Creating realistic hair in Autodesk Maya

Jonas Ludwigsson

5/29/2013

Student thesis, Bachelor, 15 HE
Computer science
Creative Computer Graphics

Examiner: Torsten Jonsson
Supervisor: Fei Liu
Creating realistic hair in Autodesk Maya

by

Jonas Ludwigsson

Faculty of Engineering and Sustainable Development
University of Gävle

S-801 76 Gävle, Sweden

Email: jonas@ludwigsson.se

Abstract
This thesis work focuses on how to create realistic looking hair using only the vanilla version of Autodesk Maya. It describes two approaches, the widely used polygon-stripe based technique and the Maya built-in nHair. It also evaluates these two approaches in terms of ease of implementation, production speed and quality of final results. The conclusion is that nHair has the potential to produce realistic looking hair but contains various bugs and is not optimized at the current stage, while the polygon-stripe based approach is robust and flexible but the realism of rendering results is heavily dependent upon the skill level of artists.

Key words: 3D, Hair, Autodesk Maya, nHair, polygon stripe hair, texture maps, simulating
Table of contents

1 Introduction............................................................................................................................................. 1
   1.1 Objectives........................................................................................................................................ 2

2 Existing methods..................................................................................................................................... 2
   2.1 Creating the hair ............................................................................................................................... 2
   2.2 Simulation .......................................................................................................................................... 3
      2.2.1 Autodesk Maya nHair .................................................................................................................. 3
      2.2.2 Super-Helices .............................................................................................................................. 3
      2.2.3 Skeletons ..................................................................................................................................... 3
   2.3 Rendering ......................................................................................................................................... 3

3 Proposed methods................................................................................................................................... 4
   3.1 Polygon-stripe based hair ..................................................................................................................... 5
      3.1.1 Creating polygon-stripe based hair .............................................................................................. 5
      3.1.2 Textures ....................................................................................................................................... 5
      3.1.3 Creating the geometry .................................................................................................................. 6
      3.1.4 Simulating the hair ....................................................................................................................... 8
      3.1.5 Rendering polygon hair ............................................................................................................... 9
   3.2 nHair .................................................................................................................................................. 10
      3.2.1 Creating the curves ..................................................................................................................... 10
      3.2.2 Creating, adjusting the hair system and adding pfx .................................................................. 11
      3.2.3 Simulating nHair ....................................................................................................................... 12
      3.2.4 Rendering nHair ....................................................................................................................... 12

4 Results................................................................................................................................................... 12

5 Discussion .............................................................................................................................................. 13

6 Conclusion.............................................................................................................................................. 15

References.................................................................................................................................................. 16
1 Introduction

Creating digital hair is a difficult task. Different hair style shapes and colors are just a few of the difficult tasks that need to be accomplished. Because of the complexity of real hair, certain optimizations have to be made when creating hair with computer software. It would not be efficient to simulate every single strand of hair due to the limited computation power of most computers we can access every day, even though doing so would give artists maximum control of the shape of the hair. Therefore, various different techniques and solutions have been developed by companies and independent persons to handle this problem but not all of them are available commercially. All of them have different advantages considering complexity, speed and the needed computation power.

The basic workflow for creating hair involves creating the hair, simulating movement and lastly rendering the hair. See Fig. 1 below.

Fig. 1. A flowchart describing the general process of creating hair in 3D software.

Creating the hair refers to creating the basic shape of the hair. Depending on what technique is used for the hair it could either be solid geometry or guides for shaping the hair. This is also where textures and shaders will be applied.

The next step is the computation of the hair movement, called simulation. It is difficult to give the hair proper weight, a contributing factor for the hair to appear realistic. It is also necessary to have the hair collide with various scene objects, most importantly the character or actor itself. If this is not done properly, the illusion of realistic physics disappears. The movement is often cached to the hard drive to provide a faster playback of the scene.

The process of calculating an actual image of 3D objects is called rendering. A dedicated program often called renderer (for offline rendering) takes into account all the lights in the scene and renders the view from a specified camera. Setting the lighting appropriately and having proper shaders for the hair are essential to produce the right shine and colors.

