# A LightWave 3D Plug-in for Modeling Long Hair on Virtual Humans

Deborah Patrick
Department of Computer Science
Rhodes University, Grahamstown, 6140
g9750082@campus.ru.ac.za

Shaun Bangay
Department of Computer Science
Rhodes University, Grahamstown, 6140
S.Bangay@ru.ac.za

## Abstract

Multimedia applications today make use of virtual humans. Generating realistic virtual humans is a challenging problem owing to a number of factors, one being the simulation of realistic hair. The difficulty in simulating hair is due to the physical properties of hair. The average human head holds thousands of hairs, with the width of each hair often smaller than the size of a pixel. There are also complex lighting effects that occur within hair. This paper presents a LightWave 3D plug-in for modeling thousands of individual hairs on virtual humans. The plug-in allows the user to specify the length, thickness and distribution of the hair, as well as the number of segments a hair is made up of. The plug-in is able to add hairs to a head model, which the user then modifies to define a hairstyle. The hairs are then multiplied by the plug-in to produce many hairs. By providing a plug-in that does most of the work and produces realistic results, the user is able to produce a hairstyle without modeling each individual strand of hair. This greatly reduces the time spent on hair modeling, and makes the possibility of adding realistic long hair to virtual humans reasonable.

**Keywords:** Hair Modeling, Explicit Model, Plug-in, LightWave

## 1. Introduction

This paper presents a plug-in that allows users to add long hair to models in LightWave 3D[1], a graphics program used for creating objects and animations. Various applications today make use of virtual humans that are created in LightWave, or a similar program. Virtual humans are often needed as agents or avatars in virtual worlds, characters in computer games, or computer generated people in films and advertisements. In many of these applications the virtual humans need to appear realistic in order to be believable. Unfortunately generating realistic humans is still a challenging problem owing to a number of factors, one being the simulation of realistic hair. Because of this, computer generated people are often bald, or have a hat placed on their head.

The difficulty in simulating hair is due to the physical properties of hair. The average human head holds thousands of individual hairs that vary in colour, length and shape. It is far too time-consuming and impractical to model each and every hair. In addition, the width of a single strand of hair is usually smaller than the size of a pixel. This leads to aliasing problems as a single colour is calculated for a pixel shared by a number of hairs.

---

[1] LightWave 3D version seven is developed by NewTek Inc. © Copyright, NewTek 2001.

Complex lighting effects also occur among the hairs. These need to be taken into account when rendering hair in order for it to appear realistic. The most noticeable effects, which play a vital role in the appearance of the hair, are the shadows cast by hairs onto other hairs and the anisotropic reflection of light off hair. Without shadows the hair does not look realistic, and the general lighting models available do not provide good approximations of anisotropic reflection. Because of the anisotropic property of hair, from a distance you do not see the individual hairs, but rather a region of hair that has a distinct pattern or texture. Hair is also slightly transparent, which results in a haloing effect when backlighting is used. Due to these factors hair is time-consuming to model and expensive to render.

This paper investigates hair modeling and the problem of creating thousands of individual hairs, focusing on modeling long hair, rather than short hair or fur. Several researchers have developed methods for modeling hair. These can be divided into three main categories: explicit models, cluster models, and volumetric models. Each category has advantages and disadvantages. The different models with their advantages and disadvantages are discussed in Section 2.

In this paper an explicit model is used because it is the most intuitive and appropriate method for modeling long hair. The explicit model allows explicit manipulation of individual hairs. It also allows hair to be rendered realistically using conventional shaders and lighting models. Because each individual hair is modeled, the anisotropic reflection of light off hair does not need to be considered. The explicit model is often avoided due to the large number of polygons needed. This paper investigates the efficiency of the explicit model, and discusses the time it takes to model and render a hairstyle.

