COMPUTER ANIMATION AND VIRTUAL WORLDS
Comp. Anim. Virtual Worlds 2011; **22**:115–124
Published online 12 April 2011 in Wiley Online Library (wileyonlinelibrary.com). DOI: 10.1002/cav.412

SPECIAL ISSUE PAPER

Meshless simulation of brittle fracture

Ning Liu, Xiaowei He, Sheng Li and Guoping Wang*

Graphics and Interactive Technology Lab, Peking University, Beijing, China

ABSTRACT

We propose a meshless method to simulate brittle fracture. For brittle solids, stress computation can be difficult because brittle materials generally require small timesteps which bring about heavy computational burden. Furthermore, treating the brittle objects as deformable bodies will cause inevitable visual artifact. We treat the brittle objects as fully rigid bodies and solve the brittle stress distribution with Meshless Local Petrov-Galerkin as a quasistatic problem, so visual artifact disppears and no timestep restriction exists. As a meshless framework, our method has the advantage of easy-resampling around high stress areas to improve computation accuracy. To generate fractured pieces, unlike previous methods which explicitly track the crack propagation, we also present a novel damage based model. Our model supports user-control of the fracture pattern which is especially useful when simulating anisotropic materials such as glass or wood. Results show that our meshless framework is physically feasible and user controllable. Copyright © 2011 John Wiley & Sons, Ltd.

KEYWORDS

meshless method; brittle fracture; rigid simulation

***Correspondence**

Guoping Wang, Graphics and Interactive Technology Lab, Peking University, Peking, China.
E-mail: wgp@pku.edu.cn

1. INTRODUCTION

Fracture is a natural phenomena which plays an important role in animated films, virtual reality and video games, it is almost an indispensable element in realistic virtual scenes with impressive visual effects. These visual 'eye-candys' were traditionally designed and synthesised by experienced artists, but recently, fracture simulation based on computational fracture mechanics [1] achieves great success in generating more realistic results with minimal manual work. Classic finite element method (FEM) is used intensively in physically based deformation and fracturing. However, the accuracy of FEM computation calls for maintaining a high quality mesh which is sometimes costly and difficult. Recent years, meshless methods were developed with the objective of eliminating part of the difficulties and have been applied to graphics successfully such as Refs. [2,3].

For brittle fracture simulation, the stress distribution is required to determine which part of the object is going to break. However, computing the brittle stress distribution proves to be difficult. Treating the brittle solids as deformable bodies with explicit methods will result in large Young's modulus and tiny timesteps which increases computational burden. Worse still, visual artifact appears because the tiny deformation will sometimes be noticeable [4,5]. In FEM methods, techniques were developed to treat brittle objects as rigid objects and compute brittle stress through quasistatic analysis [4–6]. Unfortunately, no previous work of brittle fracture has been proposed for a meshless framework. One representative framework [2] computed stress based on the gradient of displacement field and achieved good results for deformable fracture, but their explicit method was still not suitable to handle brittle stress computation. To fill in this gap, we introduce the Meshless Local Petrov-Galerkin(MLPG) method coupled with rigid dynamics to solve the brittle fracture problem.

Controllability is another requirement in computer graphics when simulating fracture. Tuning physical parameters to get satisfactory results is time consuming and needs skills, which is not very friendly to end users such as artists. Unlike previous methods which explicitly track the crack propagation, we propose a novel damage based model to generate crack and fragments, which is very convenient to take control of the fracture pattern.

Overall, our **contributions** are:

- We novelly introduce MLPG into brittle stress analysis by solving a quasistatic problem. To ensure computational accuracy near high stress areas, we presented

an iterative resampling technique, which is more convenient than remeshing in FEM.
- We propose a simple but effective damage-based fracture model. Our method generates the fragments based on a two-state damage variable, and has good control over the fracture pattern when simulating anisotropic fracture.
- We implement a meshless framework to handle brittle fracture with physical accuracy as well as controllability.

2. RELATED WORK

Our work is closely related to meshless methods and fracture simulation in computer graphics.

2.1. Meshless Methods

Meshless methods such as smooth particle hydrodynamics is quite popular in fluid simulation [7], fluid–solid interaction [8] and granular material simulation [9]. The nature of meshless methods make them good choices for simulating frequently topology-changing objects such as fluid. In recent years, researchers also tried to simulate solid with meshless methods. Reference [10] modeled soft inelastic material using particle systems. Reference [11] introduced smooth particle hydrodynamics (SPH) method to model deformable solids. By extending fluid with viscoelasticity and viscoplasticity, toothpaste-like solid can be simulated [12]. More realistic animation were achieved using computational mechanics [13,14]. Reference [14] used a moving least square method to compute deformation gradient. Their method was capable of simulating a wide range of elastically and plastically deformable objects, however, the method only worked well if each particle had at least three neighbours at non-degenerate locations. Reference [15] coupled meshless finite element formulation and keyframe targeting, motion control of the simulated deformable objects was achieved at interactive rates. Reference [16] replaced Ref. [14]'s MLS method with SPH, which has the advantage of handling coarsely sampled or even coplanar particle configuration. However, their formation was not rotationally invariant, which led to erroneous rotation. Reference [17] solved the incorrect rotation problem in Ref. [16] from a co-rotational point of view. Reference [18] introduced a interactive technique called shape matching which was later improved by Ref. [19,20]. More recently, Ref. [3] developed an accurate unified treatment of elastica. They derived a new quadrature rule for volumetric deformation fields which offered unified treatment when simulating spanning rods, shells and solids.

