

# **Interactive Hair Modeling: Techniques and Challenges**

**Edited by Ming C. Lin and Nadia Magnenat-Thalmann**

Copyright © 2006



# ORGANIZERS

Ming C. Lin  
Department of Computer Science  
University of North Carolina  
Chapel Hill, NC 27599-3175  
U. S. A.  
lin@cs.unc.edu  
PHO: 919-962-1974  
FAX: 919-962-1799

Nadia Magnenat-Thalmann  
MIRALab, CUI University of Geneva  
24 rue du General Dufour  
CH-1211 Geneva SWITZERLAND  
thalmann@miralab.unige.ch  
PHO: +41-1-379-77-69  
FAX: +41-1-379-77-80

# LECTURERS

- **Florence Bertails, INRIA Rhone-Alpes**  
[Florence.Bertails@imag.fr](mailto:Florence.Bertails@imag.fr)
- **Marie-Paule Cani, INRIA Rhone-Alpes**  
[Marie-Paule.Cani@imag.fr](mailto:Marie-Paule.Cani@imag.fr)
- **Ming C. Lin, University of North Carolina at Chapel Hill**  
[lin@cs.unc.edu](mailto:lin@cs.unc.edu)
- **Nadia Magnenat-Thalmann, University of Geneva**  
[thalmann@miralab.unige.ch](mailto:thalmann@miralab.unige.ch)
- **Kelly Ward, Walt Disney Feature Animation**  
[wardk@cs.unc.edu](mailto:wardk@cs.unc.edu)



# BIOGRAPHICAL SKETCHES OF LECTURERS

**Florence Bertails** is a PhD student in Computer Science at Institut National Polytechnique de Grenoble (INPG), France. She graduated in 2002 from the Telecommunication School of INPG and received a MSc in Image, Vision and Robotics. Her research focuses on hair animation and physically-based modeling. She presented her work on hair simulation at international conferences such as the ACM-EG Symposium of Computer Animation and Eurographics, and received the best student paper award at Graphics Interface 2005 for interactive rendering of animated hair. She is currently investigating new models for hair in collaboration with physicists and hair scientists from L'Oréal Research.

Website: <http://www-evasion.imag.fr/Membres/Florence.Bertails>

**Marie-Paule Cani** is a Professor of Computer Science at the Institut National Polytechnique de Grenoble (INPG), France. A graduate from the Ecole Normale Supérieure, she received a PhD in Computer Science from the University of Paris Sud in 1990 and the “habilitation” degree from INPG in 1995. She was nominated at the Institut Universitaire de France in 1999. She is the head of the INRIA research group EVASION which she created in 2003, and vice-director of the research lab GRAVIR (Computer GRAPHics, Computer VISION and Robotics), the joint lab of CNRS, INPG, INRIA and UJF to which EVASION belongs. Her main research interests cover physically-based simulation, implicit surfaces applied to interactive modeling and animation and the design of layered models incorporating alternative representations and LODs. Recent applications include pattern-based texturing, the animation of natural phenomena such as lava-flows, ocean, vegetation and human hair, real-time virtual surgery and interactive sculpting techniques. She co-chaired the EUROGRAPHICS Workshops on Implicit Surfaces (1995) and Computer Animation and Simulation (2001), now respectively merged into IEEE Shape Modeling International (SMI) and the ACM-EG Symposium on Computer Animation (SCA). She was the Paper co-Chair of EUROGRAPHICS 2004, Conference co-chair of SMI'2005, and is the Paper co-chair of SCA'2006. She served in the program committees of a number of international conferences, including SIGGRAPH (2002 and 2005) and EUROGRAPHICS (1996 and 2003). She belongs to the editorial board of Graphical Models (Academic Press) since 2001. She is the president of the French Chapter of EUROGRAPHICS, which she created in 2003.

Website: <http://www-evasion.imag.fr/Membres/Marie-Paule.Cani>

**Ming Lin** received her B.S., M.S., Ph.D. degrees in Electrical Engineering and Computer Science all from the University of California, Berkeley. She is currently a full professor in the Computer Science Department at the University of North Carolina (UNC), Chapel Hill. She received the NSF Young Faculty Career Award in 1995, Honda Research Initiation Award in 1997, UNC/IBM Junior Faculty Development Award in 1999, and UNC Hettleman Award for Scholarly Achievements in 2002. Her research interests include haptics, physically-based modeling, robotics, and geometric computing and has authored over 140 refereed publications in these areas. She has served as a program committee member for many leading conferences on virtual reality, computer graphics, robotics and computational geometry. She was the general

chair and/or the program chair of the First ACM Workshop on Applied Computational Geometry, 1999 ACM Symposium on Solid Modeling and Applications, the Workshop on Intelligent Human Augmentation and Virtual Environments 2002, ACM SIGGRAPH/EG Symposium on Computer Animation 2003, and ACM Workshop on General-Purpose Computing on GPUs 2004, Computer Animation and Social Agents 2005, Eurographics 2005 Tutorial, and Eurographics Symposium on Virtual Environments 2006. She also serves on the Steering Committee of ACM SIGGRAPH/EG Symposium on Computer Animation. She is an associated editor and a guest editor of several journals and magazines, including *IEEE Transactions on Visualization and Computer Graphics*, *International Journal on Computational Geometry and Applications*, *IEEE Computer Graphics and Applications*, and *ACM Computing Reviews*. She also edited the book "*Applied Computation Geometry*." She has given several lectures and invited presentations at SIGGRAPH, GDC, and many other international conferences.

Website: <http://www.cs.unc.edu/~lin/> and <http://gamma.cs.unc.edu/>

**Nadia Magnenat-Thalmann** has pioneered research into virtual humans over the last 25 years. She obtained several Bachelor's and Master's degrees in various disciplines (Psychology, Biology and Chemistry) and a PhD in Quantum Physics from the University of Geneva. From 1977 to 1989, she was a Professor at the University of Montreal and led the research lab MIRALab in Canada. She moved to the University of Geneva in 1989, where she founded the Swiss MIRALab, an internationally interdisciplinary lab composed of about 30 researchers. She is author and coauthor of a very high number of research papers and books in the field of modeling virtual humans, interacting with them and in augmented life. She has received several scientific and artistic awards for her work, mainly on the Virtual Marilyn and the film RENDEZ-VOUS A MONTREAL, but more recently, in 1997, she has been elected to the Swiss Academy of Technical Sciences, and has been nominated as a Swiss personality who has contributed to the advance of science in the 150 years history CD-ROM produced by the Swiss Confederation Parliament. She has directed and produced several films and real-time mixed reality shows, among the latest are the UTOPIANS (2001), DREAMS OF A MANNEQUIN (2003) and THE AUGMENTED LIFE IN POMPEII (2004). She is editor-in-chief of the Visual Computer Journal published by Springer Verlag and coeditor-in-chief of the Computer Animation & Virtual Worlds journal published by John Wiley.

Website: [http://www.miralab.unige.ch/2team/team\\_director.cfm](http://www.miralab.unige.ch/2team/team_director.cfm)

**Kelly Ward** is currently a software engineer at Walt Disney Feature Animation where she works on look development and hair generation tools for feature films. She received her M.S. and Ph.D. in Computer Science from the University of North Carolina, Chapel Hill in 2002 and 2005, respectively. She received a B.S. with honors in Computer Science and Physics from Trinity College in 2000 where she was named the President's Fellow in Physics in 1999-2000. During the summers of 2002 and 2003, she worked at Intel Corporation and Rhythms and Hues Studios. Her research interests include hair modeling, physically-based simulation, and computer animation. She has given several contributed presentations and invited lectures on her research related to hair modeling at international venues.

Website: <http://www.cs.unc.edu/~wardk>

# COURSE SYLLABUS

This course is designed to cover recent techniques and rising challenges in hair modeling, including hair styling, simulation and rendering. We have assembled an excellent team of researchers and developers from both academia and industry to cover topics on fundamental algorithm design and their applications. Techniques presented offer a fine balance between algorithmic performance, visual quality, and physical correctness. We will also discuss the applications and system requirements for different areas, including entertainment, VR, and CAD/CAM industry.

## HAIR RENDERING

- *Interactive Self-Shadowing using GPUs* (Florence Bertails)
- *Real-time Rendering of Wet and Stylized Hair* (Kelly Ward)

## HAIR ANIMATION

- *Free-Form Deformation Based Scalable Approaches* (Nadia Magnenat-Thalmann)
- *Adaptive Wisp Trees* (Marie-Paule Cani)
- *Simulation Levels of Detail* (Ming C. Lin)
- *Animating Hair with Water and Styling Products* (Kelly Ward)

## VIRTUAL HAIR STYLING

- *Application Requirements of VR and CAD Systems* (Nadia Magnenat-Thalmann)
- *Industrial Perspectives* (Kelly Ward)
- *Physically-based Virtual Hair Salon* (Kelly Ward & Florence Bertails)
- *Interactive 3D Hair Dressing with Force Feedback* (Nadia Magnenat-Thalmann)

## **PRE-REQUISITES**

This course is for programmers and researchers who have done some implementation of 3D graphics. Familiarity with fundamentals of computer graphics, numerical linear algebra, differential equations, numerical methods, rigid-body dynamics, collision detection and response, physics-based illumination models, real-time techniques is highly recommended but not mandatory.

## **INTENDED AUDIENCE**

Target audiences include special effects developers, technical directors, researchers, and game developers, who are looking for innovation as well as proven methodologies in simulating real-time hair. During the course, audiences will obtain knowledge on state of the art and working solutions for hair animation.

