Fracture Modeling in Computer Graphics

A survey

by

Lien Muguercia Torres

Ing., Universidad de las Ciencias Informaticas, Cuba 2007

A THESIS SUBMITTED IN PARTIAL FULFILMENT OF
THE REQUIREMENTS FOR THE DEGREE OF

Master universitario en Informatica Industrial, Automatica, Computacion y Sistemas

(MIIACS)

Advisors:
Dr. Gustavo A. Patow
Dr. Carles Bosch

Universitat de Girona

July, 2011

Abstract

This master thesis focuses on the study of the state of the art on crack and fracture modeling. This process is one of the aging phenomena with an important amount of work done, due to its complexity and its extended application area. The variety of parameters and the complex simulation (usually implying collision detection, deformations, breakage, etc), make difficult to think on a generic solution. Thus, we cannot consider it as a simple problem to solve, but as many difficult problems to deal with.

The study is divided as follows. First, we expose the theoretical background concerning aging phenomena, and their behavior in nature. Due to the amount of work already proposed in Computer Graphics, we review and classify the existing techniques according to their kind of approach. The second part is focused on physically-based methods, presenting the work that has been done along with their advantages and disadvantages. Another chapter is dedicated to the less popular, but equally important, non-physically based methods. Finally, we conclude making a brief analysis about the main characteristics of these method, and some open problems.

Table of Contents

List of Figures

Acknowledgements

I want to thank my advisor Carles Bosch for helping me and encouraging me. Also thanks to Xavier Pueyo for giving me the opportunity to be part of this group today and for all his support.

Thanks to my Cubans colleagues for encouraging me at every time. I also thank to Gus, to my *GGG* group and office co, for his every day.

De manera especial, gracias a mis padres por estar siempre presentes y disponibles para mi, por su ejemplo y apoyo. Finalmente, gracias a ti, Paco, por ser paciente y estar a mi lado apoyandome cada dia. Gracias.

Dedication

A mis padres, Ramona y Armando.

Chapter 1

Introduction

One of the challenges in computer graphics, has been to reproduce, as well as possible, natural phenomena. One phenomenon in this field that has been extensively studied is the aging process.

What makes this phenomenon interesting is the great variety and complexity of scientific concepts that are involved, since we have to deal with a huge number of factors. Sometimes, it is not necessary to understand completely the physical concepts, because with the adaptation and use of some parts of this theory to our problem is enough for the simulation. Another important point is the variety of techniques involved in the visual simulation. Similar computer graphics areas are not capable to work independently without cooperation between them. Furthermore, aging processes often involve changes in reflection properties and geometry at the same time, as for example, cracks and fractures growth.

The cracks and fracture modeling and rendering process is one of the aging processes with an important amount of work done, due to its complexity and its extended application area. A generic solution for this process is generally not possible, as materials may behave in very different ways and there is not a generic simulation technique or representation that works well in all cases. Thus, we cannot consider the crack and fracture process as a single problem to solve, but as many difficult problems to deal with.

Facing the great diversity of concepts and techniques, knowing not everything is done and there is work to improve, we will briefly describe in this thesis the current aging simulation techniques, as well as a survey of existing techniques in computer graphics concerning to the specific processes of cracks and fractures.

This thesis is divided as follows. Chapter 2 presents the background concerning aging phenomena, and more specifically to cracks and fractures.

Due to the amount of work proposed in the literature concerning to these phenomena, we propose a classification according to their approach. In Chapter 3 the physically-based methods for those that prefer very realistic results without paying less attention to the computational time, while Chapter 4 is based on non-physical methods, for those that need real-time

solutions where the realism is a secondary factor.

Finally, our conclusions are exposed in Chapter 5 along with the future work and some open problems.

Chapter 2

Background

In this chapter we first expose some definitions and classifications concerning to weathering and aging processes in general, followed by a brief review of the literature and previous work related to weathering and aging processes in computer graphics. We then concentrate on describing how cracks and fractures behave in nature. We finally expose the main concepts involved by the physical models used to simulate fracture and how they use to be implemented.

2.1 Definitions and classifications

Aging phenomena are every process that make damage over an object and deteriorates it, affecting its appearance, structure or geometry. These damages could come from:

- External factors: atmospheric conditions, usage, mechanical damage or erosion, and organic material growth.
- Internal factors: intrinsic to the material itself, mainly due to the manufacturing process.

According to the classification made by Lu et al [49], if we consider a pure and natural surface, this one could be affected by different aging processes from different sources:

- Chemical attacks: these attacks can occur both in the material and onto its surface. They usually transform the original material in a substance with different composition and physical characteristics. This new substance is generally softer and susceptible to damages than the original. They can also introduce specific structural damages, like destructive corrosion, for example.
- Mechanical damages: resulting from external factors, the material is removed from the original surface, affecting its geometry. This can occur at every geometric scale. On a large scale, visible parts of the affected object are removed, as on surfaces affected by impacts and fractures for example. At a smaller scale, invisible parts of the affected object are removed or modified, such as scratches on a surface and cracks.
- Biological: these processes can result from external organic materials attacks (such as lichen growth onto a surface) as well as to internal biological modifications of the materials themselves.

An important thing to remark is that none of the previous classifications has a defined limit. Some processes can be affected by several factors, starting due to a specific factor and then involving another one. To create realistic scenes which involve the different aging phenomena, it is thus necessary to implement and handle all theses processes.

2.2 Weathering and aging processes in Computer Graphics

In computer graphics, there are three main ways of reproducing aging and weathering phenomena:

- Simulation: Each phenomenon is treated independently, using physically-based simulations or approximations that are specific to each case.
- Measurement-based techniques (also called capture-and-transfer): Weathering data are extracted from images or more complex acquisition processes and transferred to new objects.
- Generic models: The aim of these techniques is to provide a generic control appropriate to a large amount of aging processes, and generally introduce important simplifications. They usually focus on the reproduction of the final result rather than simulating the underlying process.

In the next sections, we explain each of these approaches. For more details, see references [58] [44] [24] [97] [49].

2.2.1 Specific aging models

This section presents works related to the simulation of specific aging processes. For each process, we first describe its nature and then review the existing work in the literature.

Mechanical aging

Dust accumulation. Dust consists of small particles and fine matter deposited onto the surfaces through the effect of wind or gravity. They usually affect the appearance of the object by modifying its reflection properties. Blinn [24], first addressed this phenomenon by accounting for light interactions within the particles. Hsu and Wong [40] improved on this introducing an empirical method for simulating the dust accumulation over object surfaces taking into account the inclination and exposure. Later, Wang et al. [94] simulated the accumulation of dust using a set of dust sources, similar to light sources. These sources "emit" dust that is accumulated on the surfaces. Dust accumulation has also been investigated in other cases like the case of moving objects or impacts [96] [3].

Scratches and impacts. Objects are affected by mechanical interactions with some external tools or other objects. These interactions can generate a big variety of defects, which can be classified in two main groups. Some defects can influence the aspect of the object without altering the geometry visibly, while other alter the geometry in a visible way. In the first group we can find the work made by Merillou et al. [57] who propose a method for simulating scratches based on a combination of a 2D texture containing the scratch paths and a $BRDF$ model for simulating their reflection. They use physical measurements on real objects to derive a geometry model of scratches at a small scale. Bosch et al. [10] propose a similar method derived from physical parameters such as the scratch tools, penetration forces and the material properties. The scratch path can be controlled by texture or can be defined with curves. In the second group we can find Paquete et al. [74], who focus in the specific case of impacts. Their method uses a tool to perform impacts in the surfaces, which are then refined in the affected regions.

Cracks and fractures. Cracks can affect a large variety of materials. Usually they are defined as a result of contractions of the surface area of the object. However, they can also appear due to thermal and mechanical labor. The object cracks or breaks into pieces due to the stress differential in the material. The methods proposed to this area will be explained in deep in the next chapters, particularly concerning fracture and their animation [70].

Cracks and peeling. Painted surfaces are also affected by cracks on the paint layer after time, which depends of many parameters like environment, paint quality, and density. These kinds of cracks also provoke the peeling of the paint layer over the time. Gobron and Chiba [33] allow direct peeling by generating the crack patterns using their 3D cellular automata (CA) [81]. This crack pattern generates detached pieces as geometry, and specifies their order of detachment. Paquette et al. [75] based on Gobron's work, introduce physical corrections in the peeling simulation. In this case, the cracks appear according to the paint force and tensile stress, and peeling due to the loss of paint adhesion around these cracks.

Erosion. Erosion appears due to the combination of many processes that can be inside mechanical and chemical classification, but can also be influenced by biological processes and the environment. Liu et al. [47] has proposed a method based on flow, using 2D Navier-Stokes equations to simulate water penetration from thin films of surface flows into materials. Musgrave et al. [63] propose to erode a fractal terrain represented as a regular height field. Then, Nagashima [65] proposed a physically model based on water flow erosion and rainfall. Valette et al. [89] propose a model of soil

surface degradation by rainfall based on a 3D CA, allowing to control the different processes involved (detachment, transport, and infiltration). Ito et al. [41] propose a specific pattern concerning the weathering of joined rocks. Rocks are modelled as voxels and propose a 3D erosion model, removing these voxels depending on parameters like the friction angle, for instance. Benes and Arriaga [7] have proposed another specific study, presenting a terrain consisting on a height field of rock that erodes and transforms, in some cases, into sand.

Flows and matter projection. The flow of dirty fluids is an important effect that affects the aspect of many kinds of objects, from vehicles to building walls for example. In this area, Dorsey et al. [21], model flow as a particle system of raindrops. The flow movement is controlled by parameters like the gravity force, the friction, roughness and absorption of the material surface. They also reproduce the sedimentation effect by a set of differential equations.

Chemical attacks

Patina. Metals can be affected in very different ways by aging phenomena. Patina is one of these affections and consists on the oxidation of the material surface layer that, over time, derive on a weathered surface. Dorsey et al. [20] propose a model of patina by modeling the affected surfaces as a set of layers, where each layer is a material. The layers are represented by using textures to control the various material parameters. Chang and Shih [13] propose a physical model based on a set of L-system rules guided by gravity, curvature and soil. The patina is a layered structure where the vertices of the model are chosen randomly and passed to the L-system rules.

Destructive corrosion. The corrosion can augment the aging rate of a material and is usually previously affected by other kinds of attacks. Destructive corrosion has been investigated by Merillou et al. [57], where they propose a physical model controlled by intuitive parameters, choosing the kind of corrosion between uniform, galvanic, pitting and crevice corrosion; and then a map allow the control of the surface and the volume spreading according to the previously defined corrosion types.

Tarnishing. Metals exposed to atmospheric conditions generally develop tarnish, a thin layer of corrosion present in every metal subjected to oxidation processes. Miller [29] proposed a technique to simulate it by modulating light reflection with accessibility, defined as the radius of the major sphere touching a surface point without intersecting any surface.

Biological processes

Organic material growth. Organic growth appears on surfaces exposed to external biological processes. Because of their color and geometry aspects, these aging processes are visually important. This growing depends of humidity to start and progress. Desbenoit et al. [18] have proposed a model to generate this organic material onto surfaces, focusing on lichens. These authors introduce a method called Open Diffusion Limited Aggregation to compute the growth of the lichen. It permits to model the propagation of lichens interacting with the environment and competing for favorable conditions. Figure 2.1 show a result obtained with this method.

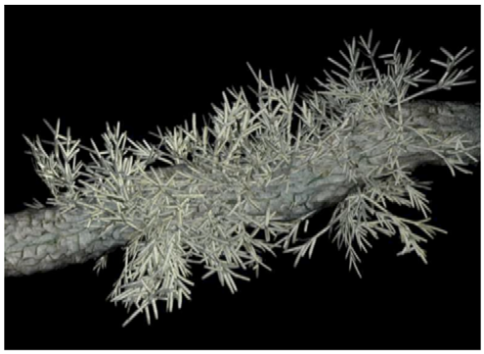

Figure 2.1: A synthetic lichen obtained with Desbenoit method [18].

Wrinkles. Wrinkles can affect many different materials in a natural way. Common examples are aged fruits and aging in human skin. Blinn [8] simulates wrinkles using its well-known bump mapping technique. Wu et al. [95] has presented an specific study of aging in human beings. Human aging, specifically facial aging, is a very active area in the computer vision research field [79].

2.2.2 Global aging models

As natural scenes are very complex, computing a realistic aged scene may involve the use of numerous techniques as shown above. The ideal solution would be do try to handle the problem as a generic problem. Some investigators have been working in this direction, but the results are not completely exacts. In this category, two main approaches can be found: simulating aging as particles and using texture synthesis.

