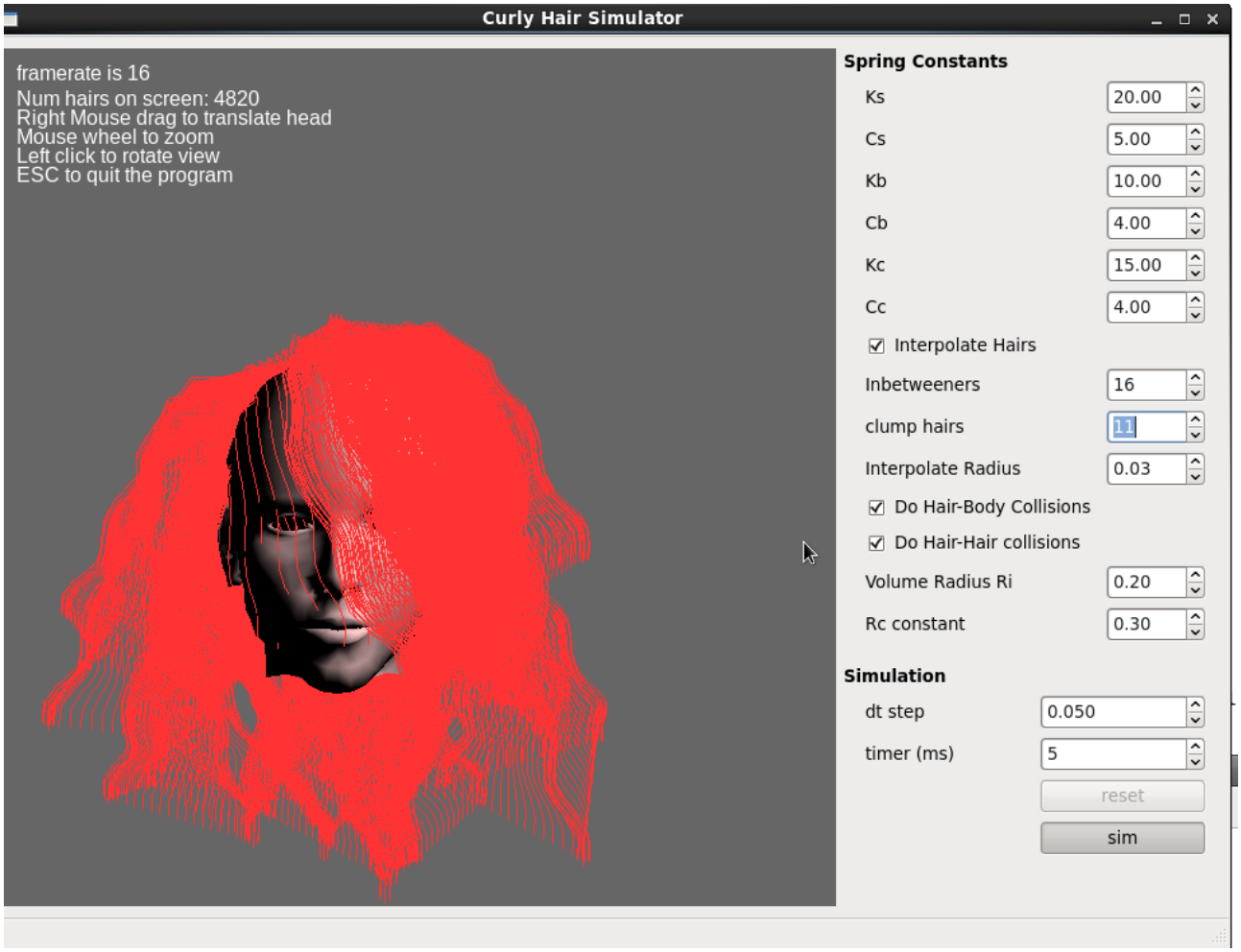
**Computer Generated Imagery Techniques**
Assignment Report – May 2013