# COURSE SCHEDULE

- 8:30am      **Introduction and Overview** (Lin & Magnenat-Thalmann)
1. Brief overview of the state of the art
  2. Roadmap for the course
  3. Introduction of the speakers

## SESSION I: TOWARD REAL-TIME HAIR ANIMATION

- 8:45am      **Animating Complex Hairstyles in Real-time** (Nadia Magnenat-Thalmann)
1. Lattice-based free-form deformation
  2. Building the hair mechanics
    - a. Mechanical model of the hair
    - b. The ether model
    - c. Handling collisions
  3. Lattice FFD for real-time hair animation

- 9:10am      **Modeling Hair Using Levels of Detail Representations** (Ming C. Lin)
1. Geometric representations and subdivision framework
    - a. Hair patches, clusters, and strands
    - b. Subdivision curves and surfaces
    - c. Continuous vs. discrete LODs
  2. Collision detection using a family of Swept Sphere Volumes
    - a. Hair mutual interaction
    - b. Hair and object collisions
  3. Contact response using simulation levels of detail
  4. Transition between different LODs
    - a. Viewing Distance
    - b. Visibility
    - c. Dynamic states

- 9:35am      **Adaptive Grouping and Subdivision of Simulated Hair** (Kelly Ward)
1. Construction of “Hair Hierarchy” based on 3 discrete hair LODs
  2. Implicit integration for stable and fast hair dynamic simulation
  3. Criteria for adaptive hair grouping and subdivision

- 9:50am      **Adaptive Wisp Trees** (Marie-Paule Cani)
1. Multiresolution hair geometry
  2. Adaptive the AWT
    - a. Splitting (separation)
    - b. Merging (fusion)
    - c. Position of non-active nodes
  3. Hair interaction
    - a. Response to mutual contacts
    - b. Collision with obstacles

10:15am **BREAK**

## **SESSION II: ACCELERATED HAIR RENDERING**

10:30am **Interactive Self-Shadowing for Animated Hair** (Florence Bertails)

1. Local illumination models
2. 3D light-oriented shadow map
3. Self-Shadowing
  - a. Filling hair density into the map
  - b. Computing transmittance
  - c. Filtering and compositing colors
4. Handling self-collision
5. Parallelization

10:55am **Real-time Rendering of Wet and Styled Hair** (Kelly Ward)

1. Influence of water and styling products on hair
2. Parameterized rendering equations
3. GPU accelerated opacity shadow map

## **SESSION III: VIRTUAL HAIR STYLING & APPLICATIONS**

11:05am **Physically-based Virtual Hair Salon** (Kelly Ward & Florence Bertails)

1. Modeling wet hair and hair with styling products
  - a. Dual-skeletal structure
  - b. Modification of physical properties
2. Cutting, curling, blowing, and other basic styling operations
3. Simulation localization techniques
  - a. Multiresolution representations
  - b. Grid-based partitioning
4. 3D interaction in virtual hair salon

11:25am **Industrial Perspectives** (Kelly Ward)

1. Needs in Production of Feature Animation
2. Practical techniques and solutions

11:40am **Real-time Virtual Hair Dressing** (Nadia Magnenat-Thalmann)

1. Requirements of VR and Soft CAD Systems
2. Interaction using Haptic Interfaces
3. Haptic Rendering Algorithm for Styling Hair
4. Interactive toolkits for virtual hair dressing

12:10pm **New Challenges and Conclusion** (Magnenat-Thalmann & Lin)

12:15pm **Q & A** (All Speakers)

# TABLE OF CONTENTS

## PREFACE

### PART I – Sample Tutorial/Survey Chapters

- 1. A Survey on Hair Modeling: Styling, Simulation and Rendering**  
*Kelly Ward, Florence Bertails, Tae-Yong Kim, Steve Marschner, Marie-Paule Cani, and Ming C. Lin*
- 2. State of the Art in Hair Simulation**  
*Nadia Magnenat-Thalmann, Sunil Hadap, and Prem Kalra*
- 3. Animating Complex Hairstyles in Real Time**  
*Pascal Volino and Nadia Magnenat-Thalmann*
- 4. Modeling Hair Using Level-of-Detail Representations**  
*Kelly Ward, Ming C. Lin, Joohee Lee, Susan Fisher, and Dean Macri*
- 5. Adaptive Grouping and Subdivision for Simulating Hair Dynamics**  
*Kelly Ward and Ming C. Lin*
- 6. Adaptive Wisp Tree - A Multiresolution Control Structure for Simulating Dynamic Clustering in Hair Motion**  
*Florence Bertails, Tae-Yong Kim, Marie-Paule Cani, and Ulrich Neumann*
- 7. A Practical Self-Shadowing Algorithm for Interactive Hair Animation**  
*Florence Bertails, Clément Ménéier, Marie-Paule Cani*
- 8. Modeling Hair Influenced by Water and Styling Products**  
*Kelly Ward, Nico Galoppo, and Ming C. Lin*
- 9. Predicting Natural Hair Shapes by Solving the Statics of Flexible Rods**  
*Florence Bertails<sup>1</sup>, Basile Audoly, Bernard Querleux, Fredric Leroy, Jean-Luc Leveque and Marie-Paule Cani*
- 10. A Simulation-based VR System for Interactive Hairstyling**  
*Kelly Ward, Nico Galoppo, and Ming C. Lin*
- 11. Interactive Virtual Hair-Dressing Room**  
*Nadia Magnenat-Thalmann, Melanie Montagnol, Rajeev Gupta, and Pascal Volino*

## **PART II – Sample Presentation Slides**

**1. Animating Complex Hairstyles in Real-Time**

*Nadia Magnenat-Thalmann*

**2. Modeling Hair Using Level-of-Detail Representation**

*Ming C. Lin and Kelly Ward*

**3. Adaptive Wisp Trees**

*Marie-Paule Cani*

**4. A Practical Self-Shadowing Algorithm for Interactive Hair Animation**

*Florence Bertails*

**5. Modeling Hair Influenced by Water and Styling Products**

*Kelly Ward*



# PREFACE

Modeling hair is essential to creating convincing virtual humans and animated characters for many diverse CG applications. However, hair modeling is a difficult task primarily due to the complexity of hair. A human head typically consists of over 100,000 hair strands. However, each individual hair strand is quite small in diameter. Considering this duality, researchers have examined whether hair should be treated as an overall volume or as individual interacting hair strands. In the real world, the structure and visual appearance of hair varies widely for each person, making it a formidable task for any one modeling scheme to capture all diversities accurately. Moreover, due to the high complexity of hair, the algorithms that provide the best visual fidelity tend to be too computationally overwhelming; while many applications have strict performance requirements. Additionally, there are still unknown properties about real hair, making the creation of a physically plausible modeling scheme elusive at this time.

This course will examine some of the latest developments on hair modeling, with focuses on balancing visual fidelity and real-time performance. We will present realistic accelerated hair rendering algorithms that take advantages of the modern programmable graphics hardware features. We will describe the geometric and physical properties of hair and how external substances (e.g. water and styling products) influence the dynamic behavior and visual appearance of the animated hair. We will discuss different novel representations and simulation techniques to significantly improve the performance of hair dynamics computation, while achieving realistic appearance of animated hair. In addition, we will also look at different application and system requirements that govern the selection of appropriate techniques for interactive hair modeling. Finally, we will examine the future needs of creating special effects and high-quality animated films, in addition to interactive applications.

*Ming C. Lin and Nadia Magnenat-Thalmann*



# **INTERACTIVE HAIR MODELING: *TECHNIQUES & CHALLENGES***

**Supplementary Course Notes**

**PART I – Tutorial and Survey Chapters**

**Ming C. Lin**  
University of North Carolina at Chapel Hill

**Nadia Magnenat-Thalmann**  
University of Geneva



# A Survey on Hair Modeling: Styling, Simulation, and Rendering

Kelly Ward   Florence Bertails   Tae-Yong Kim   Steve Marschner   Marie-Paule Cani   Ming C. Lin

**Abstract**—Realistic hair modeling is a fundamental part of creating virtual humans in computer graphics. This paper surveys the state of the art in the major topics of hair modeling: hairstyling, hair simulation, and hair rendering. Because of the difficult, often unsolved, problems that arise in all these areas, a broad diversity of approaches are used, each with strengths that make it appropriate for particular applications. We discuss each of these major topics in turn, presenting the unique challenges facing each area and describing solutions that have been presented over the years to handle these complex issues. Finally, we outline some of the remaining computational challenges in hair modeling.

**Index Terms**—Hair modeling, physically-based simulation, hardware rendering, light scattering, user-interaction

## I. INTRODUCTION

Modeling hair is essential to computer graphics for various applications, however, realistically representing hair in structure, motion and visual appearance still remains a challenging issue. Hair modeling is an important contribution towards creating convincing virtual humans for many diverse CG applications.

Hair modeling is a difficult task primarily due to the complexity of hair. A human head typically consists of a large volume of hair with over 100,000 hair strands. However, each individual hair strand is quite small in diameter. Considering this duality, researchers have examined whether hair should be treated as an overall volume or as individual interacting hair strands. Currently, there is no method that has been accepted as the industry standard for modeling hair.

In the real world, the structure and visual appearance of hair varies widely for each person, making it a formidable task for any one modeling scheme to capture all diversities accurately. Moreover, due to the high complexity of hair the algorithms that provide the best visual fidelity tend to be too computationally overwhelming to be used for interactive applications that have strict performance requirements. The diverse applications that incorporate hair modeling each possess their own challenges and requirements, such as appearance, accuracy, or performance. Additionally, there are still unknown properties about real hair, making the creation of a physically correct modeling scheme elusive at this time.

In this survey, we will discuss the primary challenges involved with modeling hair and also review the benefits

and limitations of methods presented in the past for handling these complex issues. Furthermore, we will give insight for choosing an appropriate hair modeling scheme based on the requirements of the intended application.

### A. Hair Modeling Overview

As illustrated by Magnenat-Thalmann *et al.* [1], hair modeling can be divided into three general categories: hairstyling, hair simulation, and hair rendering. Hairstyling, which can be viewed as modeling the shape of the hair, incorporates the geometry of the hair and specifies the density, distribution, and orientation of hair strands. Hair simulation involves the dynamic motion of hair, including collision detection between the hair and objects, such as the head or body, as well as hair mutual interactions. Finally, hair rendering entails color, shadows, light scattering effects, transparency, and anti-aliasing issues related to the visual depiction of hair on the screen.

While there are several known techniques for hair modeling, hair research began by viewing hair as individual strands, or one-dimensional curves in three-dimensional space [2], [3]. Building on these foundations, researchers have focused on how these individual strands interact with each other to comprise the whole volume of a full head of hair. Though several paths have been followed, the challenges that encompass modeling a full head of hair remain consistent due to the geometric complexity and thin nature of an individual strand coupled with the complex collisions and shadows that occur among the hairs. The various presented modeling schemes all have strengths and limitations which will be outlined throughout this paper. We have considered the following general questions for analyzing these methods in several categories:

- **Hair Shape:** Can the method handle long, curly or wavy hair or is it limited to simpler short, straight styles?
- **Hair Motion:** Is the method robust enough to handle large, erratic hair motion that can cause dynamic grouping and splitting of hair clusters as well as complex hair collisions?
- **Performance vs. Visual Fidelity:** Is the primary focus of the method to model visually realistic hair, to model hair quickly and efficiently, or to offer a balance between performance speed and visual fidelity of the virtual hair?
- **Hardware Requirements:** Does the method rely on specific GPU features or other hardware constraints or does it have cross-platform compatibility?
- **User Control:** To what degree does the user have control over the hair? Is the control intuitive or burdensome?