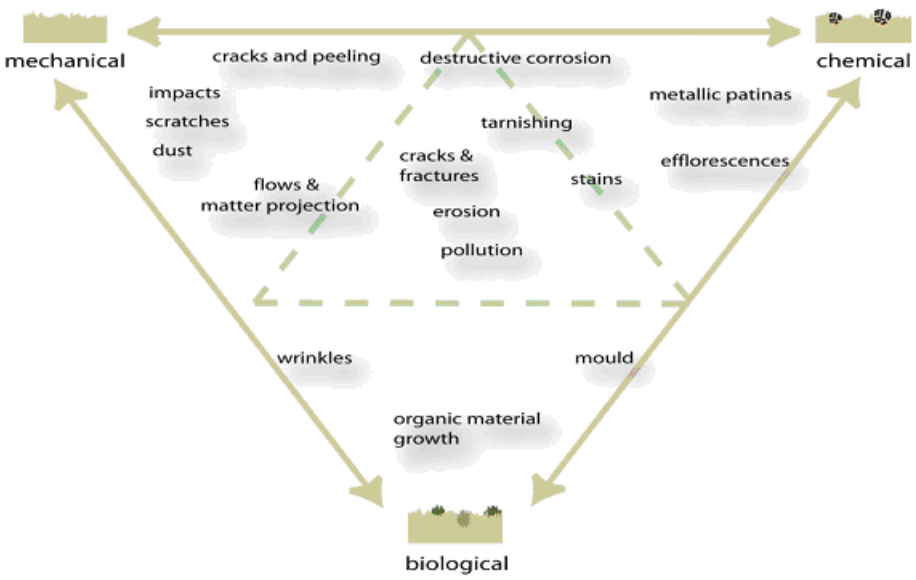

Figure 2.2: This diagram summarizes the classification of the relationship between the most common aging processes according to the three main kind of attacks. From Merillou [57].

Aging as particles

The work of Wong et al. [94] and Chen et al. [97] are included here. The first one, considers that the affected object's underlying geometry is usually still observable, and propose to use 3D texturing to handle aging processes, which is controlled by a tendency distribution, representing the potential occurrence of surface imperfections. Their framework includes dust accumulation, patinas and peeling. The second author presents a visual simulation technique that handles a variety of weathering phenomena by a particle-based approach called γ-ton tracing. These γ-tons are particles that are traced through the scene to represent some specific weathering information and enabling visual simulation of complex multi-weathering effects.

Texture synthesis approach

This set of methods avoid the difficulties of physical and mathematical modeling of the surface appearance varying modeling. This goal is effective by using techniques based on captures of time varying real models, usually

linked to texture synthesis techniques. Gu et al. [34] propose to develop a data-driven approach based in work of Matusik et al. [55] controlling weathering by $BRDF$ variations.

2.3 Cracks and fractures in nature

Cracks and fractures are abundant in nature. They can be typically found on wood, tree bark, stone, glass, ice, or dried clay. They also play an important role in the realism of a natural scene, as their presence provides a visual hint about the age of the object, its use, or indirect indications about the characteristics of the environment.

The nature of a fracture is determined by the material where the fracture occurs. The two main classifications for engineering are: brittle materials and ductile materials. In general, the main difference between brittle and ductile fracture can be attributed to the amount of plastic deformation that the material undergoes before fracture occurs. Ductile materials demonstrate large amounts of plastic deformation while brittle materials show little or no plastic deformation before fracture. [23].

Fracture is closely related to the stress that a material undergoes as it deforms. This stress depends on the resulting force at a specific area, and is computed as:

$$\sigma = \frac{F}{A} \tag{2.1}$$

where σ is the stress, F is the force and A is the cross-sectional area where the force is applied. Fractures occur when the stress on the material is larger than the energy present at a point in the material, usually where the material has some flaws.

The strain parameter is closely linked to the stress force. The strain is a normalized measure of deformation representing the displacement between particles in the object relative to a reference length, and can be expressed:

$$\varepsilon = \frac{\Delta L}{L} \tag{2.2}$$

where ε is the strain, ΔL represent the change in length per unit of the original length L. See Figure 2.3 for a typical stress-strain curve relationship. A property of the material is the yield strength, which describes the point where the material begins to deform plastically.

Crack propagation is also essential to fracture. The ways that crack propagates through the material reveal the mode of fracture. In ductile

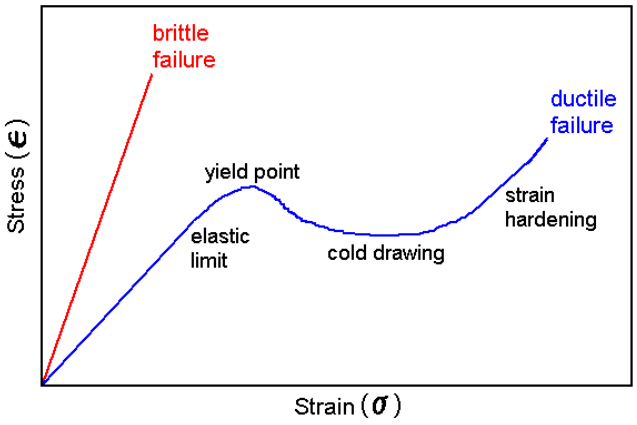

Figure 2.3: The stress-strain curve for brittle and ductile materials.

materials, the crack moves slowly with a large amount of plastic deformation. The crack will usually not extend unless an increased stress is applied. On the other hand, in brittle fracture, cracks propagate very rapidly with little or no plastic deformation.

The cracks in a brittle material continue growing once they are initiated. The way in which the advancing crack travels through the material is very important. A crack that passes through the grains within the material, following the edges of grid in a granular material and ignoring the grains in the individual grid, is known as trans-granular fracture. On the other hand, a crack that propagates following the grain boundaries, is known as inter-granular fracture.

Another important factor to determinate the amount of brittle or ductile fracture that occurs in a material is the temperature, which depends on the material nature. Basically, at higher temperatures the yield strength is lowered and the fracture is more ductile in nature. On the opposite, at lower temperatures the yield strength is greater and the fracture is more brittle in nature. At moderate temperatures, the material exhibits characteristics of both types of fracture.

Another factor is dislocation density. Dislocation density is based on the theory that sais that fracture always will be more brittle. The idea behind this is that plastic deformation comes from the movement of dislocations. Dislocations increase in the material due to stresses in the material yield point, so it is difficult for the dislocations to move because they pile into each other. Thus, a material that already has a high dislocation density can

only deform fractures in a brittle manner.

2.4 Physical models

Fracture simulation requires the computation of a set of common steps over time. A typical fracture algorithm needs to compute the following steps:

1. Compute the internal forces acting on all nodes into their stress and strain rate tensor, describing how the force act to separate the nodes,
2. determinate the location and orientation of the new fracture according to the current stress,
3. and modify the model to reflect the new discontinuity, preserving the orientation of the fracture.

Regarding the description of the material, this one can be done through a continuous or a discrete model. The continuous model sais how a material behaves and how it deforms based in a continuum mechanics approach. For more details, see the introduction made by Fung [26]. In the continuum approach, we assume that the scale of the modeled effects is bigger than the scale of the material composition. Thus, the behavior of the material particles can be modeled as a continuous model. This assumption may be not valid for certain fractures as there can be influenced by effects that happen at the small scale. However, depending of the goal of the method, a continuum model is enough to capture the real behavior. In other hand, in the discrete model the material is represented as the combination of numerous rigid-body elements. Neighboring elements are connected by springs and dashpots. The movements of the elements can be solved by the equations of motion for the whole system. The springs and dashpots are disconnected when the forces reach a critical value. This method is useful for the simulation of fragmentation of a brittle material under impact loading, for instance [67].

The Degree of Freedom (DOF) represents the allowed movement of each internal data representation of dynamic objects. A set of $DOFs$ specifies the displacement and the deformed position and orientation of the object. The rule is, each deformable object with N particles have 3N degrees of freedom.

The most important variables to update during the simulation are the position and the velocity of each DOF, but other data are also stored,

such as the link between their particles, the material parameters, etc. The physical model algorithm use these other data to control the way the force acts on each DOF and how are propagated inside the object. In the following sections, we detail detail the mass-spring system and the finite elements methods, which are the most popular models used in physical simulation. We also describe the meshless methods and other kind of approaches, even if they are not very common or popular.

2.4.1 Mass-spring models

The Mass-Spring system is one of the most simple physically-based deformation models. This representation is used for modeling soft bodies, especially cloth simulation [38] [91]. It is easy to implement, parallelizable, and cover the needs of most virtual reality applications. In this model, an object is represented as a set of masses connected by corresponding springs in a fixed topology, as shown in Figure 2.4.

The state of a mass-spring object can be defined as the position and velocity of each DOF, represented by masses. The internal forces acting on each mass is computed from the potential energy of each spring connected to the neighboring masses. The springs follow a linear force deformation law, but non-linear springs can be used to model more complex behaviors too.

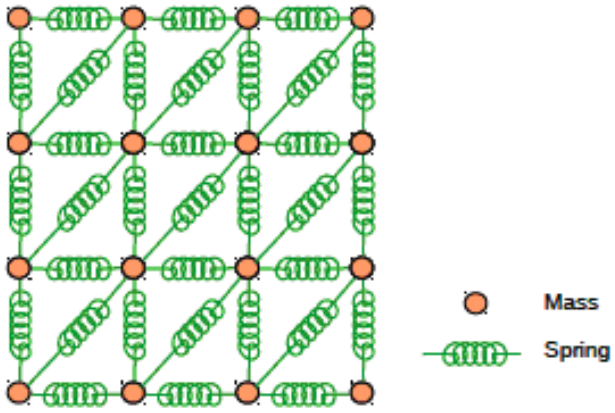

Figure 2.4: Mass-spring model: each Degree of Freedom of an object is represented by a mass. The internal force propagation is handled by the spring network connecting the masses.

Mass-spring systems have been extensively used and are still very popular, since they are easy to implement and computationally efficient. However, they have some limitations, like the difficulty to derive spring stiffnesses from elastic properties (Youngs modulus and Poissons ratio).

To overcome this deficiency, Lloyd et al. [48] proposed two ways of obtaining the parameters of a mass-spring model. The first one consists in varying the parameters until the behavior of the system is similar to the one obtained through the experiments or with the Finite Element Method. The second way is establishing an analytical reasoning for calculating the constants of the mass-spring model. For instance, starting from the definition of Youngs modulus, Poissons ration, shear and bulk modulus, apply simple tests to their mass-spring model to find the most appropriate stiffnesses. The same authors later propose a linearization of the mass-spring equations with the aim of equating the linearized stiffness matrix to the stiffness matrix of a linear finite element method. Following the same method, San Vicente et al. [92] derived a mass-spring model equivalent to a linear finite element model for maxillofacial surgery simulation. Because mass-spring systems do not derive from the equations of continuum mechanics, they have a limited capacity to model certain aspects of materials like anisotropy, viscoelasticity, etc.

2.4.2 Finite element methods

Contrary to the mass-spring model which is discrete in nature, the Finite Element Method (FEM) [80] is derived from the equations of continuum mechanics. One advantage of having a method that is a direct representation of mechanics, is that we can quantify the precision of the simulation, as the parameters of the model can be obtained from experiments on real world objects. For this reason, it is one of the most used methods for simulation [59], [31], [54] and [1].

The main idea of continuum-based deformable modeling is the minimization of the stored deformation energy all over the volume. The object gets equilibrium when its potential energy is at a minimum. In FEM the continuous model is discretized by dividing the object into small interconnected regions, called finite elements, as shown in Figure 2.5. The laws involved are then approximated by interpolation functions associated to each element.

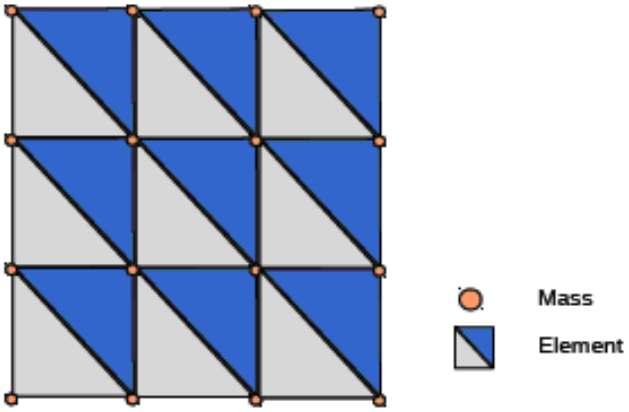

Figure 2.5: Finite Element Model: the object is subdivided into small elements. The vertices of the elements are associated to the degrees of freedom of the object.

