EG UK Theory and Practice of Computer Graphics (2005)
L. Lever, M. McDerby (Editors)

Simulating the Cumulative Effects of Multiple Impacts using 'Fracture Maps'

T. Clifton†

School of Informatics, University of Wales, Bangor

Abstract

Abstract Much research has been carried out within the computer graphics community to simulate the effects of collisions between deformable and rigid bodies, but little has been proposed to take into account the effects of one collision, on later impacts. We present a novel approach to model and retain information regarding specific impacts that can be used to better approximate the results of future collisions, taking into account non-visible effects caused as objects collide within a scene. We propose the notion of 'fracture maps' to store and evaluate stress and strain factors for polygonal meshes in order to accommodate the cumulative effects of impacts, even if these effects are not immediately visible to the human eye. We describe the initial work carried out towards this research, including the methodology to create and update these impacts and how they can be used to determine both deformation and decimation of meshes from multiple impacts. We discuss requirements for completing the research, and future directions we would extend the concept further.

Categories and Subject Descriptors (according to ACM CCS): I.3.6 [Computer Graphics]: Methodology and Techniques

1. Introduction

A wide variety research has been carried out within the Computer Graphics community to simulate the effects of deformation and decimation of polygonal meshes as a result of simulated impacts. This research has tended to focus on one-off events, with no consideration given to the cumulative effect of a series of impacts upon the structures of the simulated objects, and therefore their reactions to stresses and strains exerted in future collisions. In any simulation of a real-world impact, the majority of materials will exhibit some form of impact memory, in that their molecular structure may be affected by the impact even if these effects are not immediately visible. These molecular alterations need to be considered if future impacts with the material are to be realistic, and not be simply re-calibrated from an 'unhit' mesh state. Glass is an example of this phenomenon which is easily visualised, Initial impacts on a glass sheet may cause cracking, which affects the strength of the sheet in the particular areas where the cracks become evident. Future impacts are likely to be more destructive in these areas due to the molecular level effects produced by the forces incident on the sheet from the initial impact. Obviously glass gives a visual indication of the effects of multiple impacts, but this situation is evident in a number of other material combinations, even though the effects may not be visible to the naked human eye. This paper focuses on the determination and storage of the residual effects of an impact through the use of what we have termed 'fracture maps'. We extend the traditional notion of texture maps to maintain persistent state information for a particular polygonal mesh, or section of mesh, which can then be used in future impact calculations to more realistically simulate the cumulative effect of multiple impacts.

This paper is intended to demonstrate the theory of this ongoing research, and discuss the major issues our implementation intends to address. Much of the work towards the aim of using fracture maps to simulate cumulative impact effects has yet to be carried out. The paper is structured as follows: In section two we discuss the notion of impact simulation, previous work in the area, and also related work in the field of crack propagation which has a bearing on our ap-

† terence@informatics.bangor.ac.uk

proach. Our methodology is discussed in section three, exploring the generation of our fracture maps, and also their use at impact time to calculate and visualise deformation and decimation of meshes involved in collision. We conclude with a discussion of the current state of our development, and the avenues of future work we hope to pursue as this research takes shape.

2. Background & Previous Work

The simulation of impacts between objects within a computer-generated scene has been an appropriate area of research for many varying strands of the computer graphics community, including medical simulation, entertainment and virtual reality. Many techniques have been proposed over the years to simulate the effects of an impact on the initial mesh, and also to realistically represent the resulting debris and object destruction. Very little research has been carried out into the cumulative effects of impacts, and how one impact may affect the future stability, rigidity, or reaction of an object to future collisions. Many of the common materials that we simulate within graphical environments exhibit some form of memory in terms of alterations to their molecular structure which although invisible to the naked eye, affect the material's properties, and reaction to future events. Simulating impacts as a one-off event, with all parameters reset for each new collision, although simpler, is not, in our opinion realistic in terms of how such events occur in the real-world. We need to be able to take into account previous events and ensure that the materials in a scene react in a consistent manner with relation to these events.

Much research has been done on the subject of object deformation, whether rigid or soft-body based, with [GFG*98] providing a particularly good survey of the field up to 1997. Of specific relevance to the work we are undertaking is that of O'Brien et al on the modelling and animation of both brittle and ductile fracture [OH99, OBH02]. Although material properties are taken into account to create realistic ductile fracture within their system, reactionary aspects of the material are only considered over a single impact, with no information carried forward to future events. Effectively the mesh becomes a 'new object' which simply exhibits the same geometric properties as that which completed the previous impact. This may mean that it has been decimated or deformed, but none of the residual effects of this deformation are available to impact calculations later in the simulation.

The adaptation of the classic texture mapping algorithms to different uses in visualisation and rendering have been common research themes in the graphics community, with such application areas as light mapping and shadows being very popular. Of particular relevance in this area is the work of Wrotek, Rice and McGuire [WRM04] on the use of bump mapping techniques to simulate deformation in real-time using graphics hardware. They adopt a technique proposed by [SdS01] to create a one-to-one mapping between their bump map and objects within the scene. This differs from our approach slightly in that our approach uses key point mappings, based upon vertex positions and interpolates between these positions at collision time to locate positions in the fracture map, whereas their approach calculates all possible mappings in advance.

