CEIG - Spanish Computer Graphics Conference (2015)
Jorge Lopez-Moreno and Mateu Sbert (Editors)

Real-time Inextensible Hair with Volume and Shape

R.M. Sánchez-Banderas[1], H. Barreiro [1], I. García-Fernández[2] and M. Pérez[2]

[1]ETSE-UV, Valencia, Spain
[2]Departamento de Informática, ETSE-UV, Valencia, Spain

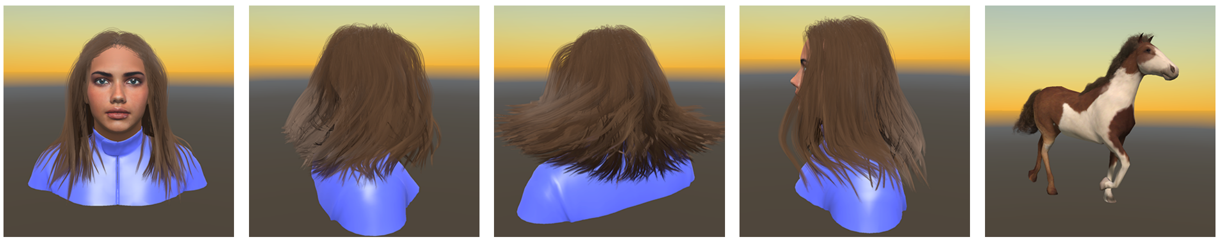

Figure 1: *Examples of the hair behaviour in two different scenarios using our method.*

Abstract

Hair simulation is a common topic extensively studied in computer graphics. One of the many challenges in this field is simulating realistic hair in a real-time environment. In this paper, we propose a unified simulation scheme to consider three of the key features in hair simulation; inextensibility, shape preservation and hair-hair interaction. We use an extension to the Dynamic Follow the Leader (DFTL) method to include shape preservation. Our implementation is also coupled with a Lagrangian approach to address the hair-hair interaction dynamics. A GPU-friendly scheme is proposed that is able to exploit the massive parallelism these devices offer, being able to simulate thousands of strands in real-time. The method has been integrated in a game development platform with a shading model for rendering and several test applications have been developed using this implementation.

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

1. Introduction

Hair and fur is an important aspect for the characterization of virtual actors. For this reason, one of the main challenges in computer graphics is to achieve a simulation that yields realistic behaviour and preserves the physical properties of the hair strands.

However, simulating hair in real-time is a non-trivial task due the high computational cost associated to the large number and detailed geometry of the strands. A wide range of methods have been developed to represent hair with varying degrees of complexity, although the research is mainly focused on modelling the behaviour of individual strands and their interactions.

Our work is specially motivated by the need of a fast simulation method for real-time applications that models the dynamics of individual hair strands while considering three key features of the hair: inextensibility, shape preservation and strand-strand interaction. In recent work, a simulation scheme proposed by Müller et al. [MKC12] guarantees inextensibility in a single simulation step, but it doesn't address shape preservation. Alternatively, Han and Harada [HH12] propose a method able to maintain stylized hair, but requires multiple iterations to achieve a stable convergence.

In this paper, we present a simulation method that simultaneously addresses both inextensibility and shape preservation, achieving convergence in a single iteration per step. This scheme is coupled with a Lagrangian approach to com-

DOI: 10.2312/ceig.20151194

pute hair-hair interactions, similar to the one by Hadap et al. [HMT01]. Moreover, a GPU implementation scheme is proposed to exploit the massive parallelism these devices offer. The resulting approach is fast and stable, capable of simulating and rendering thousands of hair strands in real-time.

2. Related Work

Traditionally, human hair has been represented in real-time applications as a simple set of triangle meshes with alpha-blended textures [KH00]. While results are visually appealing, these are not capable of conveying the actual hair behaviour and are not suitable for long hair styles nor fur. Some methods focusing on fur, render multiple extruded mesh shell layers [LPFH01], however, these approaches have very limited simulation capabilities and the trick becomes very apparent on close-up.

The simulation of one-dimensional curves such as hair strands has been extensively studied in computer graphics but these methods have not been commonly used for real-time applications due to their high computational cost. However, recent efforts achieve interactive frame rates thanks to the computing capabilities of the GPU hardware.

One of the first attempts to simulate the dynamics of hair strands used a mass-spring system [RCT91], but did not guarantee stiff strands and often lead to numerical instabilities. Techniques such as implicit integration [BW98] and projection methods [AUK92] have been used to address this problem. Although these methods allowed larger timesteps, they might require correction to limit the stretching of the springs [BFA02]. [BAC*06] and [Had06] addressed the inextensibility issue by using reduced coordinates. [BWR*08] proposed using the "fast projection" from [GHF*07] to enforce inextensibility of rods using Lagrange multipliers. Spillmann and Harders [SH10] use constraints enforcing the inextensibility of the strands.