2.4.3 Meshless methods

The meshless or mesh-free methods arised to avoid mesh reconstruction and large deformation problems that usually arise with FEM methods.

The meshless paradigm has provided different points of view about the well-known Finite Element method. Although these methods usually use the generic label "meshless", not all of them are truly meshless. Those based on the Collocation Point technique have no associated mesh, while others, such as those based on the Galerkin method, require an auxiliary mesh or cell structure.

The mesh-free Galerkin method [15] is a mesh-less technique for modeling material behavior. A set of nodes are distributed throughout the area/volume of the object being modeled. The deformed state at any given point in the material then is defined by fitting a moving least-squares approximation to the nodes in a local area around the point. This method was introduced by Belytschko in 1994 [5].

Besides the Galerkin method, there are other meshless methods used in the literature. Smoothed Particle Hydrodynamics (SPH) is based on the idea of representing fluids by a set of particles using the kernel estimate methods [32], [50], [45]. The Diffuse Element Method (DEM) was introduced by Nayroles and Touzot in 1991. The idea behind DEM was to replace the FEM interpolation within an element by the Moving Least

Square (MLS) local interpolation [11]. Another mesh-free method is the Finite Point. This was proposed by Onate et al. in 1996 [73], [72]. It was first introduced to model fluid flow problems but later applied to model many other mechanics problems. The method is formulated using the Collocation Point technique and any of the following approximation techniques: Least Square approximation (LSQ), Weighted Least Square approximation (WLS), and the above MLS, which can be used to construct the functions. The Point Interpolation method (PIM) uses the Polynomial Interpolation technique to construct the approximation. It was introduced by Liu et al. [46] as an alternative to the MLS method. The PIM, originally based on the Galerkin method, has the problem of using an interpolation matrix and the fact that it does not guarantee the continuity of the approximation function. The same author have investigated several approaches trying to overcome these problems [45].

Meshless methods have some advantages and disadvantages that will be discussed in the next chapter.

2.5 Time integration

The time integration algorithm is the engine of a physical simulation. It is responsible for integrating the motion equations of the interacting objects and particles, and computing their trajectory over time. This system is computed numerically by solving an ordinary differential equation system on the form of (2.3), which computes the new position and velocity based on the mass and the acceleration of the elements.

$$f(x, y) = M.a \tag{2.3}$$

There are mainly two groups of integration methods: the explicit methods (Explicit Euler, Runge-Kutta, Explicit midpoint) and their corresponding implicit methods (Implicit Euler, Implicit Runge-Kutta, Implicit midpoint). In the next subsections, we explain some of these common numerical integration methods.

2.5.1 Explicit methods

Explicit methods compute the state for the next time step by extrapolating the previous state. The first and the second derivatives of the current state are employed to directly create the new state. For example, in the Explicit

Euler method, we assume that the derivatives are the same over all time steps and equal to their values at the beginning.

The main limitation of this method is the time step size. There is a critical rigidity value where the numerical resolution of the system is divergent, called Courant Condition. The convergence time step that satisfies the Courant Condition is inversely proportional to the square root of the rigidity. This restriction is sometimes prohibitive for real-time physical simulation. This is why most of the works in real-time simulation using explicit integration involve soft bodies [31], [14].

2.5.2 Implicit methods

The implicit integration methods compute the state by solving an equation involving the current system state and the state at the end of the time step. Compared to the explicit method where we employ the derivatives from the values at the beginning of the step (t_0), in an implicit method the first and the second derivatives at the end of the step $t_0 + \Delta t$ are used to update the system rate.

Using an implicit method requires solving an equation to compute the derivatives of the state. Since this may be nonlinear, solving this equation requires an iterative solution method. The extra computation required by implicit time integration, and the complexity of developing an efficient equation solver, can promote the use of the explicit methods. However, many of the real life problems involve rigid objects, for which an explicit method would impose small time steps in order to avoid numerical instability. In those cases, it is preferred to use an implicit method with larger time steps, than using an explicit method with small time steps. This unconditional stability of implicit methods is the reason of their good acceptability in real-time simulation [16], [35], [76], [19].

This does not mean that an implicit method will always be better than an explicit one. Where one should use an explicit or implicit method depends on the problem to be solved. The choice of the time integration algorithm depends on the scene. Explicit methods are simpler but their use on rigid objects can come to instabilities. On the other hand, implicit methods are harder to implement and time consuming, but their result is unconditionally stable.

2.5.3 Semi-implicit methods

Another possible way to decrease the amount of time required by the simulation might be solving implicit methods by linearization of the right-hand side of the equation, usually called semi-implicit method. These Semi-implicit methods are not unconditionally stable, but they are much more stable than an explicit method because the integration steps remain within the region where the linearization is a good approximation.

Chapter 3

Physically-based methods

There are two main approaches for simulating fracture and cracking: using physically-based methods and using non-physically based approximations. The latter is often preferred by artists and developers in the game and cinema industry or those looking for real-time applications. In these situations, visual plausibility is generally enough, while physical accuracy is less significant. On the other hand, researchers and engineers often are seeking for an accurate simulation, as close to the reality as possible, then sacrificing time computation.

The majority of research on fracturing has been based on physical models; the main reason is that they produce results that are physically correct and realistic.

The models that have been found for fracturing are grouped in four groups according to the physical model used during simulation. Each one of these will be discussed in more depth throughout the following sections.

3.1 Mass-spring models

Mass-spring models are characterized for being simple and fast for computing deformations and collisions.

One of the first works was presented by Terzopoulos and Fleisher [88], where they introduced a hybrid model that represents rigid bodies as rigid and deformable components. The rigid components handle the rigid-body motion while the elastic behavior is present only in the deformable component. This approach aims at solving the conditioning of the discrete motion equations as the rigidity of the object increases.

Norton et al. [68] present a technique for animating 3D solid objects that broke under large strains. The technique employs a mass-spring system to model the behavior of the object. When the distance between two attached mass points exceeds a threshold, the simulation cuts the spring connection between them. A significant problem with this method is that, when the material fails, the exact location and orientation of the fracture are unknown. The breakage is performed by removing connection between two nodes and the current fracture orientation is not taken into account.

Mazarak et al. [56] used a voxel-based approach to model solid objects. Voxels are connected between them as showing in Figure 3.1. Theses links between them are made infinitely rigid and do not allow flexibility in the body, keeping adjacent voxels strongly attached together.

Connected voxels are stationary with respect to each other but can be grouped into more complex structures, namely bodies. These bodies can be grouped creating a desired shape, as for example, a cube. Voxels can be made as small as needed in order to reduce the stepped edges problems. The fracture of an object is simulated by breaking links between the connected voxels bodies, which imitates the crack formation inside a solid. When the number of broken links increase, the object may split into fragments.

They use the capabilities of their blast wave model to compute the pressure at each link. When pressure exceeds the link yield limit, this link is broken. For simple models, the simulation generates fracture over the real-time threshold. The main problem is the collision detection; without it the computational cost goes beyond a specified bound, where it is the number of boundary voxels of all the bodies in the scene. As for the fragmentation, the number of bodies increasing the number of boundary voxels, slowing down the simulation.

Hirota et al. presents a work [37] for simulating 3D cracks on drying clay using a spring network model. The stress in the material is caused by a deformation or a change in the distance of the nodes of the spring network.

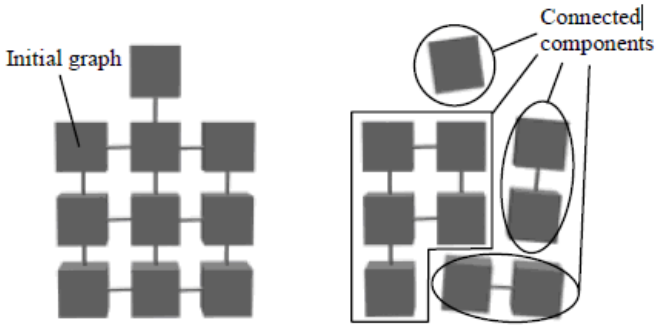

Figure 3.1: Graph representation of the connected voxel model, from Mazarak [56].

The breaking of the material is represented by cutting springs when they exceed a maximum strain threshold.

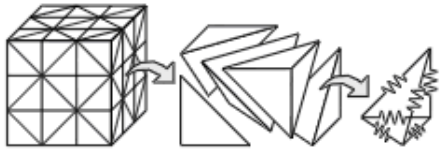

Figure 3.2: Spring network structure, from Hirota [37].

They subdivide the 3D object into a collection of tetrahedral elements, as showing in Figure 3.2. Each element is equivalent to a set of six springs placed on its six edges.

The simulation is executed in two scales of time: a small one for computing the motions of nodes, and a large time scale for the contraction of the material. When a crack appears, the elements are divided in different parts according to the affected springs, as we can see in Figure 3.3.

One of the negative aspects of their method is the computational cost. To obtain the crack simulation on a cube, they took around 8h at that time. Another disadvantage is the network structure and the resolution of the model, which does not change by the simulation. Having the same size of the elements causes undesirable effects. Another problem is that the smaller crack step is restricted by the size of the element and the new broken element can not cause cracks recursively.

This work is based on a previous work from the same authors, [36] where

they describe how static crack patterns created by drying mud can be modeled using mass-spring systems attached to a fixed support.

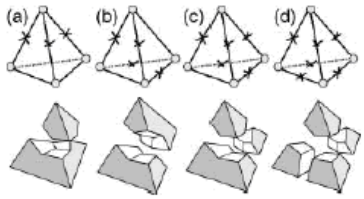

Figure 3.3: Simulation of cracks in a tetrahedra representation, from Hirota [36].

The work of Neff and Fiume [66] is based on a model of isotropic blast wave transport with an algorithm for fracturing objects in their wake. Their fracture model is based on the idea of rapid fracture, focusing on the problem of generating fracture patterns in a plane. This idea depends on an initial crack being present in the material. Given the length of the crack, a critical stress value can be determinated and if exceeded, it will cause the crack propagation. Cracks are propagated in each direction, and every time the crack structure grows, its propagating edges are separated. This process generates a crack trees structure. The edges can propagate until they hit another edge or the border of the geometry.

The crack tree is treated as a logical searchable tree structure. All the edges that are currently propagating make a leaf list. Whenever there is an intersection of two edges, a new face has to be found. Theses faces correspond to to new fracture fragments. The model used for crack propagation is a grid, whose main advantage is a very fast detection of edge collision.

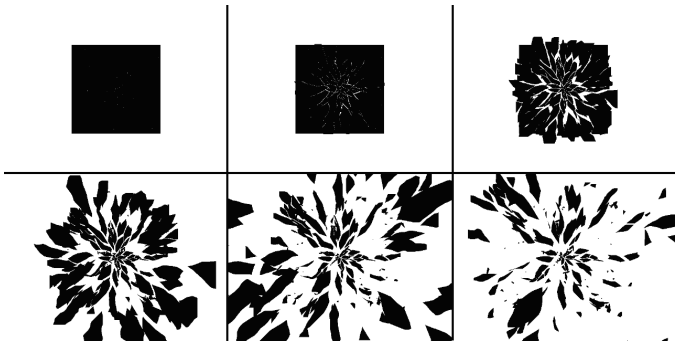

Figure 3.4: Animation of a shattering window extracted, from Neff [66].

Their algorithm is limited to thin surfaces and is not extensible to 3D solids. The animations presented took between one and two seconds per frame.

Another remarkable work was presented by Aoki et al. later on [2]. They present a method for synthesizing 3D cracking patterns in clay solids by incorporating a moisture model.

To simulate the cracking process, they use a spring network model, where a 3D object is broken into tetrahedral elements. While Hirota [37] created cracks on cubic or rectangular objects, Aoki subdivide a 3D object into a collection of tetrahedrons whose sizes are adapted in order to fit a general object. They simulate the explosions with a close-packed structure. For a given 3D object, the first step is to approximate its shape with a close-packed structure. A spring is assigned to each side or ridge of a tetrahedron. Finally, each spring is cut when it is stretched beyond the maximum strain. These steps are repeated until all the forces appearing on all nodes are less than a given threshold.

Once an edge of a tetrahedron is cut, the tetrahedron is then divided into different parts depending on the situation.