2.2. Fracture Simulation

The early work on cutting and fracturing of deformable objects dates back to Ref. [21]. Reference [22] presented a finite element model for brittle solids simulation. By analysing the stress tensors, they computed how crack initiates and propagates. Reference [4] proposed a stiff material fracture framework, they only computed the effects of impact forces at discrete collision events using finite element technique, real-time simulation was achieved. Reference [23] extended their previous work on brittle fracture to simulate ductile fracture, plastic deformation was modeled by updating the plastic strain. Geometric methods for cutting and splitting are also used in fracture simulation [24]. Built upon the work of meshless animation of elastic and plastic deformable solid [14,2] presented a new meshless framework for elastic and plastic materials that fracture. They dynamically sampled the volume and explicitly advanced the crack front to generate new fracture surface. Extending method [4,5] treated the material as fully rigid bodies in the limit of infinite stiffness, so that the computation will not suffer from the small timesteps. Their framework was also based on finite element analysis. Reference [16] presented a unified particle model for both fluid and solid. In their framework, a simple fracture strategy was proposed by checking the distances between neighbouring particles. Based on a co-rotational tetrahedral FEM, Ref. [25] described their system featuring fast deformation and fracture simulation in a real-time gaming context. Reference [26] simulated thin shell using low-resolution simulation meshes, which showed a novel combination of texture concepts and physical simulation. Targeting on sound synthesis, Ref. [6] used a fast quasistatic stress solver to resolve near-audio-rate fracture events.

3. OVERVIEW

Figure 1 shows the pipeline of our meshless simulation framework. In each timestep, objects move according to rigid body dynamics. When collision happens, collision response, such as a penalty force is loaded into our MLPG solver for stress computation (section). Meanwhile, adaptive sampling is applied to achieve better accuracy in high stress areas (section). The resampling and stress computation are called iteratively for several times to improve accuracy. After that we solve the problem of fragmentation if necessary (section). Then rigid movement

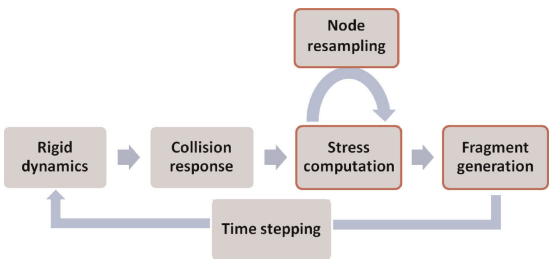

Figure 1. Overview of our simulation pipeline.

DOI: 10.1002/cav

is integrated into next timestep. We show the implementation details and results in section.

4. MESHLESS BRITTLE FRACTURE

In computational mechanics, meshless methods receive a lot of attention due to its potential in eliminating the costly effort of mesh generation, which is a common operation in finite element analysis. Among different meshless methods, the MLPG approach, first presented by Ref. [27] can be considered as a general framework for the other meshless methods. One can choose any meshfree approximations and any convenient test function for the solution [13]. MLPG is also called true meshless method without background grid for integration [27]. We choose a variation of MLPG called MLPG 5 [28] in which only boundary integration is needed for each local sub domain.

4.1. Quasistatic MLPG Formation

Explicitly simulating rigid fracture [2] required large Young's modulus and small timesteps which make the system very stiff, worse still, there can be inevitably visual artifact under the tiny deformation [4]. So with brittle fracture, we treat the object as undeformable rigid, and we use a quasistatic solver to compute the stress distribution.

The basic idea of quasistatic analysis is to approximate the stress distribution by solving the static equilibrium problem when collision happens. In this way no timestep restrictions is needed, thus we can simulate brittle object with infinite stiffness. The technique has been applied to FEM methods [4–6], but has not been presented in a meshless framework. The quasistatic formation:

$$\mathbf{Ku} = \mathbf{f_{ext}} \tag{1}$$

The displacement field $\mathbf{u} \in R^{3N}$ results from the external force $\mathbf{f_{ext}} \in R^{3N}$. $\mathbf{K}_{3N\times 3N}$ is a matrix where N is the number of simulated nodes. The whole equation reveals the idea of the technique: in order to stay equilibrium with external force $\mathbf{f_{ext}}$, displacement field $\mathbf{u}$ is needed to generate enough elastic force for canceling out the effect of $\mathbf{f_{ext}}$. As we only use the displacement field $\mathbf{u}$ for stress computation, but never for updating the particle positions, the brittle objects appear to be fully rigid without visual 'deformed' artifact.

After the displacement field $\mathbf{u}$ is solved, we then compute the stress field. As sheer rigid rotations do not generate deformation, calculating stress from displacement field with rigid transform involved will cause visual artifacts. This has been considered in finite element based methods [29,5,6] as well as meshless methods [17]. We solve the problem with the co-rotation technique used in Ref. [29,17]. We extract the rotation part $\mathbf{R}$ of the transform $\mathbf{x_0} \rightarrow \mathbf{x_0} + \mathbf{u}$ using polar decomposition, then rotate $\mathbf{u}$ with $\mathbf{R^{-1}}$ into the un-deformed frame to calculate stress, and finally rotate the stress back to current frame by $\mathbf{R}$. In this way, the rigid rotation effect is canceled out.

The equilibrium equations in a volume Ω bounded by surface Γ, are given by

$$\sigma_{ij,j} + b_i = 0 \quad \text{in } \Omega \tag{2}$$

where σ_{ij} is the stress tensor and b_i is body force. Boundary condition is presented by

$$u_i = \bar{u}_i \quad \text{on } \Gamma_u \tag{3}$$

$$\sigma_{ij} n_j = \bar{t} \quad \text{on } \Gamma_t \tag{4}$$

where $\bar{u}_i$ are the prescribed displacements, $\bar{t}_i$ are the prescribed surface tractions, n_j is the outward normal of the boundary. Equations (3) and (4) are geometry and force boundary condition, respectively. MLPG constructs the weak form of the Equation (2) over local sub domains Ω_s, which is a small region around each node inside the whole domain. These local sub domains cover the whole domain Ω and can be of any size or shape. In our implementation, we choose cubes due to their simplicity. The local weak form of Equation (2) for node I is:

$$\int_{\Omega_s^I} (\sigma_{ij,j} + b_i) v_i \mathrm{d}\Omega_s - \alpha \int_{\Gamma_{su}^I} (u_i - \bar{u}_i) v_i \mathrm{d}\Gamma = 0 \tag{5}$$

where v_i is the test function and α is a penalty parameter to enforce geometry boundary. Later we will show discretisation of this continuous formation.