- **Hair Properties:** Can the method handle various hair properties (e.g. coarse vs. fine, wet vs. dry, stiff vs. loose) and allow for these values to vary on the fly throughout the application?

Given the factors mentioned above, a hair modeling method may typically have strength in some areas, but little capability in addressing other aspects. One future research endeavor is to lessen the gap between these areas. The goal is to create an ideal unified hair modeling structure that can effortlessly handle various hair shapes, motions, and properties, while giving the desired level of intuitive user control in a manner that achieves a fast performance with photo-realistic hair. Presently, hair modeling is far from this ideal.

### B. Applications and Remaining Problems

The future research in hair modeling may be driven by applications. Cosmetic prototyping desires an exact physical and chemical model of hair for virtually testing and developing products; currently, there is little measured data on the mechanical behaviors of hair to accurately simulate how a product will influence hair's motion and structure. As a result, there is no known hair modeling method that can simulate the structure, motion, collisions and other intricacies of hair in a physically-exact manner.

In contrast, in the entertainment industry, such as with feature animation, a physically correct hair modeling scheme is not necessarily desirable. In fact, it is frequently a goal to model a physically impossible hairstyle or motion. In these cases, a high degree of user control is needed to create a desired effect. Due to the magnitude of the hair volume, manually controlling the properties of hair is a time-consuming and, thus, a costly endeavor. Methods to further accelerate and ease this process would be valued additions to hair modeling research.

Another arena that requires hair modeling is interactive systems, such as virtual environments and videogames. In these applications, the performance speed of the virtual hair is the main emphasis over its appearance. Though there have been many efforts to increase the efficiency of hair modeling algorithms, there still remains a desire to heighten the quality of the resulting hair to capture more hair shapes, motions and properties.

The remainder of this paper is organized as followed. Hairstyling techniques are reviewed in Section II. Methods for simulating dynamic hair are presented in Section III. Section IV describes the properties of hair related to its interaction with light, followed by techniques for rendering hair. Finally, Section V presents new challenges facing hair research and applications in each of these categories.

## II. HAIRSTYLING

Creating a desired hairstyle can often be a long, tedious, and non-intuitive process. Numerous hairstyling techniques have been presented that possess various user interaction requirements. In this section, the main structural and geometric properties of real hair that control its final shape are explained, followed by the methods for styling virtual hair. Techniques

for hairstyling can be categorized into three general steps: attaching hair to the scalp, giving the hair an overall or global shape, and managing finer hair properties.

### A. Hair Structural and Geometric Properties

There is a diverse spectrum of hair shapes, both natural and artificial. Depending on their ethnic group, people can have naturally smooth or jagged, and wavy or curly hair. These geometric features can result from various structural and physical parameters of each individual hair strand, including the shape of its cross-section, its level of curliness, or the way it comes out of the scalp [4], [5]. Hair scientists categorize hair types into three main groups: Asian hair, African hair, and Caucasian hair. Whereas an Asian hair strand is very smooth and regular, with a circular cross-section, an African hair strand looks irregular, and has a very elliptical cross-section. Caucasian hair ranges between these two extrema, from smooth to highly curly hair.

Furthermore, most people do not simply wear their hair naturally, but have it cut and styled in various ways, from simple (hair parting, bangs, etc.) to complex (ponytails, braids, etc.). Using cosmetic products, people can also modify the shape of their hair, either temporarily (using gel, mousse, etc.), or permanently (through permanent waving, hair straightening, etc.). Each kind of hairstyling and hair modification can lead to a wide variety of artificial hairstyles that can be synthesized to generate realistic and diverse virtual characters.

The majority of virtual styling methods used today actually do not consider the physical structure of real hair in their algorithms. Rather than trying to match the *process* of real-world hair shape generation, most virtual styling methods try to match the final *results* with the appearance of real-world hair. Consequently, virtual styling techniques are not appropriate for applications that may desire a physically-correct model for the structure of hair, but rather for applications that desire a visually-plausible solution. However, there have been recent efforts towards the creation of styling methods that more accurately reflect the real-world process of hairstyle generation by considering what is known about real physical hair properties [6] and by mimicking more natural user interaction with hair [7]. Though promising, these endeavors are still at early stages.

### B. Attaching Hair to the Scalp

Due to the high number of individual hair strands composing a human head of hair, it is extremely tedious to manually place each hair strand on the scalp. To simplify the process, a number of intuitive techniques have been developed that employ 2D or 3D placement of hairs onto the scalp.

1) *2D Placement:* In some styling approaches, hair strands are not directly placed onto the surface of the head model. Instead, the user interactively paints hair locations on a 2D map which is subsequently projected onto the 3D model using a mapping function. Spherical mappings to map the strand bases to the 3D contour of the scalp have been popular approaches [2], [8].

Alternatively, Kim *et al.* [9] define a 2D parametric patch that the user wraps over the head model, as illustrated in Figure 1. The user can interactively specify each control point of the spline patch. In the 2D space defined by the two parametric coordinates of the patch, the user can place various clusters of hair.

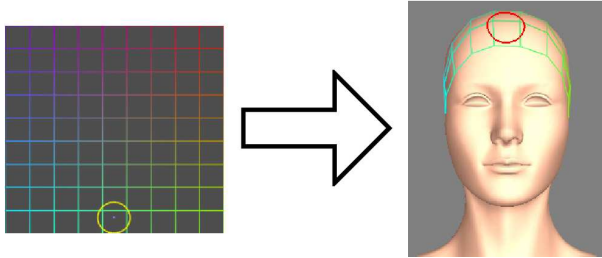


Fig. 1. 2D square patch wrapped onto the 3D model by the method of Kim *et al.* [9].

Placing hair roots on a 2D geometry is easy for the user and allows flexibility. But mapping 2D hair roots onto a 3D curved scalp may cause distortion. Bando *et al.* [10] use a harmonic mapping and compensate for the mapping distortion by distributing the root particles based on a Poisson disc distribution using the distance between corresponding points on the scalp in world space rather than their 2D map positions.

2) *3D Placement*: An alternative approach is to use direct 3D placement of the hair roots onto the scalp. Patrick *et al.* [11] present an interactive interface where the user can select triangles of the head model. The set of selected triangles defines the scalp, i.e. the region of the head mesh where hair will be attached, and each triangle of the scalp is the initial section of a wisp. Shapes of the resulting wisps thus strongly depend on the head mesh, which could be awkward if some control on the wisps' size is desired.

3) *Distribution of Hair Strands on the Scalp*: A popular approach for placing hair strands uses uniform distribution over the scalp as it makes a good approximation of real hair distribution. Some wisp-based approaches randomly distribute hair roots inside each region of the scalp covered by the root wisps sections [12], [13], [14]. But if wisp sections overlap, a higher hair density is generated in the overlapping regions, which can produce distracting results. In order to guarantee a uniform hair distribution over the whole scalp, Kim *et al.* [9] uniformly distribute hair over the scalp and then assign each generated hair root to its owner cluster.

Some approaches also enable the user to paint local hair density over the scalp [15], [13]. Hair density can be visualized in 3D by representing density values as color levels. Controlling this parameter is helpful to produce further hairstyles such as thinning hair. Hernandez and Rudomin [15] extended the painting interface to control further hair characteristics such as length or curliness.

### C. Global Hair Shape Generation

Once hair has been placed on the scalp, it has to be given a desired global shape which is commonly done through geometry-based, physically-based or image-based techniques, which are explained and evaluated in this section.

1) *Geometry-Based Hairstyling*: Geometric-based hairstyling approaches mostly rely on a parametric representation of hair in order to allow a user to interactively position groups of hair through an intuitive and easy-to-use interface. These parametric representations can involve surfaces to represent hair or wisps in the form of trigonal prisms or generalized cylinders.

a) *Parametric Surface*: Using two-dimensional surfaces to represent groups of strands has become a common approach to modeling hair [16], [17], [18]. Typically, these methods use a patch of a parametric surface, such as a NURBS surface, to reduce the number of geometric objects used to model a section of hair. This approach also helps accelerate hair simulation and rendering. These NURBS surfaces, often referred to as hair *strips*, are given a location on the scalp, an orientation, and weighting for knots to define a desired hair shape. Texture mapping and alpha mapping are then used to make the strip look more like strands of hair. A complete hairstyle can be created by specifying a few control curves or hair strands. The control points of these hair strands are then connected horizontally and vertically to create a strip. Though this method can be used for fast hairstyle generation and simulation, the types of hairstyles that can be modeled are limited due to the flat representation of the strip (see Figure 2, left).

In order to alleviate this flat appearance of hair, Liang and Huang [17] use three polygon meshes to warp a 2D strip into a U-shape, which gives more volume to the hair. In this method, each vertex of the 2D strip is projected onto the scalp and the vertex is then connected to its projection.

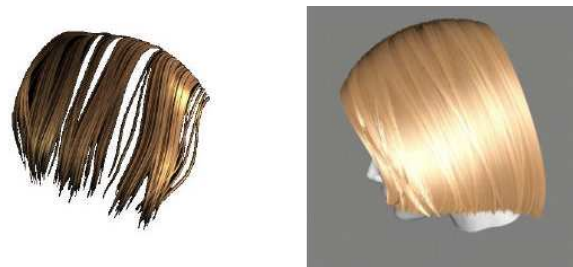


Fig. 2. Modeling hair using NURBS surfaces [16] (left). The Thin Shell Volume [19] (right)

Extra geometric detail can also be extracted from a surface representation. Kim and Neumann [19] developed a model called the *Thin Shell Volume*, or TSV, that creates a hairstyle starting from a parameterized surface. Thickness is added to the hair by offsetting the surface along its normal direction. Individual hair strands are then distributed inside the TSV (see Figure 2, right). Extra clumps of hair can be generated off a NURBS surface using the method of Noble and Tang [18]. Starting from a NURBS volume that has been shaped to a desired hairstyle, key hair curves are then generated along the isocurves of the NURBS volume. The profile curves that are extruded from the key hair curves create extra clumps, which can then be animated independently from the original NURBS surface. This approach will add more flexibility to the types of hair shapes and motions that can be captured using the surface

approach.

b) *Wisps and Generalized Cylinders*: Wisps and generalized cylinders have been used as intuitive methods to control the positioning and shape of multiple hair strands in groups [14], [20], [21], [22], [13]. These methods reduce the amount of control parameters needed to define a hairstyle. A group of hair strands tend to rely on the positioning of one general space curve that serves as the center of a radius function defining the cross-section of a generalized cylinder, also referred to as a hair cluster. The cluster hair model is created from hair strands distributed inside of these generalized cylinders, see Figure 3. The user can then control the shape of the hair strands by editing the positions of the general curve or curves.



