Eurographics/ ACM SIGGRAPH Symposium on Computer Animation (2012)
P. Kry and J. Lee (Editors)

Fast Simulation of Inextensible Hair and Fur

M. Müller T.J. Kim N. Chentanez

Nvidia PhysX Research

Abstract

In this paper we focus on the fast simulation of hair and fur on animated characters. While it is common in feature films to simulate hair and fur of computer generated actors, characters are still mostly hand-animated in computer games. A main difficulty of simulating hair is that it is perceived as inextensible by humans. Preventing an object from being stretched is a global, non-linear problem. This is the reason why simulating completely inextensible objects in real-time remains a major challenge and an open research topic.

Existing approaches typically use multiple iterations per visual frame to solve the physical equations followed by number of strain limiting iterations. Adjusting the number of iterations is a way to increase the accuracy of the simulation at the expense of more computation and vice versa. In the extreme case of one solver iteration per visual frame, most existing methods break down, either by becoming unstable or by introducing a substantial amount of stretching.

In this paper, we present a robust method that guarantees inextensiblity with a single iteration per frame. This extreme performance comes at the price of reduced accuracy. We found that for applications in graphics, it is worth to pay this price because the inaccuracies are not visually disturbing but the speed of the method allows the simulation of thousands of hairs in real-time.

Categories and Subject Descriptors (according to ACM CCS): Computer Graphics [I.3.5]: Computational Geometry and Object Modeling—Physically Based Modeling; Computer Graphics [I.3.7]: Three-Dimensional Graphics and Realism—Animation and Virtual Reality

1. Introduction

While hair and fur is common in the movie industry, those features are still mostly hand animated in computer games. The reason is that there is a lack of simulation methods that are fast and robust enough to be used in real-time environments. Finding such a method is the focus of this paper.

Hair shows a distinct global behavior. While it has relatively low bending resistance, it is perceived by humans as being inextensible along the extended dimension. In other words, in simulations, we have to consider hair as deformable only in certain directions guaranteeing zero-stretch along the main axis. However, making simulated deformable objects inextensible is a non-trivial problem.

One method to reduces the space of possible deformations is to use generalized coordinates. In the case of ropes, angles between segments are used as the degrees of freedom instead of the positions of the mass points between the segments. In this subspace of spatial configurations, a rope always keeps its rest length. However, using generalized coordinates in the presence of colliding objects and in over-constrained situations becomes a difficult problem and its solution expensive.

Most methods used in computer graphics simulate deformable objects using Euclidean coordinates. Here, all the mass points or finite elements of a discretized deformable body are allowed to move freely in space and they are held together by forces. In this case, inextensibility is achieved in the limit of forces with infinite stiffness.

However, simply increasing the stiffness constants introduces all the well known stability issues. Instead of using spring-like forces, one can solve for constraint forces that yield velocities which keep the mass points on trajectories that do not change the constraint functions. One problem with this approach is that the mass points can drift away from the conserving paths, especially when large time steps are used.

Therefore, inextensibility is typically achieved by limiting strain geometrically after the dynamics solver has updated the positions of the mass points at each time step. In the case of a mass spring system, the strain limiting step moves the mass points such that all the springs are not extended by more than a given factor. Finding such positions is a global, non-linear problem.

The classical way to solve such a system is to use the Newton-Raphson approach. First, the constraint functions are linearized at the current positions, then the resulting linear system is solved. These two steps are repeated until the error falls below a certain threshold. This approach was used by Goldenthal et al. [GHF*07]. Because these two steps are expensive, the number of iterations has to be kept small and therefore, there typically remains some stretch in the system. Another problem with this approach is that the linear system matrix becomes ill conditioned in over-constrained scenarios.

Another approach to solve the constraint system is to project each constraint one by one multiple times in a Gauss-Seidel-type fashion. This method is used in Position Based Dynamics (PBD) [MHR06]. Due to the slow convergence of Gauss-Seidel solvers and since the number of iterations has to be restricted in real-time applications, objects appear stretchy. This problem can be reduced by adding long range constraints [MÖ8]. Besides the additional effort to create the hierarchy, zero-stretch is still not guaranteed.

In this paper we present a simulation method that takes only one iteration per visible frame while guaranteeing zero-stretch. Our method extends a technique called *Follow The Leader* which as previously been used for quasy-static simulations only [BLM04].

The price we pay for requiring only one iteration is that our method introduces artificial damping. In many situations, this is not problematic because objects in the real world such as cloth are quite heavily damped. Also, our target scenario is adding simulated clothing and hair to animated characters. Those actors constantly add energy to the system when they move.

