Energized Rigid Body Fracture

XIAOKAI LI, University of Maryland, Baltimore County, United States
SHELDON ANDREWS, École de technologie supérieure, Canada
BEN JONES, University of Utah, United States
ADAM BARGTEIL, University of Maryland, Baltimore County, United States

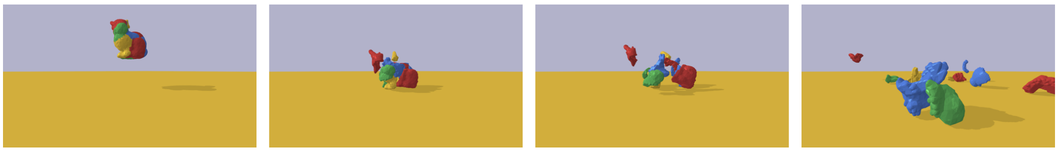

Fig. 1. Frames from an animation of an unfortunate and hollow, yet energetic, bunny fracturing on the ground.

Compelling animation of fracture is a vital challenge for computer graphics. Methods based on continuum mechanics are physically accurate, but computationally expensive since they require computing elastic deformation. In many applications, this elastic deformation is imperceptible, so simulation methods based on rigid body dynamic with breakable constraints are popular in practice. Simply deleting constraints when thresholds on force or displacement are reached ignores the elastic energy that is stored just before fracture, which is captured by continuum mechanics based methods. Our approach computes the energy stored in these constraints when they are broken, and reintroduces it to the system as kinetic energy. As a result, our method is able to animate energetic fracture scenarios with results comparable to continuum mechanics approaches, but with the computational efficiency of rigid body simulation.

CCS Concepts: • **Computing methodologies → Procedural animation**; **Physical simulation**;

Additional Key Words and Phrases: fracture, rigid body dynamics, physics simulation

ACM Reference Format:
Xiaokai Li, Sheldon Andrews, Ben Jones, and Adam Bargteil. 2018. Energized Rigid Body Fracture. *Proc. ACM Comput. Graph. Interact. Tech.* 1, 1, Article 9 (May 2018), 9 pages. https://doi.org/10.1145/3203207

1 INTRODUCTION

The creation of large-scale scenes of destruction has emerged as one of the greatest successes of physics-based animation. In these animations, man-made materials experience extreme forces that lead to failure, with the two most common failure modes being plastic deformation and fracture. In this paper, we limit our discussion to fracture.

Authors' addresses: Xiaokai Li, E-mail:l35@umbc.edu, University of Maryland, Baltimore County, Baltimore, United States; Sheldon Andrews, E-mail:sheldon.andrews@etsmtl.ca, École de technologie supérieure, Montreal, Canada; Ben Jones, E-mail:benjones@cs.utah.edu, University of Utah, Salt Lake City, United States; Adam Bargteil, E-mail:adamb@umbc.edu, University of Maryland, Baltimore County, Baltimore, United States.

2577-6193/2018/5-ART9 $$15.00
https://doi.org/10.1145/3203207

Two general approaches have emerged for animating fracture. The first approach uses continuum mechanics to model internal elastic stress and, when the stress exceeds a threshold, update the object's topology and/or geometry. The second approach foregoes elasticity computations and instead treats objects as rigid bodies. Typically, objects are "pre-scored"—broken up into a set of pieces—as a pre-process, either by an artist or with some (semi-)automatic tool. Then, during simulation each piece of the initial object is treated as an individual rigid body and constraints are added between the individual pieces that make up the object, resulting in behavior resembling a single rigid object. When the magnitude of the constraint forces exceeds a threshold, the constraints are simply removed, creating the fracture effect. Both of these approaches have been successfully applied in video games and film and resulted in Academy Awards in 2015.

Of course, there are tradeoffs between the two approaches. The primary advantages of the rigid body approach are simplicity and speed; modeling elasticity is inherently more complex and involves non-trivial computational costs. For nearly rigid man-made materials there is no visible elastic deformation and these computations are arguably wasted. On the other hand, this internal elastic energy is partly transformed into kinetic energy of individual shards when fracture occurs, which is an important aspect of realistic fracture animation. This phenomenon is missing from existing rigid-body based approaches. We address this limitation by measuring the energy stored in the constraints and explicitly converting this to kinetic energy by applying impulses when the constraints are removed. In other words, we energize rigid body fracture!

