Plausible Chainmail Rendering

Kevin Mackenzie
Glenn Smith

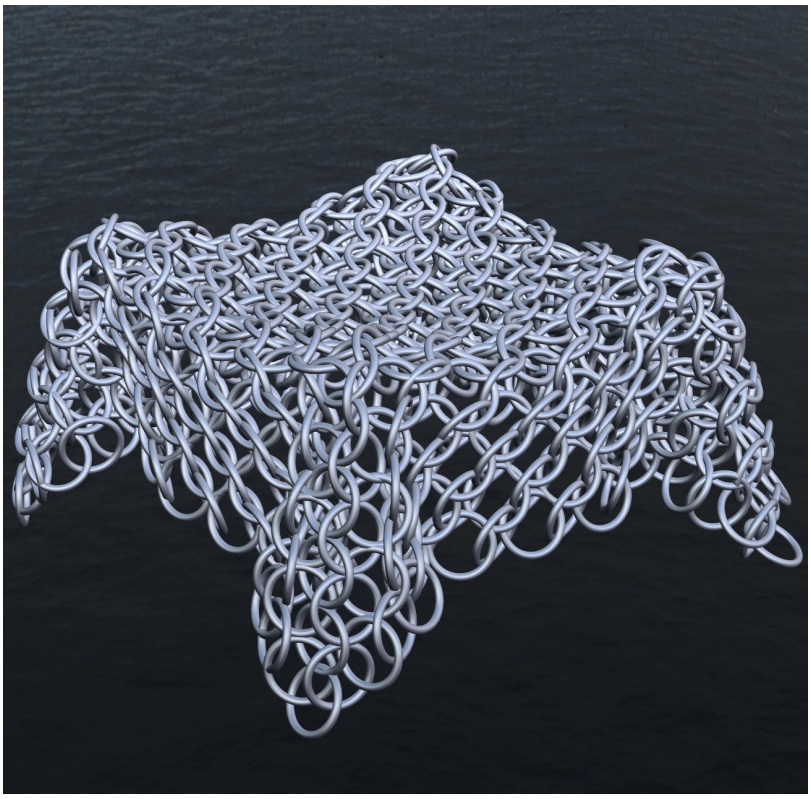

Figure 1: Table Cloth scene

Figure 2: Golden Curtain scene

ABSTRACT

We present a method to simulate interlocking-ring fabrics, commonly referred to as chainmail, in real-time as well as a method to plausibly render them. These materials exhibit unique properties that other fabrics do not adhere to and the simulation and rendering of such properties has been little studied in the field of computer graphics. Our work builds a practical and theoretical foundation for the necessary properties for visually satisfying renderings as well as a discussion on the necessary changes to achieve more physically correct results.

1 INTRODUCTION

Chainmail is a fabric consisting of many interlocked rings. Usually arranged in a grid-like pattern like in Figure 3, the rings connect to form a mesh that behaves like cloth but exhibits very little stretching and can compress considerably further. For our simulation, we wanted to focus on implementing a plausible, real-time approximation of how chainmail behaves. We used real-world examples of chainmail, such as Figure 4, to model our formulae to the real-world behavior we sought to imitate. The key features we planned to accomplish were:

- Cloth-like behavior with compressing and resisting stretching
- Rings orienting to interlock correctly
- Reflective texturing on the rings
- Real-time simulation and rendering performance

2 RELATED WORK

2.1 Mass-Spring Model

We take most of the inspiration for our work from work in simulating traditional cloth, namely Provot's work [Provot 1995] in adding deformation constraints to mass-spring models. Since rigid fabrics require very small time steps to produce a stable simulation due to the forces of high spring constants, a better way to achieve satisfying results for rigid cloths was necessary. Adding upper bounds for the distance of the structural and shear springs can strech constrains deformation and prevented unrealistic "super-elasticity" that

