Realistic soft-body tearing under 10ms in VR

Manos Kamarianakis
kamarianakis@uoc.gr
FORTH - ICS, University of Crete, ORamaVR
Greece

Antonis Protopsaltis
aprotopsaltis@uowm.gr
University of Western Macedonia, ORamaVR
Greece

Michail Tamiolakis
michalis.tamiolakis@oramavr.com
University of Crete, ORamaVR
Greece

George Papagiannakis
papagian@ics.forth.gr
FORTH - ICS, University of Crete, ORamaVR
Greece

Figure 1: (Left) The vertices of a 3D model are clustered in particles, to allow soft-body characteristics. (Middle) A continuous tear is performed on the model. (Right) The particles of the torn model are updated, allowing further proper soft-body deformations.

KEYWORDS

Real-Time, Tear, Cut, Soft-Bodies, Virtual Reality

1 INTRODUCTION

Rigged animated models are one of the most researched areas in computer graphics and have been vastly adopted in Virtual Reality (VR) applications. VR experts experiment with various animation and deformation techniques that can yield realistic real-time outputs. To cover the needs that arise from a variety of use cases, our research revolves around the ability to perform realistic tears, i.e., small cuts, on the surface of a model. Current bibliography [1, 7] describes diverse ways on how to cut a 3D model, but most of these methods are not suitable for VR, since the specific calculations must be performed in a real-time manner within a few ms to preserve user immersion. Furthermore, to avoid the uncanny valley in VR, we emulate realistic effects while performing cuts on certain materials, such as a sponge or human tissues, that one would expect to occur in real life.

Latest developments [2] allow for basic operations, such as cutting, tearing or drilling on a rigged mesh model, to be run in near real-time. Furthermore, the replication of the physical behaviour of soft bodies, when applying external forces to them in VR, would greatly increase the overall realism of the simulation [3]. The ongoing research for increased realism in virtual environments heavily impacts educational-oriented applications, especially the ones regarding VR medical training [5].

2 OUR APPROACH

Overview. Our approach is based on the techniques of [2], where the authors describe simple cut, tear and drill operations on a 3D mesh using basic geometric operations. Our optimized tearing module allows for continuous uninterrupted operation, i.e., the user can freely perform tears successively, similar to a surgeon's tearing gesture. Furthermore, we have developed a suitable particle decomposition on the model's vertices that can be used to emulate soft bodies, as in [4]. Via suitable optimizations, we are able to perform real-time continuous tears on a soft-body model and update the underlying particle decomposition to obtain high-realistic results in VR. Our methods were designed with the lowest possible complexity in terms of needed calculations that yield real-time results even in untethered Head-Mounted Displays (HMDs), with limited GPU and CPU capabilities. Lastly, proper handling and weight assignment [2] to the tear-generated vertices allow us to tear not only rigid but also skinned models, where in the latter case, further animation is still feasible.

Methodology for the Tear algorithm In order to achieve real-time tearing results, we have opted for basic geometric primitives, e.g., face-plane intersections and face ray casting, as basic building blocks for our algorithms. This approach allows for fast identification of the faces affected by the tear.

In our implementation, the tear width is user defined. In non-zero settings, a destructive tear takes place: triangles that fall in the tear-gap are completely or partially *clipped*, i.e., removed from the model. Partially clipped faces are calculated by their intersections

Figure 2: Our methods applied on a cactus rigged model. Using particles we can simulate soft-body simulations, as shown in the cactus holding the ball. The skinned model can be torn and redeformed as shown on the right cactus. The model's 2976 vertices were clustered into 110 particles, using $d = 0.1$.

with the tear-gap surrounding box which initially is considered as a finite part of a plane, defined and bounded by the intersections of the scalpel's initial and final positions with the model.

As the user moves the scalpel, freely tearing the model, we sample the scalpel's positions at specific time or distance intervals and perform multiple consecutive tears. To avoid jagged edges on the tearing path we make sure that consecutive bounding boxes do not overlap by utilizing non axis-aligned boxes instead.

Methodology for the Particle Decomposition To accomplish the so-called soft-body mesh deformation [6], the vertices of the mesh are clustered into groups, called *particles*. Each particle is positioned at the centroid of each group of vertices. A vertex may belong to multiple particles. In our method, every particle contains vertices within a model dependent, euclidean range of d units. A small range results in more particles, hence more accurate deformations, but yields worse running times. Since a high range will have the opposite effects, a balanced range should be identified per model (see Figure 2).