Movie companies today often use commercial modeling software for creating 3D characters, environments and animations in their production of animated movies. One piece of software of this kind is Autodesk Maya. It is known to many graphic artists and popular for educational purposes and the like. However, instead of solely using the software’s built-in functionality and features these companies use customized software components or extensions as well when handling more complex tasks such as hair simulation and rendering. For example in the movie “Tangled” from Disney where they had to simulate extremely long hair they chose to develop a custom tool [1].
Autodesk Maya has the possibility to create, simulate and render hair using a built-in system called nHair, although it is not so straightforward. One of the reasons is Autodesk Maya comes with a few examples of hair but provides no information on how those examples were created. The documentation contains mixed information on how to use the hair system. Sometimes it claims a certain feature can or cannot be used in a certain way but in fact the documentation is simply outdated. This usually leads to misunderstandings of the hair system, which often results in unnecessary work put on workarounds or the using of obsolete techniques. Perhaps this is also why companies disregard the build-in portion of commercial tools in favor of developing their own at the cost of extra money. However, if we stick to the built-in functionality of these tools, how good will the performance and the result of the process be? This is what I want to find out with this work.

1.1 Objectives

This thesis work focuses on how realistic and efficient it can be to create, simulate and render hair using only built-in functionality from the vanilla version of Autodesk Maya. I will show two different techniques, one using simple polygon-stripe based hair and the other one using Autodesk Maya’s built-in hair system, nHair. These two will then be compared regarding simplicity, production speed, simulation time, render time, and quality of the finished product. I will also draw a conclusion of when and why the different techniques should or should not be used.

2 Existing methods

2.1 Creating the hair

To create the hair in Autodesk Maya a variety of different techniques can be used. While it is not very efficient, one could simply place every curve by hand, providing a very accurate way to style the hair. Another way is to create NURBS planes and extracting wanted curves from those planes. A third way, which is used in Autodesk Maya, is to assign an nHair system to the scalp and let it generate hair follicles on the selected geometry. The hair follicles resemble the follicles of those on a human head and are used to keep the root of the dynamic curve in place. The number of follicles decides the number of dynamic curves that will be simulated. During the simulation, various constraints can be imposed on the hair to create the hair style.

Joseph Alter has created a plugin for Autodesk Maya called Shave and a Haircut, in which you can comb the hair into place [2]. The plugin works more like traditional hair grooming by utilizing tools similar to a comb and a pair of scissors. The plugin also provides the simulation and the rendering of hair.

A simpler and faster technique for creating hair is using polygonal surfaces to form strips of hair. This is usually used when the hair must be rendered in real time, such as in video games. It is basically a line of polygons starting on the head and continuing to the tip of the hair. The desired length of the hair is decided by the length of the strip created. Many strips are used to create the entire shape of hair, the hair style.

A method, similar to the NURBS surface approach and the polygonal surface approach, has been developed by Phung Dinh Dzung [3]. It starts with the basic approach of creating polygonal hair and then a script converts the polygon hair to curves. Lastly, Paint FX (pfx) is assigned to the curves. Pfx resembles the stroke of a traditional brush, letting the curve act like a brush stroke. The pfx is generated along the curves path at render time, which enables incredible amounts of hairs to be rendered without running out of system memory [4].
2.2 Simulation

2.2.1 Autodesk Maya nHair

In Autodesk Maya the hair uses NURBS curves for its placement. The actual curves themselves are not rendered but instead control pfx. The curves are made dynamic by assigning an nHair system to them, which will provide the simulation of the hair movements [5]. nHair works by assigning a specific number of hairs to each dynamic curve. Each dynamic curve is called a hair-clump in the nHair system. This is an optimization that must be made because simulating every string of hair on an average head (110,000 on a person with brown hair) [6] is almost computationally impossible for an ordinary computer. nHair uses Maya built-in solver for dynamics called nucleus. The Autodesk Maya hair system can simulate wavy, curly and straight hair.