# Simulation of curly hair




student ID : i7266699
student name : Fabio
student surname : Turchet

## 1. Introduction

For my assignment I implemented a paper by Pixar called “Artistic Simulation of Curly Hair ” which explains the system created for the incredible hair simulation of the characters in the recent movie “Brave”. (Iben et al. 2012) The system was initially developed just for the main character (Merida) , which posed for sure a big challenge having long curly hair that had to behave properly in a multitude of situations and movements. The choice of Pixar was in fact to avoid an exact physical model and searching instead a trade-off between believability and full artistic control. For example the artist had the ability to control the stiffness of the springs of even a single edge of the strand. The system worked extremely well in production and once set up per character allowed the simulation almost without additional tweakings for every shot. Even though the system was developed and customized for curly hair, it was adapted without problems to a variety of hair styles or to characters with straight hair. All of this made this paper and its techniques really attractive for a possible implementation. I will explain here all the steps taken to create the system , the problems I encountered and the results achieved.

## 2. Previous work

Selle et al. (2008) created a remarkable work in conjunction with ILM to produce a model that simulates every single hair, compared to other methods where for efficiency reasons a sparse model is more convenient instead (Bando 2003) which is suggested at the end of their paper as a good solution for real time applications. Selle et al.(2008) introduce a new spring called “tetrahedral altitude spring” to deal with torsion issues and maintain volume. They also add strain limitation and stiction to their model, which produces believable results.

Sparse models have been presented for example by Chang et al. (2002) in which they show a model to recover hair shape in case of movement and improve collision quality with the use of triangle strips between hair pairs.

Tae-Yong (2004) from Rythm and Hues , after explaining why the rigid joints approach is not the right way to go, show how the an implicit method can be used and how the length of the hair can be preserved by applying a non linear post-correction. Finally, a solution is proposed that uses a predictor-detector method with a two-pass implicit filtering.

The vast majority of models nowadays uses a mass-spring approach, at least at its basis.
As suggested by Catto (2011) soft constraints could be preferred to springs instead.

## 3. Types of spring

A single hair can be modelled as a series of particles disposed in a curvilinear fashion and adjacency-connected. Each particle can have one and only one incoming connection and one or zero outgoing edges.
The system makes use of 3 different springs, or constraints. The first and basic are the structural ones that is those connecting one particle to the next. The second type are the bending springs, preventing undesired bending. The third type are the core springs, needed to avoid curl shape loss.

### *3.1 Stretching springs*

These are the classic linear springs defined by the well known Hook' law F = -kx.

The general formula is

$$\boldsymbol{f}_s(k_s, c_s)_i = k_s(\| \boldsymbol{e}_i \| - \| \bar{\boldsymbol{e}}_i \|)\hat{\boldsymbol{e}}_i + c_s(\Delta \boldsymbol{v}_i \cdot \hat{\boldsymbol{e}}_i)\hat{\boldsymbol{e}}_i \qquad (1)$$

which contains also a damping term to dissipate the energy progressively. The spring behaviour is controlled by the two coefficient Ks and Kd of elasticity and damping respectively.
In the program each force is derived by an abstract class containing a virtual method apply_force() that has to be implemented, in this case returning the computed force based on current particle positions and velocities. Hair should not stretch by default so this constant was set quite high. It is important to notice anyway that explicit methods tend to become very unstable when the stiffness is too high. When this force is calculated, has to increment both particles which connects one with sign opposite to the other as this force is applied symmetrically.