When a user applies a force to a particle, its position changes and this movement affects the position of all vertices of the particle. The particle's velocity is changed, proportionally to the displacement, always pointing to its initial position (simulating elasticity); in the case of skinned models, the current pose is also considered. Physics calculations are natively handled by the employed game engine.

After a tearing operation, the clustering map is updated by adding or removing vertices to the involved particles. To allow fast updates and produce physically correct deformation results, simple directives were also introduced, e.g., vertices belonging to opposite sides of a tear, although close enough, cannot belong to the same particle.

Results Our optimized methods run on top of UNITY3D game engine and are incorporated within the MAGES SDK [6], developed by ORamaVR, publicly available for free.

Applying our methods on the Stanford bunny (3365 vertices), the tearing times were varying between 7-16ms depending on the region torn. The average time for tears as the ones shown in Figure 1 was 10ms. The results were obtained without employing GPU or parallel processing, using an Intel core i7 7700HQ at 2.8GHZ with an Nvidia GTX 1050ti m (8GB RAM) graphics card and a Desktop VR - HTC Vive. The model's vertices were clustered into 78 particles, using range $d = 0.1$, by an offline process that took 456ms.

Comparison with other methods Methods accomplishing similar results are implemented in the discontinued Nvidia Flex physics engine (developer.nvidia.com/flex) and the Flex-based Unity3D plugin uFlex that has not been updated in the past 6 years (still in beta). These methods do not support rigged models in their particle decomposition. They support cloth tearing but not surgical-like tearings as the ones described in our work. In conclusion, our method remains the only active, game-engine compatible, solution.

3 CONCLUSIONS

We have presented a novel integration of a real-time continuous tearing algorithm for 3D meshes in VR along with a suitable particle decomposition that allows soft-body deformations on both the original and the torn model. Our methods are based on simple geometric primitives and therefore are suitable even for untethered HMDs of low specifications. The proposed techniques are already implemented in the MAGES SDK, running on Unity3D, publicly available for free.

ACKNOWLEDGMENTS

REFERENCES

[1] Cynthia D Bruyns, Steven Senger, Anil Menon, Kevin Montgomery, Simon Wildermuth, and Richard Boyle. 2002. A survey of interactive mesh-cutting techniques and a new method for implementing generalized interactive mesh cutting using virtual tools. *The journal of visualization and computer animation* 13, 1 (2002), 21–42.

[2] Manos Kamarianakis and George Papagiannakis. 2021. An All-in-One Geometric Algorithm for Cutting, Tearing, and Drilling Deformable Models. *Advances in Applied Clifford Algebras* 31 (Jul 2021), 58.

[3] Miles Macklin, Matthias Müller, Nuttapong Chentanez, and Tae-Yong Kim. 2014. Unified Particle Physics for Real-Time Applications. *ACM Transactions on Graphics (TOG)* 33, 4 (2014), 104.

[4] Andrew Nealen, Matthias Müller, Richard Keiser, Eddy Boxerman, and Mark Carlson. 2006. Physically based deformable models in computer graphics. In *Computer graphics forum*, Vol. 25. Wiley Online Library, 809–836. Issue 4.

[5] George Papagiannakis, Manos Kamarianakis, Thomas C. Sauter, Alan Chalmers, Joan Lasenby, Daniele Di Lernia, and Walter Greenleaf. 2022. Editorial: New Virtual Reality and Spatial Computing Applications to Empower, Upskill and Reskill Medical Professionals in a Post-Pandemic Era. *Frontiers in Virtual Reality* 3 (2022).

[6] George Papagiannakis, Paul Zikas, Nick Lydatakis, Steve Kateros, Mike Kentros, Efstratios Geronikolakis, Manos Kamarianakis, Ioanna Kartsonaki, and Giannis Evangelou. 2020. MAGES 3.0: Tying the Knot of Medical VR. In *ACM SIGGRAPH 2020 Immersive Pavilion* (Virtual Event, USA) *(SIGGRAPH '20)*. Association for Computing Machinery, New York, NY, USA, Article 6, 2 pages.

[7] Jun Wu, Christian Dick, and Rüdiger Westermann. 2013. Efficient collision detection for composite finite element simulation of cuts in deformable bodies. *The Visual Computer* 29, 6 (June 2013), 739–749.