## 2. Related Work

There are three important aspects in hair simulation: hair modeling, hair rendering and hair animation. Hair modeling deals with the geometry and distribution of the hair strands, hair rendering involves the lighting and shading of hair, and hair animation looks at hair movement and collision detection [Magnenat-Thalmann et al. 2000]. Previous research has been done on all three of these aspects, and several models have been developed. The models, together with their strengths and weaknesses, are summarized by Magnenat-Thalmann et al.. [2000] in a paper that reviews the state of the art in hair simulation. The models can be divided into three main categories: those that model hair explicitly, those that model hair in clusters, and those that model hair as volumetric textures.

### 2.1 Explicit models

The explicit hair models generate the geometry of each individual hair strand. They are brute force methods that are suitable for modeling long hair or simulating hair dynamics. In these models, each hair strand is modeled using some sort of primitive such as a curve, cylinder, or group of connected lines [Kim and Neumann 2000].

Anjyo et al. [1992] define each hair strand as a set of connected lines and animate the hair as a cantilever beam. Daldegan et al. [1993] produce 'HairStyler', a framework that allows the user to interactively model strands of hair. The user defines a few hair strands, which are then multiplied to produce a hairstyle. The hair strands are represented as "straight cylindrical segments connected by points".

The explicit model is simple and intuitive; unfortunately it is also time consuming. Several researchers have therefore considered a cluster or group of hairs at a time to reduce the time taken in simulating a hairstyle. These models are called cluster or wisp models.

## 2.2 Cluster models

Cluster models, also known as wisp models, group hair into clusters of hair strands. They make use of the observation that real hair strands tend to form clumps due to adhesive and cohesive forces [Magnenat-Thalmann et al. 2000], making it appropriate to model hair in groups. The geometry of a cluster is modeled, and detail is added by rendering the hair strands or volume density [Kim and Neumann 2002].

Chen et al. [1999] model groups of hair using trigonal prisms. A 2D hair distribution map is then added to the prisms for the detail of the individual hairs.

Kim et al. [2000] present a method for modeling long hair, and the interaction between hairs. Groups of hair strands are modeled as a "thin shell volume" created by offsetting a surface along its normal. The individual hairs are represented as particles inside the volume. The movement of the particles is restricted by the properties of hair-to-hair interaction.

In a later paper, Kim et al. [2002] present an interactive system that provides a user with visual feedback as they model different hairstyles. In this system, clumps of hair are represented as generalized cylinders.

Koh et al. [2001], model groups of hair strands in strips using parametric surfaces. Texture maps are applied to the surfaces for detail. This method is able to render hair quickly and is therefore useful for real-time applications. Unfortunately, the method does not simulate the complex shadowing and geometry of the hairs and therefore does not produce very realistic hair.

These models are similar to explicit models, but often lack variation between hairs in a cluster, and are more difficult to use when simulating hair movement.

## 2.3 Volumetric models

These models do not explicitly model the geometry of each individual hair strand. Instead, a mathematical function or volumetric texture (volume density function) is used. Because the user does not have much control over the shape of individual hairs, and the hair must be analytically defined, these models are more suited to short hair or fur.

Perlin et al. [1989] introduce hypertextures, which are able to create the illusion of fur by evaluating a density function in a three-dimensional space. Kajiya et al. [1989] use hypertextures as a piece of texture that is tiled onto complex surfaces. Lengyel et al. [2001] then provide a method for real-time rendering of fur on complex surfaces.

The volumetric texture approach models the complexity of hair with a simple mathematical function (or functions). Unfortunately this approach does not provide the user with much control over individual hairs and is not suitable for modeling long hair. Due to the need for realism in the modeling of long hair, the explicit model is the most appropriate method to use.

## 3. Overview

In this paper a plug-in, for LightWave, that models long hair explicitly is presented. The plug-in is divided into two parts: *HairDesign*, which deals with adding hair to a head model and designing a hairstyle; and *HairClone*, which deals with the problem of modeling thousands of hairs.

