Real-Time Deformable Soft-Body Simulation using Distributed Mass-Spring Approximations

Ben Kenwright
School of Computing Science
Newcastle University
Newcastle, United Kingdom
b.kenwright@ncl.ac.uk

Rich Davison
School of Computing Science
Newcastle University
Newcastle, United Kingdom
richard.davison4@ncl.ac.uk

Graham Morgan
School of Computing Science
Newcastle University
Newcastle, United Kingdom
graham.morgan@ncl.ac.uk

***Abstract*—This paper investigates several methodologies for simulating soft-body objects using a mass-spring approach. The mechanisms are then expanded to include deformation information that can produce results suitable for use in real-time applications where *visual impact rather than accuracy is desired,* such as video games. Many methods use complex and esoteric methods to achieve physically accurate simulations; we target the mass-spring model because of its simplicity, using creative modifications for diverse visual outcomes.**

Keywords-soft-bodies; physics; deformation; mass-spring; games; voxelation.

I. INTRODUCTION

With the increase in computational power, movies and games have started to take advantage of simulated visual effects. Such effects include soft-body physics and deformation artifacts, which can be used to create more realistic skin/face movement in character simulations; deformable objects can bend and move when forces are applied. For example a metal bar bending when under stress, with enough stress eventually causing a permanent deformation of its shape. For games and the movie industry, visual effects are more important than accuracy, this means more novel methods come about that use eccentric models to create more visually pleasing results. One such novel method is to use springs to represent material rigidity, and use various topologies and tricks that produce close to the real thing. We explore some of these approximations to demonstrate various novel methods for simulating soft-body objects using simple mass-spring approximation.

The main contribution of this paper is the proposal of several models and strategies for use in soft-body simulations. We introduce various approximation techniques and their applicability to real-time applications such as games.

These methods are extended to demonstrate permanent deformation effects by taking advantage of the stress information stored in the mass-spring model.

Our method approaches the problem by keeping the simulations fast and uncomplicated by utilizing a mass-spring system in combination with various modeling approaches. As visual impact is the primary concern of the paper, the models are judged by whether they produce plausible results at interactive frame rates, rather than their ability to generate results that are physically accurate.

The remainder of the paper is organized as follows. Section II presents previous work done in the area, and introduces our integration scheme and mass-spring model. Section III describes our soft-body model approximations. Section IV explains our deformation method. Section V discusses preliminary results. Last, Section VI presents conclusions and future work.

II. BACKGROUND AND RELATED WORK

The mass-spring system is one of the simplest physically-based models developed over the past decade [1][2], its simplicity and ease of implementation making it the most likely candidate to achieve real-time performance. Prior work has been done which models soft-body simulations using the object's pressure [3], and implicit integrators with a mass-springs model [4]. A deformable soft-body is approximated by sets of masses linked together using springs in various configurations. The mass-spring is highly parallelizable, easy to implement and involves few computations.

There are various approaches to creating soft-bodies. Elasticity and viscoelasticty models have been shown to be successful [5,6], but are not suitable for real-time applications due to their computational complexity. Using a global restraining method to create soft-bodies [7,8] allows a simulation to run at more interactive rates but demonstrates less realism.

Other methods include a finite-element approach. The approach decomposes the model into separate pieces. Work by [12,13,14,15] used tetrahedral elements, while [16] employed a hexahedral composition. Bro-Nielsen [9] takes advantage of linear elasticity to achieve real-time deformable structures. A more recent and novel method has been to use neuro-animators [10], which have a learning period, after which they can emulate coupled physical systems, including cloth.

Our method is different by using an extremely low computational model that may not produce physically accurate results, but more artistic visual outcomes.

 ISBN: 978-1-61208-157-1

A. Integration Scheme

The methods explored in this paper use a semi-explicit Euler method (also called symplectic Euler):

$$v_i^{n+1} = v_i^n + F_i^n \frac{dt}{m} \quad (1)$$

$$x_i^{n+1} = x_i^n + v_i^{n+1} dt \quad (2)$$

where v is velocity, x is position, n is the current frame, F is force, dt is the timestep, and m is mass.