In summary, our two main contributions are

- The generalization of the *Follow The Leader* method to simulate the dynamic behavior of inextensible ropes.
- Techniques for handling bending resistance and over-constrained situations, for fast collision handling with a the character's skin and for hair-hair interaction.

2. Related Work

The simulation of one dimensional elastic solids such as hair and rods has been studied extensively in computer graphics. For an excellent survey of the field can we recommend [WBK*07] and [HCB*07]. More recent developments include the study of discrete elastic rods [BWR*08] and viscous threads [BAV*10] which treat the centerline as dynamic and the material frame as static. Spillman and Teschner [ST07,ST08] represent material frames with quaternions and couple them to the centerline with penalty force. Kubiak et al. [KPGF07] use PBD to simulate threads in knot tying applications. A more analytical approach was taken by Theetten et al. [TGAB08] who use dynamic splines for the simulation. Selle et al. [SLF08] propose to use altitude springs to handle torsion in hair simulation and the semi-implicit discretization of springs to make the simulation unconditionally stable. Bertails [Ber09] devised a linear time algorithm to simulate piecewise helical rods. Since the explicit handling of hair-hair interaction is expensive, [PHA05] proposed to use a regular background grid to compute a hair density field which is used to approximate mutual hair repulsion and friction. A similar approach was used by McAdams et al. [MSW*09]. They handle the bulk interaction and volume preservation of hairs using an Eulerian grid. Sueda et al. [SJLP11] propose the use of reduced degree of freedom Eulerian nodes to handle the simulation of ropes and cables in highly constrained scenarios. Often it is not necessary to simulate every individual hair. Chang et al. [CJY02] proposed to only simulate a subset of hairs called *key-hairs* and interpolated all other hair shapes from this subset.

An important problem in hair simulations is enforcement of inextensibility. For cloth, Provot [Pro95] proposed a method for limiting spring extension to a given threshold. This idea was further developed by many authors such as Bridson et al. [BFA02] who corrected the velocities to respect a given stretch limit, and [MHR06] in the context of position based dynamics. Brown et al. [BLM04] proposed to use the *Follow The Leader* (FTL) scheme for the quasy static simulation of inextensible ropes in a knot tying application. This is the method we extend in this paper to handle dynamic scenarios.

Global approaches for enforcing inextensibility solve for all the nodes at once. Examples of this approach are implicit constraint enforcement [HCJ*05], fast projection to constrained manifolds [GHF*07] and its extension based on a multi-step time integration scheme to reduce numerical damping [EB08]. To reduce discretization dependent artifacts [TPS09] used continuum based strain limiting . Multi-resolution approaches to inextensibility for reducing computational cost were proposed by [MÖ8] for PBD and [WOR10] for strain-limiting of isotropic materials.

A completely different approach to reduce stretchiness are the data driven methods [WOR10], [KGBS11]. Instead of simulating the cloth in real-time, pre-recorded simulations are synthesized at run time. In this case, inextensibility can be enforced by simply increasing the iteration count during the recording of the samples because this phase is not time critical.

Strain can also be limited by simulating a coarse mesh and add small detail with a different method. One approach is to augment rendering with a high resolution normal map

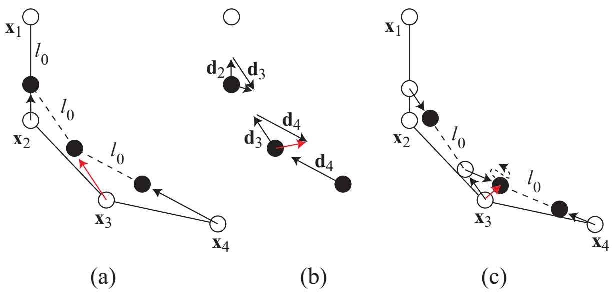

Figure 1: *(a) The static Follow The Leader projection. The white points show the positions of the particles before projection with the topmost fixed. Each particle is moved towards its predecessor to enforce their mutual distance to be l_0. The red arrow is the resulting velocity of particle 3. (b) Velocity correction: The new velocities are computed from the own projection vector minus the one from the parent. The resulting velocity for particle 3 is shown in red. (c) One iteration of PBD, the resulting velocity vector in red and the path of particle 3 with more iterations.*

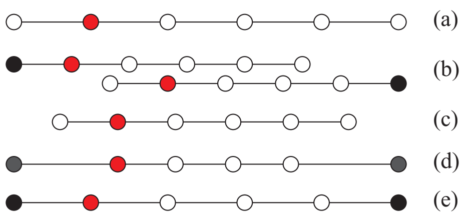

Figure 2: *(a) Overstretched rope. (b) After FTL projection started from the left and from the right. (c) Average for detached rope. (d) Averaged with two ends fixed. (e) Averaged using weighted sums.*

generated with procedural methods [HBVMT99] [KWH04], [DRM*] , [Lov06]. Müller and Chentanez [MC10] added wrinkles by running a static simulation on a fine resolution mesh attached to the coarse simulation mesh.