4.2. MLS-Based Displacement Field

In MLPG 5, the trial function is constructed using moving least square approximation. The displacement field $\mathbf{u}$ is approximated by polynomial $\mathbf{a^{Tp}}(\mathbf{x})$ with $\mathbf{p}(x,y,z) = (1,x,y,z)^T$ for linear basis, or $\mathbf{p}(x,y,z) = (1,x,y,z,x^2,y^2,z^2,xy,yz,zx)^T$ for quadratic basis. The coefficients $\mathbf{a} = \mathbf{a}(\mathbf{x})$ can be solved by minimising error term J($\mathbf{a}$)

$$J(\mathbf{a}) = \sum_{i=1}^{n} \omega(\|\mathbf{x} - \mathbf{x_i}\|) \|\mathbf{a^T p}(\mathbf{x_i}) - \mathbf{u_i}\|^2 \tag{6}$$

where the weight kernel $\omega(d)$ can be defined as a 4th order spline

$$w(d) = \begin{cases} 1 - 6\left(\frac{d}{r_I}\right)^2 + 8\left(\frac{d}{r_I}\right)^3 - 3\left(\frac{d}{r_I}\right)^4 & 0 \le d \le r_I \\ 0 & d > r_I \end{cases} \tag{7}$$

r_I represent the limited support of node I. By setting $\partial J / \partial \mathbf{a} = 0$ yields

$$\mathbf{u}(\mathbf{x}) = \sum_{i=1}^{n} \phi_i(\mathbf{x}) \mathbf{u_i} \tag{8}$$

with shape function

$$\phi_i(\mathbf{x}) = \mathbf{p}(\mathbf{x})^{\mathbf{T}} \mathbf{M^{-1}}(\mathbf{x}) \mathbf{p}(\mathbf{x_i}) \omega(\|\mathbf{x} - \mathbf{x_i}\|) \tag{9}$$

and motion matrix

$$\mathbf{M}(\mathbf{x}) = \sum_{i=1}^{n} \omega(\|\mathbf{x}-\mathbf{x_i}\|)\mathbf{p}(\mathbf{x_i})\mathbf{p}(\mathbf{x_i})^{\mathbf{T}} \tag{10}$$

This MLS approximation has one shortcoming, sufficient number of neighbouring nodes are needed to guarantee that $\mathbf{M}(\mathbf{x})$ is invertible. Similar problems exist in previous work [14,2,16] when the simulated nodes come to coplanar or colinear configuration. To handle this issue, we use the generalised moving least square [13]. Compared to traditional MLS, an additional error term Δ is added to $J(\mathrm{a})$ in Equation (6):

$$\Delta = \sum_{i=1}^{n} \sum_{j=1}^{3} \omega(\mathbf{x}-\mathbf{x_i})\|\mathbf{a}^{\mathbf{T}}\mathbf{p}_{,\mathbf{j}}(\mathbf{x_i})-\mathbf{u}_{\mathbf{i},\mathbf{j}}\|^2 \tag{11}$$

$\mathbf{p}_{,\mathbf{j}}$ represents the derivative of $\mathbf{p}$ with respect to component x_j. Minimising the new $J(\mathbf{a})$ leads to a invertible motion matrix

$$\begin{aligned}\mathbf{M}(\mathbf{x}) = &\sum_{i=1}^{n} \omega(\|\mathbf{x}-\mathbf{x_i}\|)(\mathbf{p}(\mathbf{x_i})\mathbf{p}(\mathbf{x_i})^{\mathbf{T}} \\ &+ \sum_{j=1}^{3} \mathbf{p}_{,\mathbf{j}}(\mathbf{x_i})\mathbf{p}_{,\mathbf{j}}(\mathbf{x_i})^{\mathbf{T}})\end{aligned} \tag{12}$$

Compared with the meshless fracture work [2], our method can handle colinear and coplanar configuration, which is critical when simulating small features such as rod or thin sheet.

4.3. Numerical Discretisation

According to Ref. [28], discretising Equation (5) leads to the following equation

$$\sum_{\mathbf{J}=\mathbf{1}}^{\mathbf{N}} \mathbf{K_{IJ}}\mathbf{u_J} = \mathbf{f_I} \tag{13}$$

For node I which lie inside the boundary:

$$\mathbf{K_{IJ}} = -\int_{\partial\Omega_s^I} \mathbf{NDB^J} d\Gamma,\ \mathbf{f_I} = \int_{\Omega_s^I} \mathbf{b} d\Omega \tag{14}$$

For node I on geometry boundary:

$$\mathbf{K_{IJ}} = \phi^J \delta_{IJ} \mathbf{I_{3\times 3}},\quad \mathbf{f_I} = \bar{\mathbf{u}} \tag{15}$$

For node I on force boundary:

$$\mathbf{K_{IJ}} = \mathbf{NDB^J},\quad \mathbf{f_I} = \bar{\mathbf{t}} \tag{16}$$

where

$$\mathbf{N} = \begin{pmatrix} n_1 & 0 & 0 & n_2 & 0 & n_3 \\ 0 & n_2 & 0 & n_1 & n_3 & 0 \\ 0 & 0 & n_3 & 0 & n_2 & n_1 \end{pmatrix}$$

and (n_1, n_2, n_3) is the outward normal on the boundary of local sub-domain.

$$\mathbf{B^J} = \begin{pmatrix} \phi^J_{,x} & 0 & 0 \\ 0 & \phi^J_{,y} & 0 \\ 0 & 0 & \phi^J_{,z} \\ \phi^J_{,y} & \phi^J_{,x} & 0 \\ 0 & \phi^J_{,z} & \phi^J_{,y} \\ \phi^J_{,z} & 0 & \phi^J_{,x} \end{pmatrix}$$

$\mathbf{B^J}$ is composed of the derivatives of shape function ϕ^J_k with respect to the k-component of $\mathbf{x}$.

$$\mathbf{D} = D_0 \begin{pmatrix} 1 & \frac{\nu}{1-\nu} & \frac{\nu}{1-\nu} & 0 & 0 & 0 \\ \frac{\nu}{1-\nu} & 1 & \frac{\nu}{1-\nu} & 0 & 0 & 0 \\ \frac{\nu}{1-\nu} & \frac{\nu}{1-\nu} & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & \frac{1-2\nu}{2(1-\nu)} & 0 & 0 \\ 0 & 0 & 0 & 0 & \frac{1-2\nu}{2(1-\nu)} & 0 \\ 0 & 0 & 0 & 0 & 0 & \frac{1-2\nu}{2(1-\nu)} \end{pmatrix}$$

with $D_0 = [E(1-\nu)]/[(1+\nu)(1-2\nu)]$. E and ν are the Young's modulus and Poissons ratio, respectively. As we choose simply cube as the sub domain, the boundary integration for $\mathbf{K_{IJ}}$ can be easily computed by Gaussian integration. The matrix generated by MLPG is sparse unsymmetric, we use Taucs to solve the system.