The Position Based Dynamics method (PBD) introduced by [MHHR07] is able to solve the constraint dynamics problem iteratively, suitable for real-time applications due its stability and efficiency. However, the PBD method inherits the poor convergence from its Gauss-Seidel iterative solver. To improve this, Han and Harada [HH13] propose a tridiagonal matrix formulation able to achieve convergence in two iteration steps.

On the other hand, Müller et al. [MKC12] presented the Dynamic Follow-the-Leader (DFTL) method, based on the Follow-the-Leader (FTL) algorithm used for quasi-static knot tying simulation [BLM04] and borrowing concepts from the PBD paradigm. The DFTL method achieves inextensibility in a single step at the cost of introducing numerical damping. Our work expands from this technique by including additional projections to simultaneously address hair preservation in the DFTL solver pass.

Another important challenge in hair simulation is to maintain the hair structure along time. Approaches based on shape matching are proposed by [RKN10] and [MC11] to simulate complex hair styles. Han and Harada [HH12] introduced local and global shape constraints to simulate styled hairs in real-time applications. However, to achieve stable convergence, these constraints needed to be solved in an iterative manner. We propose an integration with the DFTL solver scheme achieving convergence in a single projection step while keeping the simulation stable.

To achieve a realistic behaviour, the interaction of neighbouring strands should also be taken into account. Plante et al. [PCP01] introduced layered wisps to represent hair clusters that collide and slide each other. This method was further extended by Ward and Lin [WL03] and by Bertails et al. [BKCN03]. However, these methods model the local dynamics of a strand and don't consider the global interaction between them. [DBDB11] and [KTS*14] proposed algorithms for simulating contacts with dry friction between strands, focusing respectively on reduced and maximal coordinates models.

Models based on continuum descriptions have also been proposed using both the Eulerian [PHA05] and the Lagrangian formalism [HMT01, MSW*09]. In our work we follow the Lagrangian approach by using the simulation scheme for fluid dynamics presented by Müller et al. [MCG03]. The choice of a Lagrangian description is motivated by the simplicity of it's implementation and direct coupling with the local dynamics due to their shared representation of the strand data.

3. Simulation Method

In our approach, strands are modelled as polylines, and each of its vertices is represented as a particle. The simulation is performed in two stages, as outlined in Algorithm 1. The first stage computes pressure and viscosity forces due to hair-hair interaction (global dynamics), using Smoothed Particle Hydrodynamics (SPH). The second stage computes the local strand dynamics by using a position based approach that ensures inextensibility while maintaining hair's shape.

In this section we present this approach. For the sake of clarity, in our exposition we start with the description of the local dynamics, where we introduce the notation and the geometrical representation of the system. Then, we proceed with the description of the global dynamics.

3.1. Inextensibility and Shape Preservation (Local Dynamics)

We follow a similar scheme to the Position Based Dynamics (PBD) paradigm. Each simulation frame begins by computing a prediction step considering external and mass forces as

$$\mathbf{x}_i^* \leftarrow \mathbf{x}_i + \mathbf{v}_i \Delta t + \mathbf{a}_i \Delta t^2, \tag{1}$$

Algorithm 1: Simulation outline

```
/* Global Dynamics */
1  for each Particle $p_i$ in simulation do
2      compute spatial hash of $p_i$
3  end
4  count sort spatial buckets;
5  for each Particle $p_i$ in simulation do
6      rearrange $\mathbf{x}_i$ and $\mathbf{v}_i$ to temporary buffer;
7      find k-nearest neighbours;
8      compute density and density gradient at $p_i$;
9      compute pressure and viscosity forces at $p_i$;
10     add forces to unarranged $\mathbf{a}_i$;
11 end
/* Local Dynamics */
12 for each Particle $p_i$ after root in simulation do
13     compute external forces (wind, etc.);
14     predict position $\mathbf{x}_i^*$;
15 end
16 for each Strand level $l$ after root do
17     for each Particle $p_i$ in $l$ do
18         apply shape constraints;
19         apply inextensibility constraint;
20         handle collisions;
21         update coordinate frames;
22     end
23 end
24 for each Particle $p_i$ after root in simulation do
25     update velocity $\mathbf{v}_i$ and apply correction term;
26     update positions $\mathbf{x}_i \leftarrow \mathbf{x}_i^*$;
27 end
```

where $\mathbf{x}_i^*$ is the predicted position of the particle i, $\mathbf{x}_i$ is its current position, $\mathbf{v}_i$ and $\mathbf{a}_i$ are the velocity and acceleration respectively, and Δt is the timestep.