In Sections 4 and 5 modeling hair explicitly and LightWave plug-in development are looked at. The implementation of the plug-in is discussed in Section 6, and some results are shown in Section 7.

## 4. Modeling Hair

Creating hair on a model of a head involves placing hair on the head model, representing the hair in some way, and dealing with the large number of individual hairs.

### 4.1 Growing hair from the scalp

An important task in modeling hair is attaching the hair to the scalp of the head model and determining the distribution of the individual hair strands.

There are two ways you can add hair to the scalp of the head model. The first method makes the geometry of the hair part of the head's geometry. The advantage of this is that there will be no holes or overlapping polygons in the mesh, and the hair will move with the head. This method, however, is not practical. It requires an algorithm to join the hair geometry to the head and results in an unnecessarily large number of polygons for the scalp. The second method is more practical. It involves placing the hair on the head, but does not attach the hair geometry to the head. The advantage is that no algorithm is required for attaching the hair, and the scalp mesh remains the same. There are ways to move the hair with the head, and if the 'root' of the hair is placed close to the surface, overlapping the hairs on the head geometry will not cause unusual effects.

Several methods have been used for distributing the hair on the head. Lee et al. [2002] divide the area where hair should be placed into ten regions. The hair is distributed in each region using a growth map (probability density function) based on the density of the hair. Hadap et al. [2000] also use a probability density function, or growth map, but do not divide the head into ten regions. Kim et al. [2002] do not model the hairs directly on the surface of the head model. Instead they wrap a spline patch over the area of the head model where hair can be placed. The advantage of this method is that the hairs are distributed on a two-dimensional plane, which is easier than trying to assign the hairs to polygons on the head model. The disadvantage is that the scalp is curved. This can cause hairs to be misaligned and results in abnormal bending of the hair segments. Daldegan et al. [1993] assign an individual hair to each triangle on the head model. This hair defines the shape and size of the other hairs on the triangle. The number of hairs on each triangle is proportional to the area of the triangle.

This paper uses an approach similar to that taken by Daldegan et al. [1993], but instead of distributing the hair according to the areas of the triangles, the head mesh is divided into triangles of similar size. The same numbers of hairs are then placed randomly in each triangle. This is implemented in the plug-in, so that the user does not have to manually place hair strands on the head.

### 4.2 Representing hair strands and the hairstyle

Because the hair is modeled explicitly, it is not represented by mathematical functions, but rather by points, lines or polygons. Physically, hairs are thin, curved cylinders. Close approximations

**Figure 1:** The modeling process to "grow" hair on the scalp using the *HairDesign* plug-in. A head model is designed or imported (left), the user selects the polygons on which they want hair to be placed (middle), and run *the HairDesign* plug-in. The plug-in produces control hairs (right).

to round bent cylinders require a large number of polygons, and since hundreds or thousands of hairs have to be modeled, the number of polygons required for each hair needs to be kept to a minimum. One way is to represent the hairs as polylines, which is one of the most widely used methods. Polylines are two-dimensional and can only provide approximations of cylinders when rendering. In this paper individual hairs are first represented as curves, which are easy to use and bend. Each curve is then extruded in two directions to produce a three-dimensional approximation of a cylinder. Unfortunately this method produces a ribbon effect, and is not the most effective method for producing realistic results. A more appropriate technique is to represent groups or individual hair strands as generalized cylinders.

As in the work undertaken by Daldegan et al. [1993], the hairstyle is represented by a few 'control' hairs. The control hairs are initially curves, therefore the user is able to easily modify them, using tools in LightWave.

### 4.3 Multiplying hair

To overcome the problem of creating thousands of individual hairs, a plug-in is designed that multiplies the individual control hairs by a number specified by the user. Unlike the method in [Daldegan et al. 1993], each triangle on the head model can have more than one type of hair 'growing' from it, all the user has to do is run the *HairDesign* plug-in more than once, and select the same polygon each time. The number of hairs placed in each triangle is user-defined rather than proportional to the area of the triangle.