As our results demonstrate, transferring the internal constraint energy to kinetic energy produces much livelier simulations of fracture. We provide artists with a single parameter, similar to a coefficient of restitution, that determines how much of the internal energy is transferred. While values in the range $[0, 1]$ are physically valid, artists can create even more energetic environments by choosing values greater than 1. Our simple approach is effective for simulating brittle fracture of near rigid man-made materials.

2 RELATED WORK

Since the pioneering work of Terzopoulos and Fleischer [1988], simulation methods based on continuum mechanics have been used to animate fracture. O'Brien and colleagues [O'Brien et al. 2002; O'Brien and Hodgins 1999] proposed using the finite element method (FEM) to simulate fracture. Many later approaches have also been based on tetrahedral finite elements [Clausen et al. 2013; Wicke et al. 2010]. Bao et al. [2007] approximate brittle fracture by a quasistatic stress analysis. Zheng and James [2010] used a similar approach to synthesize sounds of brittle objects fracturing. They compute a fracture impulse model that assumes unused strain energy in an FEM simulation is converted into kinetic and sound energy.

Parker and O'Brien [2009] demonstrated that an optimized FEM implementation was suitable for interactive applications. Surface-only fracture simulations based on the boundary element method have been introduced recently [Hahn and Wojtan 2016; Zhu et al. 2015]. While not based on continuum mechanics, mass-spring models are also commonly used to animate deformable and fracturing bodies [Norton et al. 1991]. As noted by Jones et al. [2016], these approaches spend computational effort modeling objects as elastic bodies, but many objects we desire to animate do not show visually perceptible elastic deformations. We model objects primarily as rigid bodies, yet our approach is able to produce explosive fracture behavior similar to these elasticity-based approaches, but with much lower computational overhead.

To avoid simulating imperceptible elastic behavior, fracture simulators built on rigid body dynamics models are common in practice. Zafar et al. [2010] use a rigid body physics engine to create complex scenes involving thousands of bodies to simulate destruction of buildings. Numerous point-to-point constraints between objects give the appearance of a rigid structure. Once constraint displacements or constraint impulses surpass a user-defined threshold, the constraints are removed from the simulation thus giving the appearance of fracture. This is equivalent to fracture by pre-scoring the objects that has been used by others [Jones et al. 2016; Weinstein et al. 2008]. To semiautomatically generate fracture pieces, Su et al. [2009] precompute a fracture patten which is aligned with run-time impacts, while Müller et al. [2013] use a geometric approach to generate dynamic fracture patterns at run-time, achieving sharp contours and localized fracture. Our approach describes an improved method for simulating these objects after pre-scoring, and is therefore complementary to these approaches.

3 METHODOLOGY

3.1 Notation

Uppercase boldface is used to indicate a matrix quantity, whereas lowercase boldface indicates a vector. A single subscript $\cdot_i$ is used to denote column i of a matrix or element index i of a vector. A double subscript $\cdot_{i,j}$ denotes a row i and column j of a matrix.

3.2 Constrained rigid body simulation

Most modern constrained rigid body simulation engines, such as the *Open Dynamics Engine (ODE)* [Smith 2007] and *Bullet Physics* [Coumans 2014], solve the following linear system at each time step:

$$\begin{bmatrix} \mathbf{M} & -\mathbf{J}^T \\ \mathbf{J} & \Sigma \end{bmatrix} \begin{bmatrix} \mathbf{v} \\ \boldsymbol{\lambda} \end{bmatrix} = \begin{bmatrix} \mathbf{p} + h\mathbf{f}_{ext} \\ -\frac{1}{h}\boldsymbol{\Gamma}\boldsymbol{\phi} \end{bmatrix}. \tag{1}$$

The first line of the linear system in Eq. 1 is a velocity-level formulation of the Newton-Euler equations of motion where $\mathbf{M}$ is the mass and inertia matrix for all bodies in the system, $\mathbf{p}$ is the momentum, and $\mathbf{f}_{ext}$ are the external forces acting on the system (e.g., gravity). The second line are the constraint equations which couple the velocities $\mathbf{v}$ of bodies in the system. Here, $\boldsymbol{\phi}$ are the constraint equations or "gap" functions, $\mathbf{J}$ $\mathbf{J}$ is the constraint Jacobian matrix and $\boldsymbol{\lambda}$ are the Lagrange multipliers, which are the impulses that enforce the constraints.