Fig. 3. The cluster hair model [20] [21]

V-HairStudio, a tool created by Xu and Yang [21], allows a user to produce and manipulate hair clusters to generate a complex hairstyle. Intermediate results can be viewed quickly by previewing only the cluster-axes to show basic positioning of the hair. Furthermore, the clusters can be shown as surfaces to view the spatial relationship between hair clusters and the head model (as shown in Figure 3, left).

The clusters or wisps allow for the creation of many popular hairstyles from braids and twists of many African hairstyles [22] to constrained shapes such as ponytails. Some more complex hairstyles that do not rely on strands grouped into fixed sets of clusters are more difficult to achieve with these methods. Moreover, while they provide intuitive control to its users, the shaping of a hairstyle can often be tedious as the time to create a hairstyle is typically related to the complexity of the final style.

c) *Multi-resolution Editing*: The principle of representing global hair shape with generalized cylinders can be further extended to provide multi-resolution control in hair shape editing [9], [23]. Complex hair geometry can be represented with a hierarchy of generalized cylinders, allowing users to select a desired level of control in shape modeling. Higher level clusters provide efficient means for rapid global shape editing, while lower level cluster manipulation allows direct control of a detailed hair geometry – down to every hair strand. Kim and Neumann [9] further show that their multi-resolution method can generate complex hairstyles such as curly clusters with a copy-and-paste tool that transfers detailed local geometry of a cluster to other clusters (see Figure 4).

2) *Physically-based Hairstyling*: Some hairstyling techniques are strongly linked to physically-based animation of hair. Section III gives a thorough explanation of dynamic hair

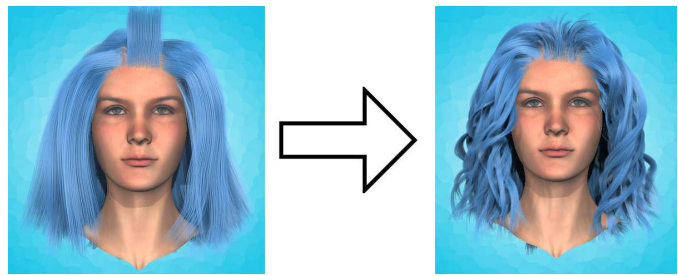


Fig. 4. Multiresolution hairstyling [9]

simulation whereas this section describes techniques for creating hairstyles based on physical simulation. These approaches rely on the specification of a few key parameters in methods ranging from cantilever beams that control individual strands to fluid flow methods that control the entire volume of hair. These methods customarily reduce the amount of direct user control over the resulting hairstyle.

a) *The cantilever beam*: In the field of material strengths, a cantilever beam is defined as a straight beam embedded in a fixed support at one end only – the other end being free. Anjo *et al.* [3] consider that it is a similar case to a human hair strand, where the strand is anchored at the pore, and the other end is free. Considering gravity is the main source of bending, the method simulates the statics of a cantilever beam to get the pose of one hair strand at rest. Combining this technique with hair-head collision handling, hair shearing, and the application of additional external forces, results in several different smooth hairstyles.

b) *Fluid Flow*: Hadap and Magnenat-Thalmann [24] modeled static hairstyles as streamlines of fluid flow based on the idea that static hair shapes resemble snapshots of fluid flow around obstacles. The user creates a hairstyle by placing streams, vortices and sources around the hair volume. For example, a vortex is used to create a curl in the hair at a desired location (see Figure 5).

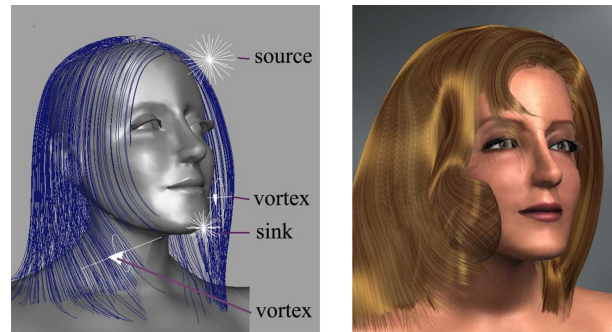


Fig. 5. Modeling hair using a fluid flow [24]

Hadap and Magnenat-Thalmann later extended this work to simulate dynamic hair, as explained in Section III-C.1.a.

c) *Styling Vector and Motion Fields*: Yu [8] observed that both vector fields and hair possess a clear orientation at specific points while both are also volumetric data; this led him to the use of static 3D vector fields to model hairstyles, see Figure 6 (left). Given a global field generated



by superimposing procedurally defined vector field primitives, hair strands are extracted by tracing the field lines of the vector field. A hair strand begins at a designated location on the scalp and then grows by a certain step size along the direction of the accumulated vector of the vector field until a desired length is reached. Similarly particles can be used in motion fields to shape strands [25]. A particle is given a fixed life-time and traced through a motion field. The history of the particle comprises the whole hair strand; changing the life-time of the particle then changes the length of the hair.

Choe *et al.* [13] also use a vector field to compute global hair position while accounting for hair elasticity. Their algorithm calculates hair joint angles that best account for both the influence of the vector field and the natural trend of the strand for retrieving its rest position. Another important feature of the approach is the ability for the user to define *hair constraints*. A hair constraint causes a constraint vector field to be generated over a portion of 3D space that later modifies the original vector field proportionally to a weight parameter. Hair deformation is computed by using the previous algorithm applied on the modified vector field. In practice, the user can specify three types of constraints: point constraints, trajectory constraints and direction constraints. Hair constraints turn out to be very useful for creating complex hairstyles involving ponytails, bunches or braids, as illustrated in Figure 6 (right).



Fig. 6. A styling vector field [8] (left) and constraint-based hairstyling [13] (right)

3) *Generation of Hairstyles from Images:* Generating a realistic hairstyle using a modeling interface such as the ones presented in Section II-C.1 generally takes hours of manual design. Physically-based approaches described in Section II-C.2 can help generate the global shape of hair automatically, but fine hair details have to be added using a procedural technique (see Section II-D.1). Recent approaches have proposed an alternative way of generating hairstyles based on the automatic reconstruction of hair from images.

a) *Hair Generation From Photographs:* Kong *et al.* were the first who used real hair pictures to automatically create hairstyles [26]. Their method is merely geometric and consists of building a 3D hair volume from various viewpoints of the subject’s hair. Hair strands are then generated inside this volume using a heuristic that does not ensure faithfulness in hair directionality. This approach is then best suited for simple hairstyles.

Grabli *et al.* introduced an approach exploiting hair illumination in order to capture hair local orientation from images [27]. Their system works by studying the reflectance of the

subject’s hair under various controlled lighting conditions. Fixing the viewpoint allows them to work with perfectly registered images. By considering a single viewpoint and using a single filter to determine the orientation of hair strands, the method reconstructs hair only partially. Paris *et al.* extended this approach [28] to a more accurate one, by considering various viewpoints as well as several oriented filters; their strategy mainly consists of testing several filters on a given 2D location and choosing the one that gives the most reliable results for that location. This method captures local orientations of the visible part of hair, and thus produces visually faithful results with respect to original hairstyles (see Figure 7). Wei *et al.* [29] subsequently improved the flexibility of the method by exploiting the geometry constraints inherent to multiple viewpoints, which proves sufficient to retrieve a hair model with no need for controlled lighting conditions nor a complex setup.



Fig. 7. Hair capture from photographs [28]

b) *Hair Generation From Sketches:* Mao *et al.* [30] developed a sketch-based system dedicated to modeling cartoon hairstyles. Given a 3D head model, the user interactively draws the boundary region on the scalp where hair should be placed. The user then draws a silhouette of the target hairstyle around the front view of the head. The system generates a silhouette surface representing the boundary of the hairstyle. Curves representing clusters of hair are generated between the silhouette surface and the scalp. These curves become the spine for polygon strips that represent large portions of hair, similar to the strips used by [16], [17].

This sketch-based system quickly creates a cartoon hairstyle with minimal input from its user. The strips, or cluster polygons, used to represent the hair, however, are not appropriate for modeling more intricate hairstyles such as those observable in the real world.

4) *Evaluation:* Each of the global hair shaping methods described in this section is appropriate for styling hair under different circumstances. Table I shows a comparison of several global shaping methods in hair shape flexibility, user control, and time for manual setup or input. The larger the range of hair shapes that can be modeled by an algorithm, the broader its applicability in practice is. The level of user control is important in order to facilitate placing exact details where desired in the hair. Moreover, while some styling methods can capture a hair shape quickly through automatic processing, others require time-consuming manual setup or input by its user.

As Table I indicates, geometry-based hairstyling techniques, such as through generalized cylinders or parametric surfaces, customarily give the user a large degree of control over the hair; however, the manual positioning of hair can be a tedious, time-consuming task due to the large intricate volume of hair. The time for a user to create a hairstyle using the multiresolution generalized cylinder approach presented by Kim and Neumann [9] ranged between several minutes to several hours depending on the complexity of the hair shape. While parametric surfaces typically provide fast methods for hairstyle creation, the results tend to be limited to flat, straight hairstyles due to the 2D surface representation. Alternatively, wisp or generalized cylinders can model many straight or curly hairstyle shapes.

Controlling the volume of the hair through physically-based techniques, such as through fluid flow or vector fields, typically requires less tedious input by the user; however, finer details of many complex hairstyles are often difficult to capture through such interaction. Many of the parameters can be non-intuitive to hairstyling and the user typically has less specific control over the hairstyle creation in comparison to the geometry-based approaches.

The generation of hairstyles from images has been shown to be a highly automatic process even with a relatively simple setup by Wei *et al.* [29]. The final hairstyles created from images can be quite impressive, but these methods are limited in that they result from hairstyles that have to exist in the real world, making the range of styles modeled generally less flexible than geometric or physically-based methods. Hairstyles generated from sketches can allow for more creativity in the resulting hair shapes, though specific finer details, such as with braided hair, can be impossible to achieve without cumbersome user involvement.

