1 Background

At least two different models for material breakage exist. The first can be described using a material's stress-strain curve and its physical yield point. A stress-strain curve shows graphically how a material will bend as varying strength loads are applied. When a certain load is exceeded, the material will break. The maximum load a material can sustain is called its yield point. The second breakage mechanism depends on an initial crack being present in the material. Given the length of the crack, a critical stress can be determined that if exceeded, will cause the crack to propagate. This type of breakage normally occurs about two orders of magnitude below the material's theoretical yield point[11]. The latter mechanism is known as *rapid* or *fast fracture*, and is the breakage mechanism that is the basis of our model. Rapid fracture occurs over small time scales and often leads to the fragmentation of the host material.

Following [1], rapid fracture can be illustrated with the example of a partially-inflated balloon. If the balloon 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 cracks on the 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 and stored elastic energy increase. 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 and it occurs below the balloon's material yield point.

During rapid fracture, cracks will propagate at high speeds and will also bifurcate. Figure 8 shows the branching pattern that occurs as a crack propagates. This is a simplified model that 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. R is defined as the the amount of energy required to form a crack divided by the area of the newly formed crack. G is the elastic energy release rate and is considered here to be proportional to the crack length. a_c is the critical crack length at which propagation begins. At this point, $G = R$. 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[3]. In this manner, a crack tree is formed.

3.2 Previous Computer Graphics Fracture Models

Previous attempts at modelling fracture for computer graphics have been based on spring mass models. Much of the work on these models was done by Terzopoulos with various collaborators [21, 22] and was applied specifically to the problem of fracture by Norton et al. [15]. The models represent materials as a grid of distributed masses connected together by springs. Often, the modelling efforts have focused on relatively thin surfaces, although they are 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. These models have been used for tearing paper, breaking a net over a sphere, breaking a solid over a sphere, and breaking a teapot. However, their effectiveness is limited by the need for a dense spring mesh. Furthermore, simulating nonpliant materials results in very stiff systems. The consequence

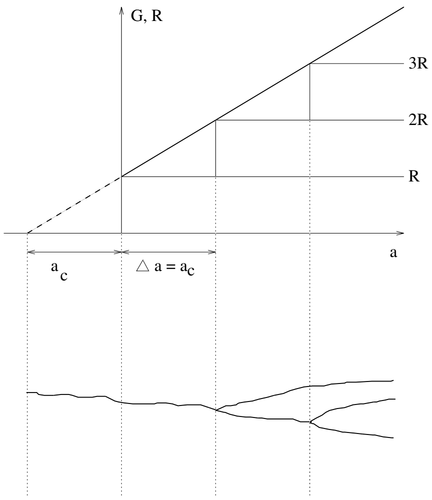

Figure 8: The crack will branch whenever G exceeds a multiple of R.

is that computation times are very slow, and when a material does shatter, it looks unrealistic both in terms of the size of the projectiles and their distribution.

3.3 Fracture Model

Our model makes use of the idea of rapid fracture. It focuses on the sub-problem of generating fracture patterns in a plane. The crack propagation algorithm is based upon the simplified relationship between G and R shown in Figure 8. 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 of its propagating edges 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. For collisions, only the colliding edge is terminated. In the base algorithm, cracks fork at a set angle specified by the user. All crack lines are straight in the base algorithm. Stochastic variations are described below.

The crack tree is maintained as a logical, searchable tree structure. A leaf list is maintained consisting of all edges that are currently propagating. Conceptually, at each time step, every leaf is extended by length a_c. After each such a_c extension, an edge is chosen to fork, creating two new edges. These are added in random order to a FIFO bifurcation queue. Elements of the queue are popped off when a new forking candidate is needed. Forking candidates could also be picked randomly from the list to yield a less coherent structure.

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. Fragments which have the object border as an edge are traced once the crack structure has finished propagating.

The fracture algorithm can be directly coupled to the blast wave model by using the calculated pressure profile to determine the critical length of the initial crack, or they can be controlled independently to maximize flexibility for an animator. 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. The examples below were intended to model a window being held in a frame, and hence the initial crack was placed near the centre of the panel.