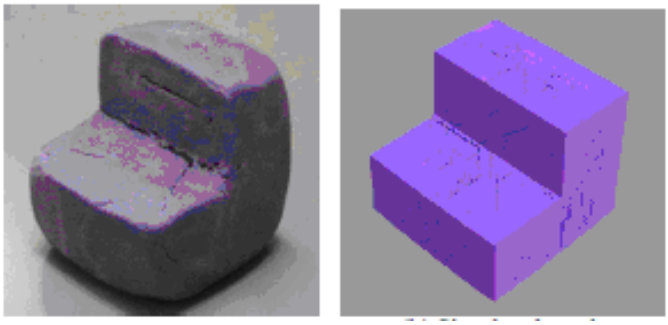

Figure 3.5: A real crack example (left) and its corresponding simulation (right), from Aoki [2].

One of its limitations is constructing the grid by keeping a similar size for the generated tetrahedrons. Using an adaptative model would allow a greater number of nodes to be filled where the model is denser and less at the sparser areas. The result would be a smaller number of nodes to animate and less tetrahedrons to model, although the grid would need to be reconstructed during fracture.

The work of Smith et al. [84] is based on a system of point-masses connected by workless, distance-preserving constraints, rather than a grid of

stiff springs. Using rigid constraints follows from an abstraction of brittle material properties that allows them to solve the forces exerted by these elements during impact more quickly than using explicit methods and an elastic mesh. They compute the solution by constructing a large, sparse, linear system that is solved using conjugate gradient methods. The constraint forces, once calculated, indicate when and where the object will break. This information is then used to construct the fragments of the broken object from the original geometry, and to solve for the final linear and angular velocities of these bodies.

Martins et al. [53] present a real-time simulation of object damage and motion due to a blast wave impact, with an improved connected voxel model for the object. Their blast wave model is a combination of simplified physical equations and experimental data. They take the blast wave simulator and the voxel model presented by [56] and extend its scope for real-time simulations, introducing arbitrary shaped voxels.

The basic voxel shape can be scaled along any axis. In addittion, the vertices of the voxels can be displaced in any arbitrary directions. Each voxel can also have unique properties according to the material it represents. This results in an object model that is much more flexible, and permits the user to create voxel shapes that better suit the needs of a particular application.

They simulate the fracture by weakening and breaking links and voxels in the model. As the object breaks apart, new fragments may be created, like in Mazarak et al. [56]. The independent objects in the scene are represented by a scene connectivity graph, where the nodes are individual voxels and the arcs are the links between voxels. Every link has an associated yield limit, which is the maximum pressure that the link can support before breaking. Any link that encounter a blast wave have its yield limits weakened, making it vulnerable to subsequent explosions.

For a small number of voxels, the performance is around 30 fps. The main bottleneck for the simulation itself is collision detection. As the number of the independent objects in the scene increases, the frame rate decreases, as more collision tests need to be made.

3.2 Finite element methods

As explained in Section 2.4.2, finite element methods divide the material into a set of disjoint elements that model the object. The material is computed from shape functions that interpolate the material within the element. This method has been extensively used for modeling fracture problems.

In this section, we focus on reviewing the FEM techniques that allow fractures to propagate in arbitrary directions, using a physically-based approach. A survey by Gibson and Mirtich [30] details much of the early work on deformable modeling using FEM, while Nealen et al. [1] survey some more recent approaches.

The pioneer method in this area was introduced by Terzopoulos and Fleisher [88]. They present a general technique for modeling viscoelastic and plastic deformations using three fundamental metric tensors. These tensors define energy functions that measured deformation over curves, surfaces and volumes. These energy functions provided the basis for a continuous deformation model that they simulated using several different discretization methods. One of their methods employed a FEM by using controlled continuity corrugation on the surface object.

In the same year, Swenson et al. [86] presented a work that describes a method for local re-meshing in a two-dimensional domain as a crack propagates on it. Their techniques make use of triangular elements. The crack tip is surrounded by an octagonal region that is triangulated from a central node located at the crack tip. While the tip advances, the central node moves with the octagonal region and the eight surrounding triangles are distorted to accommodate this motion. When the direction of the crack changes or the tip comes close to the boundary of the octagonal region, an area with twice the radius of the octagonal region is re-meshed. The authors state that their re-meshing technique is not robust and occasional user interaction is necessary when the re-meshing occurs.

O'Brien and Hodgins [69] developed a model that used a finite element model with the theory developed by Griffith and Irwin, marking a tendency [71]. They present an approach based on linear elastic fracture mechanics and non-linear finite element analysis, where 3D objects are modeled using a mesh of tetrahedral elements. The simulation determines where cracks should begin and in what directions they should propagate by analyzing the stresses created as the mesh deforms, see Figure 3.6. Although this model produces realistic results and is moderately simple to implement, the amount of computation required meant that it could not be used in real time.

Years after, the same authors extended their work to include ductile fracture [70]. The physically-based nature of the fracture criteria allows it to be easily integrated with other material behaviors, such as plasticity, under different conditions.

Although their methods represent a significant advance in the graphical modeling and animation of fracture, they still have some limitations. The simulation do not captured the real material stiffness. Another visual

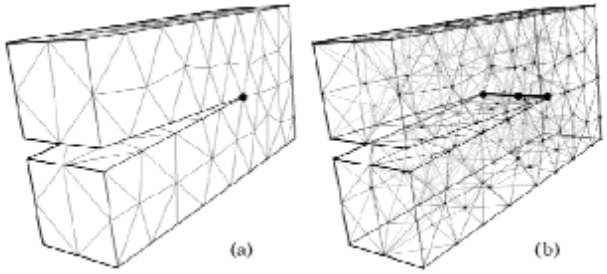

Figure 3.6: Tetrahedron model surface (a) and object interior tetrahedron (b). By O'Brien [69].

limitation is the representation of small fragments and dust created by the fracture process. This is because the size of the smallest fragments in the simulation are limited by the re-meshing thresholds and the stability considerations that they make. The amount of computation time required is another limitation; some simulations can take from a few minutes to several hours to compute a second of simulated motion.

Their implementation can switch between an Euler integration scheme and a second order Taylor integration. These two techniques are explicit integration schemes, and as such, they are subject to stability limits that require very small time steps for stiff materials (see Section 2.5). One possibility to decrease the amount of time required by the simulation could be using a semi-implicit integration scheme, as stated by the authors.

Yngve et al. [98] present a technique for animating explosions and their effects, including fractures. They simulate the propagation of an explosion through the air using a computational fluid dynamics model based on the equations for compressible, viscous flow. They use an integration method to model the numerically stable formation of shocks along blast waves. The system includes two-way coupling between solid objects and the surrounding fluid. Specifically for fracturing, the explosion simulation results in pressures, velocities, and densities for each voxel in the discretized model. The fracture simulation uses this information to compute the forces that have to be applied to a finite element representation of the objects in the scene. One of their limitations is the execution time, however. There are also some effects from explosions that they have not investigated, and some parameters are not taken into account for the simulation, like the influence of high temperatures for instance.

In 2001, they extend their model to work with ductile objects, achieved

by adding the effects of plastic distortion to their existing model. This model works quite well and has been the basis for extensive research that uses FEM for fracturing.

The same year, Muller et al. [62] present a hybrid method for simulating deformation and fracture of materials in real-time. Their hybrid idea consists in alternating between a rigid body dynamics simulation and a continuous model at the point of impact. They evaluate elastic forces only during collision events while treating the body as rigid. This allows a reasonable simplification for stiff materials, as the natural frequencies of stiff materials tend to be much higher than the rendering frame rates, producing a quick damping of the vibrations. This approximation produces good visual results.

They treat free-floating objects as rigid bodies and compute their dynamic behavior using rigid body dynamics based on explicit Euler integration. Each rigid body has four state variables: its position, its center of mass velocity, its rotational orientation, and its angular velocity. In general, these states can be initialized according to specific user inputs or according to simulation objectives. One limitation of their method is that it only considers deformation and fracture behaviors during contact. They also have the problem of real-time collision detection along a fracture line, being at least as time-consuming as the simulations.

Federl et al. [25] later present a model of fracture formation of bi-layered materials. The model allows the synthesis of patterns as a result of cracking induced by growth or shrinkage of one layer with respect to another. By applying a FEM model they synthesize crack patterns occurring in tree bark and drying mud.

The material layer of the simulated surface is discretized using 6-node wedge elements. The discretization procedure begins with a random distribution of points over the surface area, followed by the application of the particle repelling techniques to achieve a more uniform distribution. Then a Delaunay triangulation is used to construct a polygonal mesh with the vertices at the given points. See Figure 3.7.

Molino et al. [60] combine a lattice with a tetrahedal mesh similar to Aoki [2], where the lattice controls how the tetrahedrons are rendered. However, Molino et al. use the lattice more like an assistance tool to the tetrahedral model rather than a deformation model. The model works by using the lattice nodes as hidden vertices in the tetrahedrons, modeling cracks and extra geometry, along with crack propagation. A disadvantage of this model is that the geometry of the object increases by both the subdivision and the added hidden nodes, which affects the system performance.

As usual, a fracture appears when a yield stress is exceeded in the ma-

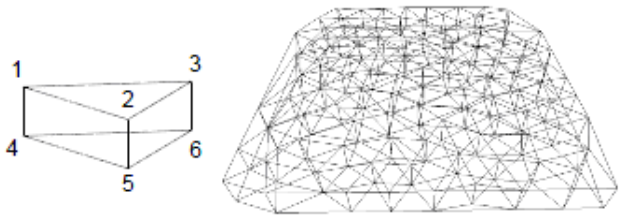

Figure 3.7: The wedge element (left) and a surface layer discretized using the wedge elements (right), by Federl [25].

terial. They simulate this fracture by removing from the model the corresponding elements. The removal process consists on first adjusting the geometrical representation of the model, and then recalculating the global stiffness coefficient matrix from the stiffness matrices of the remaining elements. Once the element is removed, the equilibrium state of the model is recomputed.

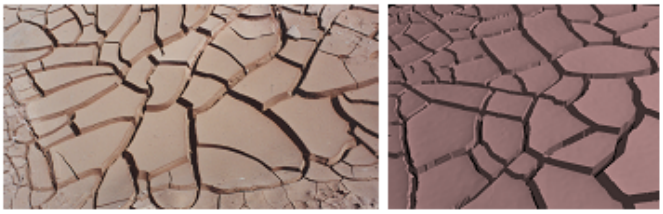

Figure 3.8: Real mud (left) and simulated mud (right), from Federl [25].

In the case of mud, the use of FEM improves the visual realism of the generated results. However, splitting material at the nodes would be a better solution than removing the elements, as stated by the authors. This would result in a less formation of elements and faster simulation times, while keeping some mesh parameters like the "wight".

Iben et al. [64] present a method for generating surface crack patterns that appear in materials like mud, ceramic and glass. To model these phenomena, their work is based on a physically-based method using a FEM with local re-meshing, as in [69]. Their algorithms generate cracks on the surface of objects but do not fracture their volume. They reduce the simulation computations due to their discretization approach and because cracks are generated from a stress field defined heuristically over a triangle discretization.

The cracks are produced by evolving this field over time and then analyzing the stress field to determinate where cracks form in the mesh. These cracks are introduced as free boundaries that affect the relaxation process and evolve the stress field. After cracks are formed, they update the stress field and repeat the process until additional cracks cannot be created or the user stops the simulation. The user can control the features and presence of the cracks using a set of simple parameters.

More recently, Wicke et al. [52] propose a finite element simulation method for modeling elastic to plastic deformations that is easy to use in plastic flow, fracture or large elastic deformations. They use a tetrahedral mesh and a dynamic re-meshing algorithm that tries to replace as few tetrahedra as possible, which limits the visual artifacts and artificial diffusion that could be introduced by repeatedly re-meshing. Their dynamic mesh is compatible with all types of specialized re-meshing used before. To demonstrate its performance they integrated their method with the fracture algorithm described by O'Brien and Hodgins [69].

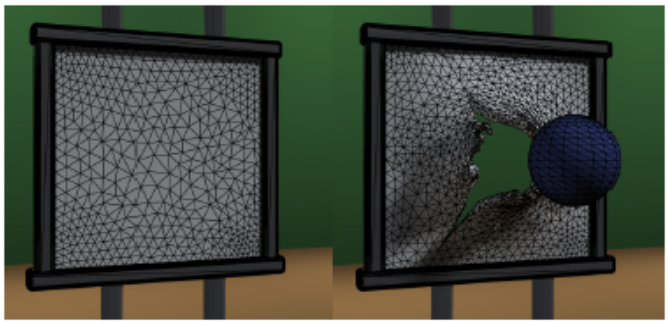

Figure 3.9: Adaptive mesh refinement helps a ball to crash through different ductile plates, from Wicke [52].