	Hair Shapes	User Control	Manual Time
Gen. Cylinders	flexible	high	slow
Surfaces	limited to straight	high	fast
Physical Volumes	limited, details hard	cumbersome	medium
Photos	limited, must exist	none	fast
Sketches	limited, details hard	medium	fast

TABLE I

ANALYSIS OF GLOBAL SHAPING METHODS *Evaluation of geometry-based generalized cylinders and surfaces, physically-based volumes and image-based using photographs and sketches in the areas of user control, flexibility of resulting hair shapes, and the time of manual input or setup.*

There are some recent techniques that build on the strengths of the different methods. For example, the work by Choe *et al.* [13] model hair in the form of wisps where the user edits the prototype strand that controls the wisp shape, but vector fields and hair constraints are also utilized to achieve intricate hair shapes such as braids, buns, and ponytails. While exact timings for manual input is not provided, the amount of user input is still considered high and the most time-consuming aspect of the whole virtual hairstyling process.

#### D. Managing Finer Hair Properties

After hair has been given a global shape, it is often desirable to alter some of the finer, more localized properties of the hair to either create a more realistic appearance (e.g. curls or volume) or to capture additional features of hair such as the effects of water or styling products. In practice, most of these techniques to control finer details have been used in conjunction with geometric or physically-based approaches for defining a global hair shape (Sections II-C.1 and II-C.2).

1) *Details of Curls and Waves*: Local details such as curls, waves or noise might need to be added to achieve a natural appearance for hair once a global shape has been defined. Yu [8] generates different kinds of hair curliness by using a class of trigonometric offset functions. Various hairstyles can thus be created by controlling different geometric parameters such as the magnitude, the frequency or the phase of the offset function. In order to prevent hair from looking too uniform, offset parameters are combined with random terms that vary from one hair cluster to another (see Figure 8, left). Similarly, a more natural look can be generated for hair shaped through fluid flow by incorporating a breakaway behavior to individual hair strands that allow the strand to breakaway from the fluid flow based on a probability function [24].



Fig. 8. Waves and curls procedurally generated by Yu [8] (left) and Choe *et al.* [13] (right)

Choe *et al.* [13] model a hairstyle with several wisps, and the global shape of each wisp is determined by the shape of a *master strand*. Within a wisp, the degree of similarity among the strands is controlled by a length distribution, a deviation radius function and a fuzziness value. The geometry of the master strand is decomposed into an outline component and a details component. The details component is built from a *prototype* strand using a Markov chain process where the degree of similarity between the master strand and the prototype strand can be controlled through a Gibbs distribution. Resulting hairstyles are thus globally consistent while containing fine variations that greatly contribute to their realism, as shown by Figure 8 (right).

These methods for localized shape variation help to alleviate the synthetic look of the virtual hair, however since most of them incorporate some form of random generation the user has less control over the finer details. This semi-automatic process helps accelerate the creation of hairstyles as these minute details could take many man-hours if performed manually. On the other hand, the random generation can also cause unwanted artifacts if strands are perturbed in a way that causes unnatural collisions. Moreover, these methods do not

account for the physical hair properties for computing the hair geometry, although it is well-known that such features, described in Section II-A, have a great influence on the hair shape [4], [5].

In order to automatically generate the fine geometry, including waves or curls, of natural hair, Bertails *et al.* [6] recently introduced a new hairstyling method using a mechanically accurate model for static elastic rods (the Kirchhoff model). The method, based upon a potential energy minimization, accounts for the natural curliness of hair, as well as for hair ellipticity (see Figure 9). Though not appropriate for direct user control, this method is promising for a more accurate hairstyle generation process and can be useful for cosmetics prototyping, for example.



Fig. 9. A real ringlet (left), and a synthetic one (right) automatically generated by the physically-based method of Bertails *et al.* [6]

2) *Producing Hair Volume*: Whereas most geometric-based hairstyling methods implicitly give volume to hair by using volumetric primitives (see Section II-C.1), physically-based methods often account for hair self-collisions in order to produce volumetric hairstyles. Approaches that view hair as a continuous medium [25], [24], [8], [13] add volume to the hair through the use of continuum properties that reproduce the effects of collisions between hair strands, such as via vector fields or fluid dynamics. As strands of hair become closer, these techniques either prevent them from intersecting due to the layout of the vector or motion fields, or foster a repulsive motion to move them apart from each other, causing the hair to appear thicker.

Since detecting collisions between strands of hair can be difficult and, in the least, very time consuming, Lee and Ko [31] developed a technique that adds volume to a hairstyle without locating specific intersections among strands. The idea is that hair strands with pores at higher latitudes on the head cover strands with lower pores. Multiple head hull layers are created of different sizes from the original head geometry. A hair strand is checked against a specific hull based on the location of its pore. A hair-head collision detection and response algorithm is then used. This method only works in the case of a quasi-static head that remains vertically oriented.

3) *Modeling Styling Products and Water Effects*: Styling products, such as hairspray, mousse, and gel, have significant effects on the hair appearance, including hairstyle recovery after the hair has moved, stiffened overall hair motion, large grouping of hair strands due to the adhesiveness of fixative products, and the change in the hair volume.

Lee and Ko [31] developed a method to model the effects of hair gel on a hairstyle. A styling force is used to enable hairstyle recovery as the hair moves due to external force or head movement. As a result, an initial hairstyle can be restored after motion. When gel is applied to the hair, the desire is to retain the deformed hairstyle rather than returning to the initial style. This algorithm preserves the deformed shape by updating the styling force during the simulation. Alternatively, breakable *static links* or *dynamic bonds* can be used to capture hairstyle recovery by applying extra spring forces between nearby sections of hair to mimic the extra clumping of hair created by styling products [32], [33].

Styling products also increase the stiffness of hair motion allowing a curled section of hair with styling products applied to retain a tight curl as the hair moves. Through the use of a *dual-skeleton* model for simulating hair, separate spring forces can be used to control the bending of hair strands versus the stretching of curls [33]. Styling products can then alter the spring stiffness' independently to create desired results.



Fig. 10. Comparison of hair (a) dry and (b) wet [33].

Water will also drastically change the appearance, shape and motion of hair. As water is absorbed into hair the mass of the hair increases up to 45%, while its elasticity modulus decreases by a factor of 10 – leading to a more deformable and less elastic material [34]. Moreover, as hair gets wet, the volume of the hair decreases because strands of hair in close proximity with each other adhere due to the bonding nature of water. In their static physically-based model, Bertails *et al.* [6] easily incorporated the effect of water on hair by simply modifying the relevant physical parameters that actually change when hair gets wet: the mass and the Young's modulus of each fiber. Ward *et al.* [33] modeled dynamic wet hair with their dual-skeleton system by automatically adjusting the mass of the hair along the skeletons as water is added to the hair. The increased mass resulted in limited overall motion of the hair and elongation of curls. A flexible geometric structure allows the volume of the hair to change dynamically by altering the radius of the strand groups that are used in simulation (see Figure 10).

An interactive virtual hairstyling system introduced by Ward *et al.* [7] illustrates how water and styling products can be used to interactively alter the look and behavior of hair dynamically through a 3D interface that allows users to perform common hair salon applications (such as wetting, cutting, blow-drying) for the purpose of intuitively creating a final hairstyle.



### III. HAIR SIMULATION

Animating hair is one of the most challenging problems when synthesizing the motion of virtual humans. Indeed, human hair is a complex material, consisting of many thousands of very thin strands interacting with each other and with the body of the avatar. It is very difficult to provide a realistic model for dynamic hair because each individual hair strand has a complex mechanical behavior and very little knowledge is available regarding the nature of mutual hair interactions. Secondly, animation of a full head of hair raises obvious problems in terms of computational costs. As a consequence, existing hair animation methods propose a tradeoff between realism and efficiency, depending on the intended application.

Before analyzing existing methods on hair animation, we briefly describe in Section III-A some real mechanical features of hair. This knowledge will be useful to evaluate the quality of various existing synthetic hair models.

Numerous methods giving the dynamics of one individual hair strand have been borrowed from existing 1D mechanical models with several levels of simplification. These models, presented and commented on in Section III-B, have subsequently been used for animating both individual hair strands and groups of hair.

While there can be over 100,000 strands of hair on a human head, it was observed that most hair strands tend to move in a similar way as their neighbors. This observation led to a number of approaches extending the single-strand method to simulate the collective behavior of hair. These methods, which range from continuum to hair wisp models, will be presented in Section III-C.

Finally, Section III-D presents most recent works that have used multi-resolution techniques in order to gain some efficiency and to achieve interactive hair simulations.

#### A. The Mechanics of Hair

A hair strand is an anisotropic deformable object: it can easily bend and sometimes twist but it strongly resists shearing and stretching. A hair strand has also elastic properties in the sense that it tends to recover its original shape after the stress being applied to it has been removed. These properties will be evaluated on each of the existing models that simulate an individual hair strand (see Figure II in Section III-B).

A human head is composed of about 100,000 hair strands interacting with each other and with the body. The nature of interactions between hair strands is very complex. This is mainly due to the surface of individual hair strands, which is not smooth but composed of tilted scales (see Figure 11).

This irregular surface causes anisotropic friction inside hair, with an amplitude that strongly depends on the orientation of the scales and of the direction of motion [35]. Moreover, hair is very triboelectric, meaning it can easily release static charges by mere friction. This phenomenon has been measured in the case of combed hair, but it seems that no study has been published regarding this effect in the case of hair-hair friction.

In addition, as previously mentioned, a hair strand can be naturally smooth or jagged, wavy or curly. The geometric hair shape, which is correlated to some structural and physical

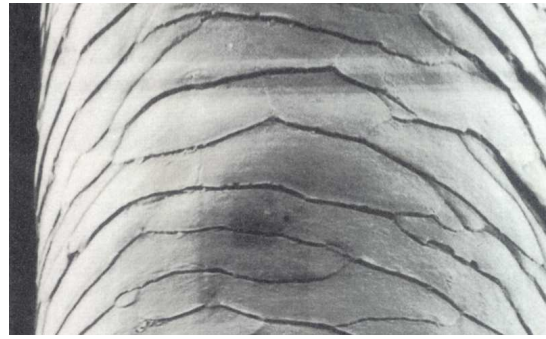


Fig. 11. An electron micrograph of a hair fiber that shows the structure of the outer cuticle surface, which is composed of thin overlapping scales [4]. In this image, the fiber is oriented with the root at the top and the tip at the bottom.

features of the hair (see Section II-A) affects the motion of hair. For example, a curly moving hair will look more “elastic” than a straight hair, because hair curls can longitudinally stretch during motion, like springs—although hair strands still remain inextensible. In addition, hair clumps are more likely to appear in curly hair, where contacts exist among hair strands, and thus the probability for them to get into tangles is greater. In fact, as it can be observed in the real world, the more a hair has an intricate geometry, the less degrees of freedom it has during motion. The feature *curliness* will be evaluated on existing approaches in Sections III-B and III-C. To our knowledge, there are no quantitative nor qualitative results published on hair grouping from the mechanics literature.