As we solve the displacement field for all the particles through the static solver, no tiny timesteps are needed to ensure brittleness, computation cost will drop when compared to the previous work on meshless fracture [2].

5. ITERATIVE RESAMPLING

In FEMs, to guarantee accuracy of the computation around high stress area, frequent remeshing is needed. Changing the resolution may be a non-trivial operation for FEM-based methods, but is quite simple and efficient in our meshless framework. We propose an iterative strategy to handle dynamic resampling to gain better accuracy in high stress areas.

Under the initial sampling configuration, we compute the stress when collision happens as in section 0. Then we resample the computational domain with higher stress through a simple octree-based strategy. In Figure 2, we show the successive particle distribution when collision happens. It is relevant that area A and B are more strain-intensive than area C, which is more likely to break. Our resampling operation also brings additional advantage when handling fracture surface, when constructing fracture surface with 3D triangulation, resampled particles generate more smooth crack surfaces.

The resampling method we presented is different with Ref. [2]. In their paper, the object is adaptively sampled according to the geometric feature. The boundary parts of the object is densely sampled while the inside parts is sparsely sampled. Our resampling method comes from another point of view, in which more particles are planted in stress intensive area. As fracture happens where stress is

DOI: 10.1002/cav

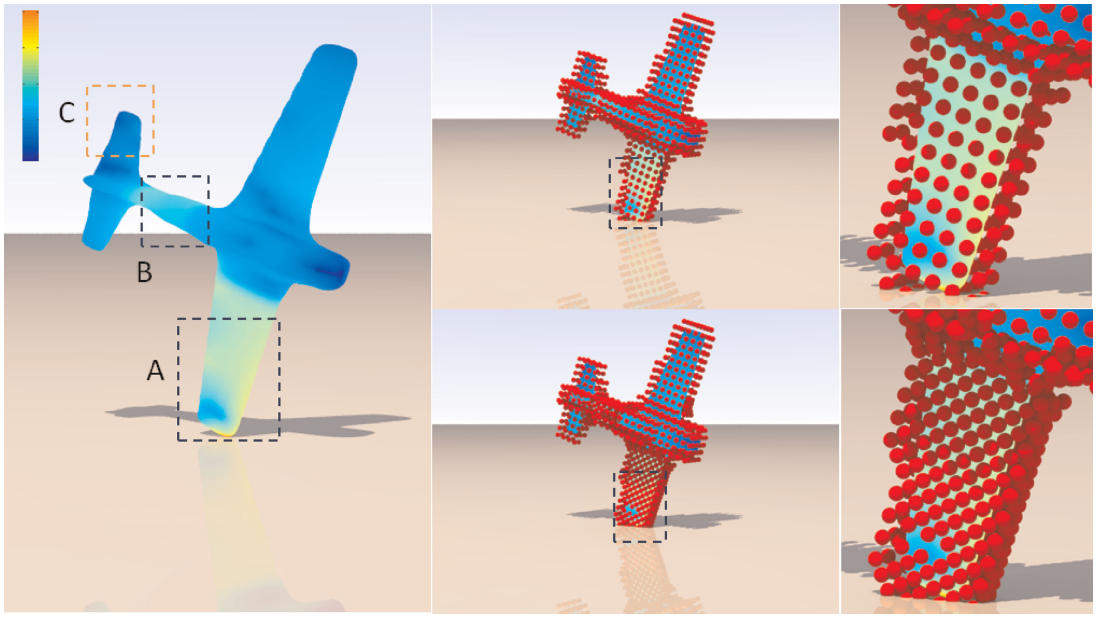

Figure 2. Stress is calculated when the plane model hits the ground. In stress intensive areas, we resample the particles to guarantee accuracy.

intensive, geometric complex parts of an object does not necessarily break. Our method puts more computational effort on high stress areas where fracture might happen, and reduces the number of particles in low stress area, thus decrease the cost of computation.

6. FRAGMENT GENERATION

According to Ref. [1], crack propagation is triggered when the maximal eigenvalue of stress tensor σ exceeds the material threshold. The cracks grow and finally fracture the object into fragments.

6.1. Damage-Based Fracture Model

Unlike previous work such as Ref. [22,2], we do not explicitly track the crack surface to generate new fragments. We solve the fragmentation problem from a damage-based point of view. Damage mechanics has gain success in engineering fields for modeling fracture [30]. The key idea is to introduce a variable D $(0 \leq D \leq 1)$ to describe the damage status of the material, for example, the stress σ for an isotropic material under tension will reduce to $(1-D)\sigma$. When material threshold is exceeded, D begins to accumulate at a constant speed C_d

$$D = D + C_d(t-t_0) \quad (0 \leq D \leq 1) \tag{17}$$

we derive our method from the damage concept by adding D_i to each particle i to represent the damage already happened to its volume. With regard to rigid fracture, the evolution of D_i is completed for an instant, so we set C_d to be infinite large ($C_d \to \infty$). Then the damage D_i can be reduced to a two state variable (whether or not totally damaged), this damage variable only changes from undamaged ($D_i \neq 1$) to damaged ($D_i = 1$) under stress which is beyond threshold.

The damaged particles no longer contribute any stress to neighbouring particles, thus separate the object into different parts. Our algorithm can be summarise into three step:

(1) Compute the maximal eigenvalue of stress tensor to compare with the material's threshold. Then each particle i is assigned as damaged or undamaged depending on whether its threshold is exceeded.
(2) Find groups of connected undamaged particles, each of which form a new fragment.
(3) Cluster the damaged particles into fragments.

We ran depth first search in step 2 with neighbouring table stored during initialisation to find connected part of the object. The disconnected damaged particles are clustered to be small debris or large fragment according to the cluster strategy (next section).