Forming the Schur complement of the linear system in Eq. 1 gives the reduced system

$$(\Sigma + \mathbf{J}\mathbf{M}^{-1}\mathbf{J}^T)\boldsymbol{\lambda} = -\boldsymbol{\Gamma}\frac{\boldsymbol{\phi}}{h} - \mathbf{J}\mathbf{M}^{-1}(\mathbf{p} + h\mathbf{f}_{ext}), \tag{2}$$

which is the linear system used by the solvers in most rigid body physics engines. Here, Σ and $\boldsymbol{\Gamma}$ are diagonal matrices with positive coefficients. Σ adds *compliance* to the system and determines the "softness" of the constraints and is often referred to as the *constraint force mixing* (CFM) term. It is clear from Equation (2) that a diagonal Σ improves the numerical conditioning of the resulting linear system. $\boldsymbol{\Gamma}$ determines how much of the constraint error should be corrected in the next time step and it is often referred to as the *error reduction parameter* (ERP). Similar terms are found in most rigid body physics engines such as Bullet and ODE.

3.3 Spring-damper constraints

Together with constraint violations $\boldsymbol{\phi}$, the matrices Σ and $\boldsymbol{\Gamma}$ permit a Baumgarte stabilization of constraints in the system. We can therefore reinterpret constraint forces as arising from a spring-damper system [Erleben et al. 2005]. With this is mind, we compute the diagonal matrices of stiffness $\mathbf{K}$ and damping $\mathbf{B}$ coefficients for the system as

$$\mathbf{K}_{i,i} = \frac{\boldsymbol{\Gamma}_{i,i}}{h\Sigma_{i,i}}, \; \mathbf{B}_{i,i} = \frac{(1 - \boldsymbol{\Gamma}_{i,i})}{\boldsymbol{\Gamma}_{i,i}} \tag{3}$$

where matrix subscript $_{i,i}$ is the diagonal entry corresponding to the ith constraint. Note that for damping to remain positive and physically plausible, the values of the ERP must be in the range $0 < \mathbf{\Gamma}_{i,i} \leq 1$.

3.4 Potential energy

The potential energy stored in a constraint at the time of fracture can be computed using the constraint violation and the stiffness as

$$E = \frac{1}{2}\boldsymbol{\phi}_i^T \mathbf{K}_{i,i} \boldsymbol{\phi}_i.$$

Recalling that constraint impulses are generated as part of a spring-damper equation, the potential energy may alternatively be computed using the approximation $\boldsymbol{\lambda}_i \approx h\mathbf{K}_{i,i}\boldsymbol{\phi}_i$, such that

$$E = \frac{1}{2}h^{-2}\boldsymbol{\lambda}_i{}^T \mathbf{C}_{i,i} \boldsymbol{\lambda}_i,$$

where $\mathbf{C} = \mathbf{K}^{-1}$ is the compliance, or inverse of the stiffness matrix. This is trivial to compute since $\mathbf{K}$ is assumed to be diagonal and non-zero entries are computed according to Eq. 3. Also, note that in the above equation the scaling factor h^{-2} is used to recover the constraint forces from the impulses $\boldsymbol{\lambda}_i$, thus giving an energy computation that is independent of the simulation time step.

3.5 Fracturing

Fracturing occurs by disabling or removing a constraint whenever the impulses at a previous time step exceed a user-defined threshold ϵ, or

$$\boldsymbol{\lambda}_i > \epsilon.$$

Our objective is therefore to compute a plausible change for the velocity of bodies that were previously coupled by the constraint before fracture. In other words, to convert the stored potential energy into kinetic energy. The kinetic energy of the system can be written as $\frac{1}{2}\mathbf{v}^T\mathbf{M}\mathbf{v}$. We wish to introduce a change in velocity $\Delta\mathbf{v}$ to the current velocities $\mathbf{v}$ such that

$$\frac{1}{2}(\mathbf{v} + \Delta\mathbf{v})^T\mathbf{M}(\mathbf{v} + \Delta\mathbf{v}) - \frac{1}{2}\mathbf{v}^T\mathbf{M}\mathbf{v} = E.$$

Expanding the first term and simplifying gives

$$\frac{1}{2}\Delta\mathbf{v}^T\mathbf{M}\Delta\mathbf{v} + \mathbf{v}^T\mathbf{M}\Delta\mathbf{v} = E, \tag{4}$$

Note that since $\mathbf{p} = \mathbf{M}\mathbf{v}$, the second term on the left-hand side of Eq. 4 is simply a dot product between the current momentum and the change in velocity $\Delta\mathbf{v}$. Our preliminary experiments indicate that, in many cases, this term has a nominal effect. Furthermore, this term is dependent on the current velocities of bodies in the simulation, which clearly should not affect the response due to an elastic potential. Also, if velocities do become very large, this could mean our method is difficult to tune and may even introduce instabilities.