### 5. LightWave 3D Plug-ins

A plug-in is a dynamically linked library (.dll) that extends the functionality of a program. Plug-ins are imported into a system and become commands that can be used like any other command in the program. Most 3D graphics programs, whether commercial or in-house, have an architecture that supports plug-in development. In this paper the method for modeling long hair is implemented as a plug-in for LightWave.

LightWave plug-ins are grouped into different categories, called 'classes'. The different classes perform different tasks and are placed in LightWave at different points. The plug-ins presented in this paper belong to the Modeler Command Sequence class. Modeler Command Sequence plug-ins create and modify geometry (The plug-in in this paper creates hair and modifies the

head geometry). They are able to issue commands[2] and have access to mesh editing functions [Wright 2001].

Mesh edits create, modify and obtain information about the points and polygons of the geometry. Point and polygon scan functions provide information or modify geometry by enumerating all points and polygons. For each point or polygon, they reference a callback function, which is supplied by the programmer [Wright 2001].
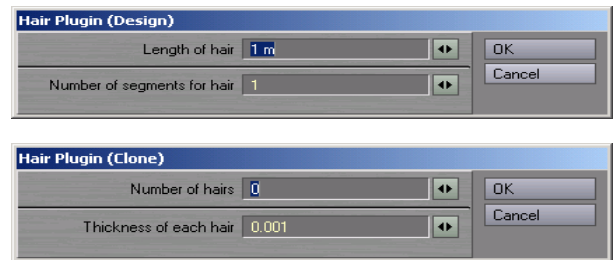


**Figure 2:** The *HairDesign* (top) and *HairClone* (bottom) plug-in interfaces.

### 6. Implementation

The hair modeling system consists of two plug-ins: *HairDesign* and *HairClone*. These plug-ins alter a head model so that hair can be placed on it, add control hairs to the head, and multiple the control hairs to produce lots of hair. Together the plug-ins greatly reduce the time required by a user to simulate a hairstyle.

### 6.1 Hair Design plug-in

The *HairDesign* plug-in creates control hairs and places them on the scalp of the head model (See Figure 1). All the user is required to do is select the polygons on the head model they would like hair to be placed on, run the plug-in and enter the length and number of segments they would like each hair to consist of. The plug-in's interface is programmed so that it appears to look like LightWave's interface (See Figure 2).

---

[2] Commands perform tasks that the user could perform in LightWave's interface [Wright 2001].

### 6.1.1 Growing hair from the scalp

Polygons are selected using a selection tool included in the program. LightWave has several tools for selecting groups or individual polygons. Due to the fact that any head model may be used, each selected polygon is checked for flatness, and all polygons that have more than three sides are subdivided into a number of smaller triangles. The result is a head with a selected area made up of triangles.

Currently it is assumed that all the triangles are of similar area. Alternative strategies would be to calculate the area of each triangle, and subdivide those triangles whose areas are greater than the average area. One could also combine triangles that are too small or assign the same hair to more than one triangle.

A single control hair is then placed in the center of each selected triangle using a mesh edit and enumerating the polygons. This hair defines the length and shape of the other hairs on that triangle. The center of each triangle is found, and this becomes the start of the control hair. The hair is represented as a curve, which is initially a straight line placed in the direction of the triangle's normal. The length of the curve and the number of segments on the curve are specified by the user and obtained through the plug-in's interface. The curves and triangles are tagged, to keep track of which hairs belong to which triangles. First the triangle is checked to see if it has a hair tag, if it does the hair curve is tagged with the same string as the triangle; otherwise the hair curve and the triangle are given a tag (the same for each). Placing tags on the triangles allows one to keep track of the triangle a hair is placed on. This is necessary for placing copies of a control hair on the same triangle. It may also be useful when animating the hair to make sure that the hair curves move with the triangles they are placed on since the hair curves are not part of the head geometry.