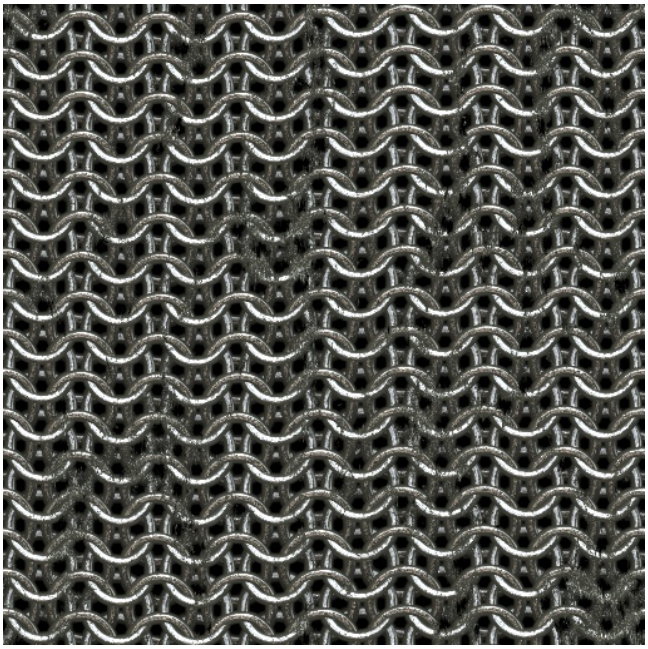

Figure 3: Chainmail - Variation 6. Texture by Constantin Malkov [CC-BY 2.0], via Filter Forge

Figure 4: Chainmail Tunic on a Mannequin. Photograph by Housing Works Thrift Shops [CC-BY-SA 2.0], via Flickr

smaller spring constants entail. Since chainmail exhibits very rigid behavior, Provot's work is well-suited for our needs.

Another important issue to consider when simulating fabrics is handling and preventing self-intersection. Provot [Provot 1997] also did work for preventing self-collision between thin sheets and arbitrary surfaces, including self-collision, by detecting collisions between points in the mesh and all triangles to handle the case where a point in the cloth is pulled through an arbitrary surface, and between edges in the cloth and arbitrary edges to handle the case of the edge of the cloth being pulled through the edge of another surface even if a point does not protrude. Due the our material's significant thickness, this strategy is not necessary. From this point forth, when Provot is mentioned by name, we are referring to his work in [Provot 1995].

Bridson [Bridson et al. 2002] also did work in handling cloth self-intersection and friction for use in animation. We take inspiration for their method for preventing self-intersections efficiently, but do not implement their specific strategy. Their method, which involves computing inelastic collisions between elements of the cloth system with each other, could be be adapted to improve the self-interaction behavior of our model.

2.2 Current Methods

There appears to be a few different methods for handling chainmail in current applications. One is to to use methods for simulating normal cloth in whatever capacity the the application requires and apply a texture to the resulting mesh. This can result in obvious artifacts like stretching and warping of the individual rings as the texture is interpolated across each primitive.

Another method, which can be used for non-real-time applications, is to do a rigid body simulation on each ring of the mesh, which has all of the drawbacks of rigid body simulations, such as unstable rest contact states. However, this does result in the most physically accurate results at an appropriate time scale.

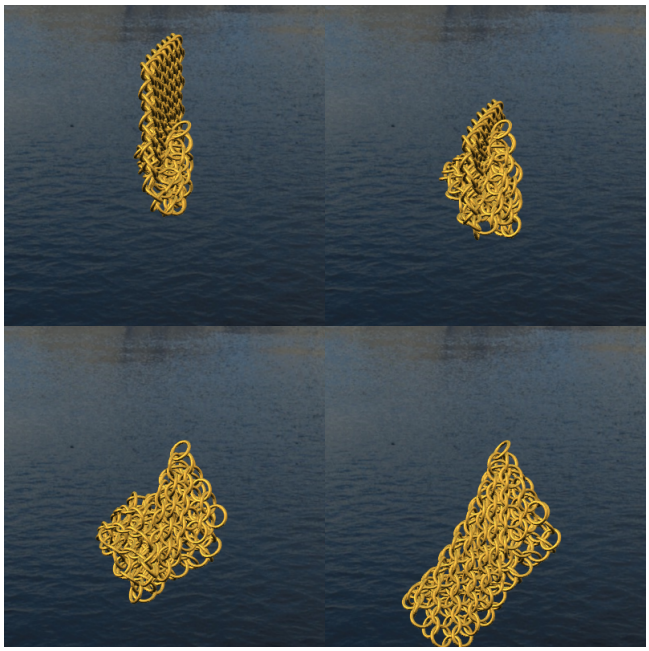

Figure 5: Several frames of chain falling over itself and avoiding apparent self-intersection

3 CLOTH

First, we must define our model for the rings in the material. We assume each ring is a perfect torus and all rings in the mesh are of uniform size. The torus has a tube radius of a and the distance

from the center of the torus to the center of the tube we call the radius c. We assume that $a << c$, otherwise the torus would not have a hole in the center.