2.2.2 Super-Helices

This is a technique which relies on the structural and mechanical features of real hair to achieve realism [7]. It simulates hair more accurately. Instead of using regular curves, it uses helix curves, which is a curve with a constant twist on the curve similar to a coil spring. Florence Bertails et al. [7] claimed that this technique was not more computationally expensive than regular techniques. This means it is superior since the result is closer to real hair without demanding additional computational resources. Just like the built-in nHair system in Autodesk Maya this technique also allows for hair-clumping, curly and straight hair.

2.2.3 Skeletons

An approach using skeletons for simulation is called Adaptive Grouping and Subdivision for Simulating Hair Dynamics [8]. It simulates the hair using different level of details (LOD) based on the visibility of the hair, the motion it has and the distance from the viewer. It uses swept sphere volumes for collision detection and has a stable simulation method that allows for large time steps. Larger time steps make for faster simulation time but result in less accuracy. Swept sphere volumes are used for collision, which means it only has to check nearby spheres for hair-to-hair collision.

2.3 Rendering

Regarding the polygonal hair, a variety of different texture maps can be used. A texture map is required for the hair to resemble the real hair. It provides the strip with color information, often with “painted on hair”. To achieve the effect of displaying individual hair strands and each strand tapering towards the tip, an alpha map is used. This technique is described briefly in the paper Dynamic Hair Effects [9]. A more suitable shading type than the common phong shader is anisotropic shading. It shades the reflection perpendicular to the fibers of the hair, instead of spreading the light evenly as phong shading does [10].

A more accurate technique for rendering hair is to use Autodesk Maya built-in pfx. It creates hairs using the dynamic curves created by the nHair system as input for the shape. It then interpolates hair strings along the curves to add more volume. Thus, fewer curves than actual number of generated pfx hair are needed. Pfx resembles hair better than the polygon approach but cannot be used in a real time environment. The pfx renders natively in the Maya Software renderer but in order to render pfx in the more preferred NVIDIA mental ray renderer, the pfx must be converted to polygons [11]. Maya also assigns a specific shader called Hair Tube Shader to the pfx hair, which works in a similar way to the anisotropic shading. It works by shading the hair based on the camera view and the shader’s tube directions attribute [12].
3 Proposed methods

I have created a character model (Fig. 2) and the hair is applied to it. The character features an entire body and has been kept to a low amount of polygons even though the rendering will not be in real time. The reasoning behind this is the focus should be on the hair and a high polygon count on the character would slow down the simulation and the render time. I will be able to compare the two different techniques by applying hair using them on the model.

Fig. 2. The character that will be used in the process of creating the hair. Notice the relatively low polygon count.

The first thing I did was to decide what technique should be used to create the hair. I should have some kind of idea and concept of what the finished product should look like. For example, there is no point in using long curves or applying a dynamic simulation to very short hair. The concept should also describe what type of hair is going to be produced, whether the hair is going to be realistic or very stylish. I have created a concept for my hair style that gives me the opportunity to show three different types of hair in the same hair style. Fig. 3 shows the concept and contains three different parts of the hair indicated by colors. The blue part will be able to move around freely and has a loose style, while the green part is being pulled and tied by the knot at the back of the head and will almost be static. The red part should not spread out too much during simulation but still be able to move around freely.
This hair style being chosen is able to prove the strengths and the weaknesses for both of the techniques I am going to show due to the constraints imposed on the different parts of the hair.

3.1 Polygon-stripe based hair

Creating polygon hair is a very straightforward task and is very light on the computer resources. However, in order for it to look acceptable, an artist is required to understand different texture maps and possess skills of how to use them. He should also keep the hair at a low amount of polygons. This is because this technique is often used in real time environments and using high amount of polygons will significantly increase the rendering and simulation time.

3.1.1 Creating polygon-stripe based hair

One can usually divide the polygon hair into three different parts, a solid part, a normal part and a thin part with individual hair strands. Before creating the geometry for the hair it is better to create the required textures first. This is because when creating the geometry stripes it is easier to lay out the UVs on one stripe and duplicate the stripe to spare the repetitive work of having to UV-map every single strand. It is also easier if you have the texture as a reference when creating the geometry so you can replicate your concept as closely as possible.