### 6.1.2 Designing a hair style with control hairs

The user is then able to bend and modify the control curves to define the hairstyle they want using the tools available in LightWave 3D (See Figure 3).
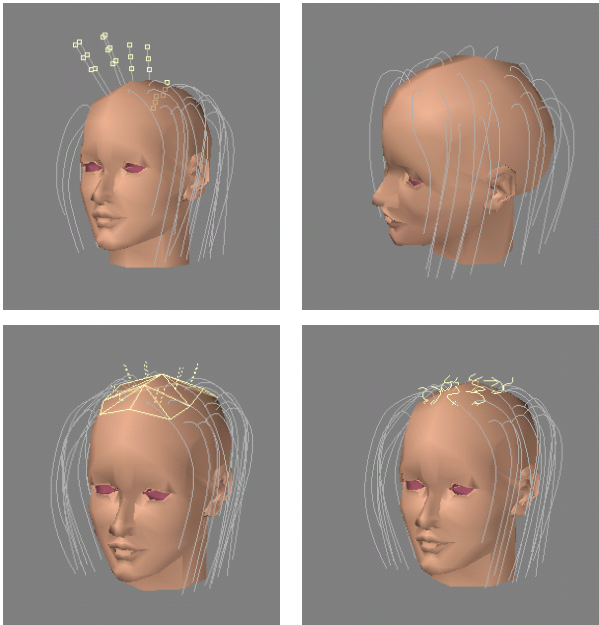


**Figure 3:** A hairstyle is designed with the control hairs (top), and a second group of control hairs are added to the hairstyle for more detail (bottom).

### 6.2 Hair Clone plug-in

After the user has designed the hairstyle, they select the control hairs they would like to multiple and run the *HairClone* plug-in (See Figure 4). To multiply different control hairs by different amounts, the user can select the control hairs individually and run the plug-in more than once.
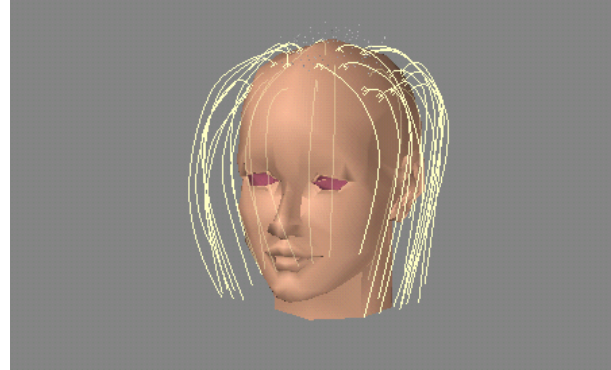


**Figure 4:** Selected control hairs before they are multiplied.

### 6.2.1 Multiplying the control hairs

The *HairClone* plug-in asks the user how many times they would like to duplicate each hair. The user is also asked for the thickness they would like each hair. Again, the plug-in's interface is programmed so that it appears to look like LightWave's interface (See Figure 2). The plug-in randomly adds the specified number of copies of the control hairs to each triangle. At the same time it extrudes the curves so that they are no longer two-dimensional, but rather three-dimensional approximations of a curved cylinder.

To add copies of a control hair, the control hair is checked to see that it is a curve. The triangle on the head model, that the control hair is attached to, is then found by matching the tags. Once found, copies of the control hair are made and translated to a random point on the triangle. This is done using Turk's algorithm [1990] for finding a random point in a triangle. The number of new hairs to be made from each control hair is obtained from the user interface.

Once all the new hairs have been added, all the hairs are extruded so that they become three-dimensional objects. To do this all geometry that is not hair geometry is hidden. A command is then issued, which extrudes all geometry in a specified direction. First the geometry is extruded along the X-axis and then along the Y-axis. The LightWave extrude command only allows geometry to be extruded along the X-, Y- and Z-axes. The amount each curve is extruded is obtained from the user through the plug-in's interface. After extruding the hair curves, a command is issued to unhide the hidden face and other geometry. Geometry that is not part of the hair is hidden because the selected hairs do not remain selected after the first extrusion. When the second extrusion along the Y-axis is performed, all the geometry is extruded including the polygons on the face. This causes unwanted results.