Then, particle positions are updated to a constraint compliant state:

$$\mathbf{x}_i^* \leftarrow SolveConstraints(\mathbf{x}_i^*). \tag{2}$$

The resulting increment is used to update the velocities and positions of the particles:

$$\mathbf{v}_i \leftarrow \frac{\mathbf{x}_i^* - \mathbf{x}_i}{\Delta t}, \tag{3}$$

$$\mathbf{x}_i \leftarrow \mathbf{x}_i^*. \tag{4}$$

As our interest is to enforce the hair strand inextensibility while preserving their shape, we need to model the dynamics of the strands with constraints that provide the desired behaviour.

On the one hand, inextensibility can be achieved by formulating a distance constraint that ensures that all particles in the strand are located at a rest distance from their predecessors. From this constraint, a position update is obtained

$$\Delta\mathbf{x}_i = \left(1 - \frac{l_i}{|\mathbf{x}_{i-1} - \mathbf{x}_i|}\right)(\mathbf{x}_{i-1} - \mathbf{x}_i), \tag{5}$$

where $\mathbf{x}_{i-1}$ is the position of the particle predecing i and l_i is the rest distance between both.

On the other hand, shape preservation can be achieved employing shape-matching constraints by guiding the particles to goal targets that keep specific hair styles. Han and Harada [HH12] propose global and local constraints to achieve this. To this end, they employ a multi-body serial chain representation of the strand in order to maintain the local coordinate systems of each particle. This way, they are able to formulate two constraints in the form:

$$\Delta\mathbf{x}_i = s_g \cdot (p_i - \mathbf{x}_i^*), \tag{6}$$

$$\Delta\mathbf{x}_i = s_l \cdot (p_i^{i-1} - \mathbf{x}_i^*), \tag{7}$$

where p_i and p_i^{i-1} are the initial positions in the local and preceding particle coordinate systems respectively, transformed to world coordinates. s_g and s_l are the two scalar coefficients that control the stiffness of the constraints.

However, solving these constraints in a serial fashion suppose a situation where the preceding particles have infinite mass, which yields strange behaviour and introduces excessive oscillation and instability. Han and Harada [HH12] address this issue through the classical PBD scheme, by symmetrizing the effect of the projection for both the particle and its predecessor. This would violate the imposed constraints over preceding particles, so, in order to achieve stable convergence it's necessary to solve them in an iterative manner.

Instead of this scheme, our approach follows the solution proposed by [MKC12] for the DFTL method. Müller et al. solves the inextensibility constraint in a single iteration per step with the introduction of a velocity correction term. We expand over this work by modifying the correction term to take into account the shape preservation constraints. Therefore, we are able to avoid the iterative projection and reduce the instabilities produced by the serial solver, at the cost of introducing additional numerical damping.

The estimated positions are then modified to follow the projection of the constraints:

$$\mathbf{x}_i^* \leftarrow \mathbf{x}_i^* + \Delta\mathbf{x}_i^{gbl}, \tag{8}$$

$$\mathbf{x}_i^* \leftarrow \mathbf{x}_i^* + \Delta\mathbf{x}_i^{lcl}, \tag{9}$$

$$\mathbf{x}_i^* \leftarrow \mathbf{x}_i^* + \Delta\mathbf{x}_i^{inex}, \tag{10}$$

where $\Delta\mathbf{x}_{i+1}^{gbl}$, $\Delta\mathbf{x}_{i+1}^{lcl}$ and $\Delta\mathbf{x}_{i+1}^{inex}$ are the projections of the global shape preservation, local shape preservation and inextensibility respectively. These are defined as:

$$\Delta \mathbf{x}_i^{gbl} = s_g \cdot (\mathrm{p}_i - \mathbf{x}_i^*), \tag{11}$$

$$\Delta \mathbf{x}_i^{lcl} = s_l \cdot (\mathrm{p}_i^{i-1} - \mathbf{x}_i^*), \tag{12}$$

$$\Delta \mathbf{x}_i^{inex} = \left(1 - \frac{l_i}{|\mathbf{x}_{i-1}^* - \mathbf{x}_i^*|}\right)(\mathbf{x}_{i-1}^* - \mathbf{x}_i^*), \tag{13}$$

following the same notation employed in equations (5), (6) and (7).

