Simplified Representations for Modeling Hair

Abstract: *We present a novel framework for modeling hair using simplified representations. The set of representations, including individual strands, clusters and patches, are derived from Dynamic-NURBS and have the same underlying base skeleton. Our framework supports automatic simplification of dynamic simulation, collision detection and graphical rendering of hair, and offers flexibility to balance between performance and quality. Furthermore, it is applicable to modeling different hair styles. We have used the framework to simulate different styles and obtained noticeable performance improvement in the simulation, with little loss in visual quality.*

Keywords: Geometric Modeling, Hair Modeling, Level of Detail Algorithms, Physically-based Modeling.

1 Introduction

The ability to model human features has become an essential aspect of computer generated images for the entertainment industry, game development, virtual reality, and other applications. Animating hair continues to be a challenging problem due to its complex nature. A human head can have over $100,000$ individual strands of hair. It is rather costly to animate hair in real-time because of the high number of primitives required to model hair accurately. Moreover, modeling long and wavy or curly hair introduces extra complexity, increasing computational costs for rendering and simulating interactions among hair and between the hair and the body.

The current commercial renderers, such as Pixar's RenderMan and other proprietary software used in movies like $Monsters,\ Inc.$ and $Final\ Fantasy$, can generate realistic appearances for hair and fur. To the best of our knowledge, these systems do not offer interactive performance for either animation or rendering of hair. Simulating and animating hair remains a slow, tedious and sometimes painful process for animators. A few existing real-time algorithms [KH01], on the other hand, often do not yield realistic hair renderings or simulation. One of the major bottlenecks in achieving real-time simulation of moving hair is collision detection among hair and between hair and the body. Due to its complex nature, this is often the dominating factor in terms of overall computational cost [PCP01].

Main Contribution: In this paper, we present a novel, unified framework for modeling hair based on simplified representations. We use a series of Dynamic-NURBS or D-NURBS [QT96] patches of varying sizes for modeling the least significant layers of hair, cluster models of variable thickness for representing the intermediate layers, and D-NURBS curves for simulating the most visible and highest resolution individual hair strands (shown in Fig. 1). Our algorithm combines the different set of representations to create different levels of detail (LODs) to simulate moving hair, perform collision detection, and accelerate graphical rendering. It automatically switches between different approximations of varying fidelity, depending on the user specified screen-space error tolerance, viewing distance, visibility, hair motion and other application dependent factors. Overall, our framework offers several advantages:

- **Unified representations** based on the D-NURBS formulation and the "base skeleton" representation;
- **Automatic simplification** of both geometric and physics models for hair animation;
- **Computational efficiency** in the overall dynamic simulation, collision detection and graphical rendering;

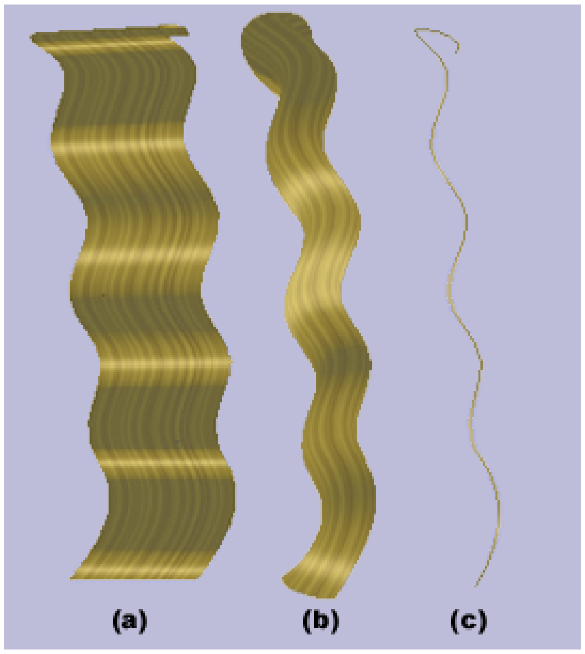

Figure 1: Simplified Representations. *(a) A patch; (b) A cluster; (c) An individual strand.*

- **Flexibility** in achieving the desired balance between simulation speed and visual fidelity;
- **Generality** in terms of modeling many different types of hair: short vs. long, straight vs. wavy, thin vs. thick, fine vs. coarse.

Using the simplified representations for modeling hair, we observed noticeable performance improvement with little degradation in visual appearance of the simulation.

Organization: The rest of the paper is organized as follows. Section 2 gives a brief survey of related work. Section 3 describes the three basic model representations of hair based on the D-NURBS formulation and the base skeleton. The dynamics model and our collision detection algorithm using the simplified representations are described in Section 4 and Section 5 respectively. Next, we describe techniques to render hair with different representations in Section 6. The criteria for automatic switching and selection of simplified representations are outlined in Section 7. Section 8 highlights the results of our implementation, and compares its performance with some earlier approaches.

2 Related Work

In this section, we briefly describe some related work in hair animation, model simplification, and simulation levels of detail.

2.1 Hair Modeling

Modeling hair has been an active area of research in computer graphics and numerous approaches have been proposed to address this problem [MTHK00, HMT01, Yu01]. Some fundamental techniques were presented to model the motion of individual hair strands in [AUK92, KAT93a, DKMTT93, KAT93b], with each strand of hair represented as a series of connected line segments and the shape of the hair determined by specifying the desired angles between segments. Forces are applied to the control points of the line segments to simulate the hair motion. To reduce the overall computation time, strands of hair that are near each other or move in a similar fashion, are bundled together as a group or as a *wisp* [KAT93a]. Using a similar philosophy, individual strands of hair are grouped together as "wisps" for animating long hair, each modeled using a spring-mass skeleton and a deformable envelope [PCP01]. A similar approach is used for interactive hair styling [CSDI99, XY01].

However, none of these techniques can perform both hair animation and rendering in real time. Recently, a thin shell volume [KN00] and 2D strips [KH00, KH01] have been used to approximate groups of hair. Such techniques enable real-time hair simulation. However, the resulting simulation lacks a realistic, voluminous appearance of the hair. Techniques for real-time rendering of fur that exploit graphics hardware were presented in [Len00, LPFH01]. However, this technique does not work well for rendering long, wavy or curly hair.

2.2 Model Simplification

Model simplification algorithms, such as automatic generation of geometric level-of-detail (LOD) representations and multi-resolution modeling techniques, have been proposed to accelerate rendering of complex geometric models [COM98, Hop96, GH97, SZ98]. A recent survey on polygonal model simplification is presented in [Lue01]. A generic framework for selecting and switching between different geometric levels-of-detail (LODs) to attain a nearly constant frame rate for interactive architectural walkthroughs was introduced in [FS93].

2.3 Simulation Level-of-Detail

The use of levels of detail has been extended to motion modeling and dynamic simulation as well. Simulation levels of detail (SLOD) are used to simplify or approximate the dynamics in a scene, similar to the way that geometric LODs are used to simplify a complex model.