3.1.2 Textures

Mainly two types of textures are used when creating polygon stripe based hair. One is diffuse texture (Fig. 4, left) and the other is alpha texture (Fig. 4, middle). Other
different textures can also be combined with these two to create extra realism, such as anisotropic texture (Fig. 4, right), which I have chosen to include.

![Diffuse texture (left), alpha texture (middle), anisotropic texture (right)](image)

These texture maps were created in Adobe Photoshop and imported to Autodesk Maya as 24-bit Targa images. The diffuse texture serves as the color information for the hair. In this case, it will make the rendered hair appear brown. The alpha texture contains the data of what part of the hair should be visible. The white region means fully visible and black means completely invisible. You can clearly see individual strands of hair forming towards the bottom part. Lastly, the anisotropic texture contains the information of where a specular color should be applied. Since the renderer ignores the alpha texture while applying the specular color, it does not know which part of the geometry has visible hair and which does not. If the anisotropic texture was not used, it would result in the entire hair strip to be shaded by the specular color even though no hair is visible on some parts of the geometry.

### 3.1.3 Creating the geometry

The best approach to creating the hair geometry is to start with the inner-most parts first, which is often the solid part. In my case this is the green part in Fig. 3. A simple sphere was used as the basis for this part. Next, vertices were removed for three reasons: 1. places where no geometry is needed (for example inside the head); 2. to create the desired shape; 3. to provide better resolution on the mesh where needed. Lastly, the mesh was UV-mapped to obtain the desired texture (see Fig. 5 for the result). Note how the UVs were not mapped to any transparent part of the texture because this part of the hair should appear solid.

![Solid mesh (left, represented green) is assigned an UV-map](image)
The next part to create is the ponytail (the red part in Fig. 3). A good way to start is creating a simple cylinder and then moving the vertices around to achieve the desired shape. The UV-map was mapped from the top of the texture to the bottom, which makes the ponytail taper as it grows farther away from the head. A real ponytail of this length, however, would not maintain such a cylindrical shape rather having individual hair strands or clumps of hair pointing out from it. This is because hair strings originating from different locations on the head have various lengths. It is now time to place the normal and thin parts of the hair to increase the realism. The best way to do this is create shorter polygon stripes and apply the texture accordingly. Although it is true that more stripes would make for a more detailed look, in many cases it is enough to use the alpha texture to achieve a satisfactory result. Thus, I created as few stripes as possible and relied on the alpha texture for realistic visual effects (see Fig. 6).

![Fig. 6. (Left) shows the simple cylindrical shape. (Middle) shows all the polygon stripes used and (right) shows what the finished ponytail looks like.](image)

The last piece of the hair style is the bangs (the blue part in Fig. 3). This was done by using only normal and thin type of hair. A total of seven stripes covering the required areas were created as well as the UV-maps accordingly. Using this many stripes in such a small area could pose a problem of depth sorting for a real time renderer. Such depth sorting is required to avoid intersections of different stripes. Fortunately, it is not a concern in my case as the offline renderer can handle depth sorting very well. See Fig. 7 for the result of the finished geometry with textures applied.
3.1.4 Simulating the hair

Simulating the movement of polygon hair can be done in different ways and there is no standard way of doing it. I have chosen an approach that actually utilizes the nHair physics simulation but in a much lighter way. I want to show the polygon hair in a physics simulation without having to write any custom code or scripts. My approach is similar to techniques used in simulating hair in real time environments, a dynamic joint chain (see Fig. 8). Computer graphics animations often handle animations on characters by binding the vertices of the character to joints, also called skeleton. When a joint rotates, it moves the assigned vertices around the joint’s pivot point. I chose to drive a chain of joints using inverse kinematics (IK). Instead of using it like regular IK where an algorithm calculates the optimal position of the joints based on a target, Autodesk Maya nHair will drive the joint chain with a dynamic curve.