Velocity estimation is changed to take into account the modified correction term:

$$\mathbf{v}_i \leftarrow \frac{\mathbf{x}_i^* - \mathbf{x}_i}{\Delta t} - s_d \frac{\Delta \mathbf{x}_{i+1}^{lcl} + \Delta \mathbf{x}_{i+1}^{gbl} + \Delta \mathbf{x}_{i+1}^{inex}}{\Delta t}, \tag{14}$$

where s_d is the scalar damping coefficient introduced in [MKC12] whose value lies between $[0, 1]$. This scalar parameter controls how much energy is reintroduced into the system to hide the uneven mass distribution. When $s_d = 1$, this uneven distribution is completely compensated. However, this comes at the cost of additional numerical damping. Therefore, Muller et al. suggest to use values near to 1 in order to reduce this damping while keeping the visual artifacts barely noticeable. A more exhaustive analysis of this term can be found in [MKC12].

3.2. Hair-hair interaction (global dynamics)

Computing the interaction between hair strands poses a challenge due to the high complexity of their physical interactions. The computational cost of geometrical methods increases with the number of simulated strands, difficulting the task of achieving interactive frame rates.

For this reason, we decide to address the problem of hair-hair interaction by considering the hair volume as a fluid continuum like [HMT01, PHA05]. This way, the field forces are computed using the SPH discretization following the scheme proposed in [MCG03]. Moreover, by using a Lagrangian description to compute interactions, we can share the particle-based representation of hair in both the local and global dynamics stages and couple them in a very natural manner.

The strands are subject to the forces described by the Navier-Stokes equation for fluid dynamics (15)

$$\rho\left(\frac{\partial \mathbf{v}}{\partial t} + \mathbf{v} \cdot \nabla \mathbf{v}\right) = -\nabla \mathrm{p} + \rho g + \mu \nabla^2 \mathbf{v}, \tag{15}$$

where ρ, $\mathbf{v}$ and p are a density, velocity and pressure fields respectively. g is an external force density field and μ is the viscosity of the medium.

Developing the equation using the SPH method, yields the three forces that drive the dynamics of the fluids: pressure ($-\nabla \mathrm{p}$), external forces (ρg) and viscosity ($\mu \nabla^2 \mathbf{v}$), which are computed following the derivation described in [MCG03].

The fluid pressure force allows us to model a repulsion term to achieve volumetric hairstyle, producing an impression of finite hair thickness. Also, the viscosity force introduces a damping term that models the friction produced due to the intersection between hair strands. Since the external forces are considered during the local stage, they are dropped from these computations.

4. Implementation

Our implementation runs entirely on the GPU using Compute Shaders. During initialization, the strand data is loaded into system memory and rest-state values are computed, such as rest distances, initial positions and coordinate frames. The interested reader can refer to [HH12] for the initialization of the strand coordinate frames.

As outlined in Algorithm 1, the global dynamics stage is computed first. The forces originated from the hair-hair interaction are transferred as accelerations to the local stage. Then, the particle positions are predicted and projected into the shape and inextensibility constraints. Finally, the new velocities are estimated and their locations updated.

In order to process a large number of hair strands, choosing an adequate memory layout that allows to reach the maximum occupancy and memory coalescence possible is very important to fully exploit the massive parallelism of GPU devices. Therefore, we need to carefully choose the strategies that are able to achieve a highly efficient simulation for each stage.

The hair local dynamics are computed with particle-level parallelism (lines 12 and 24 in Algorithm 1), except for the DFTL constraint solver (line 14 in Algorithm 1) due to the existence of a serial dependency with the results of preceding constraint projections. For this part of the solver, strand-level parallelism can be used in order to accelerate the computations. However, this approach leads to a lesser usage of the GPU capabilities.

Instead, we arrange the particle data by grouping them following the local particle index. This ensures that the particles can be executed on batches (eg. the particles with index 0, then 1, and so on) satisfying the serial dependency. In our tests, this yields an increment of performance by a factor of 2x-7x over strand-level parallelism, depending on the complexity of the simulation. Table 2 compares the two strategies under different situations.

The hair global dynamics also employs particle-level parallelism (lines 1 and 5 in Algorithm 1). However, neighbouring particle search is the bottleneck of this approach. Therefore, the memory layout proposed for the local dynamics is not suitable for this stage as it doesn't guarantee coalescent access to neighbouring particle data.