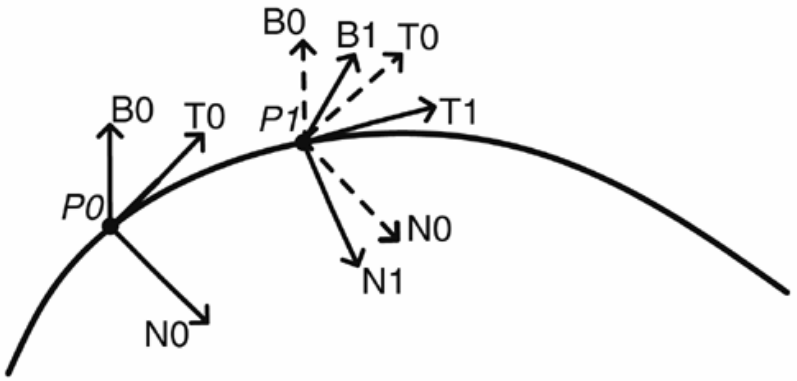
The paper indicates also that in case of excess of variation velocity over a threshold, a limit is inserted and the edges are shortened. This detail was not implemented though.

*3.2 Bending springs*

These springs have the effect of influencing the system in a way that the hair shape will restore the rest shape, in this case a curly one. The benefit of having them is visible in the fact that unwanted rotations in the curl are avoided. To achieve this, at first the hair is smoothed using an IIR filter to remove spikes and high frequencies, something common when dealing with irregular natural curly hair. The paper shows the effect of different alfa constants, in my case I found that a value of 0.07 was good enough.
The smoothing function : $\boldsymbol{d}_i = \dot{\boldsymbol{\varsigma}}(\Lambda, \alpha)_i,$

This smoothing is needed in order to perform the parallel frames transport. Essentially we move a coordinate system along a curve to perform transformation of vectors from one space to another.
In particular at every step of the simulation the reference vectors are compared and used to generate a force through a spring that will try to restore the initial curly shape. Parallel transport is explained in (Dougan 2001) , but I left the axis and angle method in favour of a simpler method (Pollard 2005) .

*Illustration 1 Image from http://www.cs.cmu.edu/afs/andrew/scs/cs/15-462/web/old/axis.gif*




Computing a reference frame from the previous frame

The parallel transport should avoid flipping, but I encountered it in some cases. Probably using the algorithm of the book should fix the problem.
The formula for the bending springs is :

$$\boldsymbol{f}_b(k_b, c_b)_i = k_b(\boldsymbol{e}_i - \boldsymbol{t}_i) + c_b(\Delta \boldsymbol{v}_i - (\Delta \boldsymbol{v}_i \cdot \hat{\boldsymbol{e}}_i)\hat{\boldsymbol{e}}_i) \qquad (4)$$

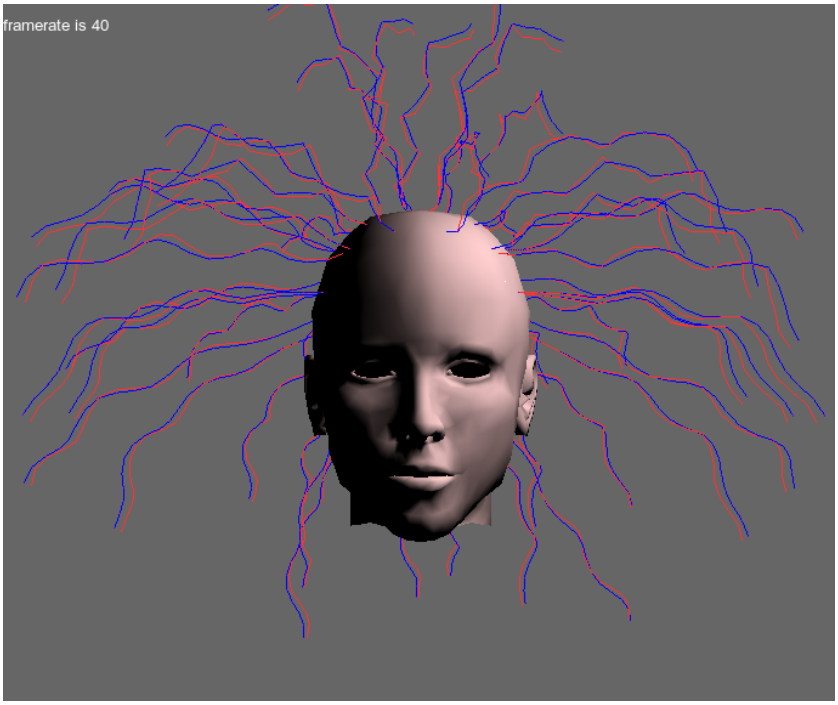



*Illustration 2 Original curves (red) and smoothed curves(blue) with alfa = 0.07*