[HTK98, HTK00] and [NTB*91], among others, propose methods to simulate crack propagation, based on a Spring-Network model, which offers a discrete representation of the objects as a series of nodes, each carrying a mass and connected by 'springs'. Stress tensors are then evaluated at each node to describe the stress distribution over the whole model, and thus the likelihood of cracking at particular node/spring locations.

3. Methodology

The approach we propose makes use of two distinct stages, map configuration and impact management. The former being a one-off process carried out during scene initialisation, and the latter being a continuous process that is evaluated as collisions are detected within the scene.

3.1. Map Configuration

The creation of a suitable 'fracture map' in memory is handled as an offline process that is initiated when the mesh is created. Maps are created to evenly cover the mesh at a suitable resolution to allow minimal disfiguring of a 256x256 texture. This is done by 'growing' a series of textures, using the polygons of the original mesh. Taking a central polygon, we add adjacent polygons incrementally in a two dimensional fashion until we have a configuration of a suitable size for a 256x256 texture, which is stored as a 256-level greyscale map, The level of gray within a particular pixel in the map corresponds to the 'stored impact value' of that particular region of the mesh, and therefore how prone that particular section will be to fracture in the future. As we accumulate impacts, so we add to this grey level, and the region of mesh becomes 'weaker' as it approaches it's deformation or fracture limit.

Once we have a suitable map, we can associate key points on a particular mesh to points within the map. This gives us a series of mappings which can be used to handle the propagation of impact stresses when we read an impact at a certain point in the mesh. Simple interpolation is used at the texture level for impacts which are registered on any section of the mesh which isn't a specified key point. We calculate the key points (which we have termed 'impact zones') using a simple rule based approach based on changes in mesh gradient, and projection of vertices. As a result, the points selected as key points form the extremities of a particular mesh, and are therefore more likely to be involved in the majority of impacts.

When configuring a fracture map for a particular element

within a scene we maintain global material characteristics which are used in the calculation of fracture map changes when impacts occur. This allows us to specify such global properties such as material type, tensile and compression strength, and deformation, crack and fracture thresholds, and possibly recovery properties. These parameters will determine the reaction of the particular material, and the underlying map to the impacts to which it is subjected. We can also specify a global formula for impact propagation through a particular map, which may reflect molecular properties of the material, or known fracture patterns evident in that particular class of object.

3.2. Impact Management

Managing impacts is carried out as a result of reading a collision between suitable objects in the scene, and follows a three-step process:

1. Determination of impact properties
2. Update of the fracture map for the affected mesh/region
3. Re-meshing of the polygonal object if deformation or decimation is required

The first two steps of this process are carried out fully at each collision, whereas the third is only carried out if the resulting changes to the fracture map mean that it has exceeded the thresholds set globally for this particular material.

3.2.1. Impact Properties

In order to calculate the effects of a particular impact in terms of the changes made to the associated fracture map for the mesh involved, we base our calculations on impact parameters read at the point of impact. This involves the initial determination of the correct point of impact on the mesh, and the corresponding location on the fracture map. For simplicity in calculation we assume impacts occur at a unique point, rather than upon an area of the mesh, and thus are able to determine a single pixel location as the 'virtual epicentre' of our impact, which we can then use to determine how our fracture map is altered as the impact propagates through it. We also determine the force of a particular impact, based upon the global characteristics of the objects involved, and the nature of the collision.. Simple calculations as to the velocity and mass of the objects involved allow us to determine a 'unit impact speed', which gives us a ratio of the current impact in relation to a an impact caused by a 1 kilogram object on a stationary object of our particular material . This ensures that we can create suitable fracture propagation characteristics, knowing that the events will be normalised to an equivalent impact force, and that the results will therefore be consistent for all impacts. In addition to these force parameters, we determine the vector of impact which represents the equivalent incident vector of the object causing the impact on our 'fracture map'. This allows us to use this vector when we determine the propagation of the impact through our map.

Figure 1: *Series of impacts using simple radial propagation*

Figure 2: *Impacts using noise formula for propagation*

3.2.2. Updating the Fracture Map

Once we have a suitable set of parameters from the impact, we are able to calculate the required changes to our fracture map to reflect this particular impact on our mesh. We make use of the global propagation formula for the particular material and map with which we are dealing, and apply the impacting force to generate a new map. This could be just a radial distribution of the impact force from the point of impact (Figure 1), or a more complex function that calculates propagation based on molecular construction, or using simple noise formulae (Figure 2). As we propagate our impact force through the map we simply add to the greyscale level at each particular point to represent the incidence of the new impact force. Prior to applying these forces, we need to factor in the global recovery parameters of the particular material to ensure we account for the fact that some materials will recover rigidity over time. For example, some plastics will be particularly weak immediately after a collision, and become ductile, but as they time passes, they will return to a more stable state and recover their rigidity. We need to take into account the time since the last map update and the recovery properties of the material to give a realistic cumulative impact effect.