To address this, we follow a spatial subdivision strategy similar to [Gre08] by discretizing the location of the particles into an spatial hash grid (line 2 in Algorithm 1). The particle

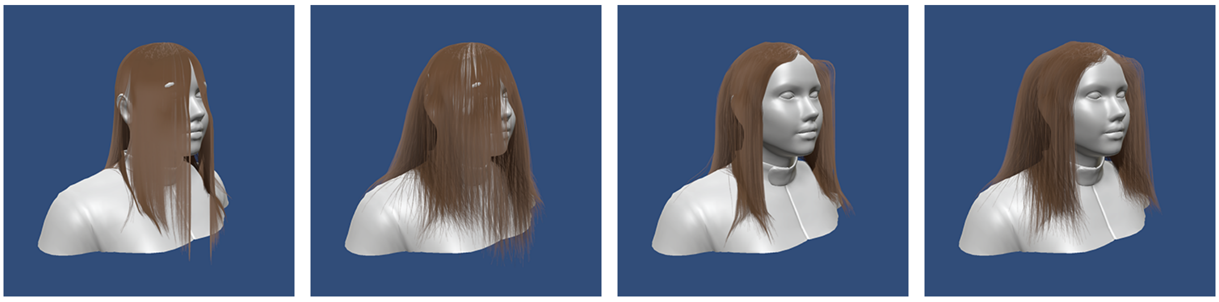

Figure 2: *Comparison of the behaviour of each technique. From left to right: (a) only DFTL, (b) DFTL with only hair-hair interaction, (c) DFTL with only shape preservation and (d) DFTL with hair-hair interaction and shape preservation. Note that in (b), hair-hair interaction grants volume but does not preserve any kind of hair style while in (c), hair preservation can reproduce volume but results in highly stiff motion. A combination of both (d) produces the most visually appealing results.*

state variables are then grouped by bucket index and stored into temporary buffers to enhance the memory coalescence and thread coherence (lines 4 and 6 in Algorithm 1). Additionally, we build a rough list of the k-nearest neighbours for each particle and use it to further accelerate the computation of the interaction forces (line 7 in Algorithm 1).

Finally, this stage is coupled with the local dynamics stage by storing the forces originated from the interaction as accelerations. These accelerations are stored at the particle's location before the rearrangement (line 10 in Algorithm 1).

5. Results

The proposed scheme has been implemented in the Unity^{TM} game engine using DirectX11 Compute Shaders. For rendering, we have implemented the Scheuermann shading model and used single strand-interpolation to raise the number of apparent strands. Moreover, transparency is addressed by using Per-Pixel Linked List Order-Independent Transparency. We have developed several demo scenes to exhibit the different models of behaviour. All the presented images are snapshots taken from these applications.

Figure 1 shows two example uses of the method; a mannequin weaving her hair, and a running horse. The mannequin demo shows long human hair with hair-hair interaction in which shape is preserved. The rightmost picture in Figure 1 exhibits an animated non-human character. Self-collisions are addressed by using geometric primitives.

Figure 2 illustrates the contribution of hair-hair interaction and shape preservation constraints. Repulsion grants hair volume but has problems overcoming the gravity in some areas like the scalp (second image from the left). Preservation constraints can reproduce hair volume but results in highly stiff motion (third image from the left). A calibrated combination of the two methods is able to overcome these limitations, as seen in the rightmost image in Figure 2.

Figure 3 shows the simulation of short fur with our method. Self-collisions are handled with the use of Signed Distance Fields. Figure 4 demonstrates how rendering techniques can be used along shape preservation to convey different hair styles. Figure 5 exhibits how the method is suitable for it's use on animals or stylized characters.

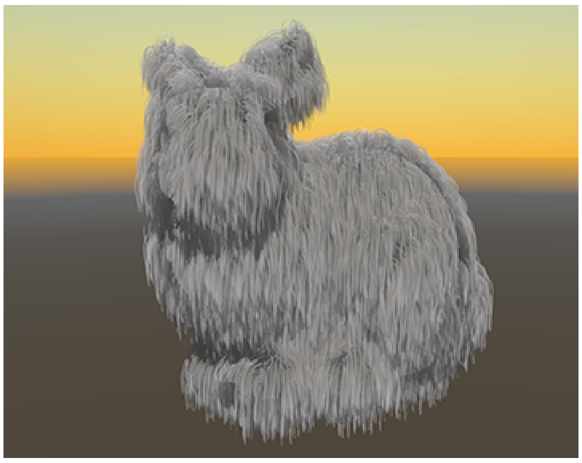

Figure 3: *The method applied to the simulation of short fur.*

We have observed that hair with low preservation and repulsion yields realistic behaviour. Overpreservation and underpreservation produce stiff motion and hair volume loss respectively, although we have also observed that hair-hair repulsion is able to compensate the later issue. We've also found that cases where the strands are largely tessellated, the oscillations produced by the shape preservation constraints becomes more noticeable as the error accumulates the further we descend into the chain.