6.2. Cluster Strategy

As the micro flaws inside the object which define the thresholds are hard to model [30], it is difficult to simulate desired fracture pattern by tuning physical parameters. However, in computer graphics, controllability over number/shape of fragments is often desirable especially for the artists. One option is to control the fracture process

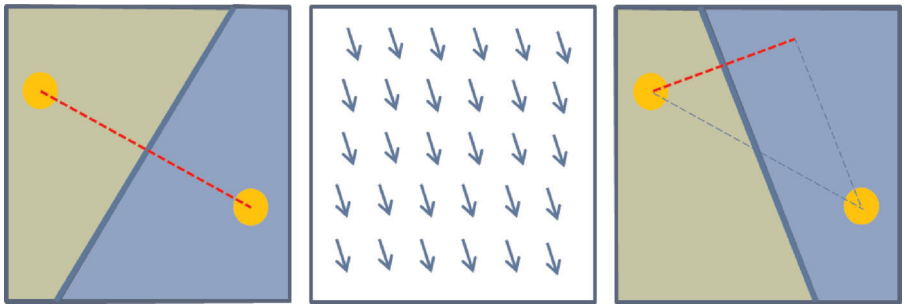

Figure 3. Taking the simplest 2D case as example, the squared object is clustered into one green part and one blue part by two seeds(orange coloured circle), with the fracture line along the perpendicular bisector of segment between the two seeds(left figure). With a vector field (middle figure) and penalty $\lambda = 0$, the components of distance perpendicular to the vector field will be suppressed, then the fracture line will conform to the vector field (right figure). The dotted red lines (left and right figure) show the original distance and its penalty distance according to the vector field.

by pre-setting the stress threshold values. Setting lower stress threshold in user interested area will lead fracture there. This strategy is simple to implement and sometimes can be sufficient for the fracture pattern design. However, for some scenario such as anisotropic fracture pattern such as wood or glass, setting thresholds will be a tedious work to achieve satisfactory results. We propose a vector-field guided cluster method to enforce controllability.

Our method can be divided into two steps of seeding and clustering. We first seed the damaged particles with desired number of fragments then cluster all damaged particles to the seeds. The seeding process can be operated as: add first seed randomly, then add seeds in a greedy way, such that the minimum distance between current seed and the already added seeds is maximised. We also apply the strategy that higher stress areas is covered with more seeds, this will generate more debris for areas such as contact point which is reasonable. This strategy can be used to generate many tiny debris which is common in real world fracture. We propose a penalty distance to seeding and clustering process to guide the fracture with a user defined vector field. With $\mathbf{P_A}$ and $\mathbf{P_B}$ the position vectors, the penalty distance $d(A,B)$ guided by a normalised vector field $\mathbf{v}$ is defined by

$$d(A,B) = \|\overleftarrow{P_AP_B} - (1-\lambda)(\overleftarrow{P_AP_B} \cdot \mathbf{v})\mathbf{v}\| \qquad (18)$$

where the penalty factor λ $(0 \leq \lambda \leq 1)$ suppresses the distance components perpendicular to the given vector field $\mathbf{v}$, enforcing the fragments to be generated along $\mathbf{v}$. This is shown in Figure 3. With this simple technique, we are able to model anisotropic fragmentation such as glass or wood fracture (Figures 8 and 9).

With complex shaped models (e.g. plane model), we use distances along neighbouring particles to ensure connectivity. For distances between particles, we find the shortest distance using the neighbouring information we have stored in the neighbouring table. Computing the shortest path can add cost, but guarantees particles from the same rigid fragment stick together. For rendering detailed triangular surfaces, we just need to cluster the surface triangles based on the already clustered particles.

7. IMPLEMENTATION AND RESULTS

Particles in our implementation are arranged into a spatial hash grid [7] to accelerate neighbour finding. At the beginning of each time step, we insert the particles into the spatial grid for later use. We also implemented a neighbour table to further accelerate the neighbour finding process.

Collision detection is handled between particles. We use penalty forces as the collision response. We assign an id to each particle to indicate to which fragment it belongs, then for rigid bodies, collision only happens between particles with different ids.

For integrating $\mathbf{K_{ij}}$, we use cubes as the sub domains, and 4-point Gaussian integration on the cube boundary is calculated with accuracy, which is the dominant cost of the stress computation process (we use as many as $4 \times 4 \times 6 = 96$ Gaussian integration points on each cube). If efficiency is the main concern, reducing the Gaussian points will help, with the loss of some accuracy.

For rendering, the point-based technique is suitable to our methods [2,31]. However, to take advantage of off-the-shelf high quality ray-tracing tools, we embed a detailed triangle mesh into the particles and the triangles move with the particles with the same rigid body parameters. When the object is fragmented into parts, we generate the fracture surface through a constrained triangulation procedure using the original surface triangles and simulation particles near the crack surfaces. All images in our experiments are rendered using POVRAY(www.povray.org) Figures 4 and 5

Figure 4 shows the dropping test of a coffee cup. When the cup hits the ground, stress computation is triggered, then fragmentation procedure is called to generate new fragments. The whole computation is based on the particle model in Figure 4. Figures 5 and Figure 6 show additional

DOI: 10.1002/cav

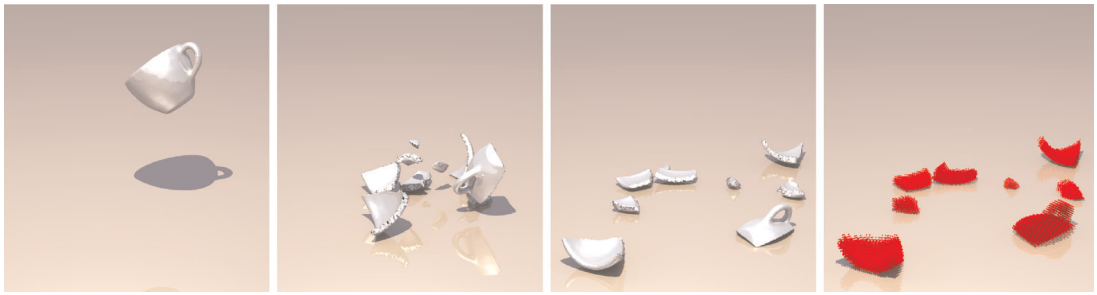

Figure 4. A coffee cup hits the ground and breaks into pieces. Simulation is computed with particles shown in the rightmost figure. we embed the detailed triangle surface into the particles, then update the surface according to the rigid dynamics.