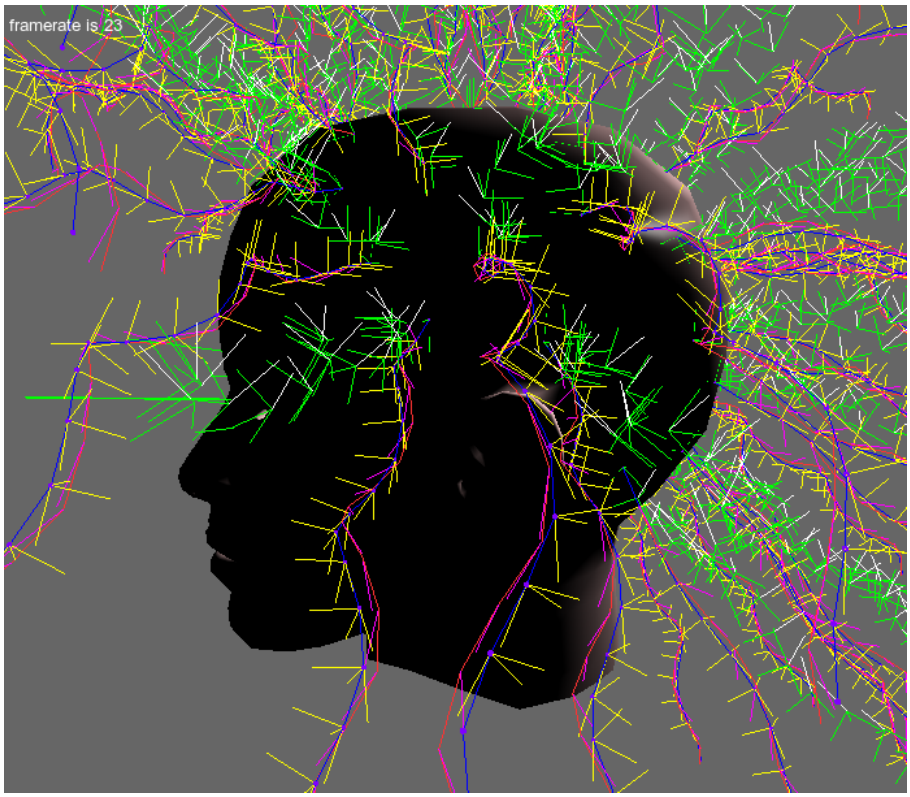



*Illustration 3 Frames in green are the fixed reference. Frames in yellow are the current ones along the smoothed curve in blue. In pink are the current target vectors t_i*

*3.3 Core springs*

In order to maintain the curly shape even more, a third type of spring is necessary : the core springs. This type will take care of avoiding stretch in case of high accelerations. Again, the method makes use of the smoothed version of the curve to calculate the length at rest of the edge vectors (b_i) in order to have the target for a spring force to aim to. The force is applied in the opposite direction to the stretch vector, but not when the hair strand is compressing, only when it is extending. If the elastic constant was set to 0 based on a simple thresholding, the behaviour would not be correct and would change abruptly. So in the paper is described how a Hermite function for cubic interpolation was used in order to ramp from 0 to the current value. This allowed also to automatically set the Kc constant without the need of the artist. This detail was not implemented anyway.
The formula of the third and last spring is shown below :

$$\boldsymbol{f}_c(k_c, c_c)_i = k_c(\| \boldsymbol{b}_i \| - \| \bar{\boldsymbol{b}}_i \|)\hat{\boldsymbol{b}}_i + c_c(\boldsymbol{\nu}_i \cdot \hat{\boldsymbol{b}}_i)\hat{\boldsymbol{b}}_i \qquad (5)$$

Notice that v_i are the smoothed velocities using the same alfa of the positions.