Carlson and Hodgins explored techniques for reducing the computational cost of simulating groups of legged creatures when they are less important to the viewer or to the action in the virtual world [CH97]. In [PC01], levels of detail, including 3D geometry, volumetric textures and 2D textures, are used to animate and render prairies in real-time. SLODs have also been proposed for the automatic dynamics simplification of particle systems [OFL01].

Other types of simulation acceleration techniques, such as view-dependent dynamics culling [CF97] and Neuro-Animator [GTH98], have also been investigated to reduce the total computational cost for simulating a large, complex dynamical system.

3 Model Representations

Our algorithm combines several previously defined methods, and extends and generalizes these techniques to create a series of simplified representations to model, simulate and render hair. We use three basic representations to model different layers of hair. They are *groups of individual strands*, *clusters*, and *patches*.

3.1 D-NURBS

All three representations of the hair are derived from the formulation of Dynamic Non-Uniform Rational B-Splines, or D-NURBS [QT96]. D-NURBS are a class of physically-based geometric representations that are designed to allow the incorporation of dynamics behavior along geometric shapes. NURBS are a popular representation for freeform surfaces. The shape of a NURBS curve or surface is determined by its control points and weights. With D-NURBS, forces can be applied directly to the control points to automatically modify the shape of any NURBS curves or surfaces in a physically intuitive manner.

Let $B_{i,k}(u)$ be the B-spline basis functions, assuming basis functions of degree $k-1$. Then, a D-NURBS curve is represented as a time-dependent function (where t is the time variable)

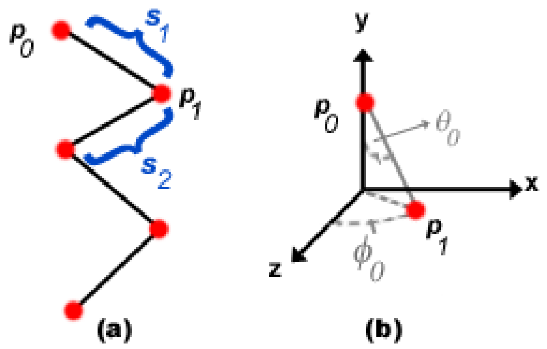

Figure 2: Basic Physics Models. *(a) The base skeleton model; (b) The parameters that define the style of hair.*

$$\mathbf{c}(u,t) = \frac{\sum_{i=0}^{n} \mathbf{p}_i(t) w_i(t) B_{i,k}(u)}{\sum_{i=0}^{n} w_i(t) B_{i,k}(u)}$$

where $\mathbf{p}_i(t)$ are time-dependent control points and $w_i(t)$ are time-dependent generalized coordinates of D-NURBS. The resulting time-varying curve function, $\mathbf{c}(u,t)$ can be expressed as $\mathbf{c}(u, \mathbf{p}(\mathbf{t}))$ to emphasize dependence on vector $\mathbf{p}(\mathbf{t})$, which is a function of time. Similarly, a D-NURBS patch can be defined using the tensor product for representing a D-NURBS surface. All bold-face symbols are used to represent vector-valued quantities.

Based on the D-NURBS representation and a base skeleton, three simplified representations for hair modeling are used. Our approach enables a powerful mechanism for automatic simplification of both the underlying dynamics and geometric representation for collision detection and graphical rendering.

3.2 The Base Skeleton

The base skeleton is comprised of a certain number of control points, or nodes, and is modeled as an open chain of line segments that connect these nodes. A spring force is used to control the angles between each node, while the distance between each node is fixed. Fig. 2(a) shows the basic setup of the skeleton. There are n control points $(\mathbf{p}_0, \mathbf{p}_1, \ldots, \mathbf{p}_{n-1})$ and $n-1$ sections $(s_1, s_2, \ldots, s_{n-1})$ to the skeleton. The number of control points is decided automatically based on certain criteria, such as the length of the hair, the waviness or curliness specified for the hair, and the desired smoothness. These control points, or nodes, of the skeleton have resting positions based on these criteria.

The waviness or curliness of the hair is specified by a parameter used for the angle and frequency of the wave. The placement of nodes is based on polar coordinates, where each node has θ_0 and ϕ_0 specified for it, which are the resting angles of the node in relation to the previous node. (See Fig. 2(b).) The angle θ_0 determines the waviness of the hair by specifying an angle between 0 and 90 degrees (straight hair will have a resting position of θ_0 = 0). Hairstyling is discussed in further detail in Section 3.6.

3.3 Patches

The patch model in Fig. 3(a) uses one skeleton model as its basis for motion. The skeleton is the center of the patch and for each node in the skeleton there are two control points that are used to define the patch. These two patch control points and the skeleton node point are colinear. The distance between the two control points from the skeleton node is dependent upon the specified size of the patch.

A skeleton made up of n points will result in a NURBS patch defined using $2n$ control points. Once the two control points have been determined from the skeleton node, they are rotated based on the orientation of the patch at that point. The patch's orientation is determined by the wave of the hair. If the patch is resting on the head, then its orientation is defined by the orientation of the head at the point where it is resting.

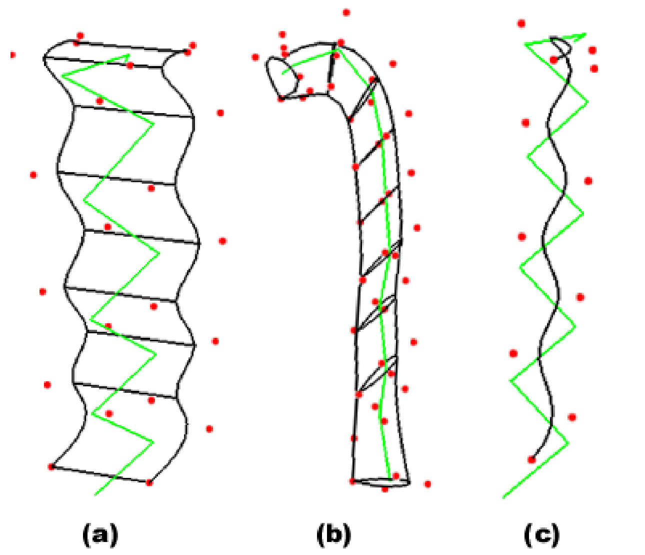

Figure 3: Model Representations Using D-NURBS and the Base Skeleton. *(a) A patch; (b) A cluster; (c) An individual strand. The red dots are the control points and the green lines are the base skeletons.*

A patch is typically used to represent the inner most layers of hair or parts of hair that are not visible to the viewer. It is the coarsest (lowest) level of detail used for modeling hair.

3.4 Clusters

The clusters are represented as generalized cylinders created with texture-mapped NURBS surfaces, as shown in Fig. 3(b). Each cluster is based on one skeleton curve located at the center of the cluster. A radius is specified for the top and the bottom of each cluster. The radius is then linearly interpolated at each skeleton node point; this allows the thickness to vary down the length of the cluster. At each skeleton node, a circular cross-section is created based on the radius value at that node. Four control points are used to create this circular shape of the cross-section for each skeleton node. Thus, a skeleton made up of n points will create a cluster of $4n$ control points.