Rather, we found it sufficient to compute the change in velocity using the approximation

$$\frac{1}{2}\Delta\mathbf{v}^T\mathbf{M}\Delta\mathbf{v} = \alpha E, \tag{5}$$

where α is a user-determined value that, similar to a coefficient of restitution, determines the amount of elastic energy that is converted to kinetic energy. Physically valid values are in the range $[0, 1]$, but users may choose values larger than 1 for artistic effect. Also, rather than computing a change in the velocities of all bodies in the system, $\mathbf{v}$, we instead take a more pragmatic approach

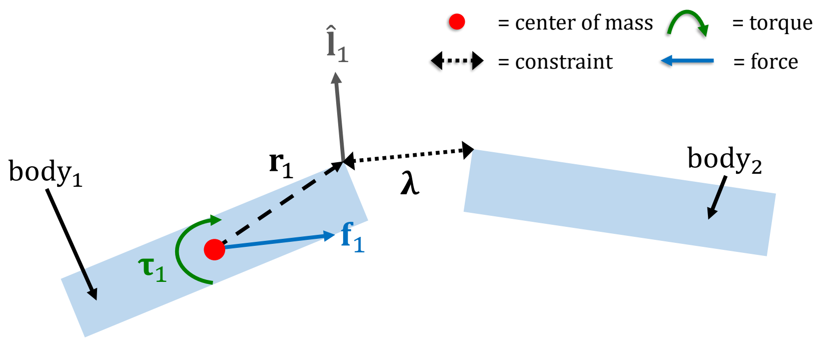

Fig. 2. A two-body system coupled by a single constraint. A free-body diagram for body_1 shows the forces and torques introduced by the constraint impulse $\boldsymbol{\lambda}$ which is applied at the attachment location. Similar forces and torques affect body_2 during the simulation.

and compute an impulse based on the mass of the participating bodies and the direction of the forces and torques applied by the constraint.

Although the change in velocity could be introduced over a short period of time (e.g. several time steps), we found it sufficient to compute an impulse μ that is applied over a single time step. We next discuss how to compute this impulse.

3.6 Computing impulses

We compute an impulse as a force to be applied at the attachment, or "pivot", point of the fractured constraint. This involves computing an impulse $\mathbf{G}^T\mu$, where matrix $\mathbf{G}$ encodes the direction of the impulse and μ is the magnitude. The change of velocity $\Delta\mathbf{v}$ due to an applied impulse is determined by the mass matrix, such that

$$\mathbf{M}\Delta\mathbf{v} = \mathbf{G}^T\mu. \tag{6}$$

Note that the right hand side essentially contains generalized forces applied to the bodies over a time step, h, which can be written as $\mathbf{G}^T\mu = h\begin{bmatrix}\mathbf{f}_1^T & \boldsymbol{\tau}_1^T & \dots & \mathbf{f}_m^T & \boldsymbol{\tau}_m^T\end{bmatrix}^T$. Here, $\mathbf{f}$ and $\boldsymbol{\tau}$ are the linear and angular forces (torques) respectively. Note that the Jacobian matrix $\mathbf{J}$ similarly maps between the constraint impulses $\boldsymbol{\lambda}$ and the generalized forces applied over a time step by $\mathbf{J}^T\boldsymbol{\lambda}$.

3.6.1 Direction of the impulse. We compute a direction based on the impulse, $\boldsymbol{\lambda}_i$, of the fracture constraint at the previous time step. Specifically, we choose a direction that is orthogonal to both the torque and linear forces applied by the pre-fracture constraint. We find that this not only produces lively motion, but produces motion in directions that correspond to the moment introduced by bending or stretching deformations, even if these deformations are very small.