The basis for our cloth simulation is a mass-spring system. Each mass in the cloth represents a ring in the chainmail and each "spring" represents a connection between two rings. Instead of the traditional structural, shear, and flexion springs, we use a simplified model that omits the shear and flexion springs since they have limited relevance for our material. Due to the macroscopic units of chainmail, some of the methods used to handle the continuous nature of cloth over a discrete mesh grid are unnecessary.

At first thought, one may think it would be suitable to have spring forces for the structural node relationships in the mesh, but traditional linear springs do not accurately model the relationship between rings in the mesh since compressing the material yields no resistance. A reasonable approximation for the springs that each ring represents would be a piecewise function that applies zero force unless over or under extended. Since metal rings have a small amount of elastic deformation, this spring constant is effectively infinite, which is not well-suited for numerical integration. Because of this, we omit all cloth spring forces and rely fully on Provot's edge constraint method to keep the material together. Springs also give the material an unnaturally bouncy appearance.

Since we omit spring forces, there is the additional problem of over-compressing these edges. Since the material has significant thickness, we must ensure that it buckles instead of falling into itself completely. While the exact nature of how close each node can get to its neighbor depends on a variety of factors, we choose to allow it to get as close as the thickness of the ring and we will discuss how we visually handle this in Section 4. We make the assumption that the cloth will not be under simultaneous compression from both axes of the node grid, which is reasonable since the only relevant compression force is from gravity, so this strategy is still plausible.

One drawback of omitting spring forces and exclusively using constrained edge lengths is that we have no internal forces to oppose gravitational forces on each node, so, despite appearing correct, the velocities actually become unbounded if not correctly updated. One solution is to increment the physics as normal, but once the positions have all been calculated, update the velocities to reflect the constrained movement of the nodes instead of the integration of acceleration. This method is visually indistinguishable to normal integration. It should be noted that momentum is not conserved with any method using Provot's edge constraints without specially accounting for it.

There are a few additional aspects that require some note. The first is that the inner diameter of the rings should not be directly used for edge constraints. Since the actual minimum distance between rings depends on their orientations, we must make an approximation. The simplest method is to constrain edge distances to an effective inner diameter that is a percentage of the true inner diameter, which is what our results are based on, but you may be able to achieve marginally better results by factoring in the a in this approximation. This allows the specified dimensions of the rings to be true to the visual appearance and reduce the amount of overlap in the contact regions of each ring under normal circumstances.

We avoid collision between non-adjacent rings using a simple spring force that all rings emit on each other when under a certain distance away [Bridson et al. 2002]. Since there is no defined surface like traditional cloth, spring forces seemed appropriate to approximate the average effect that two patches of a cloth have on each other. For larger meshes, a bounding-volume hierarchy would significantly improve the performance of this step, but for the relatively small patches (less than 30x30) we use, the naive $O(n^2)$ approach was acceptable. This produces plausibly positioned rings that avoid each other enough to not overlap, even if they are a bit distant at times. Figure 6 shows the chain avoiding intersections in the Table Cloth example where the corners cause the chain to collapse towards itself.