Once the cross-section has been defined, the four control points are rotated based on the orientation of the cluster skeleton at that point. Rotating the control points is a necessary step to give the cluster model a realistic, smooth appearance and shape, especially for wavy and curly hair.

A cluster is used to model the intermediate layers of hair and often makes up the majority of the body of visible hair. Whenever appropriate, it is far less costly to represent a group of hair using the cluster model, instead of a large number of individual strands.

3.5 Individual Strands

Each individual strand is modeled as a D-NURBS curve with n control points, as shown in Fig. 3.3 (c). A small number of individual strands can also be grouped similar to that of the wisps [KAT93a, PCP01] so that only one skeleton model is used to represent a group of individual strands.

Most human heads have a few individual strands that are separate from the body of their hair, commonly referred to as “fly-aways”. These type of small imperfections are usually only noticeable when viewing the hair closely. By including them in the model, it adds to the realism. Since these fly-away hairs are primarily noticeable around the edges of the hair model and around the silhouette of the head, they appear at the silhouette of the hair model. The ability to see these individual strands is what makes this representation a finer detailed model of hair than the clusters.

3.6 Hairstyling

The skeleton controls the motion and the shape of each hair section and is responsible for the overall style of the hair. Any shape or style of hair can be specified for the hair by stipulating the rest angles θ_0 and ϕ_0 (see Fig. 2) of each control point of the skeleton. Straight hair can be created by assigning θ_{i0} to 0 and ϕ_{i0} to 0 for each control point $\mathbf{p}_i$ of the skeleton. In addition, we can create a wavy hair style by zig-zagging the position of the control points down the length of the skeleton. A zigzag or wavy skeleton is created by assigning each θ_{i0} to a certain angle between 0 and 90 degrees and then the values of ϕ_{i0} alternate by 180 degrees. We let θ_{i0} be the angle for specifying degree of waviness, then

$$\begin{cases} \phi_{i0} = 90, & \text{if } (i\%2 == 0) \\ \phi_{i0} = -90, & \text{otherwise.} \end{cases}$$

In addition, ringlet or spiral curls can also be created through the skeleton. Similar to wavy hair, to specify a ringlet curl, the θ_{i0} value is assigned a curl factor or an angle value between 0 and 90 degrees. Then, to achieve the spiral effect, each ϕ_{i0} value increments by 90 degrees down the length of the skeleton. We let θ_{i0} be the angle for specifying degree of curliness, then

$$\begin{cases} \phi_{i0} = 0, & \text{if } (i\%4 == 0) \\ \phi_{i0} = 90, & \text{if } (i\%4 == 1) \\ \phi_{i0} = 180, & \text{if } (i\%4 == 2) \\ \phi_{i0} = -90, & \text{otherwise.} \end{cases}$$

The ringlet curl has more complexity involved than modeling straight hair since more control points are needed to define the curl. To get a tighter spiral curl, the segment size between each control point is made smaller. Decreasing the segment size between each control point requires adding more control points to keep the overall length of the hair.

A skeleton model can be used to create various hairstyles. Each LOD is generated from the skeleton model and thus, each LOD can be used to represent any of these hairstyles. Since the individual strand grouping is the finest level of detail, this representation gives the best visual results for each hairstyle. Both the strand groupings and the clusters are able to accurately depict the shape defined by the user. While the patch representation gives better visual results for straight hair, it can also be used to model wavy and curly hair, but not in as fine a detail as the other two representations. However, the patch representations are only used when fine detail cannot be observed by the viewer. Patches are used the most at distances far from the viewer when exact detail cannot be noticed, or when the hair is not in sight by the viewer. Thus, while the patch cannot give as accurate of a representation of all of the hairstyles as the other two LODs, it is not apparent to the viewer. Criteria for choosing an LOD is discussed in further detail in Section 7. Three different hairstyles, straight, wavy and curly, are shown in Fig. 4.

4 Dynamic Simulation

The use of the same underlying skeleton model for each hair representation, i.e. patches, clusters, and individual strands, provides a straightforward mechanism for switching between different levels-of-detail. In this section, we describe the dynamic model of the skeleton, and explain our hair simulation algorithm which uses a combination of these three representations.

4.1 Basic Physics Model

The physics of motion for the base skeleton is similar to those described in [AUK92, KAT93a]. The motion of the skeleton is governed by the forces that are applied to each node point. The force measured from the angular springs of the skeleton model, F_{spring}, helps hold the specified hairstyle during the simulation. As external forces, such as wind blowing through the hair, change the initial hairstyle, the spring force, F_{spring}, works to bring the style back to its resting position. Thus, if a section of hair is specified to be curly, the F_{spring} force works to maintain the positioning.

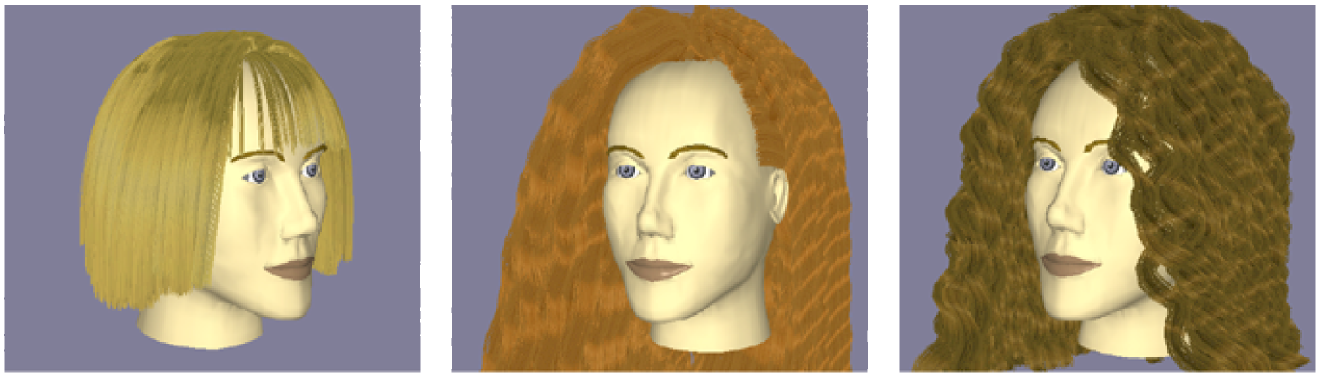

Figure 4: Three different hair styles generated using our simplified representations. *From left to right: straight, wavy, and curly*

$F_{i_{spring}}$, force for the ith node of the skeleton, is calculated by combining the forces for F_{i_θ} and F_{i_ϕ}.

$$F_{i_\theta} = -k_\theta(\theta_i - \theta_{i0}),$$
$$F_{i_\phi} = -k_\phi(\phi_i - \phi_{i0}),$$

where k_θ and k_ϕ are angular spring constants and θ_{i0} and ϕ_{i0} are initial angles for node i.