Hair is thus a very complex mechanical material. Unlike some well-known physical materials such as fluids—which have been studied for centuries and modeled by accurate equations—hair remains an unsolved problem for which there is currently no standard physically-based model. Hence, one of the challenges lies in finding an appropriate representation of hair for dynamic simulation.

#### B. Dynamics of Individual Hair Strands

Three families of computational models have been proposed for simulating the dynamics of one individual hair strand: mass-spring systems, projective dynamics, and rigid multi-body serial chains. Each one of these models is described and evaluated in terms of realism and ability to be included inside a full hair.

1) *Mass-Spring Systems*: One of the first attempts to animate individual hair strands was presented by Rosenblum *et al.* [2] in 1991. A single hair strand is modeled as a set of particles connected with stiff springs and hinges. Each particle has three degrees of freedom, namely one translation and two angular rotations. Hair bending rigidity is ensured by angular springs at each joint. This method is simple and easy to implement. However, torsional rigidity and non-stretching of the strand are not accounted for. Limiting the stretching of the strand requires the use of strong spring forces, which leads to stiff equations that often cause numerical instability, unless very small time steps are used.

Many advances in mass-spring formulation were recently made, especially in the context of cloth simulation. Baraff

and Witkin [36] showed that implicit integration methods prove very useful for the simulation of a stiff system as they ensure that the system will remain stable even with large time steps. Implicit integration was later used in the case of hair simulation [37], [38]. Other approaches [12], [39] used a constrained mass-spring model, well-suited for animating extensible wisps such as wavy or curly wisps.

2) *One Dimensional Projective Equations*: In 1992, Anjyo *et al.* proposed a simple method based on one-dimensional projective differential equations for simulating the dynamics of individual hair strands. Initially, the statics of a *cantilever beam* is simulated to get an initial plausible configuration of each hair strand. Then, each hair strand is considered as a chain of rigid sticks  $s_i$ . Hair motion is simulated as follows:

- Each stick  $s_i$  is assimilated as a direction, and thus can be parameterized by its polar angles  $\phi$  (azimuth) and  $\theta$  (zenith) (see Figure 12).
- The external force  $\mathbf{F}$  applied to the stick is projected onto both planes  $P_\phi$  and  $P_\theta$ , respectively, defined by  $\phi$  and  $\theta$  (the longitudinal projection of  $\mathbf{F}$  on  $s_i$  is neglected since it should have no effect on the rigid stick).
- Fundamental principles of dynamics are applied to each parameter  $\phi$  and  $\theta$  which leads to two differential equations that are solved at each time step. Positions of the sticks representing each hair strand are always processed from the pore to the tip, where the root stick position is first processed and following sticks are recursively defined.

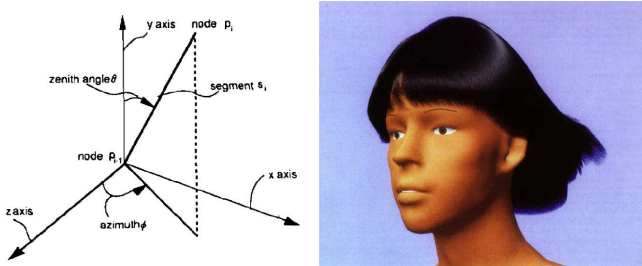


Fig. 12. Left: the polar coordinate system for a hair segment. Right: simulating individual hair strands using one dimensional projective equations for dynamics [3].

This method is attractive for many reasons. It is easy to implement, efficient (tens of thousands of hair strands can efficiently be simulated this way). Moreover, hair is prevented from stretching while hair bending is properly recovered. However, as torsional hair stiffness cannot be accounted for, this method cannot properly handle fully 3D hair motions. Furthermore, as motion is processed from top to bottom, it is difficult to handle external punctual forces properly. Issues related to the handling of external forces are discussed in Section III-B.4.

3) *Rigid multi-body serial chain*: In order to compute the motion of individual hair strands, forward kinematics have been used as a more general alternative to one-dimensional projective equations [40], [32]. Such techniques are well-known in the field of robotics, and efficient multi-body dynamics algorithms have been proposed for decades [41].

Each hair strand can be represented as a serial, rigid, multi-body open chain using the reduced or spatial coordinates formulation [41], in order to keep only the bending and twisting degrees of freedom of the chain: stretching DOFs are removed (see Figure 13). Apart from the gravitational influence, forces accounting for the bending and the torsional rigidity of the hair strand are applied on each link. Forward dynamics are processed using the Articulated-Body Method described in [41], with a linear time complexity. Hadap and Magnenat-Thalmann [40] and Chang *et al.* [32] used this technique to animate several sparse individual hair strands within an elaborate, global continuous model for hair (see Section III-C.1). Although the authors mention they can use a non-zero rest configuration for the strand in order to simulate wavy or curly hair strands, none of these methods actually show any results for simulating curly strands. The possible issues related to curly hair simulation are not explained.

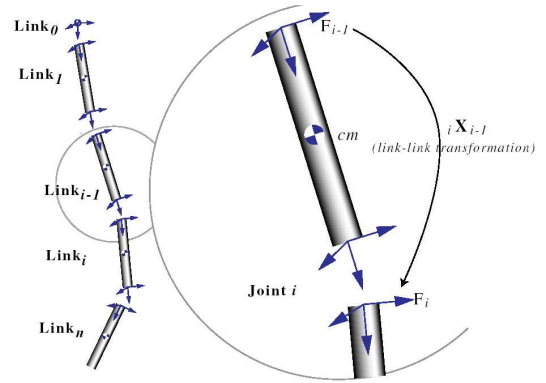


Fig. 13. Hair strand as a rigid multi-body serial chain [40].

4) *Handling external forces*: A good dynamic model for one individual hair strand is expected to yield a realistic motion of one isolated hair strand, but it should also be able to properly handle any external force, such as gravity, wind, contacts or collisions. Depending on the method chosen for the dynamics of one strand, accounting for such forces can be tricky—especially when the forces are punctual (interactions with other objects).

In methods simulating chains of rigid links (Section III-B.2), motion is processed from top to bottom in one single pass. This means that a collision detected at stick  $s_k$  only affects the following sticks  $s_j$ , where  $j > k$  without propagating the effect backward to the sticks located near the roots, which can lead to unrealistic shapes for hair. Refining Anjyo's method, Lee and Ko [31] simply fix the problem by adding an extra force that enables hair to get a proper shape when colliding with an object other than the head.

In the case of serial rigid multi-body chains, external forces can properly be accounted for when using a multi-pass forward dynamics algorithm. However, because of the high bending and torsional stiffness that are required to maintain the curved shape at rest, the simulation may lack stability if external forces are integrated within an explicit integration scheme. The major drawback of the articulated bodies model is that, unlike the mass-springs model, it is difficult to formulate

the conventional dynamics–based on reduced coordinates–using an implicit integration scheme [42]. As proposed by Baraff [43], a solution may be the use of Lagrange multipliers instead of a reduced-coordinates formulation of the problem, in order to integrate hard constraints implicitly. Probably because of its non triviality, this method has never been implemented in the case of hair dynamics.

5) *Evaluation*: The following table indicates, for each hair dynamic model given above, which are the required properties that it ensures.

	Mass-springs	Projective dynamics	Rigid multi-body serial chain
<b>Bending</b>	yes	yes	yes
<b>Torsion</b>	no	no	yes
<b>Non-stretching</b>	no	yes	yes
<b>Curliness</b>	no	no	no
<b>External forces</b>	easy	tricky	tricky

TABLE II

ANALYSIS OF DYNAMIC MODELS FOR INDIVIDUAL HAIR STRANDS  
*Evaluation of each hair dynamics model, according to the following criteria: bending rigidity, torsional rigidity, non-stretching, curliness handling, and easiness to handle external forces properly.*

According to Table II, we can notice that rigid multi-body serial chain models support more realistic features than any other model. But as mentioned above, this model can hardly be imposed with some external constraints–such as contacts or collisions with an external object–in a stable way. To take advantages of both mass-springs models and rigid multi-body serial chains, Choe *et al.* [38] recently proposed a hybrid model, made of rigid links separated by soft spring joints. This model allows for an implicit (and thus stable) integration of the dynamics, including the constraints between the hair and the body. But as mass-springs, it does not fully avoid stretching of the hair strand. Last but not least, note that all existing models for the dynamics of one hair strand are *straight* models, and none are able to account for natural hair curliness.

### C. Simulating the Dynamics of a Full Hairstyle

Handling a collection of hair strands leads to additional challenges in the field of computer graphics: the realism of the collective dynamic behavior and the efficiency of the simulation.

As mentioned in Section III-A, hair interactions are very complex, and little knowledge—including physical experiment reports, measurements and possible models—is known about the actual phenomena of interactions, and their order of magnitude. In addition, the enormous number of contacts and collisions that occur permanently *or* temporarily inside hair raises obvious problems in terms of computational treatment. Consequently, two challenging issues have to be handled when computing hair contacts and collisions: detection and response.

Early hair animation methods [2], [3] generally simulated a set of individual hair strands while neglecting hair-hair interactions for the sake of efficiency and simplicity. Thus, they failed in capturing the actual complexity of hair during

motion, and hair just looked very smooth and lacked volume. Neither did they account for the dissipation of energy due to friction between hair strands.

More recent approaches for animating hair make assumptions on hair consistency during motion to simplify the problem of collisions. Hair is essentially either globally considered as a continuous medium (Section III-C.1), or as a set of disjoint groups of hair strands (Section III-C.2.b). Specific hair-hair interaction models are proposed in both cases.

1) *Hair as a Continuous Medium*: Due to the high number of strands composing a human head of hair, simulating each strand individually is computationally overwhelming. Furthermore, strands of hair in close proximity with each other tend to move similarly. This observation led researchers to view hair as an anisotropic continuous medium.

a) *Animating Hair using Fluid Dynamics*: Considering hair as a continuum led Hadap and Magnenat-Thalmann [40] to model the complex interactions of hair using fluid dynamics. The interactions of single hair strands are dealt with in a global manner through the continuum.