The cracks are propagated using a bit bucket structure. A bit bucket is an occupancy grid or coverage mask (cf. [8, 5]) 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$\times$2k bit bucket. This allowed for the specification of very small initial cracks, leading to small fragments. A coarser grid could be used if small fragments were not desired. The grid is marked with pointers to the crack edge occupying a bucket so that when a collision occurs, the two edges involved can be quickly identified in order to trace the chunk.

The algorithm is currently limited to thin surfaces and it is not obviously extensible to three dimensional solids. This could prove interesting territory for future research, possibly involving propagating planes or surfaces.

3.4 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 9 and 10, 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.

Although these patterns are visually interesting, they have too much coherent structure to appear natural. The algorithm is extended by allowing the propagation angle to be randomly computed as the base propagation angle perturbed by up to one-half the maximum variation, which can be set by the user. 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

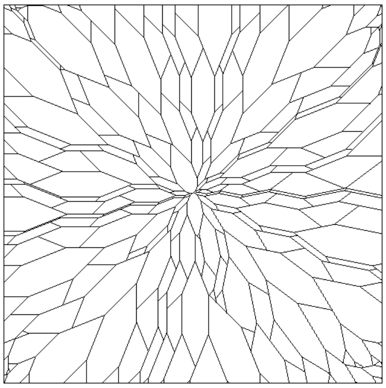

Figure 9: A crack pattern using the base algorithm and a bifurcation angle of 25 degrees.

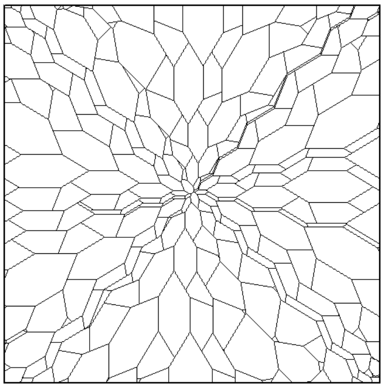

Figure 10: A crack pattern using the base algorithm and a bifurcation angle of 35 degrees.

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 by an animator. 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 parent's orientation perturbed by upt to one-half the maximum variation. There are some similarities between this work and that by Reed and Wyvill on modelling lightning propagation[17], although the details of the propagation algorithm are quite different. Figure 11 shows the combined effect of both line wiggling and varying the propagation angle for a base propagation angle of 35 degrees. Note the vastly improved, more realistic pattern as compared with the base algorithm. To provide a qualitative basis for comparison to real world crack patterns, Figure 12 shows a series of cracks in a section of pavement. 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.

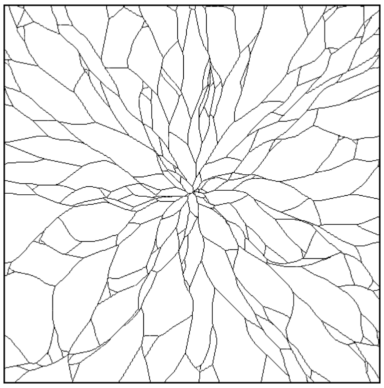

Figure 11: A crack pattern featuring random variation of the bifurcation angle and line wiggling.

4 Results

The fracture algorithm was used to generate the fragments that would be created as a window shatters. This fragment set was coupled with the blast wave model to generate an animation of an exploding window, seen from both the front and the side. In this example, fracture generation is not coupled with the blast wave model in order to maximize animator control. The fragments are first generated separately using the fracture algorithm to create a desireable pattern. The fragments are then passed as a geometry data set to the explosive model for animation. The model does not modify their shape. Inertia tensors are calculated assuming uniform density.

A simple coupling between the two models is possible where the amount of pressure exerted by the blast wave is used to calculate length of the crack which seeds the fragment growth algorithm. An interesting area for future work would be to actively use the pressure profile determined by the explosion model in steering the crack growth.

Several frames from these animations are shown in Figures 13 and 14 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. This means that larger particles experience the much higher reflected pressure for a longer duration 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 more greatly damped, allow-

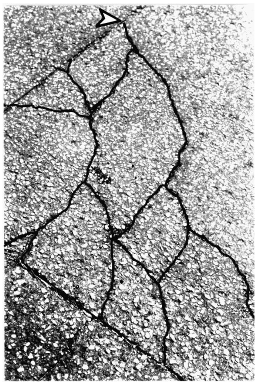

Figure 12: A photograph of cracks in a section of pavement. (The contrast of the photo has been increased and the cracks have been outlined in black to make them easier to see.) To trace the crack tree pattern, start at the arrow and trace the cracks downward. Notice that the pattern is consistent with a crack propagating forward 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.