Other forces that act on the hair are collision forces, F_{col}, which will be described in the Section 5, gravity, $F_{gravity}$, and other external forces, F_{ext}, such as wind. The total force that is to be applied to the node point is calculated by summing all of these forces together. That is,

$$F_{i_{total}} = F_{i_{ext}} + F_{i_{spring}} + F_{i_{col}} + F_{i_{gravity}},$$

calculates the total force, $F_{i_{total}}$, to be applied to a given node $\mathbf{p}_i$ of the skeleton.

4.2 Transitioning between LODs

One of the most crucial aspects of using LODs in a visual simulation is the capability to switch between different levels of simplified representations smoothly. Ideally, a switch between the levels should not be apparent to the viewer. In order to avoid a sudden jump or popping in the graphical display, it is necessary that the motion and positioning of the hair remain consistent throughout the transition.

Since the motion of each LOD is based on the same underlying skeleton model, we can use this formulation to move from one level to another seamlessly with little or no visual artifacts. Thus, when a switch is made, the skeleton of the new level of detail inherits the dynamics state of the skeleton of the previous level. When a section switches from a simplified representation to a higher level of detail, the dynamics states of the new skeletons are interpolated from the skeleton and its neighbors of the previous level.

For example, if the current LOD is a 2D patch, the algorithm refines the hair model and makes a transition to multiple clusters. During this transition, a single patch is typically split into several, say five, clusters. This implies that this section of hair transitions from a model with one skeleton to another one with five skeletons. During the transition from a patch to five clusters, the center cluster inherits the exact skeleton values as the patch since the skeleton of the center cluster is at the same location as the patch skeleton. The remaining new cluster skeletons are computed by interpolating their values directly from the patch. Each cluster skeleton interpolates values from the patch skeleton based on the position of its root, or the first skeleton control point, $\mathbf{p}_0$, in relation to the root of the patch skeleton. Moreover, the five new clusters cover the same area as the previous patch.

The transitions are even more straightforward going in the reverse direction. As we move from five clusters back to a single patch, the skeleton of the patch simply inherits the skeleton of the center cluster. Transitions between the clusters and individual strands are performed in a similar manner.

By using these simplified representations together, our framework automatically switches between different LODs of hair and simplifies the dynamics of hair as needed, using the same base skeleton. Criteria for switching between LODs is discussed in Section 7. A patch can model the largest portion of hair and its simulation requires as little computation as a single strand grouping or a cluster of hair. When appropriate, patches are used to accelerate the simulation of hair, while maintaining some high-level behavior of the hair dynamics. Clusters are used in a similar manner to accelerate the simulation.

5 Collision Detection

It is important to check whether the hair has collided with an object. The hair will always be in contact with at least the scalp of the head. Due to the high complexity of hair, this can be a costly computation. It is crucial that the collision detection is performed efficiently since it is an important part of the overall running time.

5.1 Bounding Volume Hierarchy

There are many techniques known for collision detection. Some of the commonly used algorithms for general models are based on the use of bounding volume hierarchies (BVHs). A tree of bounding volumes (BVs) is pre-computed offline to enclose sets of geometric primitives, such as triangles or NURBS patches. Different types of bounding volumes have been used for collision detection including axis-aligned bounding boxes (AABBs), oriented bounding boxes (OBBs), spheres, etc. To perform collision detection using BVHs, two objects are tested by recursively traversing their BVHs [LGLM00].

5.2 Swept Sphere Volumes

To perform collision detection on the different representations of the hair, we use the family of "swept sphere volumes" (SSV) [LGLM00] to surround the hair. SSVs are a family of bounding volumes that correspond to a core skeleton grown outward by some offset. The set of core skeletons may include a point, line, or n-gon. We have chosen to use arbitrarily oriented rectangles, instead of n-gons, as the most complex skeleton in our current framework. See Fig. 5 for examples of SSVs. More precisely, let C be the core skeleton and S be a sphere of radius r. Each SSV, B, can be defined as:

$$B = C \oplus S = \{\boldsymbol{c} + \boldsymbol{s} |\ \boldsymbol{c} \in C, \boldsymbol{s} \in S\}$$

To perform collision queries on a pair of arbitrary SSVs, we can simply perform intersection tests on the corresponding pair of core skeletons and then subtract the appropriate offset (radius of each SSV). We have chosen SSVs as BVs, because their shapes match well with our three simplified representations of hair. Different SSVs provide varying tightness. For clusters and groups of individual strands, line swept spheres are the best candidates for

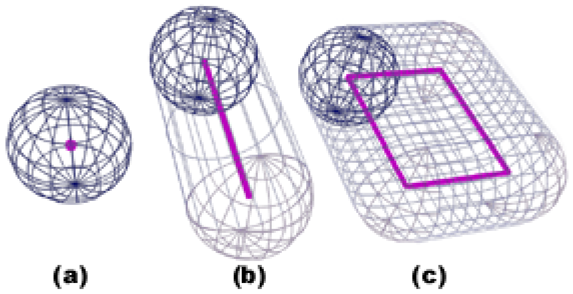

Figure 5: A Family of Swept Sphere Volumes. *a) Point swept sphere (PSS); b) Line swept sphere (LSS); c) Rectangle swept sphere (RSS). The core skeleton is highlighted in purple.*

each section, while rectangle swept spheres provide the better fit for patches, and the point swept sphere for the head at the top level. Using the family of SSVs as the underlying BVs, collision detection can be performed efficiently and easily using any combination of the three representations together.

5.3 Constructing BVHs of SSVs

For each section of the skeleton model that controls the shape of the three hair representations, we pre-compute a BV of SSVs for constructing BVHs for the hair and the nearby objects (e.g. head, neck, etc.) that interact with the hair. For a skeleton made up of n nodes, there are $n-1$ sections, and thus $n-1$ BVHs of SSVs. The variable thickness of each section defines the radius of the root SSV along its length.

In order to compute a BV for a patch, we use the four control points of the patch that outline a section to create a BV enclosing it. This is performed for each of the $n-1$ sections. Similarly for the cluster representation, we use the eight control points that define a section (four control points that make up the cross-section at the top of the section and 4 control points at the bottom) to create a BV.

For individual strands, we perform collision detection for each strand or strand grouping in a similar manner. We compute an LSS around the skeleton that defines each section with a radius defining the thickness. For each individual strand, the radius is zero. Thus, performing collision queries on each individual strand is similar to checking intersections on the skeleton curve. Moreover, the radius of each BV is varied based on the thickness of each grouping.

Once the BVs are computed for each section of the hair using our simplified representations, we construct a BVH of SSVs using top-town hierarchy construction for the hair and each object in the scene as a pre-computation. During the runtime simulation, the BVHs are used to perform collision queries, and are lazily updated on the fly.