3.2.3. Re-meshing

Once we have a fracture map for objects in our scene, we can use these to determine the requirements for deformation and fracture by using the global material properties and the

pixel values of the map. Each map has a series of threshold properties which account for the levels of stress at which it will deform, crack, or fracture. We can trace the elements of the map which exceed a particular threshold to determine where to re-mesh our objects in order to display these crack patterns, or where to deform the mesh. Once we have a series of crack lines, we can map these back to our original mesh using the impact zone mapping we stored in our initial calculations, and perform a partial re-mesh, taking into account impact parameters such as the directional vector. If, when we perform our re-mesh, we decimate a section of the original mesh, we have all the date required to simulate the ejection of this portion of the object within the scene. In addition, the nature of the fracture map system means that we do not have to recreate our map when we perform a re-mesh, as any additional propagation through elements of the map which have already exceeded the decimation threshold can be ignored, based on the map values.

4. Discussion & Future Work

This paper presents a work in progress for our research into the use of fracture maps to simulate the cumulative effects of multiple impacts, and a number of the concepts presented in this paper are still in the process of development. Our current software is capable of creating and updating fracture maps using simple fracture propagation patterns. We still have to implement the re-meshing system, and would like to approach the propagation formulae from a more physically based perspective, taking better account material properties, in terms of strength, and molecular properties which would affect crack propagation. Figures 3 and 4 demonstrate the current state of the system. Figure x shows a pre-impact screen, about to be struck by a ball. Figure x shows the screen post impact, with the detail texture removed and the fracture map visible. We can see the propagation of the impact over a small area of the mesh using a simple radial pattern perturbed with a degree of noise.

Further work would then be focused on the use of these maps to control the deformation of the objects. Once we have a simple re-meshing system in place we should be able to factor into our initial texture creation process a series mappings from greyscale thresholds to deformation values, to allow for realistic bending and flexing of materials, both at the point of impact, and also as a result of the recovery process. In theory this should allow us to realistically simulate the recoil evident in typical materials post-impact.

We hope to link this research into our existing software designed to simulate impact debris, and drive the debris system using the fracture map, whereby the level of map propagation, couple with the mesh parameters is able to completely determine the quantity, location and direction of ejected debris.

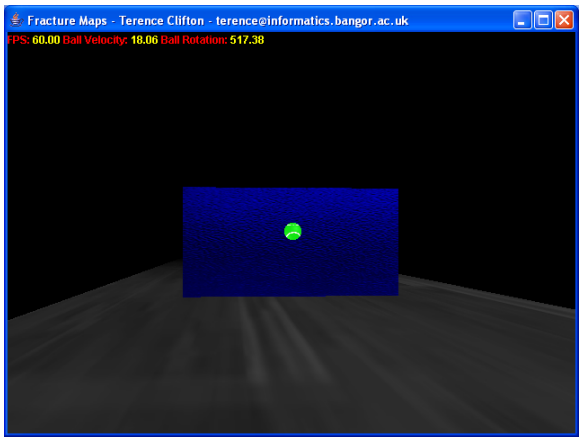

Figure 3: *Scene pre-impact.*

Figure 4: *Scene post-impact, showing the fracture map.*

References

[GFG*98] GIBSON S., FYOCK C., GRIMSON E., KANADE T., KIKINIS R., LAUER H., MCKENZIE N., MOR A., NAKAJIMA S., OHKAMI H., OSBORNE R., SAMOSKY J., SAWADA A.: Simulating surgery using volumetric object representations, real-time volume rendering, and haptic feedback. *Medical Image Analysis 2*, 2 (1998), 121–132. 2

[HTK98] HIROTA K., TANOUE Y., KANEKO T.: Generation of crack patterns with a physical model. *The Visual Computer 14*, 3 (1998), 126–137. 2

[HTK00] HIROTA K., TANOUE Y., KANEKO T.: Simulation of three-dimensional cracks. *The Visual Computer 16*, 7 (2000), 371–378. 2

[NTB*91] NORTON A., TURK G., BACON B., GERTH J.,

SWEENEY P.: Animation of fracture by physical modeling. *Vis. Comput. 7*, 4 (1991), 210–219. 2

[OBH02] O'BRIEN J. F., BARGTEIL A. W., HODGINS J. K.: Graphical modeling and animation of ductile fracture. In *Proceedings of ACM SIGGRAPH 2002* (Aug. 2002), ACM Press, pp. 291–294. 2

[OH99] O'BRIEN J. F., HODGINS J. K.: Graphical modeling and animation of brittle fracture. In *Proceedings of ACM SIGGRAPH 1999* (Aug. 1999), ACM Press/Addison-Wesley Publishing Co., pp. 137–146. 2

[SdS01] SHEFFER A., DE STURLER E.: Parameterization of faceted surfaces for meshing using angle based flattening. *Engineering with Computers 17*, 3 (2001), 326–337. 2

[WRM04] WROTEK P., RICE A., MCGUIRE M.: Real-time bump map deformations. In *Proceedings of the 31st annual conference on Computer graphics and interactive techniques* (2004). 2