Figure 2 shows a simple two-body system with an applied constraint impulse that we use in our analysis. Although the figure represents a 2D example, it is straightforward to extend the analysis to 3D. From the torque $\boldsymbol{\tau}_1$ and linear force $\mathbf{f}_1$ affecting the first body, a direction for the post-fracture impulse is simply computed as

$$\hat{\mathbf{l}}_1 = -\frac{\boldsymbol{\tau}_1 \times \mathbf{f}_1}{\|\boldsymbol{\tau}_1 \times \mathbf{f}_1\|},$$

which gives a normalized 3D vector. In the case that $\boldsymbol{\tau}_1$ is small, we simply compute a random direction orthogonal to $\mathbf{f}_1$. The process is similar for the second body.

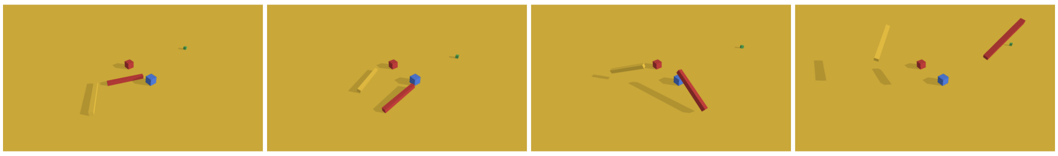
Fig. 3. A beam pinned between two posts as a block strikes at high velocity. Each frame shows the aftereffects of fracture for various values of α (left to right): α of 0, 0.5, 1, and 1.5. Larger values exaggerate the effect.

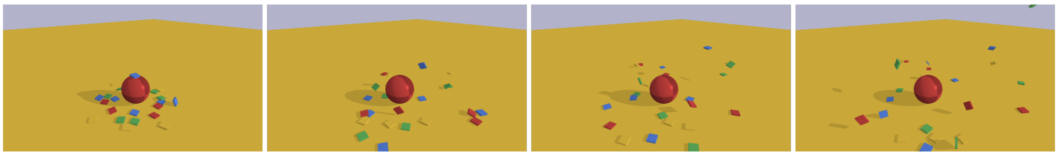
Fig. 4. A plate made up of 25 cubes lands on a sphere and breaks. Each frame shows the aftereffects of fracture for α values (left to right): 0, 0.5, 1, and 1.5. Larger values exaggerate the effect.

An impulse that is applied at the constraint attachment point in the direction $\hat{\mathbf{l}}_1$ produces a torque due to moment arm $\mathbf{r}_1$. This gives a direction $\mathbf{G} = \begin{bmatrix} 0 \dots \hat{\mathbf{l}}_1 & (\hat{\mathbf{l}}_1 \times \mathbf{r}_1) \dots 0 \end{bmatrix}$, where only entries pertaining to the first body are non-zero. From Eq. 6, we get$\Delta\mathbf{v} = \mathbf{M}^{-1}\mathbf{G}^T\mu$. Substituting into Eq. 5, we get the quadratic equation

$$\frac{1}{2}\mu^T\mathbf{G}\mathbf{M}^{-1}\mathbf{G}^T\mu = \alpha\,. \tag{7}$$

Since we only apply an impulse in a single direction, $\mathbf{G}$ is a row vector and the solution to Eq. 7 is simply

$$\mu = \sqrt{2\alpha\, m_G\, E}$$

where the effective mass “seen" by the impulse at the attachment point is

$$m_G = (\mathbf{G}\mathbf{M}^{-1}\mathbf{G}^T)^{-1}. \tag{8}$$

Note that m_G is a scalar quantity and may be computed efficiently since it only involves a single body in the system, which means that $\mathbf{M}$ in Eq. 8 is a 6x6 matrix with a special form. Also, in considering just a single body in the impulse calculation, we ignore coupling due to any remaining constraints in the system; we also ignore damping. Therefore, a change in velocity $\Delta\mathbf{v}$ is not guaranteed. However, these approximations do simplify the computation and improve the tractability of our method.

4 RESULTS

To demonstrate our approach, we present three didactic examples and one more practical example.

Beam. In the first example a beam is pre-scored into two pieces and is placed in front of two goal posts as a block is pushed through the goal. The collision leads to the removal of the constraints holding the pieces together and the beam fractures. The accompanying animation shows the default behavior as well as several animations with different values for our energy transfer parameter, α.

Thin Plate. In our second example a plate is pre-scored into a number of blocks and dropped on a sphere. The impact leads to fracture. Again we show results for different parameter values in the accompanying video.