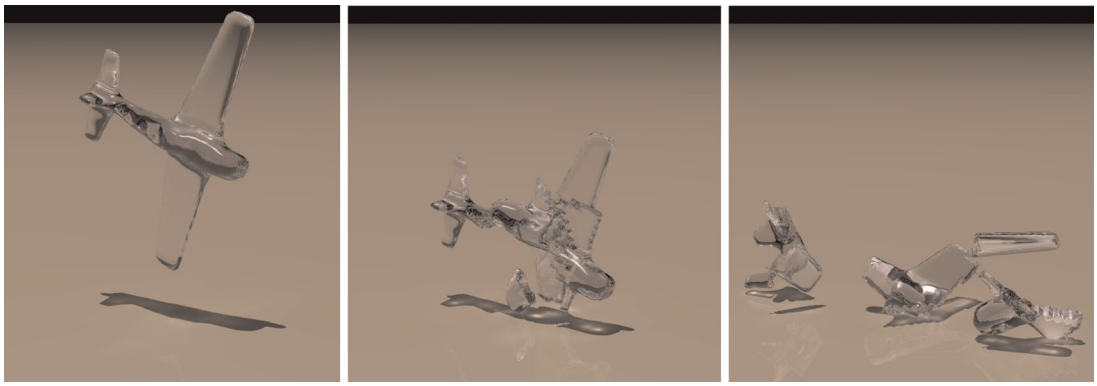

Figure 5. The plane hits the ground and the stress computation is triggered as in Figure 2.

dropping test of a coffee plate and a plane model. Notice that when the plate hits the ground, our fragmentation method generates more debris in the high stress area near the contact point. Multiple fracture is supported in our method, after the first collision, new fragments may break into more pieces in successive collisions. Figure 7 shows the isotropic plaster wall broken by a rigid ball. Two variations are showed in Figures 8 and 9 with anisotropic control. Figure 8 uses a radial vector field around the hitting point while Figure 9 uses a vector field along the wood grain to guide the fracture.

Table 1 shows the geometry complexity and timing of our method. The geometry statistics shows that detailed triangle surfaces can be embedded in particles of much lower resolution. Grid and table represents the cost of building the spatial hash and neighbouring table. Dynamics and collision measures the time of rigid dynamics and collision handling. Stress calculation is the most time consuming part of our method because of the computation intensive Gaussian integration. The fragmentation cost basically changes with the number of particles, except that when shortest distances are computed for complex models (e.g. plane).

General physical parameters are shown in Table 2. Our C++ code runs on PC with 2.93 GHz CPU, NVIDIA GT240 graphics card and 3G RAM memory.

Comparing to previous mesh-based fracture methods such as in Refs. [5,6,22], our method needs minimal work on mesh processing. To prepare the input to our method, particles are inserted through a simple coarse voxelisation,

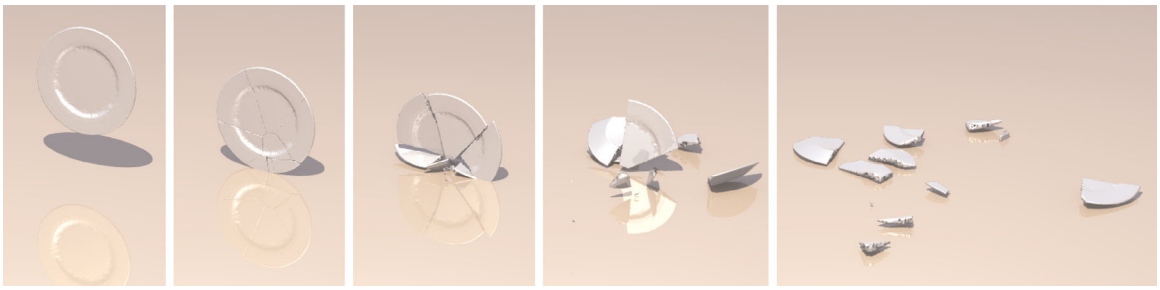

Figure 6. The coffee plate hits the ground and breaks. Notice the small debris generated near the contact point. After the first hit, new fragments may break into more pieces in the successive collisions.

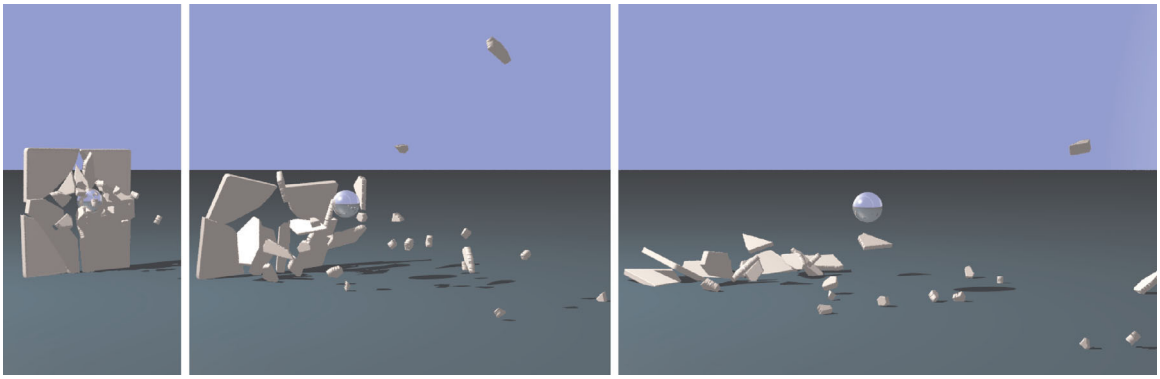

Figure 7. The plaster wall is supposed to be an isotropic material. The ball hits the wall and generate more fragments near the hitting point.