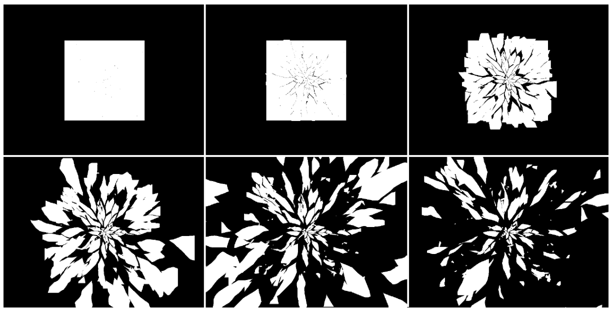

Figure 13: Six equally spaced frames from an animation of a shattering window. Front view.

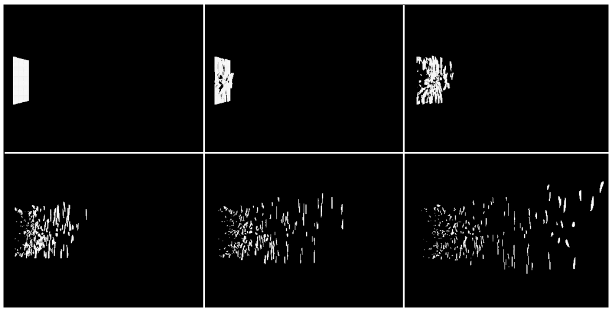

Figure 14: Six equally spaced frames from an animation of a shattering window. Side view.

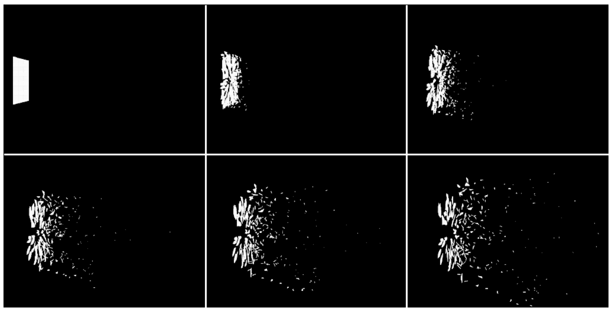

Figure 15: Six equally spaced frames from an animation of an over damped shattering window. Damping is 1000 times normal. Side view.

ing 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 15. Notice that the larger particles undergo more rapid initial acceleration, but are overtaken by the smaller particles. Animators can adjust the damping in the system.

The crack patterns can be grown at interactive rates. The glass window animation takes less than two seconds a frame (on a 300 MHz Pentium II workstation). The brick wall example requires less than one second a frame.

5 Conclusions and Future Work

New methods for modelling fracture and blast waves have been presented. These methods have been successfully employed to generate an animation of a shattering window. There are several interesting directions in which this work can be taken, including: extending the fracture algorithm to run on arbitrary surfaces and solids, combining the blast wave model with an explosive cloud model, and finding a physical basis for "steering" the propagating cracks.

6 Acknowledgements

Michiel van de Panne and James Stewart both provided useful input during the course of this work. The authors gratefully acknowledges the financial support received through a Natural Science and Engineering Research Council Post Graduate Scholarshipi, Research Grant. We also thank CITO for their financial support.

7 References

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

[2] 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.

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

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

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

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

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

[8] 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.

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

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

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

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

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

[14] Michael Neff. A visual model for blast waves and fracture. Master's thesis, Department of Computer Science, University of Toronto, 1998.

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

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

[17] Todd Reed and Brian Wyvill. Visual simulation of lightning. In *Proceedings of SIGGRAPH '94 (Orlando, Florida, July 24–29, 1994)*, pages 359–364, July 1994.

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

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

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

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

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

[23] Nicolas Tsingos and Jean-Dominique Gascuel. Soundtracks for computer animation: Sound rendering in dynamic environments with occlusions. In *Proceedings: Graphics Interface '97*, pages 359–364, May 1997.

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

[25] 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.