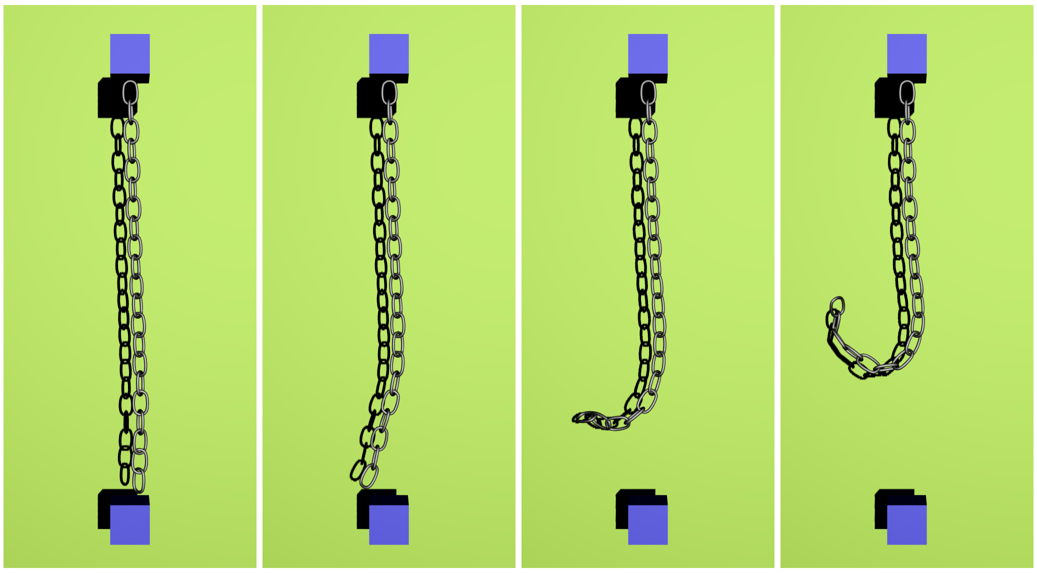

Fig. 5. A chain is attached at each end to a pair of blue cubes that are slowly driven apart until the chain breaks, creating a recoil motion. Each frame shows the aftereffects of fracture for α values (left to right): 0, 0.5, 1, and 2. Larger values exaggerate the effect.

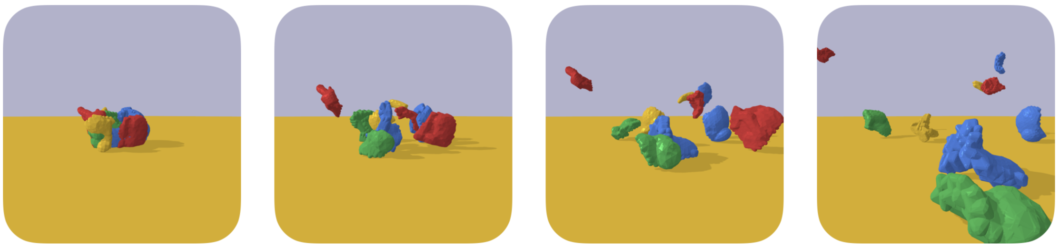

Fig. 6. A hollow bunny lands on the floor and breaks into 10 pieces. Left to right, the parameter values α used by our method are 0, 0.5, 1, and 2. Larger values exaggerate the effect.

Chain. Our final didactic example is of a chain breaking. In this example, there is no torque induced by the constraint forces so the direction of the applied impulse is chosen arbitrarily in the plane tangent to the constraint force. Figure 5 demonstrates that using our method results in the chain recoiling, which is a plausible response for this type of fracture.

Bunny. Our final example is of a hollow bunny falling on a sphere. To create this example, we began with a coarse volumetric tetrahedralization of the bunny and removed tetrahedra from the interior. To pre-score the mesh we randomly chose a number of seed tetrahedra and performed a round-robin flood fill from those seeds to create ten disjoint pieces. Constraints were placed at the vertices of faces incident to two pieces. The complexity of the example is roughly analogous to what one might expect in current video games.

The beam, plate, and bunny examples were performed with Bullet Physics, while the chain example was performed in Vortex, demonstrating that our approach can easily be incorporated into a variety of constrained rigid-body simulators.

Limitations. One limitation of our approach is that the effects are less perceivable as the stiffness of constraints is reduced. However, we note that in these cases the kinetic behavior at the time of fracture is typically dominated by error build in the other constraints. For instance, consider modeling an elastic rod as series of bodies and compliant constraints. As the rod stretches and fracturing eventually occurs, constraint impulses acting to reduce Φ will dominate the overall motion.