3.3 Meshless methods

Meshless methods come as an alternative to enhance $FEMs$. There are a number of features of these methods that make them favorable for fracture simulation, such as the complex re-meshing operations and the problems associate to element cutting and mesh alignment common in FEM. (for more details see Belytschko et al. [5]). In the field of mechanics, Sukumar et al. [85] propose a particle-based approach to model the physical behavior

around a crack; while Belytschko et al. [6] entirely resort to meshless methods. These works were the base for the later works in Computer Graphics.

Muller et al. [51], for instance, present a method for modeling and animating a wide range of volumetric objects, with material properties ranging from stiff elastic to highly plastic. They use a meshless method based on particles to represent both the volume and the surface. Their physical model is derived from continuum mechanics, which allows the specification of common material properties such as the Young modulus and Poisson ratio of elasticity. At each step, they compute the spatial derivatives of the discrete displacement field using a Moving Least Squares (MLS) procedure. From these derivatives, they obtain strains, stresses and elastic forces at each simulated point. They solve the equations of motion based on these forces, with both explicit and implicit integration schemes.

More recently, Pauly et al. [77] present a meshless animation framework for elastic and plastic materials with fracture. Their main idea is a dynamic surface and volume sampling method that supports arbitrary crack initiation, propagation and termination, while avoiding some of the stability problems of traditional mesh-based techniques. They model crack fronts and associated fracture surfaces in the simulation volume. When cutting through the material, the crack tip directly affects the link between the simulation nodes, requiring a dynamic adaptation of the nodal shape functions, which is one of their contributions, as show in Figure 3.10. To model the initial surface they use the point-based representation proposed two years before [78].

Their contributions include the dynamic creation and maintenance of fracture surface by continuously adding samples during crack propagation; a meshless initial sampling of the volumetric domain and local dynamic re-sampling that adapts the nodal sampling resolution to handle fracturing and large deformations; and the handling of the topological events associated with multiple branching and merging cracks.

Steinemann et al. [17] propose an algorithm for splitting deformable solids along arbitrary piecewise linear crack surfaces for cutting and fracture simulations. They propose the use of a meshless discretization of the deformation field, and a visibility graph for fast update of shape functions in the discretization. They decompose the splitting operation into a first step where they synthesize crack surfaces as triangle meshes, and a second step where they use the newly synthesized surfaces to update the visibility graph, and thus the meshless discretization of the deformation field. The separation of the splitting operation into two steps enables flexibility and control over the splitting trajectories, provides fast dynamic update of the meshless

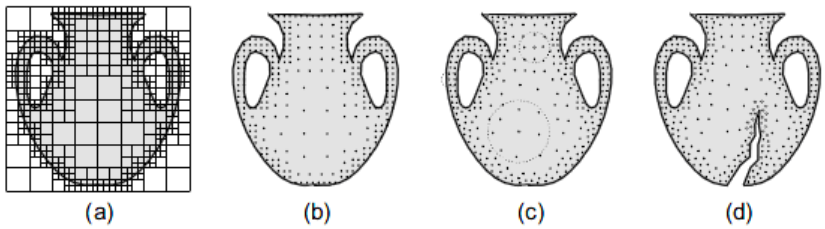

Figure 3.10: Volume sampling: octree decomposition (a), initial adaptative octree sampling (b), sampling after local repulsion (c), and dynamic resampling during fracture process (d), from Pauly [77].

discretization, and facilitates the implementation. This makes their algorithm suitable for a large range of applications, from computer animation to interactive medical simulation. However, the quality of the surface mesh degrades when the same region is splitted multiple times. Local surface remeshing is probably a viable solution, as it does not affect the discretization of the simulation in their algorithm. See Figure 3.11 for some results.

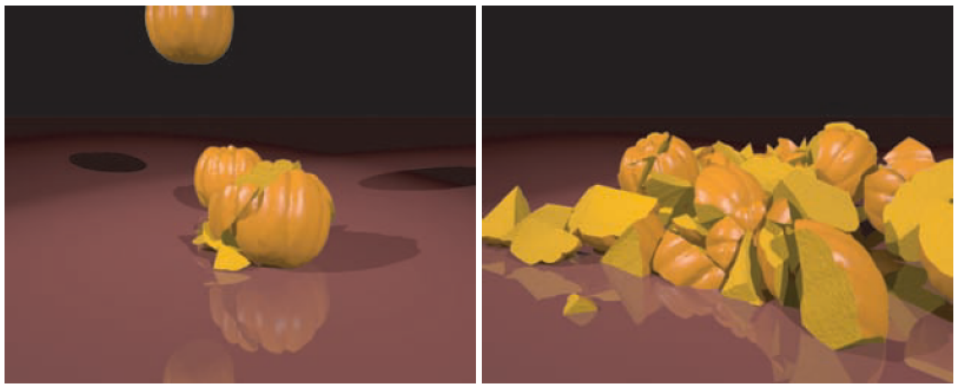

Figure 3.11: Using the algorithm for synthesizing crack surfaces for fracture. From Steinemann [17].

The work presented by Martin et al. [82] includes a section dedicated to fracturing objects. They handle fracturing adapting the idea the connectivity graph of Steinemann et al. [17] and the virtual node approach of Molino et al. [60] using a point-based discretization.

A general limitation of meshless approaches is that even very small fragments need to be sampled sufficiently dense in order to obtain a stable evalu-

ation of the shape functions. This increases the number of simulation nodes when an object is fractured excessively, which increases the computation time.

3.4 Other methods

In this section, we describe the methods that do not belong to any of the above classifications for some reason. For instance, Gobron and Chiba [81] describes a method for modeling the propagation of cracks on a 3D surface based on a multi-layer Cellular Automaton (CA). The main advantage of their model is that it proposes a semi-physical solution, giving more control to the user and being easily extensible.

Crack propagation is determined by the systematic stress release of all the unstable cells on the input object. Unstable cells are those cells that contain a stress intensity higher than the material resistance. The size of the crack depends on the stress release, in order to make the crack pattern look more realistic.

One limitation of their work is that they do not consider the stress interaction between layered models, thus producing an even worse solution to the velocity relationship between cracks. They also simulate crack formation on layered materials consisting of mixed types of materials.

More recently, Heo et al. [43] present a fracturing-aware stable collision detection method for large-scale fracturing models, which supports geometric and topological changes. They use a dual-cone theorem to check wherever a surface can have self-collisions or not, designing a Bounding Volume Hierarchy (BVH) that uses these dual-cones for each node. They also propose a selective restructuring method that only modifies the sub-hierarchies with low culling efficiencies. Finally, in order to reduce performance degradations at fracturing events, they propose a BVH construction process that builds multiple levels of the hierarchy for each iteration.

Their dual-cone method combined with BVH is approximates and can miss collisions, being this a limitation. Also, the memory required is relatively high due to the number of bytes required for each bounding volume node. Another limitation is the culling efficiency metrics for self-collision and inter-collision, and that their selective restructuring method does not always improve the performance of the collision detection. However, by combining these methods, they achieved a faster and stable collision detection performance.

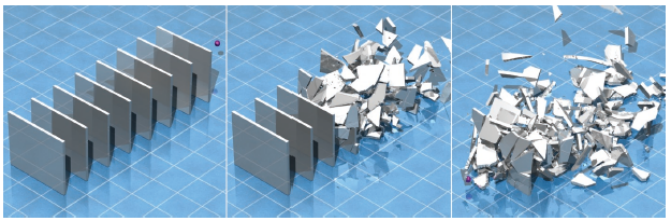

Figure 3.12: Breaking-Wall benchmark. From Heo [43].

3.5 Conclusions

A large range of physically-based methods have been proposed for simulating fracture. Each of these methods is based on a specific representation, a simulation/mechanical model, and a set of approximations. It is thus difficult to state which one is better suited on a general basis. If we are looking for a simple implementation method or a fast one, we should better use mass-spring; if we are seeking for more accurate simulations and the computation time is less important, we can choose FEM methods; but if we also want to avoid the mesh treatment and obtain results similar to FEM, we can use the meshless approach.

One of the problems that is common on these methods is the quality of the final simulation results. Some effects, like the representation of small fragments and dust, which are very common in the real world, are not well represented. In FEM-based methods, the size of the fragments in the simulation is limited by the re-meshing threshold imposed to maintain the stability.

The amount of computation time required is another limitation. Most physically-based approaches can take from a few minutes to several hours to compute each frame. This makes these methods non-applicable to interactive applications, like video games for instance. Almost all of the reviewed works propose, as future work, either increase the accuracy of the results or improve the speed of their simulation. Another limitation is the limited control that is given to the users, especially for animations.

Chapter 4

Non-Physically based methods

Non-physically based methods try to reproduce real fracture patterns without considering the underlying physical process. They tend to be fast and sometimes interactive, but usually require more manual intervention for computer animation, while one of the main advantages of physically-based modeling is that they avoid the need of specifying how fracture behaves and propagates. As in some physical approaches, some of these techniques make use of regular grids and viewers tend to be sensitive to the directional artifacts, which result from forcing the fracture lying on a specific path.

In this chapter, we explain the existing non-physically based methods proposed in the literature to simulate crack and fracture patterns.

As it is not possible to keep the same classification as in the previously chapter, we classify these approaches into two new categories: image-based methods and procedural techniques.

4.1 Image-based methods

Image-based methods make use of an input image or texture in order to capture and transfer real fracture patterns. This is an area that extends to both computer vision and computer graphics. In the case of computer graphics, and more specifically in cracks and fracture generation, image-based techniques are used to guided the crack propagation, extracting either the path or some parameters to help the path generation. For a comprehensive review of image-based techniques in general, see Shum and Kang [83].

Wang et al. [93] presented an interactive approach for modeling tree bark. They take as input a bark image and produce a textured height field representing a bark pattern, being an easy-to-use technique to interactively model a variety of photo-realistic bark surfaces. This method, however, is limited to patterns that can be represented through a height field.

Mould et al. [61] propose an image filter that transforms an input line drawing into an image of a fractured surface, where the cracks reproduce the input drawing. Their algorithm is based on the Voronoi diagram of a weighted graph, where instead of partitioning a surface they partition nodes in a graph. The distance metric is then the path cost within the graph. The graph edges are weighted, and the edge weights provide some control over the region shapes. By modifying the edge cost, they can control the placement of the cracks.

The cracks distribution can be set by both the number of sites (global density), and their distribution (local density) affecting the final appearance of the crack pattern. They can create a pattern similar to pavement cracks by using a few sites distributed randomly across the image; alternatively, they can deviate from a uniform distribution in the interest of other effects, like in Figure 4.1.

The rendering of the final crack maps is made either by texture synthesis or by modulation of an un-cracked texture. One shortcoming of the system is that the graph must be computed using the same texture resolution. However, according to the authors, no aliasing is observed in the synthesized image. In addition, they only generate cracks on flat surfaces; but the method could be adapted to general surfaces by directly sampling it with Voronoi diagrams. This would avoid the classical distortions associated to texture mapping.

Hsien et al. [39] propose an approach to render crack patterns and animate their propagation on 3D object surfaces. This method provides flexibility by means of some controllable parameters.

The pattern is extracted by image processing operations from natural

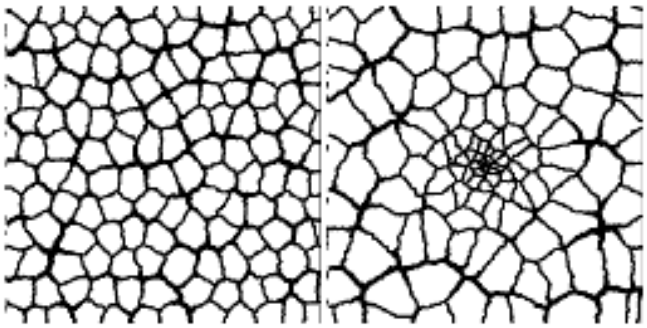

Figure 4.1: Different crack patterns by manipulating the site distributions. By Mould [61].

crack images. By tracking the pixel connectivity in the pattern image, the image is vectorized and simplified as a planar graph, called a feature pattern. To generate a new crack pattern on an object surface, the planar graph is mapped into the surface using a parabolic bounding volume onto the o surrounding the target object. This volume is then projected onto the surface in order to map the crack pattern. Finally, some animations of cracking propagation are produced by script-based traversal schemes.