## 4. Collisions

There are two kinds of collisions to consider: hair-hair contacts and hair-head contacts. Pixar shows some methods that allow to improve the performance in both situations

*4.1 Hair particles pruning*

Each particle has a radius that indicates its volume in relation to its neighbours. By observing that many spheres can potentially overlap, a pruning is performed that can generally remove 50 % of the points. This means having less collision tests to check. By using an iterative method that goes along the hair starting from the root and sums the length of the edges, it is possible to implement this optimization. The related formula to do that is :

$$\sum_{i=j}^{k-1} \| \bar{\boldsymbol{e}}_i \| > s(r_j + r_k) \qquad (6)$$

where r_j and r_k are the sphere radius. As shown in the paper, the result is significative.

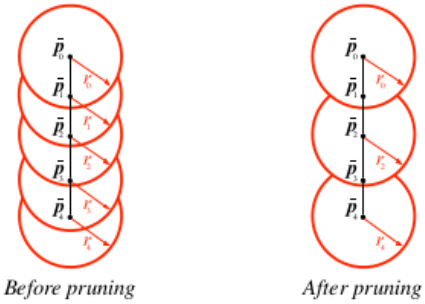



*Illustration 4 effect of particles pruning is beneficial for complexity. (image from the original paper)*

*4.2 Hair pairs pruning*

In order to execute less collision tests, it is observed how close hairs can show a good behaviour also if they are a bit far from each other because in-between hairs will participate as a bridge in order to propagate the impacts.
Based on this observation, the method proposed in the paper counts the number of contacts between each hair pair and if this number is less than a threshold, then it Is assumed those two strands are not in contact. Pixar shows also how they used a graph where each node is connected to another if an hair can collide with it. By doing this they can cluster and parallelize the simulation in different processors. It is important to note that this graph is generated at initialization time and therefore only once. In my implementation I used an sparse adjacency matrix where edges are indexed.

*4.3 Collision implementation and response*

At each step all the particles are stored in a octree for fast neighbourhood check. Pixar uses a uniform grid instead.  A collision test is done only if a particle of the original hair was not pruned and if the colliding particles are in hair indices which correspond to an edge in the adjacency matrix. If so, a simple sphere-sphere test is performed. The same test is done to test compenetrations between hairs and the head. Because of its shape, a sphere-ellipsoid test was also implemented, but the results were not as expected.
As explained in (Parent 2012 ,pp285-286) a penalty method was used to respond to the collisions, that is a spring with rest length null in order to have an equilibrium only at the point of contact.

## 5. Interpolation

In the last part of the implementation I scratched the surface of hair interpolation. The system is sparse so has a limited number of guide hairs which are simulated. At render time these hairs become more and get created in-between.
The first kind of interpolation I implemented uses linear interpolation between pairs of neighbour hairs. At initialization time a edge matrix is created in a k-nearest neighbour fashion using euclidean distance as feature. Example shown below :