Individual strand dynamics is computed to capture geometry and stiffness of each hair strand (see Section III-B.3). Interaction dynamics, including hair-hair, hair-body, and hair-air interactions, are modeled using fluid dynamics. Individual hair strands are kinematically linked to fluid particles in their vicinity. In this model, the density of the hair medium is defined as the mass of hair per unit occupied volume and the pressure and viscosity represent all of the forces due to interactions of hair strands. As hair strands are compressed together, the pressure increases resulting in the strands being moved apart.



Fig. 14. Simulation of hair blowing in the wind using fluid fbw [40].

Using this setup, it is possible to model hair-body interactions by creating boundary fluid particles around solid objects (see Figure 14 which shows hair blowing in the wind). A fluid particle, or *Smooth Particle Hydrodynamics* (SPH), then exerts a force on the neighboring fluid particles based on its normal direction. The viscous pressure of the fluid, which is dependent on the hair density, accounts for the frictional interactions between hair strands.

Utilizing fluid dynamics to model hair captures the complex interactions of hair strands. However, since this method makes the assumption of a continuum for hair, it does not capture dynamic clustering effects that can be observed in long, thick real hair. Moreover, computations required for this method are



quite expensive; using parallelization, it took several minutes per frame to simulate a hair model composed of 10,000 individual hair strands.

b) *Loosely Connected Particles*: Bando *et al.* [10] have modeled hair using a set of SPH particles that interact in an adaptive way. Each particle represents a certain amount of hair material which has a local orientation (the orientation of a particle being the mean orientation of every hair strand covered by the particle), refer to Figure 15.

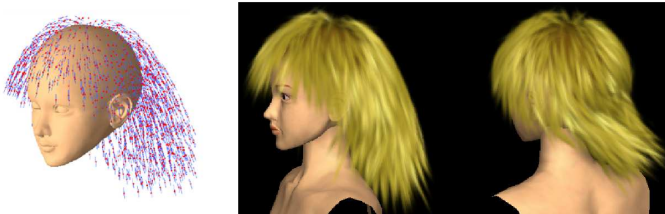


Fig. 15. (left) Particles defining hair, line segments indicate direction (right) Animation of hair with head shaking [10].

Initially, connected chains are settled between neighboring particles being aligned with local hair orientation: two neighboring particles having similar directions and being aligned with this direction are linked. This initial configuration is kept during the motion because it represents the spatial consistency of interactions between particles. During motion, each particle can interact with other particles belonging to its current neighborhood. The method proposes to handle these interactions by settling breakable links between close particles; as soon as the two particles are not close enough, these links vanish. Thus, this method facilitates transversal separation and grouping while maintaining a constant length for hair. At each time step, searching the neighborhood of each particle is done efficiently by using a grid of voxels.

c) *Interpolation between Guide Hair Strands*: Chang *et al.* [32] created a system to capture the complex interactions that occur among hair strands. In this work, a sparse hair model of *guide strands*, which were first introduced in [44], [45], is simulated. A dense hair model is created by interpolating the position of the remaining strands from the sparse set of guide strands. Using multiple guide hair strands for the interpolation of a strand alleviates local clustering of strands.

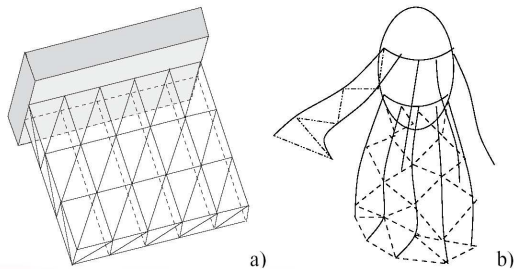


Fig. 16. Triangle strips used for collision detection setup for (a) brush (b) head of hair [32].

The sparse set of guide strands is also used to detect and handle mutual hair interactions. Since detecting collisions only among the guide strands is inefficient, an auxiliary

triangle strip is built between two guide strands by connecting corresponding vertices of the guide strands (see Figure 17). A collision among hair strands is detected by checking for intersections between two hair segments and between a hair vertex and a triangular face. Dampened spring forces are then used to push a pair of elements away from each other when a collision occurs. Figure 17 shows the sparse and dense hair models, respectively.

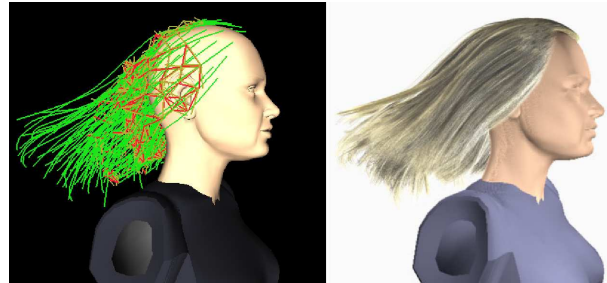


Fig. 17. (left) Sparse hair model with static links and (right) Rendered image of interpolated dense hair model [32].

The use of guide strands can lead to missed collisions when the interpolated strands collide with an object with which the guide strands do not.

d) *Free Form Deformation*: To achieve hair simulation of complex hairstyles in real-time, Volino *et al.* [46] proposed to use a global volumetric free form deformation (FFD) scheme instead of considering an accurate mechanical model related to the structure of individual hair strands. A mechanical model is defined for a lattice surrounding the head. The lattice is then deformed as a particle system and hair strands follow the deformation by interpolation. Collisions between the hair and the body are handled by approximating the body as a set of metaballs.

This method is well-suited for animating various complex hairstyles, when the head motion has a low magnitude. For high deformations, hair discontinuities observed in real hair (e.g., see Figure 20) would not be reproduced because only continuous deformations of hair are considered through the lattice deformation.

2) *Hair as Disjoint Groups*: In order to reduce the complexity of hair, an alternative approach consists of grouping nearby hair strands and simulating these disjoint groups as independent, interacting entities. This representation of hair was especially used to save computation time in comparison with the simulation of individual strands, and even reach interactive frame rates. It also captures realistic features of hair as it accounts for local discontinuities observed inside long hair during fast motion; these local discontinuities cannot be captured using the continuum paradigm.

a) *Real-time Simulation of Hair Strips*: As discussed in Section II-C.1.a, the complexity of hair simulation has been simplified by modeling groups of strands using a thin flat patch, referred to as a *strip* (see Figure 18) [16], [19], [47], [48], [17], [49], [50]. A simple dynamics model for simulating strips is presented in [47] that is adapted from the projective angular dynamics method introduced by Anjyo *et al.* [3] (see

Section III-B.2); dynamics is applied to the control point mesh of the NURBS surface.

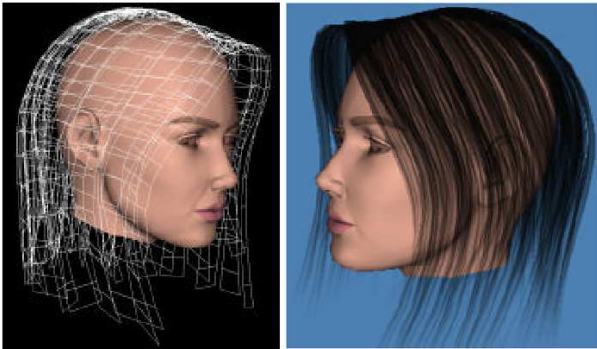


Fig. 18. Hair strips as an approximate hair model [16].

Using strips to model hair results in significantly faster simulation because fewer control points are required to model a strip in comparison to modeling individual strands. In [47], collision avoidance between hair strips and external objects, such as the head or the body, is achieved by using ellipsoids to approximate the boundaries of these objects. When a control point of the strip is inside the ellipsoid a reaction constraint method is used to move it back to the boundary. Furthermore, collisions between hair strips are avoided by introducing springs within the strips and between neighboring strips. The springs are used to prevent neighboring strips from moving too far apart or too close together. Moreover, springs are also used to prevent a strip from overstretching or over-compressing. The result is that the hairstyle remains relatively consistent throughout the simulation. Similarly, Guang and Zhiyong [48] presents a strip-based hair structure for modeling short hair where the strips of texture-mapped hair are simulated using a mass-spring model and 3D morphing.

By using a single strip to represent tens or hundreds of hair strands, hair simulation, including hair-hair collision avoidance, can be achieved in real-time. This process, however, is limited in the types of hairstyles and hair motions it can represent; the flat shape of the strips is most suited to simulating simple, straight hair.

*b) Simulation of Wisps:* One of the first methods to take advantage of grouping hair was presented by Watanabe and Suenaga in [51]. They animate a set of trigonal prism-based wisps. During motion, the shape of a wisp is approximated by *parabolic trajectories* of fictive particles initially located near the root of each wisp. At each time step, the trajectories of the particles are estimated using initial velocities and accelerations, such as gravitational acceleration. This method amounts to simulating only approximate kinematics without considering the inertia of the system, which appears to be limited to slow hair motion. Moreover, interactions between different wisps are not taken into account.

A similar process of grouping neighboring strands together into wisps was used by [45], [44]. In these works, a wisp of strands is modeled by simulating the motion of a single *typical* strand and then generating other strands by adding random displacements to the origin of the typical strand.

The number of overall strands that need to be simulated is reduced significantly. Again, in this work, interactions among strands, or between wisps, is not considered.

To account for complex interactions being observed in real hair during fast motion, Plante *et al.* [12], [52] have represented hair using a fixed set of deformable, volumetric wisps. Each wisp is structured into three hierarchical layers: a *skeleton curve* that defines its large-scale motion and deformation, a deformable volumetric envelope that coats the skeleton and accounts for the deformation of the wisp sections around it, and a given number of hair strands that are distributed inside the wisp envelope and used only at the rendering stage of the process (see

Figure 19).

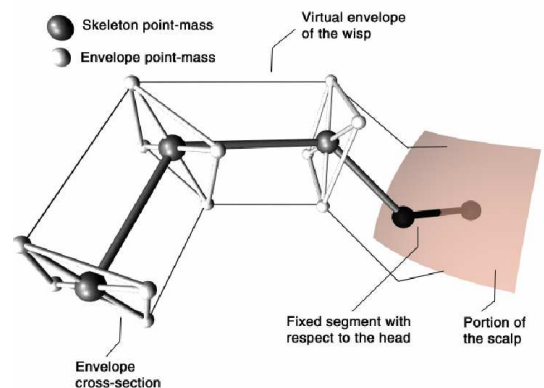


Fig. 19. Elements defining a deformable volumetric wisp [12].

As the skeleton approximates the average curve of a wisp, it is likely to stretch or compress a bit while the wisp is not completely straight. The mass-spring simulation can thus be well-suited for simulating wavy or curly wisps.