Their work presents some limitations. First of all, the projection produces some distortions on the crack pattern. In addition, cracks are rendered using simple bump mapping, thus the final detail is limited.

4.2 Procedural methods

Procedural techniques are known for providing efficient and flexible ways of representing both appearance and surface details. In this case, procedural methods are used to model the crack pattern and the size and shape of the obtained fragments from a simple set of parameters.

Lefebvre et al. [87] propose a semi-empirical model of bark generation which produces either geometry or texture. This method runs at interactive frame rates and allows automatic bark generation with intuitive parameters. They model the bark with a set of strips parallel to the growing direction. A strip is a set of alternating elements: original epidermal elements and fracture elements. The epidermal elements are semi-rigid, while the fracture elements are soft. When the bark breaks at a given location, they introduce a new fracture element between the edges. This splits the fracturing

epidermal element into two parts. The fracture criterion is based upon the relative lengthening of elements, which corresponds to the Griffith energetic approach. They state that the threshold has to be chosen in a reasonable range; otherwise an infinite fracture loop may occur.

Figure 4.2: Real (left) and synthetic (right) bark, from Lefebvre [87].

The obtained results look quite realistic, the visual aspect of their bark could be improved by adding other natural phenomena, like lichens for instance.

Martinet et al. [4] present a procedural method for modeling cracks and fractures in solid materials such as glass, metal and stone. They use a procedural approach that provides to the designer the tools to control the pattern of the cracks and the size and shape of fragments given a few parameters. Their approach is phenomenological and it is related to the aging technique proposed by Paquette et al. [74]. Based on the observation, they create some patterns and parameters to guide the procedural technique.

Their crack and fracture modeling system relies on an hybrid representation of shapes that combines skeletal implicit surfaces and triangle meshes.

Cracks are modeled using Boolean difference operations between the original input model and the carving volume defined by the cracks. The carving volume is defined as the union of complex implicit skeletal primitives composed of tetrahedral. Fractures are created using Boolean intersection and difference operations between the original input model and fracture masks.

In 2004, Wyvill et al. [12] present an algorithm for simulating the crack pattern found in Batik wax painting. Their method is based in the Distance

Figure 4.3: A real clay vase (left) and a synthetic model (right). From Martinet [4].

Transformation algorithm, being easy to implement and computationally efficient.

In their method, the user generates one or more wax images that will be combined to create the final visual work. These wax images are loaded into the simulator, the cracks are produced, the simulated stain is applied and the result is create with the final images. Their crack algorithm produce convincing patterns that capture many of the characteristics of the crack patterns found in real Batik paintings. One of their limitations is that the crack patterns do not behave in the same way compared to real examples. In addition, their patterns take inverse direction of propagation with less frequency, and thus the final results can be not very realistic eventually.

In 2004, Taubman et al. [28] presents a fracture method for exploding structures based on Martins et al. [53] work. They use a connected voxel method as a basis for fracturing and approximates blast wave forces using the Friendlander equation [22] to achieve real-time performance. They pre-compute the fracture patterns into arbitrary meshes of convex polyhedral instead of using a warped voxel grid, and then propagate force along the connections between these fractured components. This adds a greater level of realism to the simulation because pieces broken off of the structure are of arbitrary shape and resemble fractured material. Material attributes are simulated by restricting the splitting of polygons to minimum sizes, allowing some materials (like brick walls) to fracture less than others (like glass).

As the materials have no physical properties and only one behavioral observation is used, the results are not very realistic. However, the method is simple and easy to implement on any polygonal mesh.

Valette et al. [90] model and visualize the main characteristics of cracks produced on the surface of a soil. For this, they use a height map to model the terrain and a procedural method to generate cracks in soil. The cracks

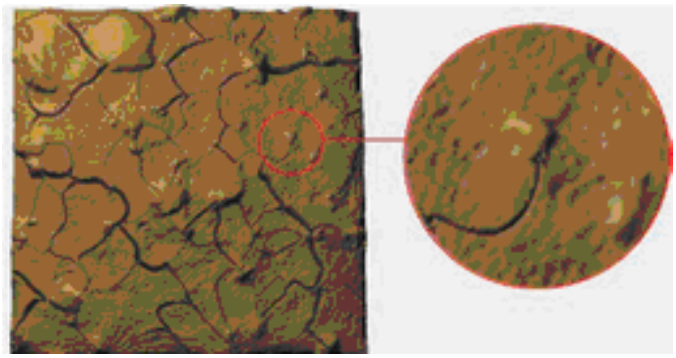

Figure 4.4: Cracks applied to a simulated crusted soil. From Valette [90].

are generated by combining a voxelised representation of the terrain with the crack path, generating the crack geometry, along with the original terrain model. The method is not interactive, but the results are very realistic, Figure 4.4. As in Wang et al. [93], this method is restricted to height fields. In the other hand, their method is dynamic and can thus simulate the propagation of the cracks. The input data are simple: some of them are related to the physical properties of the soil, while the parameters controlling the simulation are intuitive. The computation of the crack-path can be done automatically using different algorithms provided, but it is possible too use any path provided by the user.

4.3 Conclusions

There is an important amount of work done on non-physically based methods. Although visual quality could be improved considerably, they, can provide us with a simple and fast solution for simulating fracture and cracks, suitable for such applications like video games and entertainment, or even as a pre-visualization step, where interactivity is mandatory.

The particularity of non-physically based methods is given by the way they determine the formation of the cracks. Some works base their criterium in information obtained from patterns extracted from images, while others are based on the observation or some simplified rules.

Chapter 5

Conclusions and Future Work

One of the main goals of computer graphics is to reproduce the real world by creating realistic models and images. The animation of this virtual world is also essential, especially for virtual reality, games and similar applications. To reproduce this synthetic motion in a 3D virtual scene, many methods have been developed as an inherit part of computer graphics.

It is well known that creating a good and realistic motion technique, whatever the field, is a challenging task due to its complexity. This thesis address the particular problem of simulating and animating cracks and fractures, giving a survey of the most relevant techniques proposed in the literature. In this thesis, we have reviewed these techniques divided into three main chapters, which involves the main concepts and related Background, Physically-based techniques and Non-physically based methods.

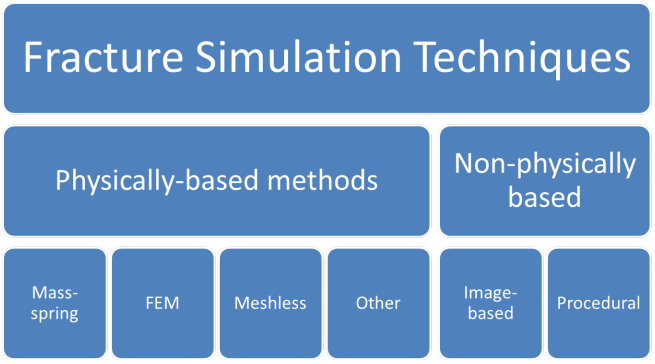

Figure 5.1: Overview of our classification of techniques.

In Figure 5.1 we can see an overview of the classification of techniques exposed in this thesis. As we can see in Figure 5.2, for physically-based solutions, FEM methods represent the 36% of all the cited methods, followed by the mass-spring methods 30%, and the meshless methods, which repre-

sent the 27%. According to this, the FEM approach is the most popular techniques used in referenced works. This is probably given by the FEM ability to correctly represent cracks and fracture simulations. As FEM is derived from the equations of continuum mechanics, it is possible to quantify the precision of the simulation, since the parameters of the model can be obtained from experiments on real world objects. Some authors prefer to work on representations that limit the crack path, avoiding expensive cost of computations but sacrificing the veracity of the crack; and other authors bet for an accurate calculation of the crack path, in which case the simulation can take minutes or even hours to compute.

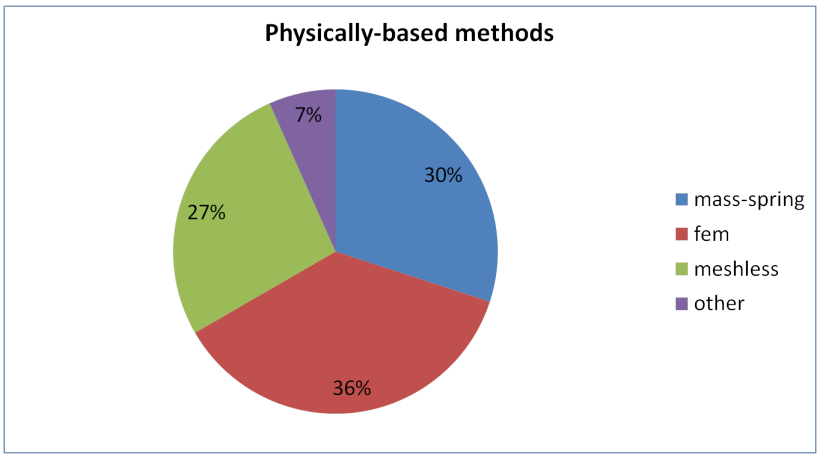

Figure 5.2: Based on the articles cited in this thesis, this graphic represents the contribution of each technique to the physically-based methods.

Analyzing the figure corresponding to the contribution of each cited technique for the non-physically based methods 5.3, we can see that the tendency is to use procedural methods. These kind of methods allow to the author add the desire characteristics manipulating the effect until reproduce the crack exact as they need it.

These figures 5.2 and 5.3 are based on the previous work cited in this thesis, but it helps in getting an idea about the most popular methods used in Computer Graphics and the reasons.

It is important to note that simulating fracture is a challenging task in computer graphics due to its complexity. This could bring in two different ways of modeling the process depending on the kind of realism that is wanted, as defined by Ferwerda et al., in 2003 [42]:

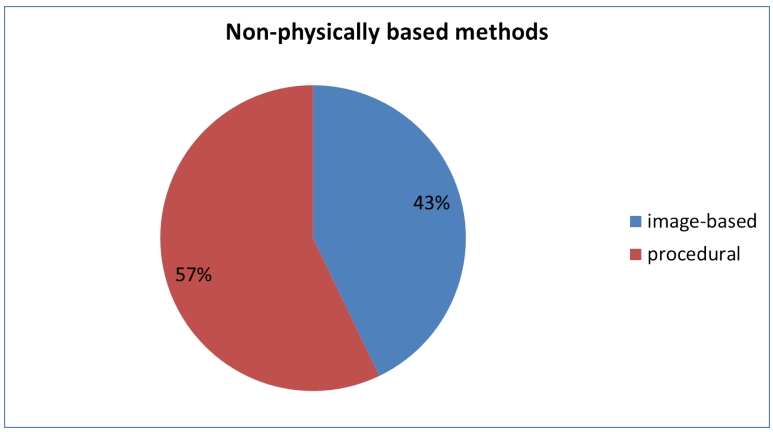

Figure 5.3: Based on the articles cited in this thesis, this graphic represents the contribution of each technique to the non-physically based methods.

- If the first choice is plausability, we can do physical simplifications while keeping realistic results. Thus, models can be intuitive without needing many parameters.
- If we prefer to obtain an accurate simulation of the phenomenon, the best choise is to work on a physical approach, trying to use measurable parameters.

After analyzing the existing techniques, we can realize that research and development efforts are as active and constant as in the beginning. Although there are proposed techniques that are very close to a realistic approach, there does not exist one ideal model for all kinds of aplications. The graphics community still have a long way to do until we can easily simulate plausible and interactively natural phenomena as in our real life/environment. For simulating fracture, many parameters and considerations have to be taken into account, like model representations, topological changes, realtime or interactive simulation, and many others.

There is a main problem often detected while analyzing the state of the art, in both types of approaches, which is the quality of the simulation results. One example is the representation of small fragments and dust, which are very common in the real world. In almost all of the cases the size of the smallest fragments in the simulation are limited by the re-meshing threshold imposed by some considerations of stability.

The amount of computation time required is also one the main limitations in physically-based techniques. The simulation can take from a few

minutes to several hours to compute each frame. This makes these methods non-applicable to interactive applications, like video games for instance. Another limitation is the limited control the user may have over the animations. On the other hand, in non-physically based methods allow easy control of fracture patterns and are usually simple and fast, but they rarelly provide realistic results.

Although we know there is not a perfect method to simulate physical behaviors, we are nowadays able to obtain very good visual results close to the reality. Furthermore, despite all their limitations, we see a potential in coupling more intuitive user interfaces with physically-based simulations.