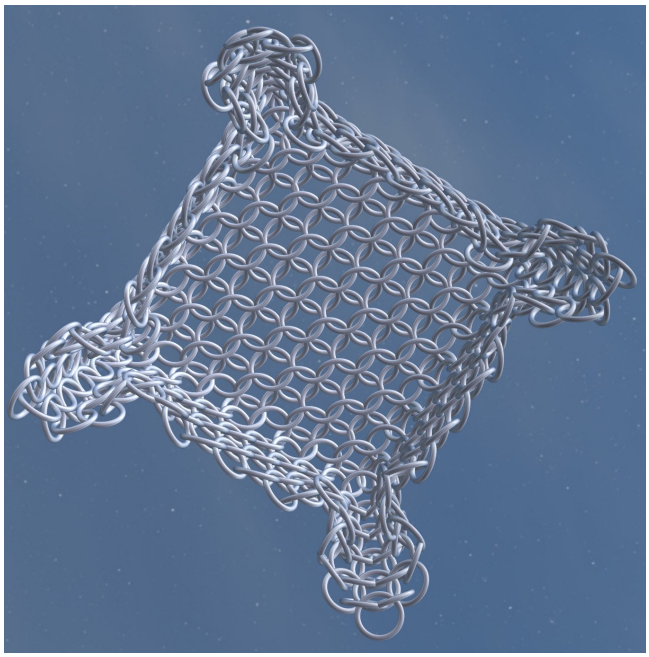

Figure 6: Table Cloth scene demonstration from below, highlighting the repulsive forces

4 ORIENTATION

To orient the rings we used a visually plausible approach that, while not physically accurate, looked good from a distance and was efficient enough for real-time performance. The general strategy was this: rotate the rings to face their neighbors, forming a sort of sheet of rings, then swivel the rings in a pattern to create the interlocking effect of chainmail.

Rotating the rings was done with a single matrix transformation, constructed by finding the vectors to neighboring rings and using them as the bases of the matrix. A visual example of this is shown in Figure 7. For internal rings, these vectors were from one neighbor to the opposite neighbor, and for edge rings they were just from that ring's center to its one neighbor. Because our ring model was oriented along the X/Z plane, these neighbor vectors were used as the X and Z bases for the transformation. Using their cross product

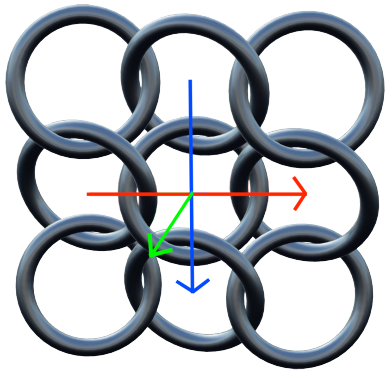

Figure 7: Determining x, y, and z bases for rings

as the Y basis yielded the full set of bases for the final transformation matrix in Equation 1.

Although this method correctly connected the rings in a sheet, it suffered from artifacts when the rings were not at perfect angles and distances. Because the X and Z bases were based on neighboring rings, they weren't necessarily orthogonal and often caused the rings to warp and stretch to fit. The solution to this was to rotate the X and Z vectors outwards to form a 90 degree angle. This was done with a series of cross products and averages that eventually yielded three orthogonal basis vectors which were used in forming the final matrix to transform the rings.

$$\begin{aligned} \vec{v_{45}} &= |\,\vec{x} + \vec{z}\,| \\ \vec{v_{135}} &= \vec{y} \times \vec{v_{45}} \\ \vec{x}' &= |\,\vec{v_{45}} + \vec{v_{135}}\,| \\ \vec{z}' &= |\,\vec{x} \times \vec{y}\,| \end{aligned}$$

$$M = \begin{bmatrix} x_0 & y_0 & z_0 & 0 \\ x_1 & y_1 & z_1 & 0 \\ x_2 & y_2 & z_2 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{1}$$

To add the necessary texture to the material that gives it its distinctive weave, each ring must be swiveled by some angle θ along some axis $\vec{S}$ (the swivel axis). While there are many physically-based factors that affect θ and $\vec{S}$, we came across a good approximation based on a few easily calculated factors. The first is the distance between connected rings. The second is the radius of the wire each ring is composed of. Intuitively, $\vec{S}$ can be defined as the as vector composed of the average distance between connected rings along both direction of the grids. This has the effect of having the swivel axis be based partially on the axis of compression. We then specify a minimum swivel that is based on a and c such that the material sits in a satisfying orientation under normal conditions. We can approximate this angle by $\frac{a}{c}$, which the gives us half the angle that the wire thickness offsets the two rings that have radius c. This approximation only diverges significantly when a approaches c, which is not a reasonable condition for this material. We chose to linearly interpolate between the upper bound ($\frac{\pi}{2}$) and the lower bound ($\frac{a}{c}$) based on the ratio of the average distance between a ring and its neighbors. One limitation to this strategy is that it only allows for one weave pattern across the entire mesh, so some drapes have artifacts when there is significant slack near the fixed points.