3. Rope Simulation

We model a single hair by a chain of particles that is attached at one end (see Figure 1). Let $\mathbf{x}_1,\ldots,\mathbf{x}_n$ be the positions of the particles with particle 1 being attached and let us assume that the rest distances between adjacent particles are all l_0. Starting with positions that violate the distance constraints we want to move the particles such that all the constraints are satisfied but with the restriction that we are only allowed to iterate through all the particles once.

3.1. Static Follow-The-Leader

One way to do this is to process the particles in the order from 2 to n. Particle 2 has to be on a sphere with radius l_0 around particle 1. A natural choice for its position is to choose the point on the sphere that is closest to the original position $\mathbf{x}_2$, i.e. to move it in the direction of particle 1. In case of a collision, the two remaining degrees of freedom of particle 2 within the sphere can be used to resolve the collision simultaneously. In this case, one chooses the position closest to $\mathbf{x}_2$ that resolves the collision. Once the new position of particle 2 is determined, the algorithm continues with particle 3 preforming the same steps as before with particle 2 taking the role of particle 1. Figure 1(a) shows the algorithm for a chain of four particles.

This algorithm that we just derived from the structure of the constrained system and the restriction that we are only allowed to iterate through the particles once is called *Follow The Leader* (FTL). In the field of physically based simulation it was used by [BLM04] for the quasy static simulation of knot tying. One of the main reasons why it has not become popular in the field is that making it work in a dynamic simulation is challenging. It is this problem that we will investigate next.

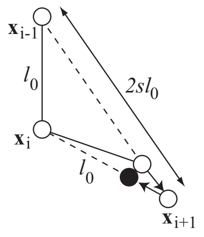

Figure 3: *Bending resistance is modeled by additional unilateral distance constraints between particles separated by two edges.*

3.2. Dynamic Follow-The-Leader

For a dynamic simulation, the particles need to store and update velocities along with the positions. Let $\mathbf{v}_1,\ldots,\mathbf{v}_n$ be the velocities of the particles. Since we manipulate positions to drive the simulation we need a way to derive the velocity updates from the position updates. Position Based Dynamics (PBD) [MHR06] provides such a way. The algorithm to perform one time step of PBD has the following simple structure

$$\mathbf{p} \leftarrow \mathbf{x} + \Delta t\,\mathbf{v} + \Delta t^2\,\mathbf{f} \tag{1}$$
$$\mathbf{p} \leftarrow \text{SolveConstraints}(\mathbf{p}) \tag{2}$$
$$\mathbf{v} \leftarrow \frac{\mathbf{p}-\mathbf{x}}{\Delta t} \tag{3}$$
$$\mathbf{x} \leftarrow \mathbf{p}, \tag{4}$$

where Δt is the time step size and $\mathbf{f}$ external forces. To make FTL dynamic we could simply perform static FTL inside the SolverConstraints() method. This way, we would still only need one iteration per time step because the four steps can be performed together in a single pass.

However, this straight forward method yields the strange behavior shown in Figure 4 and in the accompanying video for a horizontal chain attached on the left and falling under gravity. To understand why this simple approach does not work we have to have a closer look at the physical system we are simulating. In traditional PBD to solve a distance constraint between two particles, they are moved towards or away from each other by distances proportional to their inverse masses. Therefore, moving just the second particle corresponds to the situation in which the first particle has infinite mass. In general, by using FTL as the projection step in a dynamic simulation, a system with masses $m, sm, s^2m, \ldots s^{n-1}m$ with $s \to 0$ is simulated – independent of the integration method.

We have not found a discussion of this interesting, physically impossible system in the literature but came to the following intuition about its behavior. At a given point in time as shown in any image of Figure 4, a series of particles starting from the attachment downwards has come to rest due to damping. One particle below them contains all the current kinetic energy of the system and swings around all the particles below it without effort because the sum of all the masses of the tail is infinitely smaller than its on mass. In turn, nothing of its motion is propagated to the parents because they, in turn, have infinitely more mass.