5.4 Collision Response

Our algorithm based on the BVHs of SSVs automatically detects collision between the hair and the head or other nearby objects in the scene. Collision response is then computed using the output of our collision detection method. Since the same skeleton model is used for each representation, whenever a collision is detected we calculate response forces using the same method for all of the representations. Once a collision between the hair and head has been detected, we first determine a direction to apply the response force. The BVH used on the head for collision detection finds the triangles on the head that are colliding with the the section of the hair. After that, we compute the average normal direction, $\mathbf{N}_{avg}$, of these triangles by summing the normal associated with each triangle and dividing them by the total number of intersected triangles, n. We use this average normal, $\mathbf{N}_{avg}$, as the direction along which we apply the response force. We define $\mathbf{N}_{avg}$ as,

$$\mathbf{N}_{avg} = \frac{\sum \mathbf{N}_i}{n}; \quad for \;\; i = 1, \ldots, n,$$

where $\mathbf{N}_i$ is the normal of the ith intersecting triangle on the head that has collided with the section of hair.

The magnitude of the response force is based both on the speed of the given hair section, and the speed of the head. Each hair section is made up of a rigid section of the skeleton model. This rigid skeleton section has two skeleton control points, one at the top of the section and the other one at the bottom. The ith SSV of a grouping of hair has control points $\mathbf{p}_i$ at the top and $\mathbf{p}_{i+1}$ at the bottom. The velocity of the section of hair is calculated at the top of the section, $\mathbf{v}_t$, and at the bottom of the section, $\mathbf{v}_b$, by computing the change in position of each control point over the last time step.

$$\mathbf{v}_t = (\mathbf{p}_i(\tau) - \mathbf{p}_i(\tau - \delta_t))$$
$$\mathbf{v}_b = (\mathbf{p}_{i+1}(\tau) - \mathbf{p}_{i+1}(\tau - \delta_t)),$$

where i is the hair section, τ is the current time and δ_t is the time step.

These calculated speeds are used to compute the magnitude of the response force. Since each skeleton control point is connected to two BVs (each of them is a SSV), except the first and last control points of each skeleton, we disperse the magnitude of the force between each control point. The resulting response forces are computed as:

$$F_i = 0.5 * \mathbf{v}_{ti} * \mathbf{N}_{avg}$$
$$F_{i+1} = 0.5 * \mathbf{v}_{bi} * \mathbf{N}_{avg}$$

to be applied on the control points, $\mathbf{p}_i$ and $\mathbf{p}_{i+1}$, respectively. A frictional force is also applied to each skeleton control point when a section of hair collides with the head. The average normal vector, $\mathbf{N}_{avg}$, calculated for the collision response force is used to compute the plane tangent to the head at the point where the collision occurs. The forces applied to a skeleton control point, including internal forces, gravity, and outside forces such as wind, are then summed. This resulting force vector is projected onto the tangent plane to obtain the component of the forces acting upon the control point in the tangent direction. Since the frictional force will be opposing the motion of the hair along the head, the direction of the component force is then reversed to obtain a direction to apply the frictional force. The magnitude of the component force is multiplied by a frictional constant value, and the frictional force is applied to the corresponding skeleton control point.

6 Rendering

Various techniques for hair and fur rendering have been investigated, such as [KK89, Len00, KH00]. In our system, a mixed model of discrete and continuous geometric levels-of-detail is used to render the hair representations. Each representation has a distinct type of geometric representation: each strand is rendered as a NURBS curve, each cluster is rendered as a ruled NURBS surface (i.e. surface generated by sweeping a curve around an axis), and each strip is rendered as a NURBS surface. By varying the tessellation parameters as a function of distance to the viewer, an almost continuous level-of-detail is provided for each geometric hair representation.

One of the key aspects of rendering hair is simulating the interaction of light. Traditional lighting models depend on the surface normal. However, given an individual strand to render, there is no clear choice of a normal vector. Moreover, hair is an anisotropic surface. Lighting of anisotropic surfaces, often seen with hair, satin, and brushed metals, changes according to the viewing direction. We simulated anisotropic lighting based on the the methods proposed by [HS98, Ban94]. These methods use a texture map, indexed by the eye space light direction and the view direction to simulate anisotropic lighting.

Traditional lighting is not applied to our hair representations. Instead, the simulated anisotropic lighting is blended with the underlying color of the primitive during a single OpenGL rendering pass. For strands, random shades in a range of hair color are assigned to each individual strand. The NURBS surfaces that are used for clusters as well as patches, are assigned a texture map using a similar color range.

Transitions between strand, cluster, and patch representations cause little visual disturbance. This smooth transitioning has to do with the restrictions placed on when patches are used, and due to the design of the hairstyle. Patches are only used when the hair is completely occluded, or at a great distance from the user. In addition, hair sections around the part of the hairstyle and around the face are set to not transition below a cluster representation. By not allowing these hair sections to transition to a patch representation, the volume and silhouette of the hairstyle are preserved.

7 Choosing Hair Representations

Given the three representations of a section of hair, a single representation for modeling and simulating that section is computed on the basis of several criteria. We have used the following components in our current implementation:

- Visibility
- Viewing distance
- Hair motion

7.1 Visibility

If a viewer cannot see a section of hair, that section does not need to be simulated or rendered at its highest resolution of individual strand groupings. Hair cannot be seen by the viewer if it is not in the field of view of the camera or if it is completely occluded by the head or other objects in the scene.

If a section of hair in strand groupings is normally simulated using five skeletons but is occluded by other objects, we simulate that section of hair using one larger patch, and therefore, one skeleton. Simulating only one skeleton instead of fives cuts the computation time by a factor of five. When that section of hair comes back into view, it is important that the placement and action of the hair are consistent with the case when no levels-of-detail are used at all. As a result, we need to continue simulating the hair section that is currently not visible. For example, this can happen when the occluded hair is blown back into view by a gust of wind. Also, when the hair comes back into view by either the camera moving or the head moving, it is necessary that the hair be represented in a manner that is consistent with the viewer's understanding of the simulated scene. Simulating the occluded hair at a lower level-of-detail allows the simulation to remain visually consistent while reducing the computation time.

In addition, when hair section is occluded, its rendering can also be simplified. Hair that is completely occluded does not need to be rendered at all. Therefore, when a section of hair is occluded, we simulate the hair that might normally be represented as either clusters or strand groupings as patches that use fewer skeletons and these sections are not rendered.

The way we compute whether a section of hair is occluded is important. In our current implementation, we perform a simple conservative occlusion test. We put a bounding box around the head and fit the box so that it is slightly smaller than the head. We then shoot a ray from the camera to the top and bottom of each hair representation. If the hair is in cluster form, then the rays originate from the camera and end at the top and the bottom of the cluster. We check whether the rays intersect with any of the six faces of the box. If both of the rays intersect the box before reaching the hair section, then we conclude that the entire section of hair is occluded. It is possible to use more sophisticated occlusion culling algorithms, such as [ZMHH97], to perform these tests.

7.2 Viewing Distance

The next factor to consider in choosing a hair representation is the distance from the viewer to the hair. Hair that is far from the viewer cannot be seen in great detail. As a result, we do not use the highest resolution representation for it. We can estimate the amount of detail that will be seen by the viewer by computing the screen space area that the hair covers.