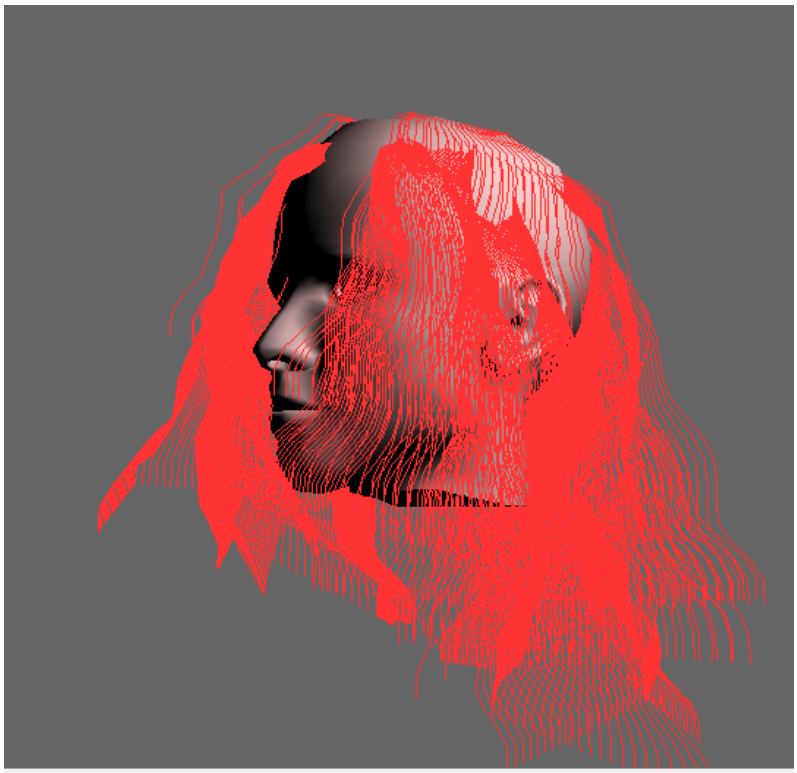


*Illustration 5 Linear interpolation of neighbours guide hairs*

The second type creates multiple hairs around a guide one , like clumps. In order to do that, an array of possible directions is defined at initialization and the hair particles are displaced by an amount specified by the user in that direction. Result is shown below :

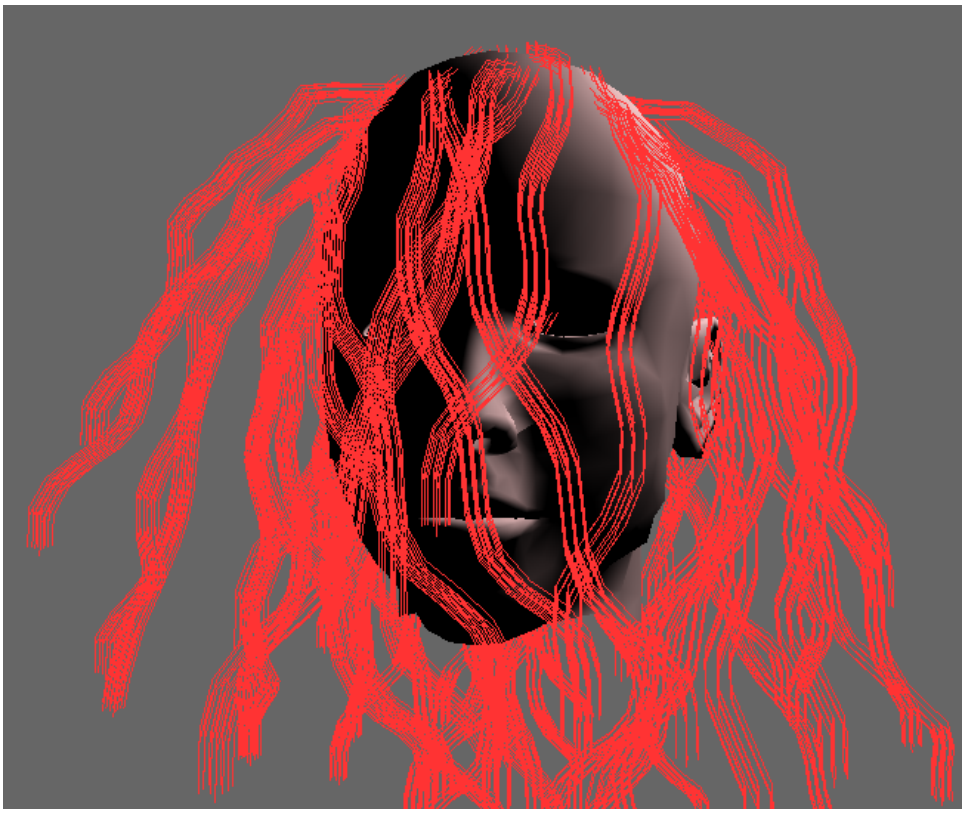


*Illustration 6 clumps around every guide hair*

In order to maximize efficiency during the communication with the GPU, a single Vertex Array Object is used which contains all the hairs in the scene so they can be drawn “at once” calling subsequent glDrawArrays() with proper offsets. Interpolated hairs are then calculated in the same draw function :

```
void HairSystem::draw(true , 20 , true , 15 , 0.03);
```