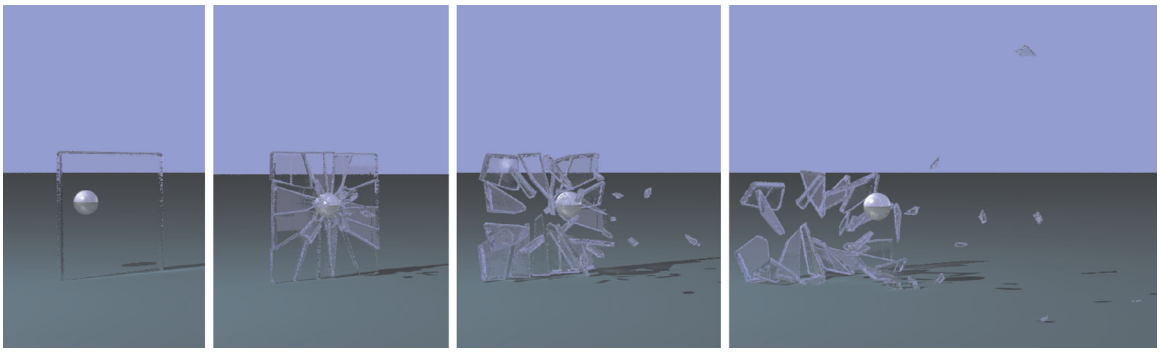

Figure 8. A radial vector field is used to guide the fracture, this gives us more realistic results of the glass breaking effect.

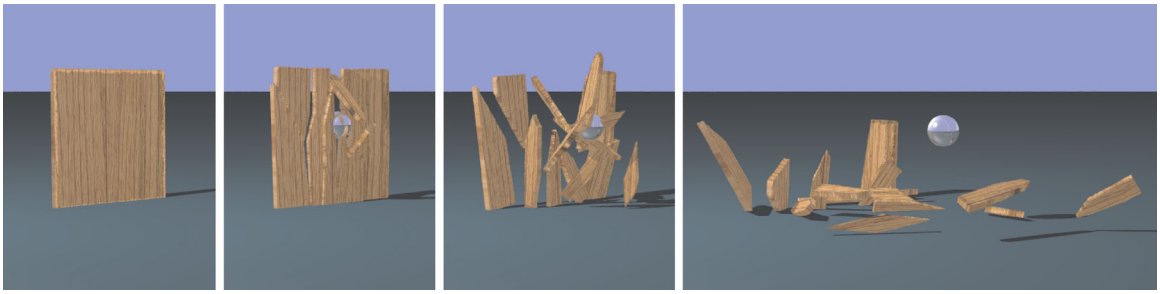

Figure 9. Wood is one type of anisotropic material, the particles along the wood grain tend to hold together. Our vector-guided cluster algorithm can be used to simulate this fracture pattern.

Table 1. Geometry complexity and timing of the test models.

Model	# Rigid	# Triangles	# Particles	Grid & table (milliseconds)	Dynamics & collision (milliseconds)	Stress	Fragmentation	Render (seconds/ frame)
Plaster wall	1→37	154.6k/161.8k	968	9	0.3	5.650 seconds	9.240 milliseconds	3
Glass wall	1→37	154.6k/162.7k	968	9	0.3	5.649 seconds	8.397 milliseconds	50
Wood wall	1→15	154.6k/161.8k	968	9	0.3	5.741 seconds	2.690 milliseconds	4
Coffee plate	1→10	115.1k/118.5k	4857	10	2.0	47 seconds	5.0 seconds	27
Coffee cup	1→10	220.5k/228.5k	12,988	20	5.5	3 minutes 4 seconds	18.0 seconds	68
Plane	1→7	52.5k/91k	2739	14	1.1	30.7 seconds	1 minutes 50 seconds	7

Table 2. General physical parameters.

Parameters	Value
Timestep	$1.0\,E^{-3}$
Gravity	-9.8 m/second^2
Kernel range	0.008 m
Initial spacing	0.0048 m
Young's modulus/Poisson ratio	$5.0\,E^{9}$/0.43
Particle mass	$4.0\,E^{-3}$kg

then resampling is handled easily when necessary. However, to prepare and maintain a high quality FEM mesh can be a non-trivial work.

Comparing to previous meshless fracture methods [2], our method is more suitable to simulate brittle fracture because no timestep constraint is required to ensure the stiffness of the object, computation cost is minimised, and visual artifact is avoided.

8. CONCLUSION AND FUTURE WORK

We have presented a new solution to the brittle fracture problem. By treating the brittle object as fully rigid bodies, no visual artifact of small deformation will appear. When computing the brittle stress, previous explicit meshless methods are not suitable because tiny timesteps are needed which lead to much more computational cost. We introduce MLPG method from the computational mechanics to solve brittle stress through a quasistatic problem, we also propose a resampling method to improve accuracy.

To generate fractured fragments, we introduce a novel damage-based fracture model, which clusters the particles based on a two-state damage variable. Our method is simple but effective to offer control over the fracture pattern.

As a meshless framework, the limitation of our method is that more simulated nodes are needed than FEM methods, which may consume more memory and processing time. One of the possible future work is accelerating the integration in stress computation. Although reducing the number of Gauss integration point will help, we seek to parallel the integration process with graphics hardware. Also, we look forward to applying the MLPG method to more interesting graphics-specific problems.

ACKNOWLEDGEMENTS

We would like to thank the anonymous reviewer for their valuable suggestions. This work was supported by grant No.2010CB328002 from the National Basic Research Program of China, and by Nos. 60833007, 60925007 and 90915010 from National Natural Science Foundation of China. It was also supported by grants No. 2009AA01Z324 from the National High Technology Research and Development Program of China and research fund from the Doctoral Program of Higher Education (No. 20070001024).