After multiplying the control hairs the user is able to modify newly created hairs using the tools provided by LightWave.

### 7. Results

Hair created using the proposed plug-in can be seen in Figure 5. The hair is created using two sets of control hairs. The first set contains 30 control hairs. Each of these control hairs has 4

segments and an initial length of 70mm (The length needed depends on the size of the object and the units being used). The second set of control hairs consists of 12 hairs. Each of these hairs has 3 segments and an initial length of 30mm. The model of the head is approximately 150mm wide and 255mm high. Each control hair is multiplied 30 times. The hair curves are extruded by 0.27mm (Again this depends on the size of the object and the units being used).

The hair is rendered using the default lighting models and rendering techniques provided by LightWave. The hair transparency is achieved using the built-in raytracer, and shadows are produced using shadow mapping. While other approaches to hair modeling [4, 6] require development of specialised shaders for approximating the illumination of hair, use of an explicit model allows shading which is completely consistent with the underlying geometry. This is particularly relevant for long hair where the relative densities, and relationship to the surrounding objects, changes substantially over the length of the hairs.

The hairstyle takes approximately 1 hour to model and 150 seconds to render. The system used is an 800 Mhz Intel Pentium III system with 128 MB RAM.

In Figure 6 the same control hairs are used, with different hair densities. The rendering times for each example can be seen in the table below:

| Control hairs | Copies of each control hair | Total number of hairs | Rendering time |
|---|---|---|---|
| 30 | 10 | 330 | 42 seconds |
| 30 | 20 | 630 | 90 seconds |
| 30 | 30 | 930 | 150 seconds |
| 30 | 40 | 1230 | 214 seconds |

The hairstyles become similar in appearance as the density of hair is increased. Thus, although use of the explicit model requires large numbers of polygons to represent hair, there is an upper bound on the amount of detail required to produce a realistic appearance. Generating more hair strands than needed will increase the time taken to render the hair without increasing the visual quality of the hair. The number of hairs required is well below the number actually present on the human head.



**Figure 5:** The same hairstyle rendered with different hair colour values.

## 8.   Conclusion

This paper describes the modeling and rendering of hair using an explicit hair model. A process is devised that is suitable for the creation of hairstyles containing thousands of hairs. This is embodied in a plug-in for LightWave, which adds hair to an arbitrary head model, allows a user to manipulate control hairs to produce a skeleton structure of the desired hairstyle, and converts the skeleton structure into a hairstyle with the desired hair density. The plug-in allows the user to specify the thickness, length and distribution of hairs, as well as the geometric complexity of each hair strand. A number of limitations of the plug-in SDK for LightWave are discussed, and methods for overcoming them are described.

The results show that use of the explicit model for representing hair can produce realistic results without excessive storage or computational overheads. The number of polygons required to produce quality hair models is bounded, and increasing geometry detail beyond this point results in marginal improvements in the appearance of the rendered hairstyles. The benefits of using the explicit model are substantial: existing rendering and shading techniques can be used, and reflection, refraction and shadowing are all matched to the actual geometry.