5 RENDERING

To complete our description of how to make visually plausible chainmail, we must outline how to render it in an appealing way. This section will serve as a practical guide for how to achieve similar results to ours.

The most fundamental component is having a well proportioned ring model for each ring in the mesh. Since we can define a point on a torus parametrically over two angles u, and v, it is trivial to generate a model of arbitrary resolution at run-time. You can scale the resolution by the size of the ring in the context of the application to get visually appealing results without unnecessary computation. With the methods described in 4, you can find the orientation for each of the rings and use methods like instanced rendering if your graphics API supports it to increase performance.

In the interest of performance, we render the rings using environment mapped reflections off a static skybox. Though we considered using ray tracing for more accurate reflections, the runtime hit would have been significant and negated our efforts to optimized the simulation for real-time performance. The environment mapping involved a basic reflection of the camera ray off the surface into into a cubemap texture of the skybox. While a basic reflection shader would have sufficed, we opted to make the chainmail look rougher by adding blur to the reflection color. Since the texture coordinates of the reflection were on the surface of a cubemap, we took a one-dimensional Gaussian noise algorithm and applied it to all axes. Using a fast algorithm from [DesLauriers 2015], we sampled 13 pixels of the cubemap per axis, and averaged them to create a decent-looking and efficient blur effect. The result, when combined with a customizable material color, creates a pleasing reflective surface that can be seen in figure 8.

6 CONCLUSIONS AND FUTURE WORK

When compared to pictures of real chainmail, we found our method to be fairly convincing. When put in isolation it actually becomes very believable. The rings rarely intersect with their neighbors and when they do, the patch is under compression. When under asymmetric compression or in regions of high curvature, the pattern becomes less defined as it should and these potential artifacts go easily unnoticed. For dynamic scenes the intersections become even more difficult to spot, even for configurations that have many intersecting rings.

There are no large-scale intersections where large portions of the cloth pulls through itself in all but the most extreme scenarios. If the mesh is fixed at the corners and forced to support its own weight from the bottom, it will not behave in a desirable fashion. We found that these scenes needed to be intentionally fabricated to fail.

One artifact that is noticeable under low tension scenarios is due to the diagonal row pattern. Since this diagonal pattern is global to the mesh, suspending corners of the square sheet can often look

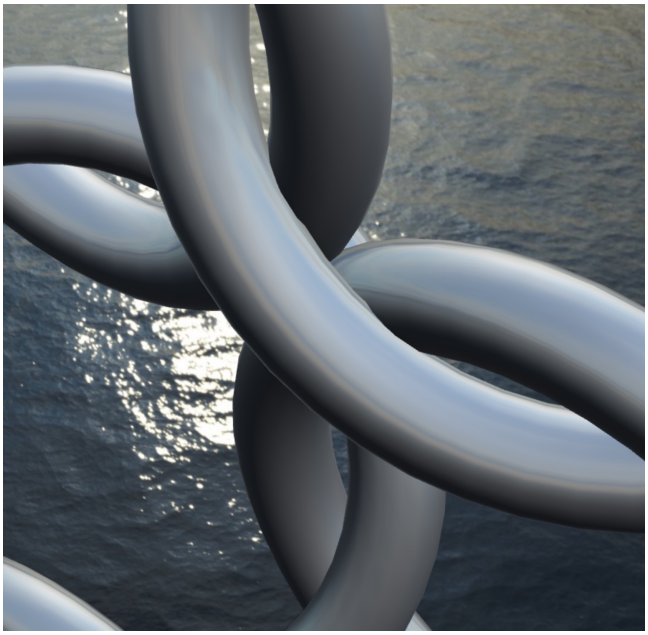

Figure 8: Up-close view of the environment mapping on the rings

odd (as seen in the top-right side of figure 2). This could be solved by propagating orientation information (as simple as a boolean) throughout the mesh from the fixed points, but we did not pursue this problem.