For example, a given section of hair may be represented by a single patch, a group of three clusters, or a group of five strand groupings. Each of these representations is designed to cover a similar amount of world space. Since all the of these representations occupy a similar amount of world space, we can use the control skeleton of the patch as an estimate to the amount of screen space area a given hair section occupies. For example, to determine the screen coverage of a strip, we project a line from the first skeleton node to the last skeleton node into the screen space. If the amount of screen space covered by this line exceeds the pre-determined maximum allowable size for a strip, then the given hair section will be rendered as a cluster. Similarly, if the amount of screen space covered by the line exceeds the maximum allowable size for a cluster, then it will be rendered as individual strands.

7.3 Hair Motion

Another aspect to consider in choosing the appropriate representation is the current motion of the hair. If the hair is not moving at all, then a large amount of computation is not needed to animate it and we can, therefore, use a lower level of detail. When the person makes sudden movement, e.g. shaking his or her head, or a large gust of wind blows, a higher-detailed simulation is used. When a large force is applied to the hair, such as wind, often individual strands can be seen even by a person who is normally too far away to see individual strands of hair that are not in motion. Also, if the hair is moving around a lot, then it is likely to have many collisions with the objects (e.g. the body) around it. When the hair is colliding due to motion, using a higher level-of-detail results in more accurate collision detection and response.

We choose the particular LOD based on hair motion by first determining the skeleton control point in the current representation that has the strongest force acting on it. This value is compared to certain thresholds defined for strands or clusters. If the force acting on the skeleton is not high enough to be represented as either strands or clusters, then the hair is modeled as a patch.

7.4 Combining Criteria

At any given time during a simulation, a head of hair is represented by multiple LODs. Each section of hair uses its own parameter values to trigger a transition. The sections of hair that have a root location at the top of the head, and therefore more viewable, remain at the individual strands grouping level longer than the sections of hair that are located at the base of the neck. Thus, even if these two sections are at the same distance from the camera and have the same motion, it is more important that the top layer be represented as individual strands instead of clusters, since it is directly in the view. When determining an appropriate LOD to use, we first test that section of hair for occlusion. If the hair is not visible to the viewer then we automatically simulate it as a patch and do not render it. In this case, no other transition tests are needed. If the section of hair is visible, we perform the motion and distance tests described above. The LOD representation is chosen based on whichever of these two tests requires a higher detail and fails to satisfy the user-specified screen-space error tolerance. Fig. 6 shows an example of multiple LODs being used at once for a simulation. The use of different representations for the hair is virtually unnoticeable to the viewer.

8 Results and Comparisons

We have implemented our automated simplification algorithm for hair modeling in C++. We modified and extended the publically available proximity query package, PQP [LGLM00], to perform collision detection. The simulation results are displayed using OpenGL.

8.1 Implementation Issues

In order to speedup the LOD transitions at runtime, many components are precomputed. An interactive hairstyling tool was created to allow the user to place the skeletons on the head at the desired locations. The styling tool also lets the user set parameters for curly

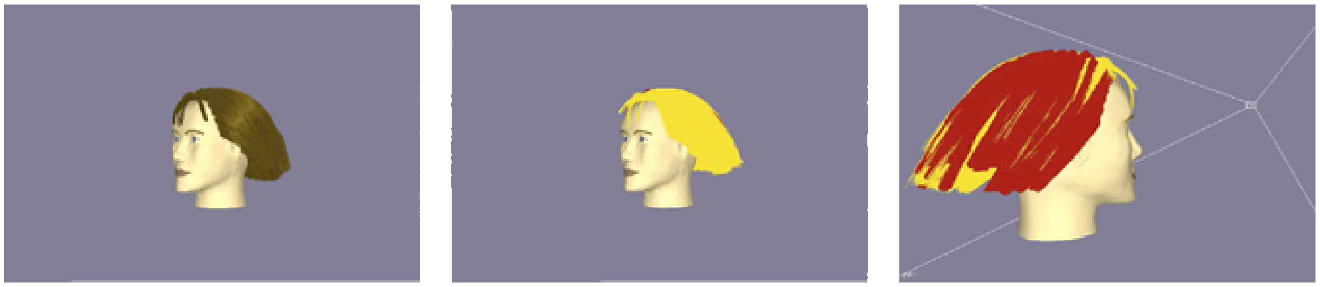

Figure 6: Simulation of wind blowing through the hair as the camera zooms away from the head. *The left image shows actual simulation using LODs. The center image shows the same simulation but each representation is color coded - strands are shown in yellow and clusters are shown in red, patches are representing occluded hair and are not visible at this point in the simulation. The right image shows the same simulation from the rear using the same color scheme as the center image. The view frustum of the front camera is shown in white.*

or wavy hair automatically by setting the curl factor or wave factor, which are the angles that control the degree of the curl or wave, respectively. The size (length, width, radius, etc.) of each representation is also set using this styling tool.

We also precompute the corresponding BVH for each representation of hair to be used for collision detection. Therefore, during an LOD transition, the only values that need to be updated are the positions of the skeleton nodes. Moreover, we use a simplified representation of the head model in performing collision detection between the head and the hair in our simulation.

8.2 Performance Comparisons

We have tested our implementation of our framework on various scenarios. (See the supplementary videotapes for these simulation.) Fig. 7 shows a sequence of snapshots taken from a hair simulation using our simplified representations.

We also compared the performance for the overall dynamic simulation (not including collision detection), collision detection, and rendering using different representations. Table 1 gives a detailed comparison of the running time using a combination of simplified representations (indicated as LODs) against each one of the simplified representations (Strands, Clusters, and Patches). For this simulation, a curly hairstyle (shown in Fig. 8) was used and the timings were taken as the camera moves away from the hair. Table 2 shows results from a similar simulation using the same hairstyle and camera motion, but in this simulation a wind force was applied to the scene. For this benchmark, we used 455 individual strand groupings, which contained a total of 37,765 strands of hair, which were represented using only 91 patches or only 273 clusters. A combination of all three representations was automatically determined by our framework at any given time during the simulations. Timings were taken on a PC equipped with an Intel Pentium IV 1.6 GHz processor, 512 MB main memory and GeForce 2 graphics card.

As expected, patches provide the best overall performance in both simulation and rendering time, since it is the coarsest (lowest) LOD of hair. But, a combination of simplified representations using our framework offers significant performance advantages over the use of individual strands alone. While it renders images of almost as good visual quality as that of individual stands, our LOD implementation gave better timing performances than modeling with individual strands in simulation, collision detection, as well as rendering.

8.3 Analysis and Discussion

The impetus of this research is to explore the use of simplified representations for modeling hair to automatically generate its *aggregate* behavior, while preserving the visual fidelity of the overall simulation. It is difficult to meaningfully quantify the computational errors introduced by the use of simplified representations for modeling hair. However, we can subjectively evaluate the resulting simulation by performing side-by-side comparison on the visual quality of rendered images.