Since the most important aspect of a stylized hair is preserving the shape of the strands around the scalp and face, we can circumvent this issue by adjusting the preservation coefficients s_g and s_l of (6) and (7) per particle and linearly

Scenes	Strands	Vertices	Colliders	Average FPS		
				DFTL	DFTL + Shape Preserv.	DFTL + Preserv. + Hair-Hair
Mannequin A	15,000	349,232	15	130.3	124.0	21.5
Mannequin B	500	8,000	15	174.2	172.6	147.4
Horse Louis	3,358	32,041	18	115.0	108.5	83.2
Bunny	2,902	34,824	1	169.2	164.0	126.1
Hectroll	496	5,710	1	315.0	305.3	270.1

Table 1: *Performance results for different scenes and method combinations. Scenes "Mannequin A" and "Mannequin B" correspond to the mannequin with different levels of detail, scene "Horse Louis" to the horse animation, scene "Bunny" to fur animation and scene "Hectroll" to a highly preserved doll hair. The following 3 columns show the number of element involved. The last three columns show the average FPS during the simulation as provided by Unity.*

fading their values in respect to the distance from the root. The loss of shape at the hair tips can be compensated with the aforementioned rendering techniques.

Another limitation of the proposed method is some lack of dynamism in some simulation scenarios, which is due to numerical damping introduced by the correction term. Adjusting the scalar damping coefficient s_d can reduce this issue but it's not completely mitigated. However, as characters are constantly introducing energy into the system (either from their motion or from external phenomena), this might not be a significant drawback.

In our test scenes, the scalar coefficient s_d grants stability to the simulation with values around 0.9. The local and global shape preservation coefficients offer the best results when they are below 0.3 and 0.1 respectively or the strands might become too stiff and converge to the initial shape. The repulsion and viscosity coefficients are within $[0, 0.5]$ and $[0, 200]$ range respectively, and the time step is $0.016s$. These parameters can be quickly calibrated by trial & error to obtain the desired behaviour.

Table 1 contains the results of some performance tests. All the tests have been executed on an Intel® Core^{TM} i7-3770 CPU @ 3.40 GHz with a NVIDIA GeForce GTX 650 GPU. We have measured simulation and rendering frame rates for the head, the horse and fur with the different features enabled. The measurements reveal that hair shape preservation constraints yield little overhead over the regular DFTL method, while hair-hair interaction becomes the most important bottleneck. Finding an optimal smoothing length, along with a fast neighbour search algorithm, are the key to reach interactive frame rates. The results confirm that the method can simulate the three desired features with thousands of inextensible hair strands at interactive rates.

Table 2 compares the performance of strand-level and particle-level parallelism for the DFTL constraint projection, highlighting the benefits of the adopted strategy. The improvement in coalescent memory access and occupancy yields a significant boost in the simulation's performance, specially in highly complex scenes. According to these data, our method is comparable, in terms of performance, to the work by Müller et al. [MKC12] while we add shape preservation to the simulated behaviours in comparison to their work.

Strands	Vertices	Average FPS		Speed Up
		SLP	PLP	
1,024	8,190	330	570	1.72
1,024	32,768	250	520	2.08
4,096	131,072	84.8	370	4.36
4,096	262,144	40.1	223	5.56
16,384	1,048,576	10	74	7.40

Table 2: *Performance comparison between strand-level parallelism (SLP) and particle-level parallelism (PLP). The first two columns show the number of strands and particles for every test. The two following columns show the average FPS. The last column shows the speed up achieved with PLP respect to SLP.*

6. Conclusion

In this paper we have addressed three relevant features in hair simulation; hair inextensibility, shape preservation and hair-hair interaction. Our constraint projection strategy combines the DFTL for inextensibility with shape preservation constraints, providing convergence in a single iteration. Hair-hair interaction is achieved using the SPH formalism.

The resulting method is able to simulate the dynamics of thousands of strands in real-time with visually satisfying results. To improve the performance of our implementation, we execute the algorithm on the GPU. Per particle parallelism has been enabled with a proper data arrangement, obtaining an speed up compared to strand-level parallelism.

We demonstrated the feasibility of its integration within an existing game development platform and its flexibility to convey realistic and stylized behaviours in different real-world scenarios.

As future research, we would like to study the impact of the introduced numerical damping on the energy of the system. Furthermore, it could be interesting to couple the method with other particle systems, such as water.

Acknowledgements

We would like to thank AMD for the TressFX sample hair guides and the Ruby model, Cem Yuksel for additional hair guides, Ilove3D2 for the female head model and Dootsy Development for the animated horse model used in Figure 1. Also we would like to thank the reviewers for their kind comments and suggestions, which have been very useful for improving our work.