First, a skeleton was created in order to bind the hair to its joints. The skeleton was placed in such a way that different parts of the hair could move independently (Fig. 9). Binding a mesh to skeletons is a tedious process and in order to alleviate it I chose to use the smooth bind method to connect the vertices to the joints. This smoothly bound the mesh to the joints by letting more than one joint drive a single vertex. The maximum allowed distance from a joint to a vertex was decided by a dropoff rate. However, the root of the hair must stay connected with the scalp. Thus, it must be left out from the simulation. This was achieved by assigning the vertices of the hair closest to the scalp to a root joint which is not included in the simulation.
Fig. 9. The hair is assigned to a skeleton. The seven different chains can move independently.

Next, the curves for the simulation were created. A vertex was created on each joint with the curve following the joint chain. This means that no unnecessary resolution was created for the curve simulation, since the joint chain can only bend at its specific joints. The curve was made dynamic by assigning it to an Autodesk Maya nHair system. The hair system holds the dynamic attributes for the curves in the simulation. This curve was then used by Maya IK solver to make the joint chain behave like simulated hair. The simulation was so fast that the hair systems attributes could be tweaked in real time for immediate response of the changes. These attributes include, for example, hair stiffness and simulation dampening. Usually the dampening has to be increased or else the hair will behave as if it were in a vacuum. Curves for each joint chain were created and the previous steps repeated, effectively creating seven different hair systems.

Lastly, the hair should be made to collide with the character. This will be done automatically if the body is assigned as a rigid body to the same nucleus (dynamics solver) as the hair in Maya. I let the hair simulate 100 frames before starting to animate the character in order to let hair come to its resting position before starting the animation. When character movement has been applied and the hairs attributes have been tweaked to achieve the desired result, the hair system should be cached to the hard drive so that the animation can be rendered on many computers at the same time. The reason for doing so is that the hair system can perform the calculation differently on different computers and caching allows many computers to render different parts of the same sequence with the exact same hair simulation.

3.1.5 Rendering polygon hair

First, the scene must be set up properly to achieve good rendering results. Nvidia mental ray was chosen as the render engine, since it produces more realistic results than Maya software renderer. I set up three cameras from different angles (see Fig.10).
I chose these three views to cover as much of the hair as possible while still preserving the possibility to study details of the result. I assigned the anisotropic shading to the hair and tweaked the specular shading attributes to match those of realistic hair. An important thing to keep in mind is not to make the specular light spread too much. By default the spread is set to a wide angle. We can simply increase the “spread x” to make the angle smaller, thus making the specular shading spread less. I connected the diffuse texture to the color attribute in the shader, the alpha texture to the transparency attribute and the anisotropic texture to the specular color attribute.

Next, the lights in the scene were set up. I used standard 3 point lighting as a basis and added an extra light directed towards the face for additional brightness to make the rendered result seen clearly. Ray tracing was enabled on all lights and the brightness of each individual light was adjusted so the rendered image would not overexpose. The settings I used for rendering was the production: fine trace preset as a basis, final gathering and ambient occlusion enabled.

Lastly, the entire sequence and all the cameras were to be rendered. This was done by running a batch render. I let it run through the sequence of frame 100-365 for all three cameras. The batch renderer automatically read the stored cache of each frame. Each frame was outputted as a targa image at a resolution of 1280x720 pixels and later put together as a seamless sequence.

3.2 nHair

Unlike polygon hair, this technique requires more computation power, more memory and cannot be used in a real time environment.

3.2.1 Creating the curves

Since I had already created the polygon-stripe based hair, I could use most of the stripes for creating the curves. The easiest way to extract a curve is to do it from a NURBS surface, which means the polygon stripes have to be converted to NURBS. This cannot be done directly but can be achieved by first converting the polygon to a subdiv and from a subdiv to NURBS. This imposes such a constraint on the polygon as it must keep the same resolution across the entire mesh. Namely, the polygon mesh must have the exact same amount of vertices at the top and the bottom of the mesh and
no edges can intersect or be merged in the middle. The same applies to the left and the right of the mesh as well.