## 6. Design

As skeleton for the system I followed the useful Siggraph notes by Pixar (Baraff and Witkin, 2003 ) , translated to an OOP language (C++).
In brief the Hairsystem object creates the hairs on the scalp object passed and on the faces specified. An hair will be oriented in the direction equal to the average normal of the face vertices. The Hairsystem constructor also initializes the integrator which is an RK4 with fixed time step in this case. Other integrators should be tested like Verlet for example which should give good results without compromising efficiency too much. Pixar in its paper uses a semi-implicit Euler scheme which is also worth trying and should achieve a good stability also for stiff springs.
At each simulation step all the spring and external forces are computed and accumulated in an attribute for each particle object. At this point also the collision detections have been performed and their force contribution has been added as well. The integrator can then compute the new position

and velocity for the next frame. The external forces included in the system are gravity and drag, which is useful to increase numerical stability.

## 7. Discussion

Hair simulation is a non trivial non linear problem. Results achieved are promising but the system needs to be further developed. In particular it seems that the bending springs cause some instability issues. Also, it seems difficult to find some good values of the coefficients that can bring the right stiffness , something noticeable when dragging the head with the right mouse : there is an annoying delay that affects the solver unnaturally.
I have to say that the system presented by Pixar was not intended to be run real time at all. For this reason the number of guide hairs had to be restricted. (Merida has 579 hairs instead).
In summary the implementation can be improved in many ways (for example by interpolating on the GPU instead that on the CPU like done now), but it was rewarding being able to implement one of the latest research papers in the field used in production.

**References**


Bando, Y. , Chen,Y. , Nishita, T. 2003. Animating Hair with Loosely Connected Particles . Eurographics 2003 proceedings. 22(3)

Baraff, D. Witkin, A. 2003. Physically based modelling. Siggraph 2003 course notes.

Catto, E. 2011. Soft Constraints. Presentation at GDC 2011. Available from : http://box2d.googlecode.com/files/GDC2011_Catto_Erin_Soft_Constraints.pdf [Accessed 01/05/2013]

Chang, J. T. , Jin, J. , Yu, Y. 2002. A practical model for hair mutual interactions. ACM Siggraph 2002/ Eurographics symposium on Computer Animation proceedings , 73-80.

Dougan, C. , 2001. The Parallel Transport frame in Game Programming Gems Volume 2. edited by DeLoura, M. p. 215-217 . Charles River Media; Har/Cdr edition

Iben, H. , Meyer, M, Petrovic, L. , Soares, O. , Anderson, J. , Witkin, A. 2012. Artistic Simulation of Curly Hair. Pixar Technical Memo #12-03a . Available from : http://graphics.pixar.com/library/CurlyHairA/paper.pdf [Accessed 30 March 2013 ]

Parent, R. 2012. Computer Animation Algorithms and Techniques – Third Edition. USA : Elsevier.

Pollard, N. 2005. Graphics Project 2: Camera Movement . Available from : http://www.cs.cmu.edu/afs/andrew/scs/cs/15-462/web/old/asst2camera.html [Accessed 25 April 2013]

Selle , A. , Lentine, M. , Fedkiw R. 2008. A Mass Spring Model for Hair Simulation. ACM Siggraph 2008 papers, 64:1-64:11

Tae-Yong, K. 2004. Writing a hair dynamic solver. 2004 Game|Tech Summit and Seminars . Available from : http://www.rhythm.com/~tae/Kim_Simulation.ppt [Accessed 30/04/2013]