This method of integration is computationally cheaper than implicit integration, but suffers from stability issues with large forces, where the time step squared must be inversely proportional to the stiffness. This is not such a problem with off-line computation, but is an issue when used in real-time applications since very small time steps are required to ensure stability. As a way to overcome this problem and ensure that our simulations remain robust and reliable, we clamp the maximum forces and allow for small visual artifacts to keep at real-time frame rates.

We also extend this checking of the forces at limits to introduce deformation artifacts.

B. Mass-Spring Model

Springs are not a perfect physical model for soft-body objects, but provide a good visual approximation.

The mass-spring system relies on the principle of Hooke's law, which states that the force applied by a spring is proportional to the load it is under.

Linear: $$F_{spring} = -k_s X_{diff} + k_d v \quad (3)$$

Square: $$F_{spring} = -k_s X_{diff}^2 + k_d v \quad (4)$$

where F is force, k_s is the spring constant, k_d is the spring damping constant, X_{diff} is the current displacement distance, and v is the velocity of the mass-spring.

Hooke's law is a useful approximation of how a real-life spring acts when compressed or expanded by external forces – it always tries to return to its basal 'restitution' length, with a greater force the further from this restitution length the spring becomes. The soft-body approximation methods described here use mass-springs between the vertices of the model, allowing flexible movement that reacts realistically to forces imparted upon it. However, it is not enough to simply recreate the model mesh using springs – the spring model is likely to collapse in on itself and become unstable under minor forces. Section III investigates methods for solving this while still allowing for real-time soft-body approximation.

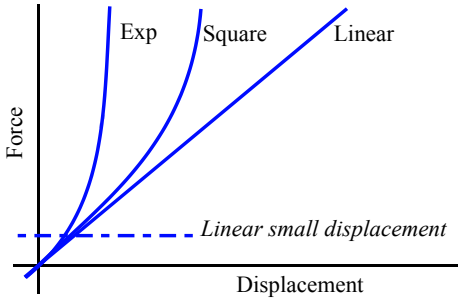

Figure 1. Ideal spring model.

Using a squared displacement-correcting spring gives us a more rigid body that still shows good flexibility for small changes, such as simulating a skin surface, or any small surface ripples while reducing the overall bendiness of the shape. Mass values chosen initially where equal and the sum of the total mass of the object. This was arbitrary, but we explored variations of the mass selection, such as making masses at the centre different, or making the mass on one edge to be larger so that object appeared to self-balance, as shown in Figure 3. We can expand on the idea of varying the mass-spring properties to create alternative effects. Other spring modifications would include different spring coefficients for expansion and contraction to produce more outlandish artistic visual effects.

III. METHODS

This section introduces various soft-body models and their applications.

A. Brute Force Method

This method is usually the first and most intuitive way of creating a soft-body object from a mesh, where spring constraints connect every vertex to every other vertex to form a rigid structure as shown in Figure 3. This produces a sufficiently stable model for soft-body simulation, but has a high cost of (n*n-1)/2 spring constraints per object.

B. Mass-Spring Voxelation Method

Splitting a model into an array of voxels is useful for the

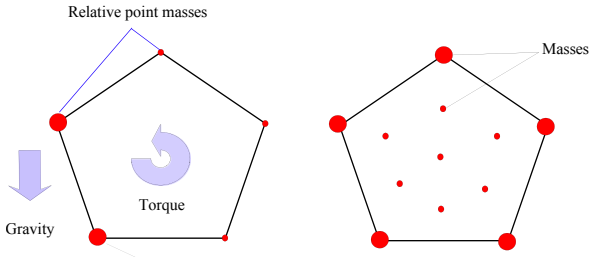

Figure 2. *Left*: Biasing the weight on one side of an object to make it balance. *Right*: Mass variation, using larger masses for the surface and smaller masses for the volume.