By providing a plug-in that does most of the work and produces realistic results the user is able to produce a hairstyle without modeling each and every strand of hair. This greatly reduces the time spent on hair modeling, and allows realistic long hair for virtual humans to be modeled and rendered.
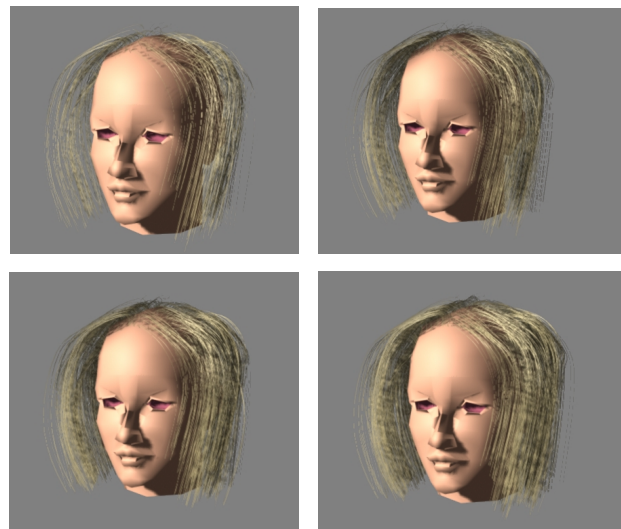


**Figure 6:** The same control hairs are multiplied 10, 20, 30, and 40 times (From left to right, top to bottom).

## References

ANJYO, K., USAMI, Y., AND KURIHARA, T. 1992. A Simple Method for Extracting the Natural Beauty of Hair. *In Computer Graphics (Proceedings of ACM SIGGRAPH 92), 26 (2)*, ACM, 111-120.

CHEN, L., SAEYOR, S., DOHI, H., AND ISHIZUKA, M. 1999. A System of 3D Hair Style Synthesis Based on the Wisp Model. *The Visual Computer*, 15 (4), 159-170.

DALDEGAN, A., MAGNENAT-THALMANN, N., KURIHARA, T., AND THALMANN, D. 1993. An Integrated System for Modeling, Animating and Rendering Hair. *Computer Graphics Forum (Eurographics 93), 12 (3)*, 211-221.

GOLDMAN, D. 1997. Fake Fur Rendering. In *Proceedings of ACM SIGGRAPH 97*, ACM Press / ACM SIGGRAPH, 127-134.

HADAP, S., AND MAGNENAT-THALMANN, N. 2000. Interactive Hair Styler based on Fluid Flow. *Proceedings of Eurographics Workshop on Computer Animation and Simulation 2000*, 87-99.

KAJIYA, J., AND KAY, T. 1989. Rendering Fur with Three Dimensional Textures. In *Computer Graphics (Proceedings of ACM SIGGRAPH 89) , 23 (3)*, ACM, 271-280.

KIM, T., AND NEUMANN, U. 2000. A Thin Shell Volume for Modeling Human Hair. *IEEE Computer Animation*, 104-111.

KIM, T., AND NEUMANN, U. 2002. Interactive Multiresolution Hair Modeling and Editing. *ACM Transactions on Graphics, 21, 3*, 620-629.

KOH, C.K., AND HUANG, Z. 2001. A Simple Physics Model to Animate Human Hair Modeled in 2D Strips in Real Time. *Proceedings of Eurographics Workshop 2002*, 127-138.

LEE, C., CHEN, W., LEU, E., AND OUHYOUNG, M. 2002. A Rotor Platform Assisted System for 3D Hairstyles. *Journal of WSCG, 10 (1-3)*, 271-278.

LENGYEL, J., PRAUN, E., FINKELSTEIN, A., AND HOPPE, H. 2001. Real-Time Fur over Arbitrary Surfaces, In *Proceedings of Interactive Symposium on 3D Graphics*, 227-232.

MAGNENAT-THALMANN, N., HADAP, S., AND KALRA, P. 2000. State of the Art in Hair Simulation. *International Workshop on Human Modeling and Animation*, 3-9.

PERLIN, K., AND HOFFERT, E. 1989. Hypertexture. In *Computer Graphics (Proceedings of ACM SIGGRAPH 89), 23 (3)*, ACM, 253-262.

TURK, G. 1990. Generating random points in triangles. In *A. S. Glassner, editor, Graphics Gems*. Academic Press, 24-28.

WRIGHT, E. 2001. LightWave Development Documentation. Available online at: http://www.lightwave3d.com/developer/.