References

[AUK92] Anjyo K.-i., Usami Y., Kurihara T.: A simple method for extracting the natural beauty of hair. In *SIGGRAPH'92* (1992), vol. 26, pp. 111–120. 2

[BAC*06] Bertails F., Audoly B., Cani M.-P., Querleux B., Leroy F., Lévêque J.-L.: Super-helices for predicting the dynamics of natural hair. In *ACM Transactions on Graphics (TOG)* (2006), vol. 25, ACM, pp. 1180–1187. 2

[BFA02] Bridson R., Fedkiw R., Anderson J.: Robust treatment of collisions, contact and friction for cloth animation. In *SIGGRAPH'02* (2002), vol. 21, pp. 594–603. 2

[BKCN03] Bertails F., Kim T.-Y., Cani M.-P., Neumann U.: Adaptive wisp tree: a multiresolution control structure for simulating dynamic clustering in hair motion. In *ACM SIGGRAPH/Eurographics symposium on Computer animation* (2003), pp. 207–213. 2

[BLM04] Brown J., Latombe J.-C., Montgomery K.: Real-time knot-tying simulation. *The Visual Computer 20*, 2 (2004), 165–179. 2

[BW98] Baraff D., Witkin A.: Large steps in cloth simulation. In *SIGGRAPH'98* (1998), pp. 43–54. 2

[BWR*08] Bergou M., Wardetzky M., Robinson S., Audoly B., Grinspun E.: Discrete elastic rods. In *ACM Transactions on Graphics (TOG)* (2008), vol. 27, ACM, p. 63. 2

[DBDB11] Daviet G., Bertails-Descoubes F., Boissieux L.: A hybrid iterative solver for robustly capturing coulomb friction in hair dynamics. In *ACM Transactions on Graphics (TOG)* (2011), vol. 30, ACM, p. 139. 2

[GHF*07] Goldenthal R., Harmon D., Fattal R., Bercovier M., Grinspun E.: Efficient simulation of inextensible cloth. *ACM Transactions on Graphics (TOG) 26*, 3 (2007), 49. 2

[Gre08] Green S.: Cuda particles. *nVidia Whitepaper 2*, 3.2 (2008), 1. 4

[Had06] Hadap S.: Oriented strands: dynamics of stiff multi-body system. In *Proceedings of the 2006 ACM SIGGRAPH/Eurographics symposium on Computer animation* (2006), Eurographics Association, pp. 91–100. 2

[HH12] Han D., Harada T.: Real-time hair simulation with efficient hair style preservation. In *VRIPHYS'12* (2012), The Eurographics Association, pp. 45–51. 1, 2, 3, 4

[HH13] Han D., Harada T.: Tridiagonal matrix formulation for inextensible hair strand simulation. In *VRIPHYS'13* (2013), pp. 11–16. 2

[HMT01] Hadap S., Magnenat-Thalmann N.: Modeling dynamic hair as a continuum. In *Computer Graphics Forum* (2001), vol. 20, pp. 329–338. 2, 4

[KH00] Koh C. K., Huang Z.: *Real-time animation of human hair modeled in strips*. Springer, 2000. 2

[KTS*14] Kaufman D. M., Tamstorf R., Smith B., Aubry J.-M., Grinspun E.: Adaptive nonlinearity for collisions in complex rod assemblies. *ACM Transactions on Graphics (TOG) 33*, 4 (2014), 123. 2

[LPFH01] Lengyel J., Praun E., Finkelstein A., Hoppe H.: Real-time fur over arbitrary surfaces. In *Interactive 3D graphics* (2001), pp. 227–232. 2

[MC11] Müller M., Chentanez N.: Solid simulation with oriented particles. In *SIGGRAPH'11* (2011), vol. 30, p. 92. 2

[MCG03] Müller M., Charypar D., Gross M.: Particle-based fluid simulation for interactive applications. In *ACM SIGGRAPH/Eurographics symposium on Computer animation* (2003), pp. 154–159. 2, 4

[MHHR07] Müller M., Heidelberger B., Hennix M., Ratcliff J.: Position based dynamics. *Journal of Visual Communication and Image Representation 18*, 2 (2007), 109–118. 2

[MKC12] Müller M., Kim T.-Y., Chentanez N.: Fast simulation of inextensible hair and fur. In *VRIPHYS'12* (2012), The Eurographics Association, pp. 39–44. 1, 2, 3, 4, 6