After the polygon stripes were converted to NURBS surfaces, the curve extraction was the next step. An important thing to remember is that the curves resolution depends on the number of vertices the NURBS has. Thus, the resolution should be kept at minimum while the desired shape is maintained. When the hair is being rendered, it can instead be subdivided to fake a higher resolution. However, the simulation will always depend on the number of vertices the curve has. Keeping a high amount of vertices will slow the simulation down significantly but provides better simulation results. Extracting curves from a NURBS surface is easy; just select where on the surface a curve should be extracted and choose the duplicate surface curve menu option. The curves should be evenly distanced from each other but not too far since a hair clump will be assigned to each curve. If the curves were too far from each other, it would mean the hair clump must be widened and thus have too many hair strings assigned to it, making it clearly visible that the hair is made of separate clumps. This would break the illusion of realistic hair.

This process is applicable to both the blue and the red color parts in Fig. 3. For the green part, however, the polygon mesh does not have the correct geometry. So I had created a new NURBS surface from scratch. Simply creating a NURBS sphere and moving its vertices to match the shape of the concept did a sufficient job. Then the curve extraction proceeded as usual. See Fig. 11 for the finished curve placement.

![Fig. 11. All the required curves created and put in place to match the concept of the hair style](image)

### 3.2.2 Creating, adjusting the hair system and adding pfx

The generated curves in the previous step were first constrained to the skull in the character rig to make them follow the head movement of the character. Next, they were made dynamic by assigning hair systems to them. I chose to use eight different hair systems to provide the individual setups required by the constraints put on the different parts of the hair: the bangs in the front became one hair system; the bangs on the side became one; the smaller bangs on the side also had to become its own separate system because it needed to be more rigid than the bangs on the side; the hair on the top tied in the knot became a hair system because it needed to be almost
completely rigid and finally, the ponytail became four different hair systems, which are one on the very top for the short hair sticking out from the knot, one for the part that was a bit longer but not reaching to the very bottom of the ponytail, one following the entire ponytails length and one for filling in the middle of the ponytail.

By assigning different hair systems to each part of the hair I could assign specific values to different parts of the hair without affecting the entire hair. This made it possible to make the hair behave and look the way I wanted it to. Pfx was assigned by using the menu option assign paint effect brush to hair. This let me set up the hair shading and set how many hair strings per clump the hair system should have. For the hair shading I only changed the color and specular color values. The appropriate shading style (anisotropic) was already handled by Maya. Since nHair cannot be displayed properly in real time I had to render it every time I wanted to see the changes of the shading.

3.2.3 Simulating nHair

The dynamic properties of the nHair were configured in each of the hair systems. It is possible to have collisions between the different hair systems: hair to hair collision, self collision and collision between every object assigned to the same nucleus as the hair system. The right values for a good looking simulation vary and to get them is a trial and error process. Since the playback of the simulation is very slow, a playblast is often required to see if the hair behaves in the desired way or not. A playblast records each frame sent to the graphics card by Maya and saves it to a video file. This video file can then be played back in real time for easier viewing of the results. The dynamic values were changed a little at a time and a playblast were made to see the results. This process was repeated until I was satisfied. When the desired result was achieved, the simulation was cached to hard drive. In my case the desired result was realistic movement of the hair.

3.2.4 Rendering nHair

The exact same scene setup for rendering polygon stripe based hair was used for rendering nHair. There was no need to adjust the nHair shaders, since this was done in the previous step when assigning pfx to the hair system. The same frame span was rendered (100-365) and the cache was automatically read from the hard drive. The frames were outputted as targa files and later put together as a seamless sequence.

4 Results

The polygon stripe method is much easier to use than nHair because what you see in the preview is what you get when you render. The nHair approach can produce much more realistic results if it is used in the right way but is difficult to master and takes long time to get any kind of results. The polygon stripe method simulated the sequence of frame 1-365 in real time whereas it took nHair 6 minutes and 1 second to do so. That is almost 24 times slower than the polygon stripe method. The render times between the two were almost similar. It took the polygon hair 15 hours and 32 minutes to render the sequence of frame 100-365 whereas the nHair took 15 hours and 52 minutes to render the sequence. The quality of the rendered hair from the polygon stripe method depends on the texture quality and the amount of stripes used, while the result of nHair looks more realistic because of its ability to shade each individual hair differently. The rendered results are shown in Fig. 12 below but is best viewed in motion at the following URL’s: http://youtu.be/PaRVqzfZGk for polygon stripe based hair and http://youtu.be/IJk8F2zArTw for Autodesk Maya nHair.
5 Discussion