3.3. Velocity Correction

Deriving the dynamic behavior of a chain of particles with equal masses after an FTL projection step is surprisingly difficult because the extremely uneven mass distribution is deeply inherent to the FTL projection. After investigating various methods that all failed, we found a very simple solution. It completely hides the uneven mass distribution at the price of introducing some numerical damping.

Let $\mathbf{d}_i$ be the correction vector computed by FTL for particle i. When considering constraint $(i-1, i)$, FTL updates the positions of particles $i-1$ and i as

$$\mathbf{p}_{i-1} \leftarrow \mathbf{p}_{i-1} \text{ (no update)} \tag{5}$$
$$\mathbf{p}_i \leftarrow \mathbf{p}_i + \mathbf{d}_i. \tag{6}$$

To generate the behavior of two particles with the same mass we would have to move particle $i-1$ as well by the same amount as

$$\mathbf{p}_{i-1} \leftarrow \mathbf{p}_{i-1} - \mathbf{d}_i \tag{7}$$
$$\mathbf{p}_i \leftarrow \mathbf{p}_i \quad + \mathbf{d}_i \tag{8}$$

in the update step of particle i. However, this would violate the constraint between particle $i-2$ and particle $i-1$. The idea to circumvent this problem is to use this symmetric position of particle $i-1$ only to compute the new velocity in Eq. (3) at the end of the time step, not for the position of the particle. We, thus, modify Eq. (3) as follows:

$$\mathbf{v}_i \leftarrow \frac{\mathbf{p}_i - \mathbf{x}_i}{\Delta t} + s\frac{-\mathbf{d}_{i+1}}{\Delta t}. \tag{9}$$
$$\tag{10}$$

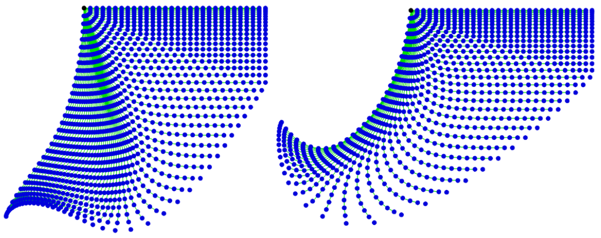
Figure 5: *Rope simulation with velocity correction. Left: Correction scale $s = 1$. Right: Correction scale $s = 0.9$.*

The resulting velocities are shown in Figure 1(b). In Figure 1(c) we show the velocity for particle 3 after one iteration of PBD and the estimated location when the number of iterations is increased. As the three images in Figure 1 show, the corrected velocity in (b) resembles the velocity in (c) more but is tilted more against the pulling direction which introduces more damping.

In Eq. (9) we introduced the scalar parameter $s \in [0, 1]$. For $s = 1$ the uneven masses are completely compensated for but with the introduction of numerical damping (see Figure 5 left). For s smaller but close to 1, damping is reduced while the artifact of the uneven masses is still hardly noticeable (see Figure 5 right and the accompanying video). Reducing s is a simple and computationally very cheap way to reduce the damping problem.

3.4. Backward Propagation

Due to the nature of FTL, information is only propagated slowly against the projection order in ropes that are attached on one end only. This property can produce visual artifacts as shown in the left image of Figure **??**. In most of our experiments, especially in the fur and hair scenes, this was not problematic. In cases where one way wave propagation is a problem, we run FTL in both directions similar to the case of detached ropes. However, instead of averaging the results, we first perform the backward projection pass from the detached to attached end and then perform the regular forward pass using the updated positions. Due to the order, the rope is correctly attached and all distance constraints satisfied after the two passes. The right image of Figure **??** shows a simulation with two pass projection. In contrast to the left image, the waves and wrinkles propagate further upwards from the small sphere.