[MSW*09] McAdams A., Selle A., Ward K., Sifakis E., Teran J.: Detail preserving continuum simulation of straight hair. In *ACM Transactions on Graphics* (2009), vol. 28, p. 62. 2

[PCP01] Plante E., Cani M.-P., Poulin P.: *A layered wisp model for simulating interactions inside long hair*. Springer, 2001. 2

[PHA05] Petrovic L., Henne M., Anderson J.: Volumetric methods for simulation and rendering of hair. *Pixar Animation Studios* (2005). 2, 4

[RCT91] Rosenblum R. E., Carlson W. E., Tripp E.: Simulating the structure and dynamics of human hair: modelling, rendering and animation. *The Journal of Visualization and Computer Animation 2*, 4 (1991), 141–148. 2

[RKN10] Rungjiratananon W., Kanamori Y., Nishita T.: Chain shape matching for simulating complex hairstyles. In *Computer graphics forum* (2010), vol. 29, pp. 2438–2446. 2

[SH10] Spillmann J., Harders M.: Inextensible elastic rods with torsional friction based on lagrange multipliers. *Computer Animation and Virtual Worlds 21*, 6 (2010), 561–572. 2

[WL03] Ward K., Lin M. C.: Adaptive grouping and subdivision for simulating hair dynamics. In *IEEE Computer Graphics and Applications'03* (2003), pp. 234–243. 2

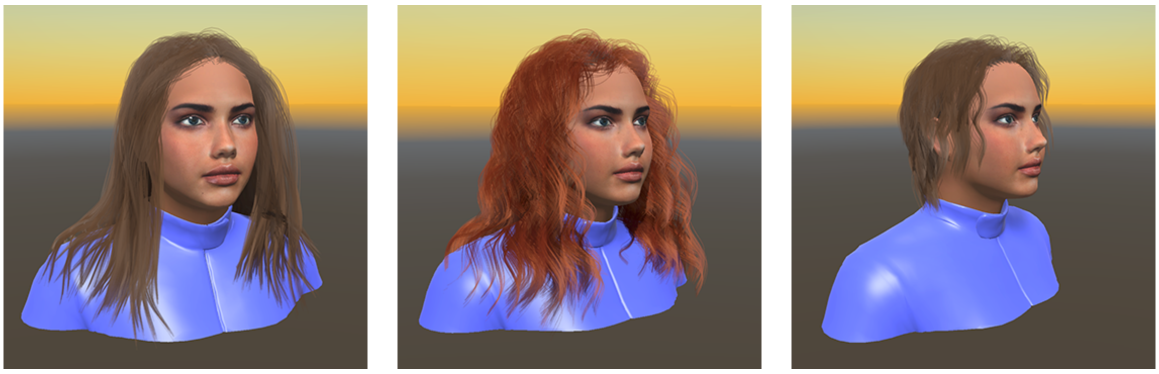

Figure 4: *By using render techniques along with shape preservation, we are able to convey different hairstyles.*

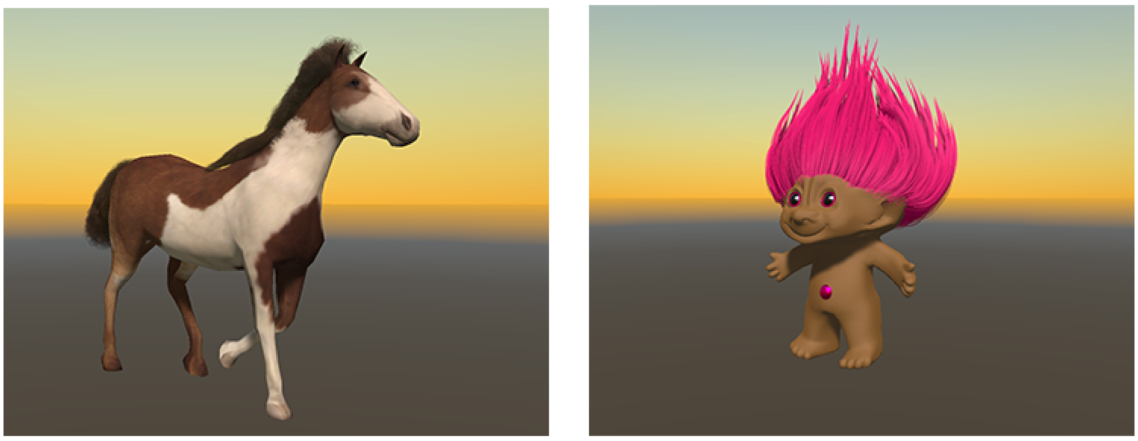

Figure 5: *The proposed method is also suitable for it's use on animals and stylized characters.*