The first impression of these two methods is their respective render times, which are not remarkably different. Being able to simulate something in real time but failing to render it in real time makes the technique an offline one. In fact, the polygon-stripe based hair can be used in real time, just not with Autodesk Maya. If the character and the hair were placed in a game engine it would easily be able to render in real time on a recent computer. The game engine would also use a similar technique to simulate the hair in real time.

Regarding the simplicity of using the different systems, my personal opinion is that the polygon-stripe based hair is easier to understand and to use. It is reliable and with the right texture maps it could generate very realistic results. The method works in nearly all renderers, both offline and online. What’s more, the result can be seen in preview of Maya’s viewport while creating the hair. There will of course be some differences when you render it. For example, shadows on underlying geometry will look better and the edges will be sharper due to the anti-aliasing Nvidia mental ray uses. With nHair, however, it is impossible to see any kind of actual shading before rendering. You can however get a rough preview during the creation of where the generated hair might end up when it is rendered.

A downside with polygon hair is that it cannot simulate the hair too freely because it is difficult to be controlled and fixed when the geometry intersects with the character geometry or other hair stripe geometries. This can be seen in the polygon stripe result video at around frame 71 (see Fig. 13). The solution is to restrain the hair from moving too far from its starting position or to swing realistically by applying a lot of dampening to it. The dampening would force the hair to move less by restraining the motion gained from the input movements. However, this would in turn make the hair behave less realistically. At last, this method relies heavily on the artist’s skills of 2D applications to create convincing texture maps and to know how to place the stripes in order to make a solid looking hair style.
Simulating with nHair using just a few curves is fine. When many curves are used, the simulation does not use the computation resources optimally. On my computer it would only use around 8% of my processor’s capacity. This is because the processor has 6 cores and supports hyper-threading (each physical core addressed as two virtual cores by the operating system) and nucleus only recently started supporting multithreading with some of its solver features. If the nHair was optimized, the simulation would go roughly 11 times faster than what it currently does on my system. Even though the simulation of nHair is so slow, one of the popular workflows of styling the hair is to use simulation and various constraints. Autodesk Maya even suggests it [13]. I, on the other hand, would not recommend using this workflow unless Autodesk speeds up the simulation by optimizing the nucleus solver.

The overall experience with nHair is that it is an unfinished product filled with bugs and badly optimized. Maya crashed frequently during the usage of nHair. For example, when I cached the simulation, Maya could crash when the timeline was rewound to frame one. This would waste 6 minutes of simulation time since there was no opportunity to save the work. When I first tried to render with nHair, the Maya batch did not read the cache and thus did not render at all. In another case the hair was simulated fine and written to the cache but when I played the animation using the cache, several bugs occurred, such as single hair strands pointing in weird directions (Fig. 14) or being completely static. This forced me to render without cache. Normally when one does the rendering without cache and the rendering starts at a non-zero frame, the Maya batch automatically does a run-up simulation to that frame before rendering the rest. With nHair this was not the case, forcing me to render 100 unwanted frames for each camera before the actual animation started, effectively losing 6 hours.
6 Conclusion

Autodesk Maya nHair could be useful and produce realistic results for rendering an image, provided the artist has a lot of time at his disposal. However, the horrible optimization, the difficulty of use and the lack of reliability make it hard to be recommended. These could also be the reasons why companies today choose to develop their own tools for the task. On the other hand, for a lighter usage, such as the one described in my polygon-stripe based hair technique, the nHair might be a good option. So the bottom line here is, if the artist needs to do a lot of hair renderings, I would suggest him to invest in other third-party solutions. As for the technique of polygon hair stripes, it is very robust and flexible and the application of it is really only limited by the skill level of the artist.
References