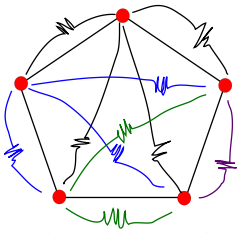

Figure 3. Brute force interconnection of surface vertices using mass-springs.

simulation of soft-bodies on highly tessellated shapes where using other methods would result in an inordinate number of springs. An energy efficient adaptive space deformation version was demonstrated by [11]. Our model utilizes voxelation for soft-body simulation by decomposing the shape area into voxels, then assigning each vertex to its surrounding voxels. A weighting factor determining how much each voxels change in orientation-position influences vertex movement – similar to the blend weights used in skeletal animation systems.

Figure 4 shows how these voxels are connected via mass-springs; when the voxelated approximation is deformed, the vertices of the original model are moved according to their attached voxel weightings.

This method provides advantages in performance over the brute-force method when used with high poly meshes, since the number of springs is determined by the voxel resolution and not the number of vertices. It is less suitable for low-poly models, where the underlying voxels will be able to form shapes the model vertices cannot accurately reproduced. This can be countered by tessellating the model, increasing the number of vertices and allowing greater freedom of movement.

C. Uniform Grid Mass-Spring Distribution

Another method of combining model rigidity with soft-body dynamics is using a uniform grid. Via this grid, mass is equally distributed across the shape, with extra vertices added to the model at points where the grid and model intersect. Neighboring vertices are then joined using mass-springs to form a rigid model.

This method offers the advantage of having a mass-spring distribution throughout the object, allowing a closer approximation to real soft-body physics. The point-mass distribution is uniformly through-out compared with the voxel method previously which uses a random scattering approach.

D. Internal / external shape scaffold

When creating surface springs, the model lacks rigidity to keep its original shape, due to it having multiple positions where its springs are still the same length, or lack enough strength to keep its original shape. Using a shape with a reduced number of vertices to attach the surface vertices to increase rigidity, either having the virtual rigidity model inside our outside the original.

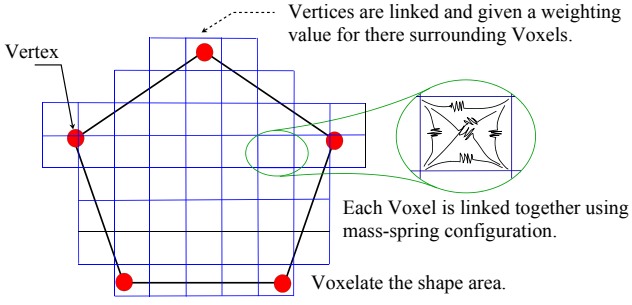

Figure 4. Mass-Spring Voxelation Method.

E. Shell Method

Complex and concave shapes, such as tubes give problems where additional mass-springs are added to increase rigidity. This additional rigidity causes the model to bend and twist un-naturally. Using a duplicated virtual surface shell mesh with springs, and attached to the original surface points, gives us a more rigid surface which mimics the behavior we would expect for more complex shapes. One such example of where this method gives good results is the crushing of a tube pipe.

This method extends the uniform approach, by only includes voxels intersecting the surface or within the shape.

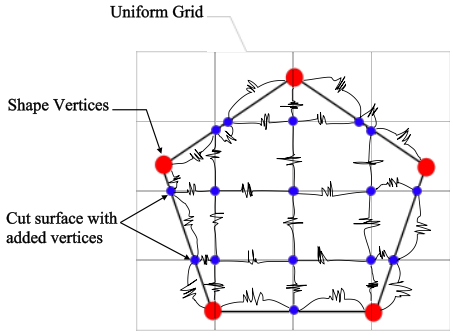

Figure 5. Uniform mass-spring grid design.

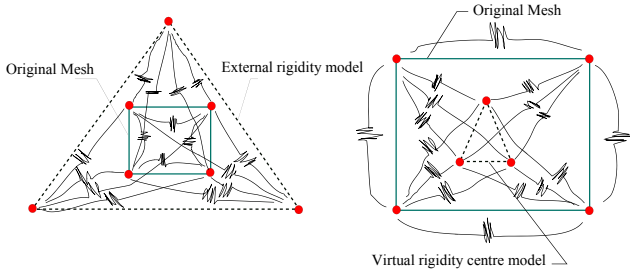

Figure 6. Using a triangle as an inner and outer skeleton for rigidity.