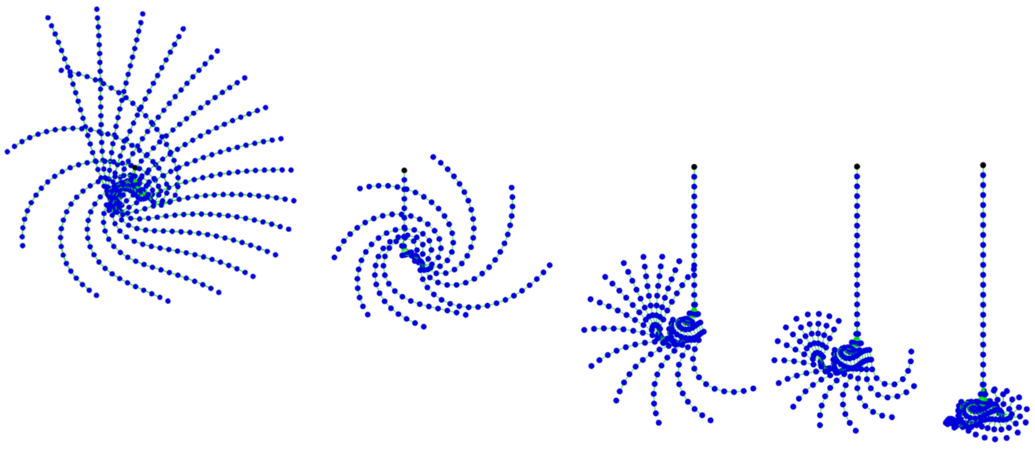

Figure 4: *Follow The Leader projection yields the dynamic behavior of a chain of particles each of which has infinitely more mass then its successor. The figure shows a simulation of such a chain starting as a horizontal line attached at the left end and falling under gravity. Each image corresponds to a different point in time and shows a set of overlayed successive frames.*

3.5. Bending Resistance

To handle bending resistance, we consider distances between particles two edges apart. Bending resistance is controlled by a parameter $s_b \in [0 \dots 1]$. While walking along the rope, we make sure that

$$|\mathbf{p}_{i-1} - \mathbf{p}_{i+1}| \geq 2\, s\, l_0. \tag{11}$$

This is a unilateral constraint. Before we project constraint $(i, i+1)$ we check the bending condition. If it is violated we move $\mathbf{p}_{i+1}$ along the line between $\mathbf{p}_{i-1}$ and $\mathbf{p}_{i+1}$ such that bending constraint is met (see Figure 3). We choose this order to give stretching resistance priority over bending resistance. Theoretically, we could solve for both constraints simultaneously. However, in moving particles along their common lines only, we make sure that the projections in Eq. (7) and Eq. (8) do not introduce any torque. The parameter s is related to the lower angle limit ϕ_{min} between two segments via

$$\phi_{\text{min}} = \frac{\pi}{2} - 2 \arccos s. \tag{12}$$

Due to fact that we only need one iteration to guarantee inextensibility, there is no need to only simulate a subset of key-hairs [CJY02] because our method can handle thousands of hairs in real time as Figure 6 shows. For off-line applications, our method even allows the simulation of real world hair numbers in the order of 10^5 in reasonable time. However, with an increasing number of hair, handling hair-hair interactions and collisions with the character's skin become the bottleneck. We speed up both processes by using a regular background grid.

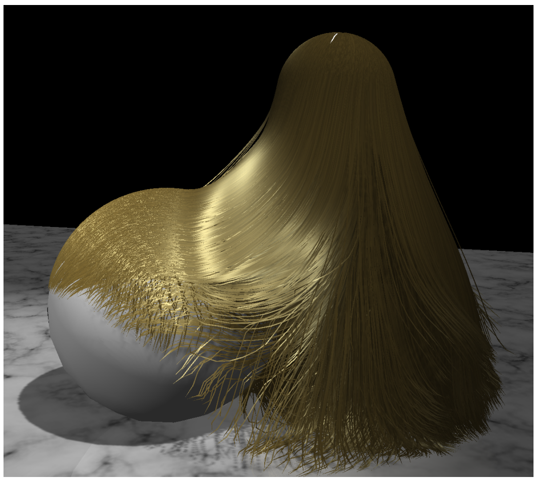

Figure 6: *A simulation of 10k hair with 30 particles each at interactive rates.*

3.6. Hair-Hair Interaction

To handle hair-hair friction and repulsion, we use a method similar to the one proposed by Petrovic et al [PHA05]. At each time step we first compute a particle density field on a regular background grid. Each particle adds its tri-linear interpolation weights to the density values of the 8 nodes of the surrounding cell. It also adds its velocity multiplied by the tri-linear interpolation weights to the velocities of the 8 nodes. Dividing the velocities by the densities stored at the grid nodes yields an averaged velocity field that can be used to imitate friction.

For each particle, we interpolate this velocity field at the particle's position. We then replace the particle's velocity with a weighted average of its own velocity and the velocity interpolated from the grid at each time step.

$$\mathbf{v} \leftarrow (1 - s_{\text{friction}})\mathbf{v} + s_{\text{friction}}\mathbf{v}_{\text{grid}} \tag{13}$$

By tuning the weight s_{friction} we can control the amount of friction.