We also observed that impulses added to the system by our method sometimes caused undesired fracturing. In such cases an effective solution was to simply limit the rate at which constraints are removed. For instance, in the chain example, this rate is one constraint per ≈ 0.3 s of simulation time. This also prevents the entire chain from breaking at once, since tension in the chain is uniform across all constraints at the time of fracture, and therefore all constraints become candidates for fracture. A more realistic approach would be to remove a constraint, restore the configuration of bodies (positions, velocities, and forces) to the beginning of the time step, and re-run the simulation. However, this comes with an increased computational and storage overhead.

5 CONCLUSION AND FUTURE WORK

Our method could be extended to offer more control over the final motion of fractured bodies in order to create other effects. Although we choose an impulse direction based on the moment and linear force of $\mathbf{J}^T \boldsymbol{\lambda}$, other directions could be computed and impulses applied. For example, the original constraint direction could also be used. The user could then use a convex weighting scheme to distribute the elastic energy in each direction. For instance, this might allow the user to control the outward motion versus recoil in the chain example. One potential difficulty with using the constraint direction is that multiple constraints often work in the same direction, as with a taut chain where each the constraints on each link are all aligned. In this case, an impulse in the constraint direction would be immediately cancelled. Similarly, recent work has addressed instabilities for stiff multibody simulation by computing a damping term which is applied in the nullspace directions of the constraints [Andrews et al. 2017; Tournier et al. 2015]. These directions could also be used by our energized fracture method.

It would also be interesting to explore simulations where fracture is computed dynamically, i.e. the geometry is *not* pre-scored. Our general approach should be applicable in such scenarios driven by rigid body simulation, though one would need to estimate the energy that is lost during fracture. The approach could even be used to add non-physical energy to elastic simulations, which already transfer the stored elastic enery into kinetic energy.

Another interesting avenue for future work would be to couple our approach for rigid body fracture with the example-based rigid body plasticity of Jones and colleagues [2016]. The two methods are complementary and presumably would work well to achieve both brittle fracture and plastic deformation, however it may prove challenging to more closely couple the two approaches to achieve ductile fracture effects.

ACKNOWLEDGMENTS

This work was partially supported by the National Science Foundation under Grant No. IIS-1314896.

REFERENCES

[Andrews et al. 2017] Sheldon Andrews, Marek Teichmann, and Paul G. Kry. 2017. Geometric Stiffness for Real-time Constrained Multibody Dynamics. *Comput. Graph. Forum* 36, 2 (May 2017), 235–246. https://doi.org/10.1111/cgf.13122

[Bao et al. 2007] Zhaosheng Bao, Jeong-Mo Hong, Joseph Teran, and Ronald Fedkiw. 2007. Fracturing rigid materials. *IEEE Transactions on Visualization and Computer Graphics* 13, 2 (2007), 370–378.

[Clausen et al. 2013] Pascal Clausen, Martin Wicke, Jonathan R. Shewchuk, and James F. O'Brien. 2013. Simulating Liquids and Solid-liquid Interactions with Lagrangian Meshes. *ACM Trans. Graph.* 32, 2, Article 17 (April 2013), 15 pages. https://doi.org/10.1145/2451236.2451243

[Coumans 2014] Erwin Coumans. 2014. Bullet Physics Library. http://bulletphysics.org/.

[Erleben et al. 2005] Kenny Erleben, Jon Sporring, Knud Henriksen, and Kenrik Dohlman. 2005. *Physics-based Animation (Graphics Series).* Charles River Media, Inc., Rockland, MA, USA.

[Hahn and Wojtan 2016] David Hahn and Chris Wojtan. 2016. Fast Approximations for Boundary Element Based Brittle Fracture Simulation. *ACM Trans. Graph.* 35, 4, Article 104 (July 2016), 11 pages. https://doi.org/10.1145/2897824.2925902

[Jones et al. 2016] Ben Jones, Nils Thuerey, Tamar Shinar, and Adam W. Bargteil. 2016. Example-based Plastic Deformation of Rigid Bodies. *ACM Trans. Graph.* 35, 4, Article 34 (July 2016), 11 pages. https://doi.org/10.1145/2897824.2925979

[Müller et al. 2013] Matthias Müller, Nuttapong Chentanez, and Tae-Yong Kim. 2013. Real Time Dynamic Fracture with Volumetric Approximate Convex Decompositions. *ACM Trans. Graph.* 32, 4, Article 115 (July 2013), 10 pages. https://doi.org/10.1145/2461912.2461934

[Norton et al. 1991] Alan Norton, Greg Turk, Bob Bacon, John Gerth, and Paula Sweeney. 1991. Animation of fracture by physical modeling. *The visual computer* 7, 4 (1991), 210–219.