Breakdown	LODs	Strands	Clusters	Patches
Dyn Sim	0.183	1.071	0.099	0.030
Col Detect	0.077	0.200	0.122	0.040
Rendering	0.385	1.687	0.656	0.158

Table 1: Performance Comparison. *Simulation for a camera zooming out away from the head. The performance numbers are measured in seconds per frame.*

Breakdown	LODs	Strands	Clusters	Patches
Dyn Sim	0.356	1.134	0.105	0.032
Col Detect	0.107	0.168	0.101	0.034
Rendering	0.785	1.734	0.665	0.158

Table 2: Performance Comparison. *Simulation for wind blowing through the hair as the camera zooms away from the head. The performance numbers are measured in seconds per frame.*

Fig. 8 shows the images of the simulation results using a combination of simplified representations vs. each of the three representations. (The timings were given in Tables 1 and 2.) We notice little degradation in the visual quality of the rendered image using LODs. However, while they offer the best computational performance, the image of hair simulated by patches appears sharp and angular, lacking a realistic appearance. The performance of our LOD combination is comparable to that of clusters. However, our approach can automatically place the computing resources at places where the hair is most visible to the viewer, and thus offer a much higher visual quality for the resulting simulation, as shown in Fig. 8.

One limitation of our algorithm is the slight popping that can occur if aggressive LOD transitions are used. This popping may be alleviated with motion warping between the three hair representations.

9 Summary and Future Work

In this paper, we present the use of levels of detail for modeling hair to accelerate dynamics computation, simplify collision detection and reduce rendering cost. We have demonstrated our system on animating different types of hair with varying styles. There are several possible directions to extend this research:

- Interactively modify the dynamics of the hair in the presence of other substances, such as water, styling gel, hair spray, etc.;

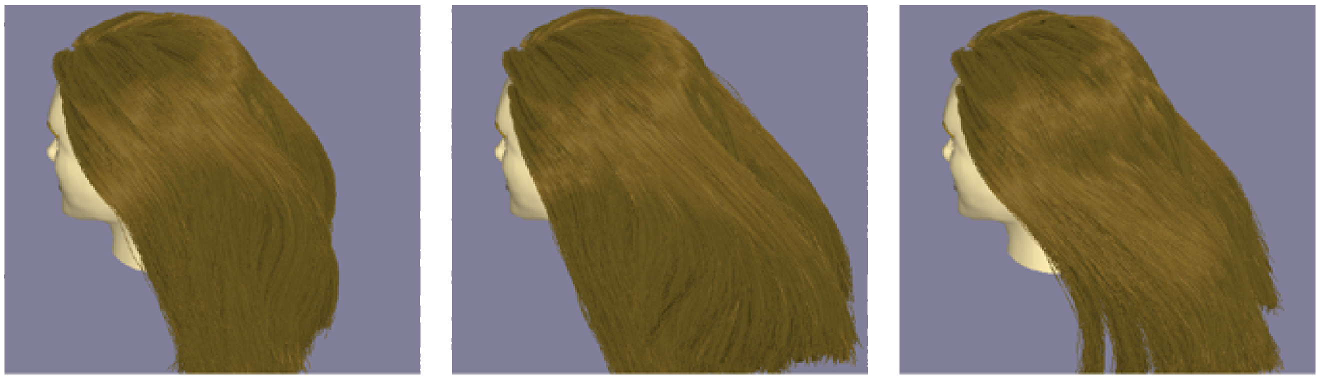
Figure 7: Dynamic Simulation of Hair Using LODs. *A sequence of snapshots (from left to right) are taken from a simulation of long, straight hair flowing in the wind.*

- Dynamically change the hair style, as the user combs or brushes the hair with a 3D user interface;
- Automatically generate desired simulation outcomes, given high-level user guidance.

References

[AUK92] K. Anjyo, Y. Usami, and T. Kurihara. A simple method for extracting the natural beauty of hair. *Computer Graphics*, 26(2):111–120, 1992.

[Ban94] David C. Banks. Illumination in diverse codimensions. In Andrew Glassner, editor, *Proceedings of SIGGRAPH '94 (Orlando, Florida, July 24–29, 1994)*, Computer Graphics Proceedings, Annual Conference Series, pages 327–334. ACM SIGGRAPH, ACM Press, July 1994. ISBN 0-89791-667-0.

[CF97] S. Chenney and D. Forsyth. View-dependent culling of dynamic systems in virtual environments. In *Proc. of ACM Symposium on Interactive 3D Graphics*, 1997.

[CH97] D. Carlson and J. Hodgins. Simulation levels of detail for real-time animation. In *Proc. of Graphics Interface 1997*, 1997.

[COM98] J. Cohen, M. Olano, and D. Manocha. Appearance preserving simplification. In *Proc. of ACM SIGGRAPH*, pages 115–122, 1998.

[CSDI99] L. H. Chen, S. Saeyor, H. Dohi, and M. Ishizuka. A system of 3d hair style synthesis based on the wisp model. *Visual Computer*, 15(4):159–170, 1999.

[DKMTT93] A. Daldegan, T. Kurihara, N. Magnenat-Thalmann, and D. Thalmann. An integrated system for modeling, animating and rendering hair. *Computer Graphics Forum (Proc. of Eurographics*, 12(3):211–221, 1993.

[FS93] Thomas A. Funkhouser and Carlo H. Séquin. Adaptive display algorithm for interactive frame rates during visualization of complex virtual environments. In James T. Kajiya, editor, *Computer Graphics (SIGGRAPH '93 Proceedings)*, volume 27, pages 247–254, August 1993.

[GH97] M. Garland and P. Heckbert. Surface simplification using quadric error bounds. *Proc. of ACM SIGGRAPH*, pages 209–216, 1997.

[GTH98] R. Grzeszczuk, D. Terzopoulos, and G. Hinton. Neuroanimator: Fast neural network emulation and control of physics-based models. *Proc. of SIGGRAPH*, pages 9–20, 1998.

[HMT01] S. Hadap and N. Magnenat-Thalmann. Modeling dynamic hair as a continuum. *Computer Graphics Forum (Proc. of Eurographics 2001)*, 20(3), 2001.

[Hop96] Hugues Hoppe. Progressive meshes. In Holly Rushmeier, editor, *SIGGRAPH 96 Conference Proceedings*, Annual Conference Series, pages 99–108. ACM SIGGRAPH, Addison Wesley, August 1996. held in New Orleans, Louisiana, 04-09 August 1996.

[HS98] W. Heidrich and H.-P. Seidel. Efficient rendering of anisotropic surfaces using computer graphics hardware. *Proc. of Image and Multi-dimensional Digital Signal Processing Workshop (IMDSP)*, 1998.

[KAT93a] T. Kurihara, K. Anjyo, and D. Thalmann. Hair animation with collision detection. In *Models and Techniques in Computer Animation*, pages 128–38. Springer-Verlag, 1993.

[KAT93b] T. Kurihara, K. Anjyo, and D. Thalmann. Hair animation with collision detection, models and techniques. *Proc. of Computer Animation*, pages 128–138, 1993.

[KH00] C. K. Koh and Z. Huang. Real-time animation of human hair modeled in strip. *Eurographics CAS Workshop*, pages 101–112, 2000.