6.1 Physically Accurate Constraints

While the goal of this paper was to produce physically plausible results, we explored the possibility of physically-correct, constraint-based orientation and position that explicitly prevents self intersection. Much of this work co-evolved with the work described in Section 4 with the "swivel" axis. First, we need to define a more formal relationship between the position and orientations of two tori in the mesh. We can define a torus with a parametric function $\vec{F}$ over two variables u, and v and a position $\vec{P}$

$$F_x(u,v) = (c + a \times cosv)cosu + P_x \tag{2}$$

$$F_y(u,v) = (c + a \times cosv)sinu + P_y \tag{3}$$

$$F_z(u,v) = a \times sinv + P_z \tag{4}$$

We can transform these Cartesian coordinates by two angles: θ and ϕ, where ϕ is the angle to rotate the vertical axis around the torus normal and θ is how much to rotate around this "swivel" axis. Now, we can define every point on the torus with two variables and two parameters.

$$F_{xT} = (F_z sin\theta + F_x cos\theta)cos\phi - F_y sin\phi \tag{5}$$

$$F_{yT} = (F_z sin\theta + F_x cos\theta)sin\phi + F_y cos\phi \tag{6}$$

$$F_{zT} = F_z cos\theta - F_x sin\theta \tag{7}$$

Each ring has a position in space and an orientation (represented as a normal). If we transform one ring into the coordinate space of the other and align one axis with the vector between their positions, we can relate the two rings by a single distance measurement, and a mutual swivel required to form contact (if two tori have $(\phi, \theta) = (0, 0)$ in this coordinate space). Each normal can be translated into a swivel (ϕ, θ) that is used to aid temporal and spacial coherence, but is not directly used in any of the equations. Using the above transformed parametric equations (5-7), we can determine if there is any contact or intersection. If there is none, then the two rings already satisfy the interlocking and non-intersecting constraint, so the next step can be skipped.

If the rings are intersecting, we must determine a mutual minimum swivel to mitigate the intersection. There are many solutions to these parametric equations since we have three equations ($\vec{F_{T1}} = \vec{F_{T2}}$) and six variables ($u_1, u_2, v_1, v_2, \theta, \phi$) (all other variables being constrained). Since we are only interested in the contact between the surfaces of the two tori, we could add the constraint that the surface normals are antiparallel, which should result in a 1-fold infinity of solutions in most cases, but the current parameters θ and ϕ of each ring can be used to aid coherence. Future work is required to determine the best method to achieve this, but we suspect it will involve biasing numerical methods of solving our system of equations in the direction of our current swivels.

Using similar strategy as Provot for preventing superelasticity, we can define an inverse of this system: given a θ and ϕ for two rings, we need to solve how far apart they must be to satisfy the contact conditions. Luckily, since we have the additional constraint that the two rings will be at the maximum distance for this θ and ϕ, there are only two solutions. However, an analytical solution may be difficult to come to.

This method serves to replace the Provot correction, but position, velocity, and acceleration integration can be performed as normal.

7 CONTRIBUTIONS

Glenn wrote the base cloth simulation code and Kevin adapted it to meet the needs of the chainmail material. Kevin wrote the base code for rendering the scene and Glenn wrote the code for realistic shading. Glenn worked on orienting the rings based on the simulated mesh and both Kevin and Glenn worked on the swivel logic. Kevin developed a theoretical model for more physically accurate position and orientation simulation.

REFERENCES

Robert Bridson, Ronald Fedkiw, and John Anderson. 2002. Robust Treatment of Collisions, Contact and Friction for Cloth Animation. *ACM Trans. Graph.* 21, 3 (July 2002), 594–603. https://doi.org/10.1145/566654.566623

Matt DesLauriers. 2015. glsl-fast-gaussian-blur. (2015). https://github.com/Jam3/glsl-fast-gaussian-blur

Xavier Provot. 1995. Deformation Constraints in a Mass-Spring Model to Describe Rigid Cloth. (1995).

Xavier Provot. 1997. Collision and self-collision handling in cloth model dedicated to design garments. (1997), 177–189 pages.