Hair-hair repulsion is needed to get volumetric hairstyles and to get the impression of finite hair thickness. We use a simple approach to achieve this effect. At each time step we compute the normalized gradient $\mathbf{g} = \nabla\rho/|\nabla\rho|$ of the density field at the particle's location and update the velocity using

$$\mathbf{v} \leftarrow \mathbf{v} + s_{\text{repulsion}}\mathbf{g}/\Delta t. \tag{14}$$

Both velocity correction steps are executed after PBD integration.

3.7. Collision with the Character

Because we use high resolution cloth and hair it is important that collisions against the character's skin is accurate. Instead of replacing the body with primitives such as capsules and spheres as is common in games, we re-create a signed distance field every time step using Fast Marching [Set95]. Once this relatively time consuming step is done, collision handling is fast enough to collide hundreds of thousands of particles in real-time. We simply sample the distance field at the location of the particle and multiply it by the normalized gradient to get the offset to the surface.

However, static collision handling is not robust enough when the character moves fast. Therefore we compute a velocity field simultaneously. When initializing the distances near the surface, we also initialize the velocities based on the previous and current animation frame. Then, whenever a new distance is computed by Fast Marching, we update the velocity at the same location by averaging all the known adjacent velocities. This way, the velocity field is propagated the same way as the distance field. We then take the velocity of the surface into account to compute proper restitution of the particles.

4. Results

Figure 7 shows a comparison of dynamic FTL (blue), PBD (green) and symplectic Euler (red). A rope composed of 30 particles is dragged around by the user at the top. The time step size, particle masses, segment lengths and gravitational acceleration are $0.01s$, $0.01kg$, $0.02m$ and $-10m/s^2$, respectively. In a first experiment (left) we let all three methods spend about same amount of time, i.e. 2 iterations for PBD and 2 sub-steps for symplectic Euler. The maximum stiffness allowed to keep the latter stable was $k = 100N/m$. Both PBD and Euler show a substantial amount of stretching. In a second experiment we adjusted the parameters such that all three ropes showed similar behavior. To achieve this 25 PBD iterations where needed. For Euler integration we had to increase the stiffness to $k = 3000N/m$ which could only be simulated stably with 20 substeps.

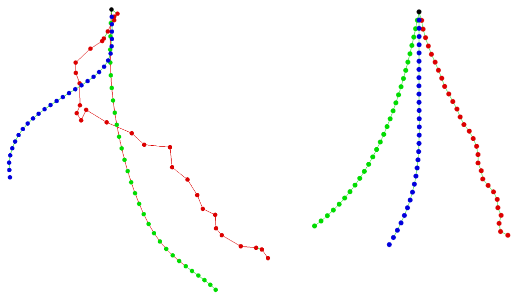
Figure 7: *Screen shots of an animation in which the top particle of a rope is dragged around using three simulation methods: dynamic FTL (blue), PBD (green) and symplectic Euler (red). Left: all three methods spend the same amount of time per frame. Right: Time spend is adjusted to yield similar results.*

All our 3D examples were simulated at interactive rates on a single core of an Intel Core i7 CPU at 3.1 GHz and rendered on an NVIDIA GeForce GTX 480 GPU. The timings are summarized in Table 1. Figure 8 shows frames from an animation of an Afghan dog. The fur is simulated with 6600 fibers and 47k particles. On average, the resolution of the density and distance field grids is 60 x 30 x 60 cells. Those grids are recomputed every frame so their sizes change during the simulation. We demonstrate the effect of hair-hair repulsion with the furry monster shown in Figure 9. Here we used 29k fibers and 247k particles in total.

The final scene shown in Figure **??** demonstrates how our method can be used to simulate hair on a character...

Scene	All	Solve	Density	Dist	Rendering
Monster	435.3	80.2	57.2	47.9	250.0
Afghan	89.2	14.8	11.2	33.2	30.0
Girl	227.0	20.0	27.0	40.0	140.0

Table 1: *Timings of the samples in milliseconds. Solve: Solving for all constraints. Density: Compute density map and handle hair-hair repulsion and friction. Dist: Compute distance field and handle character collisions.*

5. Conclusion and Future Work

We have presented a method to simulate inextensible hair and fur which guarantees zero-stretch in a single iteration per visual frame. We also proposed a variety of additional techniques that allow the simulation of hair and fur on animated characters based on our new approach. As our results