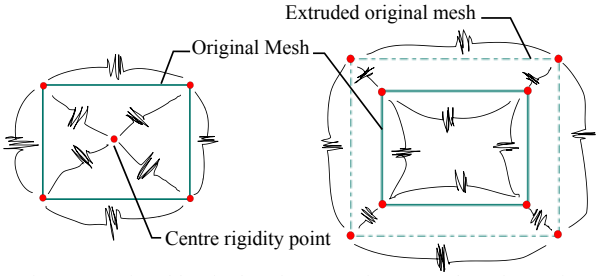

Figure 7. *Left*: Add a single reference point to attach surface points to help the soft-body hold its shape. *Right*: Using an extruded or intruded shell surface to introduce rigidity to the soft-body model

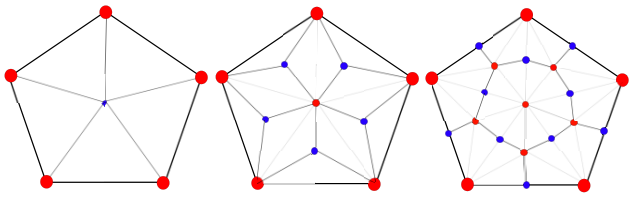

Figure 8: An example of a shape being recursively split into subsections

F. Tetrahedron Mass-Spring Distribution

This method recursively subdivides the shape into subsections - tetrahedron sections in 3D, or triangles in 2D. At each level we subdivide the largest subsections into smaller subsections until a minimum size volume is reached. Because simple shapes are used for each subsection, the mass and volume can be calculated accurately. This can then be used to scale the mass-spring constraints to give a more uniform force distribution across the body. Each subsection is then split into mass-springs along its edges.

G. Voronoi Regions and Delaunay Triangluation

The shape is partitioned using Delaunay triangulation to get a uniform random distribution of springs to help give the body rigidity.

IV. DEFORMATIONS

As materials are stretched, they reach an *elastic limit*, beyond which they will become permanently deformed, and Hooke's law will no longer accurately approximate its elastic properties.

We extended our simulation to accommodate this elastic limit, so that once a mass-spring reached a specific force threshold its stiffness and restitution length is recalculated, permanently deforming the model.

V. RESULTS

The results do not physically represent how a soft-body would squish and stretch in the real world, but that was not the aim of our work. Instead we have focused on the visual results for real-time, interactive applications such as games. We have shown diverse techniques for creating approximate soft-body simulations with extension of the underlying principles to handle deformation by taking advantage of the mass-spring stress information between frames.

For comparison, Table 1 gives a brief summary outlining the different methods.

Initial methods have been simulated in 2D to test their viability and assist in quick prototyping, while the principles can easily be extended into 3D. The 2D methods gave us simple building blocks which helped us take the designs in more outlandish directions.

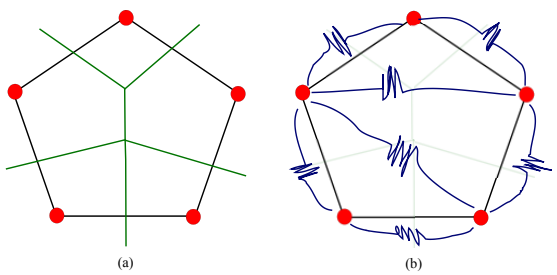

Figure 9. (a) Voronoi Regions, (b) Delaunay Triangle are used to determine where springs are placed between masses.

VI. CONCLUSION AND FUTURE WORK

The results show flexible methods to simulate soft bodies. We used the basic Euler mass-spring model for the simulations to get early results and see how they compare. We would hope to test out more reliable integration methods such as Verlet. The Early work has been with 2D shapes but hope to extend it to 3D. We would also like to vary the mass distribution throughout the models, varying the mass properties at different points and see how the results compare to realistic models. It would be interesting to enhance the simulation to display force and energy distribution throughout the objects. Additionally, the easily parallelizable nature of mass-springs makes these methods ideal candidates for GPGPU processing.