[KH01] C. K. Koh and Z. Huang. A simple physics model to animate human hair modeled in 2d strips in real time. *Proc. of Eurographics Workshop on Animation and Simulation*, 2001.

[KK89] James T. Kajiya and Timothy L. Kay. Rendering fur with three dimensional textures. In Jeffrey Lane, editor, *Computer Graphics (SIGGRAPH '89 Proceedings)*, volume 23, pages 271–280, July 1989.

[KN00] T.-Y. Kim and U. Neumann. A thin shell volume for modeling human hair. *Computer Animation*, 2000.

[Len00] J. Lengyel. Real-time fur. *Proc. of Eurogrpahics Workshop on Rendering*, 2000.

[LGLM00] E. Larsen, S. Gottschalk, M. Lin, and D. Manocha. Distance queries with rectangular swept sphere volumes. *Proc. of IEEE Int. Conference on Robotics and Automation*, 2000.

[LPFH01] J. Lengyel, E. Praun, A. Finkelstein, and H. Hoppe. Real-time fur over arbitrary surfaces. *Proc. of ACM Symp. on Interactive 3D Graphics*, 2001.

[Lue01] D. Luebke. A developer's survey of polygon simplification algorithms. *IEEE CG & A*, pages 24–35, May 2001.

[MTHK00] N. Magnenat-Thalmann, S. Hadap, and P. Kalra. State of the art in hair simulation. *Int. Workshop on Human Modeling and Animaton*, pages 3–9, 2000.

[OFL01] D. O'Brien, S. Fisher, and M. Lin. Simulation level of detail for automatic simplification of particle system dynamics. *Proc. of Computer Animation*, pages 210–219, 2001.

[PC01] F. Perbet and M. Cani. Animating prairies in real-time. *Proc. of ACM Symposium on Interactive 3D graphics*, 2001.

[PCP01] E. Plante, M. Cani, and P. Poulin. A layered wisp model for simulating interactions inside long hair. *Proc. of Eurographics Workshop on Animation and Simulation*, 2001.

[QT96] Hong Qin and Demetri Terzopoulos. D-NURBS: A physics-Based framework for geometric design. *IEEE Transactions on Visualization and Computer Graphics*, 2(1):85–96, March 1996. ISSN 1077-2626.

[SZ98] P. Schröder and D. Zorin. Subdivision for modeling and animation. *ACM SIGGRAPH Course Notes*, 1998.

[XY01] Z. Xu and X. D. Yang. V-hairstudio: An interactive tool for hair design. *IEEE Computer Graphics and Applications*, 21(3):36 –43, 2001.

[Yu01] Y. Yu. Modeling realistic virtual hairstyles. *Pacific Graphics*, 2001.

[ZMHH97] H. Zhang, D. Manocha, T. Hudson, and K. Hoff. Visibility culling using hierarchical occlusion maps. *Proc. of ACM SIGGRAPH'97*, 1997.

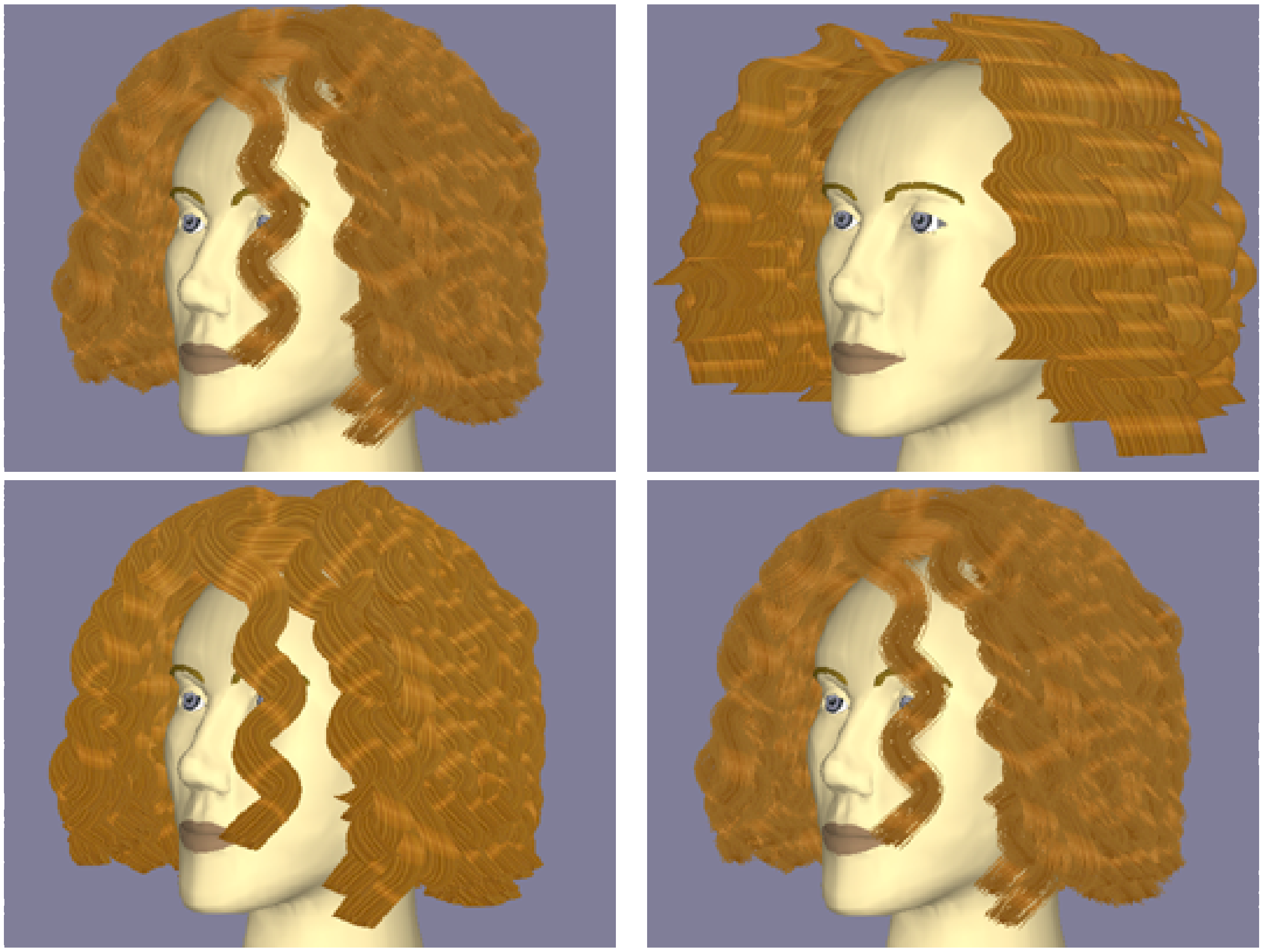

Figure 8: Comparison of Rendered Images. *From left to right, top to bottom: (a) LODs; (b) Patches; (c) Clusters; (d) Individual Strands. Notice the hair generated using patches alone looks very unrealistic with "flat" edges and sharp angles and the hair generated with clusters only looks less natural without the individual strands in its silhouette.*