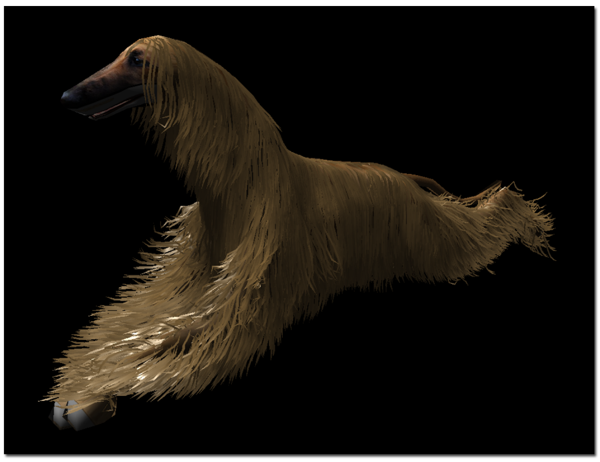

Figure 8: *This afghan dog model shows how our method can be used to efficiently simulated fur on characters.*

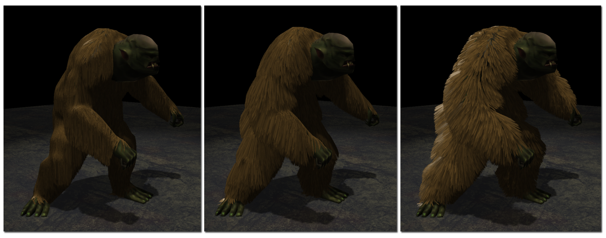

Figure 9: *A furry monster demonstrating the effect of hair-hair repulsion.*

show, our method makes possible the simulation of thousands of hair in real-time including character collision handling and hair-hair interaction. The high performance comes at the price of additional numerical damping. We did not find this to be a significant drawback because characters constantly add energy when they are in motion and objects in the real-wold are significantly damped. We also proposed a simple method to reduce artificial damping.

References

[BAV*10] BERGOU M., AUDOLY B., VOUGA E., WARDETZKY M., GRINSPUN E.: Discrete viscous threads. *ACM Trans. Graph. 29* (July 2010), 116:1–116:10. 2

[Ber09] BERTAILS F.: Linear time super-helices. *Comput. Graph. Forum 28*, 2 (2009), 417–426. 2

[BFA02] BRIDSON R., FEDKIW R., ANDERSON J.: Robust treatment of collisions, contact and friction for cloth animation. *Proceedings of ACM Siggraph* (2002), 594–603. 2

[BLM04] BROWN J., LATOMBE J.-C., MONTGOMERY K.: Real-time knot-tying simulation. *Vis. Comput. 20* (May 2004), 165–179. 2, 3

[BWR*08] BERGOU M., WARDETZKY M., ROBINSON S., AUDOLY B., GRINSPUN E.: Discrete elastic rods. *ACM Trans. Graph. 27* (August 2008), 63:1–63:12. 2

[CJY02] CHANG J. T., JIN J., YU Y.: A practical model for hair mutual interactions. In *Proceedings of the 2002 ACM SIGGRAPH/Eurographics symposium on Computer animation* (New York, NY, USA, 2002), SCA '02, ACM, pp. 73–80. 2, 5

[DRM*] DUQUE C., REIS G., MARIO J., MARTINO D., BATAGELO H. C.: Real-time simulation of wrinkles. In *Proceedings of WSCG*. 3

[EB08] ENGLISH E., BRIDSON R.: Animating developable surfaces using nonconforming elements. *ACM Trans. Graph. 27*, 3 (2008). 2

[GHF*07] GOLDENTHAL R., HARMON D., FATTAL R., BERCOVIER M., GRINSPUN E.: Efficient simulation of inextensible cloth. *ACM Trans. Graph. 26* (July 2007). 2

[HBVMT99] HADAP S., BANGERTER E., VOLINO P., MAGNENAT-THALMANN N.: Animating wrinkles on clothes. In *Proceedings of the conference on Visualization '99: celebrating ten years* (Los Alamitos, CA, USA, 1999), VIS '99, IEEE Computer Society Press, pp. 175–182. 3

[HCB*07] HADAP S., CANI M.-P., BERTAILS F., LIN M. C., WARD K., MARSCHNER S. R., KIM T.-Y., KACIC-ALESIC Z.: *Strands and Hair: Modeling, Animation, and Rendering*, vol. 33. ACM SIGGRAPH Course Notes, 2007. Award: Best Course Notes for a New Course. 153 pages. 2