**A LightWave 3D Plug-in for Modeling Long Hair on Virtual Humans:** Deborah Patrick and Shaun Bangay



**Figure 1:** The modeling process to "grow" hair on the scalp using the *HairDesign* plug-in. A head model is designed or imported (left), the user selects the polygons on which they want hair to be placed (middle), and run *the HairDesign* plug-in. The plug-in produces control hairs (right).
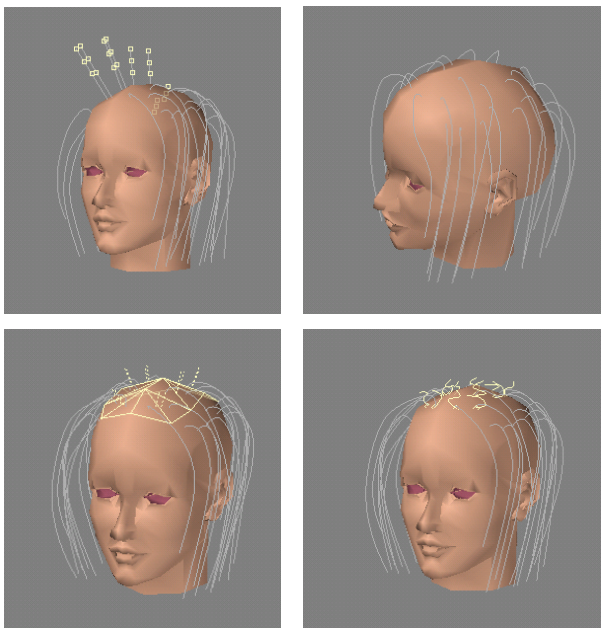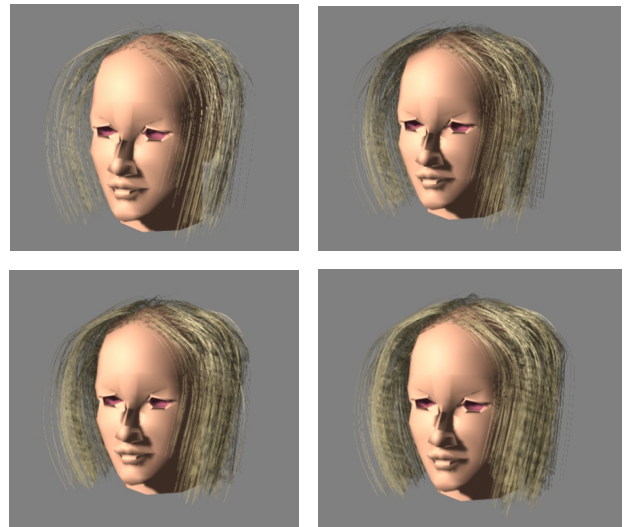


**Figure 3:** A hairstyle is designed with the control hairs (top), and a second group of control hairs are added to the hairstyle for more detail (bottom).
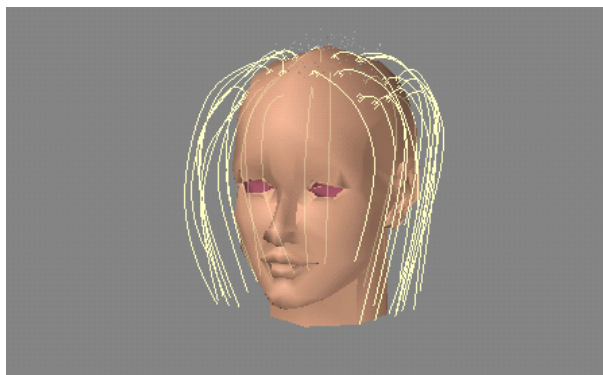


**Figure 4:** Selected control hairs before they are multiplied.



**Figure 5:** The same hairstyle rendered with different hair colour values.



**Figure 6:** The same control hairs are multiplied 10, 20, 30, and 40 times (From left to right, top to bottom).