A general question that could emerge is about the validation of the methods. These techniques need to be validated from a scientific point of view but also from a perceptual significance. Lu et al. [49] present a resume about validation techniques with different models. Theses validations are performed using visual rendering quality, making a comparison with real scenes and other similar models. For accurate simulations, this kind of validation is not enough and should be compared with experiments on real surfaces, so that the results can be validated with measurements. Valette et al. [89], for instance, did they comparisons with real samples prepared in lab conditions. Another work in this direction was presented by Ramanarayanan et al. [27], who provide a framework to link the physically accurate models to visually accurate ones using the program Visual Equivalence Predictor.

When we see the graphic 5.4 we can see there is not an equilibrium between the use of physically-based and non-physically based methods. The use of physically-based methods is superior, which shows that most researchers are usually seeking for reproducing the physical behavior of fracture in order to obtain high-quality results.

Cracks and fractures simulation techniques are very common, but reproducing a specific pattern is a challenging task, since this usually requires the fine tuning of parameters, which may become very tedious. On the other hand, the techniques based on examples mostly focus on the reproduction of a specific pattern, without providing enough flexibility. An open problem is the combination of both techniques, in order to get the best of them. An example of this can be found on the recent work by Bosch et al. [9], where they propose an image-guided simulation for flow stains. This method represents a methodology to extract parameters and detail maps from existing imagery in a form that allows new target-specific flow effects to be produced. Based on this idea, one could try to extend it to other imperfections, including phenomena that may affect the geometry of objects such as the case of fracture. In the specific case of reproducing fracture using this

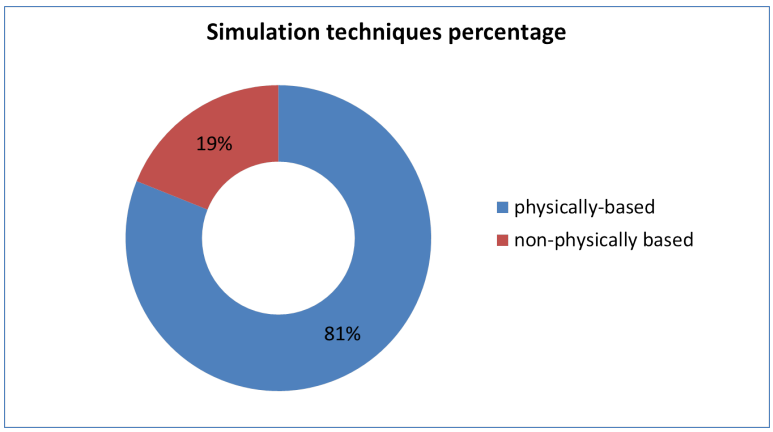

Figure 5.4: Based on the articles cited in this thesis. This graphic represent the percentage of use of both, physically-based and non-physically based methods.

image-guided simulation as a base, the use of a single image would not be probably enough, especially for large fracture, where the use of geometry might be relevant.

This master thesis represents the first step towards a PhD on this topic, which we plan to pursue afterwards. The PhD goal is to propose an inverse model of fracture from either images or other kinds of acquired data. The method will be based on an existing simulation model, and the purpose will be to find the simulation parameters that led to a specific fracture. To do this, it was necessary to deeply study the state of the art on fracture modeling, not just for knowing which kind of simulation could be better for the inverse model, but to know the work done in this field.

Our idea is to base our work in urban virtual environments as well. Current existing models focus on reproducing fracture caused by impacts, explosions, or similar. In this case, the fracture process tends to be a bit differently. We think that the most common kind of materials in these environments are construction materials, where fractures are caused by indirect external factors, such as structures movements, detachments not caused by the men, displacement for organic growth, etc. Furthermore, these materials are usually resistent and less deformable. For this reason, an accurate deformation model is probably not necessary. In those cases, it is probably not necessary to take into account collisions either. Given our ambit of work, and the fact that we want to work with large-scale environments, looking for a fast simulation model rather than a physically accurate one

is also more suitable. After analyzing the state of the art, we believe this methodology has yet to be explored and will provide many advantages over existing approaches. Figure 5.5 represent a general overview of our inverse model process.

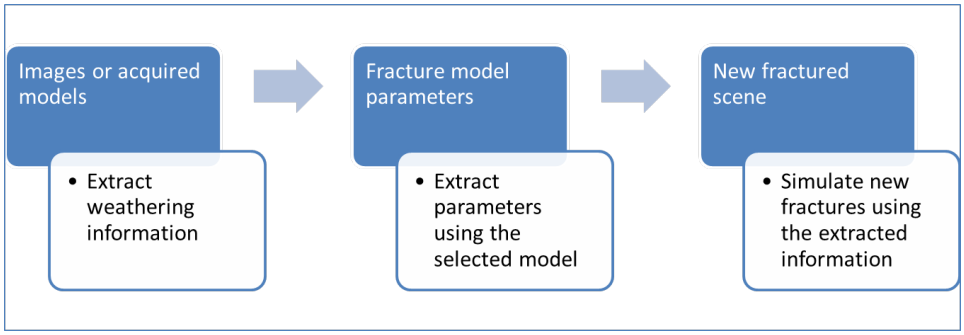

Figure 5.5: A general overview of our inverse model process.

Bibliography

[1] Nealen Andrew, Mueller Matthias, Keiser Richard, Boxerman Eddy, and Carlson Mark. Physically Based Deformable Models in Computer Graphics. *Computer Graphics Forum*, 25(4):809–836, December 2006.

[2] Kimiya Aoki, Ngo Hai Dong, Toyohisa Kaneko, and Shigeru Kuriyama. Physically based simulation of cracks on drying 3d solids. *Computer Graphics International Conference*, pages 357–364, 2004.

[3] G. Ashraf and W. Cheong. Dust and water splashing models for hopping figures. *Pacific Conference on Computer Graphics and Applications*, page 177, 1998.

[4] Martinet Aurélien, Galin Eric, Desbenoit Brett, and Hakkouche Samir. Procedural modeling of cracks and fractures. In *Shape Modelling International (Short Paper)*, pages 346–349, Genova, Italy, 2004.

[5] Lu T. Belytschko, D. Organ, and Y. Krongauz. A coupled finite element-elementfree galerkin method. In *Computational Mechanics*, pages 17–186–195, 1995.

[6] Lu T. Belytschko and M. Tabbara. Element-free galerkin method for wave propogation and dynamic fracture. In *Computer Methods in Applied Mechanics and Engineering*, pages 126–131–153, 1995.

[7] Bedrich Benes and Xabier Arriaga. Table mountains by virtual erosion. In Pierre Poulin and Eric Galin, editors, *Eurographics Workshop on Natural Phenomena*, pages 33–39, 2005.

[8] James F. Blinn. Simulation of wrinkled surfaces. In *SIGGRAPH '78-Proceedings of the 5th annual conference on Computer graphics and interactive techniques*, pages 286–292, New York, NY, USA, 1978.

[9] Carles Bosch, Pierre-Yves Laffont, Holly Rushmeier, Julie Dorsey, and George Drettakis. Image-guided weathering- a new approach applied to flow phenomena. *ACM Transactions on Graphics*, 2011. Accepted-to be presented at SIGGRAPH 2011.

[10] Carles Bosch, Xavier Pueyo, Stphane Mrillou, and Djamchid Ghazanfarpour. A physically-based model for rendering realistic scratches. *Computer Graphics Forum*, 23(3):361–370, 2004.

[11] P. Breitkopf, A. Rassineux, J.M. Savignat, , and P. Villon. Integration constraint in diffuse element method. *Computer Methods in Applied Mechanics and Engineering*, 193:1203–1220, 2004.

[12] Wyvill Brian, van Overveld Kees, and Carpendale Sheelagh. Rendering cracks in batik. In *Proceedings of the 3rd international symposium on Non-photorealistic animation and rendering*, NPAR '04, pages 61–149, New York, NY, USA, 2004. ACM.

[13] Yao-Xun Chang and Zen-Chung Shih. Physically-based patination for underground objects. *Computer Graphics Forum*, 19(3), 2000.

[14] Stéphane Cotin, Delingette Hervé, and Ayache Nicholas. Efficient Linear Elastic Models of Soft Tissues for Real-time Surgery Simulation. Technical Report RR-3510, INRIA, October 1998.

[15] E. Cueto, N. Sukumar, B. Calvo, J. Cegonino, and M. Doblare. Overview and recent advances in natural neighbor galerkin methods. *Computational Methods in Engineering*, pages 307–384, 2002.

[16] Baraff David and Witkin Andrew. Large steps in cloth simulation. In *Proceedings of the 25th annual conference on Computer graphics and interactive techniques*, SIGGRAPH '98, pages 43–54, New York, NY, USA, 1998. ACM.

[17] Steinemann Denis, Otaduy Miguel A., and Gross Markus. Fast arbitrary splitting of deforming objects. In *Proceedings of the 2006 ACM SIGGRAPH/Eurographics symposium on Computer animation*, SCA '06, pages 63–72. Eurographics Association, 2006.

[18] Brett Desbenoit, Eric Galin, and Samir Akkouche. Simulating and modeling lichen growth. *Computer Graphics Forum*, pages 341–350, 2004.

[19] Mathieu Desbrun, Peter Schröder, and Alan Barr. Interactive animation of structured deformable objects. In *Proceedings of the 1999 conference on Graphics interface '99*, pages 1–8, San Francisco CA, USA, 1999. Morgan Kaufmann Publishers Inc.

[20] Julie Dorsey and Pat Hanrahan. Modeling and rendering of metallic patinas. In *SIGGRAPH '96 Proceedings of the 23rd annual conference on Computer graphics and interactive techniques*, pages 387–396, 1996.

[21] Julie Dorsey, Hans Khling Pedersen, and Pat Hanrahan. Flow and changes in appearance. In *In SIGGRAPH 96 Conference Proceedings*, pages 411–420, 1996.

[22] BAKER W. E. Explosion in air. 1973.

[23] GDOUTOS E. E. Fracture mechanics. Technical report, Netherlands, 1993.

[24] Blinn James F. Light reflection functions for simulation of clouds and dusty surfaces. *SIGGRAPH Computer Graphics*, 16:21–29, July 1982.

[25] Pavol Federl and P. Prusinkiewicz. Modelling fracture formation in bilayered materials, with applications to tree bark and drying mud. In *Proceedings of Western Computer Graphics Symposium*, 2002.

[26] Y. C. Fung. *A First Course in Continuum Mechanics*. 1969.

[27] Ramanarayanan G, Ferwerda J, Walter B, and Bala K. Visual equivalence- towards a new standard for image fidelity. *ACM SIGGRAPH*, page 26, 2007.

[28] Taubman Gabriel and Chang Edwin. A fast fracture method for exploding structures. In *ACM SIGGRAPH 2004 Posters*, SIGGRAPH '04, pages 88–, New York, NY, USA, 2004. ACM.

[29] Miller Gavin. Efficient algorithms for local and global accessibility shading. In *Proceedings of the 21st annual conference on Computer graphics and interactive techniques*, SIGGRAPH '94, pages 319–326, New York, NY, USA, 1994. ACM.

[30] Sarah F. F. Gibson and Brian Mirtich. A survey of deformable modeling in computer graphics. Technical report, 1997.

[31] Debunne Gilles, Desbrun Mathieu, Cani Marie-Paule, and Barr Alan H. Dynamic real-time deformations using space & time adaptive sampling. In *Proceedings of the 28th annual conference on Computer graphics and interactive techniques*, SIGGRAPH '01, pages 31–36. ACM, 2001.

[32] R. A. Gingold and J. J. Monaghan. Smoothed particle hydrodynamics-theory and application to non-spherical stars. Technical report, 1977.

[33] Stephane Gobron and Norishige Chiba. Crack pattern simulation based on 3d surface cellular automaton. *Computer Graphics International Conference*, 0:153, 2000.

[34] Jinwei Gu, Chien-I Tu, Ravi Ramamoorthi, Peter Belhumeur, Wojciech Matusik, and Shree Nayar. Time-varying surface appearance- acquisition, modeling and rendering. *ACM Transaction Graphics*, 25:762–771, July 2006.

[35] Michael Hauth, Olaf Etzmuss, and Universitt Tbingen. A high performance solver for the animation of deformable objects using advanced numerical methods. In *In Proc. Eurographics 2001*, pages 319–328, 2001.

[36] Koichi Hirota, Yasuyuki Tanoue, and Toyohisa Kaneko. Generation of crack patterns with a physical model. *The Visual Computer*, 14(3):126–137, 1998.