[O'Brien et al. 2002] James F. O'Brien, Adam W. Bargteil, and Jessica K. Hodgins. 2002. Graphical Modeling and Animation of Ductile Fracture. *ACM Trans. Graph.* 21, 3 (July 2002), 291–294. https://doi.org/10.1145/566654.566579

[O'Brien and Hodgins 1999] James F. O'Brien and Jessica K. Hodgins. 1999. Graphical Modeling and Animation of Brittle Fracture. In *Proceedings of the 26th Annual Conference on Computer Graphics and Interactive Techniques (SIGGRAPH '99).* ACM Press/Addison-Wesley Publishing Co., New York, NY, USA, 137–146. https://doi.org/10.1145/311535.311550

[Parker and O'Brien 2009] Eric G. Parker and James F. O'Brien. 2009. Real-time Deformation and Fracture in a Game Environment. In *Proceedings of the 2009 ACM SIGGRAPH/Eurographics Symposium on Computer Animation (SCA '09).* ACM, New York, NY, USA, 165–175. https://doi.org/10.1145/1599470.1599492

[Smith 2007] Russell Smith. 2007. Open Dynamics Engine. http://www.ode.org.

[Su et al. 2009] Jonathan Su, Craig Schroeder, and Ronald Fedkiw. 2009. Energy Stability and Fracture for Frame Rate Rigid Body Simulations. In *Proceedings of the 2009 ACM SIGGRAPH/Eurographics Symposium on Computer Animation (SCA '09).* ACM, New York, NY, USA, 155–164. https://doi.org/10.1145/1599470.1599491

[Terzopoulos and Fleischer 1988] Demetri Terzopoulos and Kurt Fleischer. 1988. Modeling Inelastic Deformation: Viscolelasticity, Plasticity, Fracture. *SIGGRAPH Comput. Graph.* 22, 4 (June 1988), 269–278. https://doi.org/10.1145/378456.378522

[Tournier et al. 2015] Maxime Tournier, Matthieu Nesme, Benjamin Gilles, and François Faure. 2015. Stable Constrained Dynamics. *ACM Trans. Graph.* 34, 4, Article 132 (July 2015), 10 pages. https://doi.org/10.1145/2766969

[Weinstein et al. 2008] Rachel Weinstein, Frank Petterson, and Brice Criswell. 2008. Destruction System. In *ACM SIGGRAPH 2008 Talks (SIGGRAPH '08).* ACM, New York, NY, USA, Article 71, 1 pages. https://doi.org/10.1145/1401032.1401123

[Wicke et al. 2010] Martin Wicke, Daniel Ritchie, Bryan M. Klingner, Sebastian Burke, Jonathan R. Shewchuk, and James F. O'Brien. 2010. Dynamic Local Remeshing for Elastoplastic Simulation. *ACM Trans. Graph.* 29, 4, Article 49 (July 2010), 11 pages. https://doi.org/10.1145/1778765.1778786

[Zafar et al. 2010] Nafees Bin Zafar, David Stephens, Mårten Larsson, Ryo Sakaguchi, Michael Clive, Ramprasad Sampath, Ken Museth, Dennis Blakey, Brian Gazdik, and Robby Thomas. 2010. Destroying LA for "2012". In *ACM SIGGRAPH 2010 Talks (SIGGRAPH '10).* ACM, New York, NY, USA, Article 25, 1 pages. https://doi.org/10.1145/1837026.1837059

[Zheng and James 2010] Changxi Zheng and Doug L. James. 2010. Rigid-body Fracture Sound with Precomputed Soundbanks. *ACM Trans. Graph.* 29, 4, Article 69 (July 2010), 13 pages. https://doi.org/10.1145/1778765.1778806

[Zhu et al. 2015] Yufeng Zhu, Robert Bridson, and Chen Greif. 2015. Simulating Rigid Body Fracture with Surface Meshes. *ACM Trans. Graph.* 34, 4, Article 150 (July 2015), 11 pages. https://doi.org/10.1145/2766942