[HCJ*05] HONG M., CHOI M.-H., JUNG S., WELCH S., TRAPP J.: Effective constrained dynamic simulation using implicit constraint enforcement. In *Robotics and Automation, 2005. ICRA 2005. Proceedings of the 2005 IEEE International Conference on* (april 2005), pp. 4520 – 4525. 2

[KGBS11] KAVAN L., GERSZEWSKI D., BARGTEIL A., SLOAN P.-P.: Physics-inspired upsampling for cloth simulation in games. *ACM Trans. Graph. 30*, 4 (2011), 93:1–93:9. 2

[KPGF07] KUBIAK B., PIETRONI N., GANOVELLI F., FRATARCANGELI M.: A robust method for real-time thread simulation. In *Proceedings of the 2007 ACM symposium on Virtual reality software and technology* (New York, NY, USA, 2007), VRST '07, ACM, pp. 85–88. 2

[KWH04] KIMMERLE S., WACKER M., HOLZER C.: Multilayered wrinkle textures from strain. In *VMV* (2004), Girod B., Magnor M. A., Seidel H. P., (Eds.), Aka GmbH, p. 225Ð232. 3

[Lov06] LOVISCACH J.: Wrinkling coarse meshes on the gpu. *Comput. Graph. Forum 25*, 3 (2006), 467–476. 3

[MÖ8] MÜLLER M.: Hierarchical position based dynamics. *Proceedings of Virtual Reality Interactions and Physical Simulations* (2008). 2

[MC10] Müller M., Chentanez N.: Wrinkle meshes. In *Proceedings of the 2010 ACM SIGGRAPH/Eurographics Symposium on Computer Animation* (Aire-la-Ville, Switzerland, Switzerland, 2010), SCA '10, Eurographics Association, pp. 85–92. 3

[MHR06] Müller M., Hennix B. H. M., Ratcliff J.: Position based dynamics. *Proceedings of Virtual Reality Interactions and Physical Simulations* (2006), 71–80. 2, 3

[MSW*09] McAdams A., Selle A., Ward K., Sifakis E., Teran J.: Detail preserving continuum simulation of straight hair. *ACM Trans. Graph. 28* (July 2009), 62:1–62:6. 2

[PHA05] Petrovic L., Henne M., Anderson J.: Volumetric methods for simulation and rendering of hair. In *Tech. rep., Pixar Animation Studios* (2005). 2, 5

[Pro95] Provot X.: Deformation constraints in a mass-spring model to describe rigid cloth behavior. *Proceedings of Graphics Interface* (1995), 147Ű–154. 2

[Set95] Sethian J. A.: A fast marching level set method for monotonically advancing fronts. In *Proc. Nat. Acad. Sci* (1995), pp. 1591–1595. 6

[SJLP11] Sueda S., Jones G. L., Levin D. I. W., Pai D. K.: Large-scale dynamic simulation of highly constrained strands. *ACM Trans. Graph. 30* (Aug. 2011), 39:1–39:10. 2

[SLF08] Selle A., Lentine M., Fedkiw R.: A mass spring model for hair simulation. *ACM Trans. Graph. 27* (August 2008), 64:1–64:11. 2

[ST07] Spillmann J., Teschner M.: M.: Corde: Cosserat rod elements for the dynamic simulation of one-dimensional elastic objects. In *In Proc. ACM SIGGRAPH/Eurographics Symposium on Computer Animation* (2007), pp. 63–72. 2

[ST08] Spillmann J., Teschner M.: An adaptive contact model for the robust simulation of knots. *Comput. Graph. Forum 27*, 2 (2008), 497–506. 2

[TGAB08] Theetten A., Grisoni L., Andriot C., Barsky B.: Geometrically exact dynamic splines. *Comput. Aided Des. 40* (January 2008), 35–48. 2

[TPS09] Thomaszewski B., Pabst S., Strasser W.: Continuum-based strain limiting. *Comput. Graph. Forum 28*, 2 (2009), 569–576. 2

[WBK*07] Ward K., Bertails F., Kim T.-Y., Marschner S. R., Cani M.-P., Lin M. C.: A survey on hair modeling: Styling, simulation, and rendering. *IEEE Transactions on Visualization and Computer Graphics 13* (March 2007), 213–234. 2

[WOR10] Wang H., O'Brien J., Ramamoorthi R.: Multi-resolution isotropic strain limiting. *ACM Transactions on Graphics (SIGGRAPH Asia 2010) 29*, 6 (December 2010), 156:1–156:10. 2