[37] Koichi Hirota, Yasuyuki Tanoue, and Toyohisa Kaneko. Simulation of three-dimensional cracks. *The Visual Computer*, pages 371–378, 2000.

[38] Donald House, Richard W. Devaul, and David E. Breen. Towards simulating cloth dynamics using interacting particles. *International Journal of Clothing Science and Technology*, 8:75–94, 1996.

[39] Hsien-Hsi Hsieh and Wen-Kai Tai. A straightforward and intuitive approach on generation and display of crack-like patterns on 3d objects. In *Computer Graphics International*, pages 554–561, 2006.

[40] Siu-Chi Hsu and Tien-Tsin Wong. Simulating dust accumulation. *IEEE Computer Graphics and Applications*, 15:18–22, 1995.

[41] Tomoya Ito, Tadahiro Fujimoto, Kazunobu Muraoka, and Norishige Chiba. Modeling rocky scenery taking into account joints. In *Computer Graphics International*, pages 244–247, 2003.

[42] Ferwerda JA. Three varieties of realism in computer graphics. *SPIE human vision and electronic imaging*, page 290, 2003.

[43] Heo Jae-Pil, Seong Joon-Kyung, Kim DukSu, Otaduy Miguel A., Hong Jeong-Mo, Tang Min, and Yoon Sung-Eui. Fastcd- fracturing-aware stable collision detection. In *Proceedings of the 2010 ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, SCA '10, pages 149–158. Eurographics Association, 2010.

[44] Dorsey Julie, Rushmeier Holly, and Sillion Franois. *Digital Modeling of Material Appearance*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 2007.

[45] G. R. Liu. Mesh free methods moving beyond the finite element method. *Applied Mechanics Reviews*, 2003.

[46] G. R. Liu and Y. T. Gu. A point interpolation method for two dimensional solids. *Numer. Methods Eng.*, 50:937–951, 2001.

[47] Y. Q. Liu, H. B. Zhu, X. H. Liu, and E. H. Wu. Real-time simulation of physically based on-surface flow. *The Visual Computer*, 21(8-10):727–734, 2005.

[48] B. A. Lloyd, G. Szkely, and M. Harders. Identification of spring parameters for deformable object simulation. *IEEE Transactions on Vizualisation and Computer Graphics*, pages 1081–1094, 2007.

[49] Jianye Lu, Athinodoros S. Georghiades, Andreas Glaser, Hongzhi Wu, Li-Yi Wei, Baining Guo, Julie Dorsey, and Holly Rushmeier. Context-aware textures. *ACM Transactions Graphics*, 26(1):3, 2007.

[50] B. L. Lucy. A numerical approach to testing the fission hypothesis. 82:1013–1924, 1977.

[51] Müller M., Keiser R., Nealen A., Pauly M., Gross M., and Alexa M. Point based animation of elastic plastic and melting objects. In *Proceedings of the 2004 ACM SIGGRAPH/Eurographics symposium on Computer animation*, SCA 2004, pages 141–151. Eurographics Association, 2004.

[52] Wicke Martin, Ritchie Daniel, Klingner Bryan M., Burke Sebastian, Shewchuk Jonathan R., and OBrien James F. Dynamic local remeshing for elastoplastic simulation. *ACM Transaction Graphics*, 29:49–1–49–11, July 2010.

[53] Claude Martins, John Buchanan, and John Amanatides. Visually believable explosions in real time. *Computer Animation, 2001. The Fourteenth Conference on Computer Animation*, 2001.

[54] Müller Matthias, Dorsey Julie, McMillan Leonard, Jagnow Robert, and Cutler Barbara. Stable real-time deformations. In *Proceedings of the 2002 ACM SIGGRAPH/Eurographics symposium on Computer animation*, SCA '02, pages 49–54, New York, NY, USA, 2002. ACM.

[55] Wojciech Matusik, Hanspeter Pfister, Matt Brand, and Leonard McMillan. A data-driven reflectance model. *ACM Transaction Graphics*, 22:759–769, July 2003.

[56] Oleg Mazarak, Claude Martins, and John Amanatides. Animating exploding objects. In *In Proceedings of Graphics Interface*, pages 211–218. Morgan Kaufmann Publishers Inc, 1999.

[57] Stephane Merillou, Jean-Michel Dischler, and Djamchid Ghazanfarpour. Corrosion- simulating and rendering. In *GRIN'01- No description on Graphics interface 2001*, pages 167–174, Toronto, Ont., Canada, Canada, 2001. Canadian Information Processing Society.

[58] Stéphane Mérillou and Djamchid Ghazanfarpour. A survey of aging and weathering phenomena in computer graphics. *Computers and Graphics*, 32(2):159–174, 2008.

[59] Matthias Mller, Matthias Teschner, and Markus Gross. Physically-based simulation of objects represented by surface meshes. In *Computer Graphics Int*, pages 156–165, 2004.

[60] Neil Molino, Zhaosheng Bao, and Ron Fedkiw. A virtual node algorithm for changing mesh topology during simulation. *ACM Transactions Graphics (SIGGRAPH Procceding*, 23:385–392, 2004.

[61] David Mould. Image-guided fracture. In *GI '05 - Proceedings of Graphics Interface 2005*, pages 219–226, School of Computer Science, University of Waterloo, Waterloo, Ontario, Canada, 2005. Canadian Human-Computer Communications Society.

[62] Matthias Müller, Leonard McMillan, Julie Dorsey, and Robert Jagnow. Real-time simulation of deformation and fracture of stiff materials. In *Proceedings of the Eurographic workshop on Computer animation and simulation*, pages 113–124, New York, NY, USA, 2001. Springer-Verlag New York, Inc.

[63] F. K. Musgrave, C. E. Kolb, and R. S. Mace. The synthesis and rendering of eroded fractal terrains. *SIGGRAPH Computer Graphics*, 23:41–50, July 1989.

[64] Iben Hayley N. and O'Brien James F. Generating surface crack patterns. *Graphics Models*, 71:198–208, November 2009.

[65] K. Nagashima. Computer generation of eroded valley and mountain terrains. In *The Visual Computer*, volume 13, pages 456–464, 1997.

[66] Michael Neff and Eugene Fiume. A visual model for blast waves and fracture. In *In Proceeding of Graphics Interface*, pages 193–202, 1999.

[67] T. Nishioka. Computational dynamic fracture mechanics. *International Journal of Fracture*, 1997.

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

[69] J. O'Brien and J. Hodgins. Graphical model and animation of brittle fracture. In *In Proceedings of SIGGRAPH'99*, pages 137–146, 1999.

[70] James F. O'Brien, Adam W. Bargteil, and Jessica K. Hodgins. Graphical modeling and animation of ductile fracture. *ACM Transactions Graphics*, 21(3):291–294, 2002.

[71] James F. O'Brien and Jessica K. Hodgins. Animating fracture. *ACM Transactions Graphics*, 43(7):68–75, 2000.

[72] E. Onate, S. Idelsohn, O. C. Zienkiewicz, , and R. L. Taylor. A finite point method in computational mechanics. application to convective transport and fluid flow. 39:3839–3866, 1996.

[73] E. Onate, S. Idelsohn, O. C. Zienkiewicz, R. L. Taylor, , and C. Sacco. A stabilized finite point method for analysis of fluid mechanics problems. *Computer Methods in Applied Mechanics and Engineering*, 139:315–346, 1996.

[74] Eric Paquette, Pierre Poulin, and George Drettakis. Surface aging by impacts. In *Proceedings of Graphics Interface 2001*, June 2001.

[75] Eric Paquette, Pierre Poulin, and George Drettakis. The simulation of paint cracking and peeling. In *Graphics Interface 2002*, pages 59–68, May 2002.

[76] Volino Pascal and Thalmann Nadia Magnenat. Implementing fast cloth simulation with collision response. In *Proceedings of the International Conference on Computer Graphics*, CGI '00, pages 257–, Washington DC USA, 2000. IEEE Computer Society.

[77] Mark Pauly, Richard Keiser, Bart Adams, Philip Dutré, Markus Gross, and Leonidas J. Guibas. Meshless animation of fracturing solids. *ACM Transactions Graphics*, 24(3):957–964, 2005.

[78] KEISER R. KOBBELT L. P. PAULY M. and M. GROSS. Shape modeling with point-sampled geometry. In *ACM Transactions Graphics*, volume 22, pages 641–650, 2003.

[79] Narayanan Ramanathan and Student Member. Face verification across age progression. In *IEEE Conf. Computer Vision and Pattern Recognition*, pages 462–469, 2005.

[80] J.N. Reddy. *An introduction to the finite element method*. McGraw-Hill New York, 1993.

[81] Gobron S. and Chiba N. Crack pattern simulation based on 3d surface cellular automaton. In *the Visual Computer*, pages 287–309, 2001.

[82] Martin Sebastian, Kaufmann Peter, Botsch Mario, Grinspun Eitan, and Gross Markus. Unified simulation of elastic rods, shells, and solids. In *ACM SIGGRAPH 2010 papers*, SIGGRAPH 2010, page 39. ACM, 2010.

[83] Heung-Yeung Shum and Sing Bing Kang. A review of image-based rendering techniques. *IEEESPIE Visual Communications and Image Processing VCIP*, 2003.

[84] Jeffrey Smith, Andrew Witkin, and David Baraff. Fast and controllable simulation of the shattering of brittle objects. In *In Proceeding of Graphics Interface*, pages 27–34. Blackwell Publishing, 2000.

[85] N. Sukumar, B. Moran, T. Black, and Lu T. Belytschko. An element-free galerkin method for three-dimensional fracture mechanics. In *Computational Mechanics*, pages 170–175, 1997.

[86] D. V. Swenson and A. R. Ingrafiea. Modeling mixed-mode dynamic crack propogation using finite elements- theory and applications. In *Computational Mechanics*, volume 3, pages 381–397, 1988.

[87] Lefebvre Sylvain and Neyret Fabrice. Synthesizing Bark. In Simon Gibson and Paul E. Debevec, editors, *13th Eurographics Workshop on Rendering Techniques (EGSR '02)*, volume 28, pages 105 – 116, Pisa, Italy, 2002. Eurographics Association.

[88] D. terzopoulo and k. Fleischer. Modelling inelastic deformation- viscoelasticity, plasticity, fracture. *Computer Graphics*, 22(4):269–278, 1988.

[89] Gilles Valette, Michel Herbin, Laurent Lucas, and Joël Léonard. A preliminary approach of 3d simulation of soil surface degradation by rainfall. In *Eurographics Workshop on Natural Phenomena*, pages 41–49, 2005.

[90] Gilles Valette, Stéphanie Prévost, Laurent Lucas, and Joël Léonard. A dynamic model of cracks development based on a 3d discrete shrinkage volume propagation. *Computer Graphics Forum*, 27(1):47–62, 2007.

[91] T. Ivanov Vassilev, B. Spanlang, and Yiorgos Chrysanthou. Fast cloth animation on walking avatars. *Computer Graphics Forum*, 20:260–267, 2001.

[92] G. San Vicente, C. Buchart, D. Borro, and J. Celigeta. Maxillofacial surgery simulation using a mass-spring model derived from continuum and the scaled displacement method. *International journal of computer assisted radiology and surgery*, 4(1):89–98, 2009.

[93] Xi Wang, Lifeng Wang, Ligang Liu, Shimin Hu, and Baining Guo. Interactive modeling of tree bark. *11th Pacific Conference on Computer Graphics and Applications*, 2003.

[94] Tien-Tsin Wong, Wai-Yin Ng, and Pheng-Ann Heng. A geometry dependent texture generation framework for simulating surface imperfections. In *Rendering Techniques*, pages 139–150, 1997.

[95] Y. Wu, P. Kalra, L. Moccozet, and N. Magnenat-Thalmann. Simulating wrinkles and skin aging. *The Visual Computer*, 15(4):183–198, July 1999.

[96] J. x. Chen and Xiaodong Fu. Integrating physics-based computing and visualization- modeling dust behavior. *Computing in Science and Engineering*, 1999.

[97] Chen Yanyun, Xia Lin, Wong Tien-Tsin, Tong Xin, Bao Hujun, Guo Baining, and Shum Heung-Yeung. Visual simulation of weathering by y-ton tracing. *ACM Transactions Graphics*, 24:1127–1133, July 2005.

[98] Gary D. Yngve, James F. O'Brien, and Jessica K. Hodgins. Animating explosions. In *Proceedings of ACM SIGGRAPH 2000*, pages 29–36, August 2000.