VII. REFERENCES

[1] J.E. Chadwick, D.R. Haumann, and R.E. Parent, "Layered construction for deformable animated characters," ACM SIGGRAPH Computer Graphics, vol. 23, Jul. 1989, pp. 243-252.

[2] G. Miller, "The motion dynamics of snakes and worms," Computer Graphics, Volume 22, Number 4, August 1988, vol. 22, 1988, pp. 169-178.

[3] M. Matyka and M. Ollila, "Pressure model of soft body simulation," SIGRAD2003, November, 2003, p. 20–21.

[4] J. Mesit, R. Guha, and S. Chaudhry, "3D Soft Body Simulation Using Mass-spring System with Internal Pressure Force and Simplified Implicit Integration," Journal of Computers, vol. 2, Oct. 2007, pp. 34-43.

[5] D.T. and A. Witkin, "Physically based model with rigid and deformable components," 1988, p. IEEE Computer Graphics and; Applications; pages 41.

[6] D. Terzopoulos, J. Platt, A. Barr, and K. Fleischer, "Elastically deformable models," ACM SIGGRAPH Computer Graphics, vol. 21, Aug. 1987, pp. 205-214.

[7] A. Pentland and J. Williams, "Good vibrations: Modal dynamics for graphics and animation," Computer, vol. 23, 1989.

[8] A. Witkin, "Fast Animation and Control of Nonrigid Structures."

[9] M. Bro-Nielsen and S. Cotin, "Real-time Volumetric Deformable Models for Surgery Simulation using Finite Elements and Condensation," Computer Graphics Forum, vol. 15, Aug. 1996, pp. 57-66.

[10] R. Grzeszczuk, "NeuroAnimator : Fast Neural Network Emulation and Control of Physics-Based Models NeuroAnimator : Fast Neural Network Emulation and Control of Physics-Based Models," Science, 1998.

[11] M. Botsch, M. Pauly, M. Wicke, and M. Gross, "Adaptive Space Deformations Based on Rigid Cells," Computer Graphics Forum, vol. 26, Sep. 2007, pp. 339-347

[12] E.G. Parker and J.F. O'Brien, "Real-time deformation and fracture in a game environment," Proceedings of the 2009 ACM SIGGRAPH/Eurographics Symposium on Computer Animation - SCA '09, 2009, p. 165-175.

[13] G. Irving, J. Teran, and R. Fedkiw, "Invertible finite elements for robust simulation of large deformation," Proceedings of the 2004 ACM SIGGRAPH/Eurographics symposium on Computer animation - SCA '04, 2004, p. 131-140.

[14] M. Müller, J. Dorsey, L. McMillan, R. Jagnow, and B. Cutler, "Stable real-time deformations," Proceedings of the 2002 ACM SIGGRAPH/Eurographics symposium on Computer animation - SCA '02, 2002, p. 49-54.

[15] A.W. Bargteil, C. Wojtan, J.K. Hodgins, and G. Turk, "A Finite Element Method for Animating Large Viscoplastic Flow," vol. 26, 2007, pp. 291-294.

[16] V. Baudet, M. Beuve, F. Jaillet, B. Shariat, and F. Zara, "Integrating Tensile Parameters in Hexahedral Mass-Spring System for Simulation," WSCG'2009, 2009

 ISBN: 978-1-61208-157-1

A. Brute Force	Every vertex (point-mass), link all-to-all
B. Voxelation	Voxels (or bricks) form a weighted skeleton
C. Uniform Grid	Uniform grid interior, surface sliced and joined
D. Internal/External Scaffolding	Skin linked to exterior/interior reduced poly shape
E. Shell Method	Shells (or layers), extruded surface normal, linked together to form a rigid surface structure
F. Tetrahedron	Partition into Tetrahedrons
G. Voronoi Regions/ Delaunay Triangulation	Partition using Voronoi Regions and Delaunay Triangulation

Table 1. Comparing the various methods.

 ISBN: 978-1-61208-157-1