REFERENCES

1. Anderson TL. *Fracture Mechanics: Fundamentals and Applications*, CRC Press: Cleveland, OH, 1995.
2. Pauly M, Keiser R, Adams B, Dutré P, Gross M, Guibas LJ. Meshless animation of fracturing solids. *ACM Transactions on Graphics* 2005; **24**(3): 957–964.
3. Martin S, Kaufmann P, Botsch M, Grinspun E, Gross M. Unified simulation of elastic rods, shells, and solids. *ACM Transactions on Graphics* 2010; **29**(4): 1–10.
4. Müller M, McMillan L, Dorsey J, Jagnow R. Real-time simulation of deformation and fracture of stiff materials. In *Proceedings of the Eurographic Workshop on Computer Animation and Simulation*, 2001; 113–124.
5. Bao Z, Hong J-M, Teran J, Fedkiw R. Fracturing rigid materials. *IEEE Transactions on Visualization and Computer Graphics* 2007; **13**(2): 370–378.
6. Zheng C, James DL. Rigid-body fracture sound with precomputed soundbanks. *ACM Transactions on Graphics* 2010; **29**(4): 1–13.
7. Müller M, Charypar D, Gross M. Particle-based fluid simulation for interactive applications. In *Proceedings of the 2003 ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, 2003; 154–159.
8. Keiser R, Adams B, Gasser D, Bazzi P, Dutré P, Gross M. A unified lagrangian approach to solid-fluid animation. In *Proceedings of the Eurographics Symposium on Point-Based Graphics*, 2005; 125–134.
9. Bell N, Yu Y, Mucha PJ. Particle-based simulation of granular materials. In *Proceedings of the 2005 ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, 2005; 77–86.
10. Desbrun M, Gascuel M-P. Animating soft substances with implicit surfaces. In *SIGGRAPH*, 1995; 287–290.
11. Desbrun M, Gascuel MP. Smoothed particles: a new paradigm for animating highly deformable bodies. *Computer Animation and Simulation* 1996; **96**: 61–76.
12. Clavet S, Beaudoin P, Poulin P. Particle-based viscoelastic fluid simulation. In *Proceedings of the 2005 ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, 2005; 228.
13. Fries TP, Matthies HG. *Classification and Overview of Meshfree Methods*. Department of Mathematics and Computer Science, Technical University of Braunschweig, 2003.
14. Müller M, Keiser R, Nealen A, Pauly M, Gross M, Alexa M. Point based animation of elastic, plastic and melting objects. In *Eurographics/SIGGRAPH Symposium on Computer Animation*, 2004.
15. Adams B, Ovsjanikov M, Wand M, Seidel HP, Guibas LJ. Meshless modeling of deformable shapes and their motion. In *Proceedings of the 2008 ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, 2008; 77–86.

16. Solenthaler B, Schläfli J, Pajarola R. A unified particle model for fluid-solid interactions. *Computer Animation and Virtual Worlds* 2007; **18**(1): 69–82.
17. Becker M, Ihmsen M, Teschner M. Corotated sph for deformable solids. In *Proceedings of Eurographics Workshop on Natural Phenomena*, 2009.
18. Müller M, Heidelberger B, Teschner M, Gross M. Meshless deformations based on shape matching. *ACM Transactions on Graphics* 2005; **24**(3): 478.
19. Rivers AR, James DL. Fastlsm: fast lattice shape matching for robust real-time deformation. *ACM Transactions on Graphics* 2007; 26.
20. Steinemann D, Otaduy MA, Gross M. Fast adaptive shape matching deformations. In *Proceedings of the 2008 ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, 2008; 87–94.
21. Terzopoulos D, Fleischer K. Modeling inelastic deformation: viscoelasticity, plasticity, fracture. In *Proceedings of the SIGGRAPH*, 1988; 269–278.
22. O'Brien JF, Hodgins JK. Graphical modeling and animation of brittle fracture. In *SIGGRAPH*, 1999; 137–146.
23. O'Brien JF, Bargteil AW, Hodgins JK. Graphical modeling and animation of ductile fracture. In *SIGGRAPH 2002 Conference Proceedings*, 2002; 291–294.
24. Steinemann D, Otaduy MA, Gross M. Fast arbitrary splitting of deforming objects. In *Proceedings of the 2006 ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, 2006; 63–72.
25. Parker EG, O'Brien JF. Real-time deformation and fracture in a game environment. In *Proceedings of the ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, 2009; 156–166.
26. Kaufmann P, Martin S, Botsch M, Grinspun E, Gross M. Enrichment textures for detailed cutting of shells. *ACM Transactions on Graphics* 2009; **28**(3): 50.
27. Atluri SN, Zhu T. A new meshless local Petrov-Galerkin (MLPG) approach in computational mechanics. *Computational mechanics* 1998; **22**(2): 117–127.
28. Li Q, Shen S, Han ZD, Atluri SN. Application of meshless local Petrov-Galerkin (MLPG) to problems with singularities, and material discontinuities, in 3-D elasticity. *Computer Modeling in Engineering and Sciences* 2003; **4**(5): 571–586.
29. Müller M, Dorsey J, McMillan L, Jagnow R, Cutler B. Stable real-time deformations. In *Proceedings of the 2002 ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, 2002; 21–22.
30. Grady DE, Kipp ME. Continuum modelling of explosive fracture in oil shale. *International Journal of Rock Mechanics and Mining Sciences and Geomechanics Abstracts* 1980; **17**(3): 147–157.
31. Gross M, Pfister H. *Point-Based Graphics*. Morgan Kaufmann Headquarters: Burlington, Massachusetts, 2007.

Authors' Biographies:

Ning Liu is a PhD candidate in Graphics and Interaction Lab, Peking University. His research interests include image synthesis and physically based animation.

Xiaowei He is a master student of the School of AAIS, Peking University. His research interests include computer graphics, physically based simulation.

Sheng Li received his PhD degree in 2005 from the Institute of Software, CAS. Currently, he is an Associate Professor of the School of Electronics Engineering and Computer Sciences, Peking University. He works as a member of the Graphics and Interaction Lab. and publishes over 20 papers in prestigious international conferences and journals. His research interests include computer graphics, virtual reality, GPU computing.

Guoping Wang received his BS and MS degrees from Harbin Institute of Technology, Harbin, China and his PhD degree from Fudan University in 1987, 1990 and 1997, respectively, all in mathematics. Dr. Wang has been a faculty member of the School of Electronics Engineering and Computer Science, Peking University, Beijing, China since 1999. He is now a full professor and the Director of Graphics and Interaction Lab. He is also the Associate Director of the Institute of Soft ware. His research interests include computer graphics, multimedia, virtual reality, geometrical modeling, CAD, and human-computer interaction.