Assuming that the local discontinuities inside hair are caused by collisions between wisps of different orientations, this method provides a model of anisotropic interactions between wisps. Wisps of similar orientations are allowed to penetrate each other, and are submitted to viscous friction, whereas wisps of different orientations actually collide in a very dissipative way.



Fig. 20. The layered wisp model [12] (bottom) captures both continuities and discontinuities observed in real long hair motion (top).

As illustrated in Figure 20, the approach has led to convincing results for fast motions, capturing the discontinuities



that can be observed in long, thick hair. Nevertheless, very expensive computations were required for the examples shown, which was mainly due to the high cost for detecting collisions between the deformable wisps. Moreover, the high number of contacts that needed to be computed between each wisp at rest caused some visible artefacts in the rest state.

Choe *et al.* [38] have recently improved the stability of this kind of approaches. Collisions between the wisps and the body are robustly handled by using constrained dynamics. Moreover, to avoid undesired oscillations when computing wisp-wisp interactions, they propose an empiric law for controlling the amplitude of penalty forces. A cohesive force is also used to preserve the initial hairstyle during the simulation.

#### D. Multi-resolution Methods

Recently, researchers have begun to explore adaptive representations for hair. These methods can be used to alleviate unnatural clumping of hair strands that can be common in other approaches or to accelerate simulation while preserving realistic features in hair motion.

1) *Level-of-Detail Representations*: To better capture natural clustering of hair, a multi-resolution hair modeling scheme may be used to accelerate both the simulation and rendering of hair while maintaining a high visual quality. Ward *et al.* [53], [37] use three different levels of detail (LODs) for modeling hair – *individual strands*, *clusters* and *strips* represented by subdivision curves, subdivision swept volumes, and subdivision patches, respectively (see Figure 21, left). By creating a hair hierarchy composed of these three discrete LODs along with an efficient collision detection method that uses the family of *swept sphere volumes* (SSVs) [54] as bounding volumes to encapsulate the hair, this method was able to accelerate hair simulation up to two orders of magnitude.

During simulation, the hair hierarchy is traversed to choose the appropriate representation and resolution of a given section of hair. The transitions between the LODs occur automatically using a higher resolution simulation for the sections of hair that are most significant to the application based on the hair’s visibility, viewing distance, and motion, relative to the viewer. If an object in the scene occludes a section of hair or if the hair is outside of the field-of-view of the camera then the section is simulated at the coarsest LOD (a strip) and is not rendered. If the hair can be viewed, the distance of the viewer from the hair and its motion determine its current resolution. As the distance from the hair to the viewer decreases, or as the hair moves more drastically, there is more observable detail and a need for a more detailed simulation within the hair, thus the hair is simulated and rendered at higher resolutions. Figure 21 shows LOD representations (left) used for simulating hair blowing in the wind (right).

2) *Adaptive Clustering*: In order to *continuously* adjust the amount of computations according to the local complexity of motion, techniques for adaptive clustering and subdivision of simulated hair have been proposed recently [39], [37]. Bertails *et al.* [39] introduced an adaptive animation control structure, called *Adaptive Wisp Tree* (AWT), that enables the



Fig. 21. Left: Level-of-detail representations for hair (a) strip (b) cluster (c) strand. Right : Curly, long hair blowing in the wind using LOD representations [53].

dynamic splitting and merging of hair clusters. The AWT depends on a complete hierarchical structure for the hair, which can either be precomputed—for instance using a hierarchical hairstyle [9]—or computed on the fly. The AWT represents at each time step the wisps segments of the hierarchy that are actually simulated (called *active* segments). Considering that hair should always be more refined near the tips than near the roots, the AWT dynamically splits or merges hair wisps while always preserving a tree-like structure, in which the root coincides with the hair roots and the leaves stand for the hair tips.

At each time step, different wisps segments of the global hierarchy, that is, different LOD, can thus be active, while only the finest levels of details are used at the rendering stage. The splitting process locally refines the hair structure when a given wisp segment is not sufficient for capturing the local motion and deformation. The merging process simplifies the AWT when the motion becomes coherent again. Splitting and merging criteria are linked to the local motion of hair (for example, the magnitude of velocity of the wisps segments) at each time step.

One of the key benefits of the AWT is that it implicitly models mutual hair interactions so that neighboring wisps with similar motions merge, mimicking the static friction in real hair. This avoids subsequent collision processing between these wisps, thus increasing efficiency as well as gaining stability from the reduced number of primitives. In addition, the splitting behavior models wisps deformation without the need of the complex deformable wisp geometry used in [12]. For collision processing, active wisp segments of the AWT are thus represented by cylinders, which greatly simplifies collision detection tests.

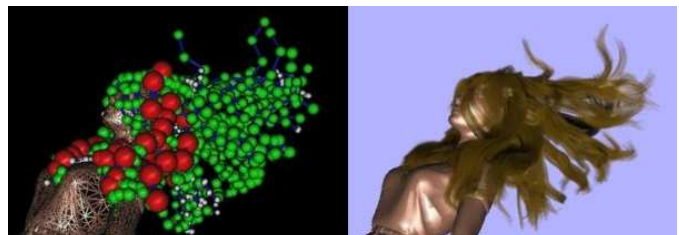


Fig. 22. Illustration of the AWT on long hair (left) and its final rendered version (right) [39].

Ward and Lin [37] proposed a similar, but a more top-down approach, for animating hair. Their continuous multi-resolution structure, called *hair hierarchy* [37], is coupled with the level-of-detail representations [53], instead of wisps [12].

#### IV. HAIR RENDERING

Realistic rendering of human hair requires the handling of both local and global hair properties. To render a full hairstyle, it is necessary to choose an appropriate global representation for hair. Implicit and explicit representations are presented and discussed in Section IV-A. Local hair properties define the way individual hair fibers are illuminated. Section IV-B describes the scattering properties of hair and reviews the different models that have been proposed to account for those properties. Global hair properties also include the way hair fibers cast shadows on each other; this issue of self-shadowing, handled in Section IV-C, plays a crucial role in volumetric hair appearance. Rendering hair typically requires time-consuming computations, Section IV-D reviews various rendering acceleration techniques.

##### A. Representing Hair for Rendering

Choices of hair rendering algorithms largely depend on the underlying representations for modeling hair geometry. For example, explicit models require line or triangle-based renderers, whereas volumetric models need volume renderers, or rendering algorithms that work on implicit geometry.

1) *Explicit Representation*: With an explicit representation, one has to draw each hair fiber. A hair fiber is naturally represented with a curved cylinder. The early work by Watanabe and Suenaga [51] adopted a trigonal prism representation, where each hair strand is represented as connected prisms with three sides. This method assumes that variation in color along the hair radius can be well approximated by a single color. Others use ribbon-like connected triangle strips to represent hair, where each triangle always faces towards the camera. Ivan Neulander [55] introduced a technique that adaptively tessellates a curved hair geometry into polygons depending on the distance to the camera, curvature of hair geometry, etc. At large distances, a hair strand often resembles many hairs. Kong and Nakajima [56] exploited this property to reduce the number of rendered hairs by adaptively creating more hairs at the boundary.

Difficulties arise with explicit rendering of tessellated hair geometry due to the unique nature of hair – a hair strand is extremely thin in diameter (0.1 mm). In a normal viewing condition, the projected thickness of a hair strand is much smaller than the size of a pixel. This property causes severe undersampling problems for rendering algorithms for polygonal geometry. Any point sample-based renderer determines a pixel’s color (or depth) by a limited number of discrete samples. Undersampling creates abrupt changes in color or noisy edges around the hair. Increasing the number of samples alleviates the problem, but only at slow convergence rates [57] and consequently at increased rendering costs.

LeBlanc *et al.* [58] addressed this issue by properly blending each hair’s color using a pixel blending buffer technique. In

this method, each hair strand is drawn as connected lines and the shaded color is blended into a pixel buffer. When using alpha-blending, one should be careful with the drawing order. Kim and Neumann [9] also use an approximate visibility ordering method to interactively draw hairs with OpenGL’s alpha blending.

2) *Implicit Representation*: Volumetric textures (or *texels*) [59], [60] avoid the aliasing problem with pre-filtered shading functions. The smallest primitive is a volumetric cell that can be easily mip-mapped to be used at multiple scales. The cost of ray traversal is relatively low for short hairs, but can be high for long hairs. Also when hair animates, such volumes should be updated for every frame, making pre-filtering inefficient.

The rendering method of the cluster hair model [20] also exploits implicit geometry. Each cluster is first approximated by a polygonal boundary. When a ray hits the polygonal surface, predefined density functions are used to accumulate density. By approximating the high frequency detail with volume density functions, the method produces antialiased images of hair clusters. However, this method does not allow changes in the density functions, making hairs appear as if they always stay together.

##### B. Light Scattering in Hair

The first requirement for any hair rendering system is a model for the scattering of light by individual fibers of hair. This model plays the same role in hair rendering as a surface reflection, or local illumination, model does in conventional surface rendering.

1) *Hair Optical Properties*: The composition and microscopic structure of hair are important to its appearance. A hair fiber is composed of three structures: the cortex, which is the core of the fiber and provides its physical strength, the cuticle, a coating of protective scales that completely covers the cortex several layers thick (see Figure 11 in Section III-A), and the medulla, a structure of unknown function that sometimes appears near the axis of the fiber. As already mentioned in Section II-A, the cross sectional shape varies from circular to elliptical to irregular [4].

Much is known about the chemistry of hair, but for the purposes of optics it suffices to know that it is composed of amorphous proteins that act as a transparent medium with an index of refraction  $\eta = 1.55$  [4], [61]. The cortex and medulla contain pigments that absorb light, often in a wavelength-dependent way; these pigments are the cause of the color of hair.

2) *Notation and Radiometry of Fiber Reflection*: Our notation for scattering geometry is summarized in Figure 23. We refer to the plane perpendicular to the fiber as the *normal plane*. The direction of illumination is  $\omega_i$ , and the direction in which scattered light is being computed or measured is  $\omega_r$ ; both direction vectors point away from the center. We express  $\omega_i$  and  $\omega_r$  in spherical coordinates. The inclinations with respect to the normal plane are denoted  $\theta_i$  and  $\theta_r$  (measured so that 0 degree is perpendicular to the hair). The azimuths around the hair are denoted  $\phi_i$  and  $\phi_r$ , and the relative azimuth  $\phi_r - \phi_i$ , which is sufficient for circular fibers, is denoted  $\Delta\phi$ .

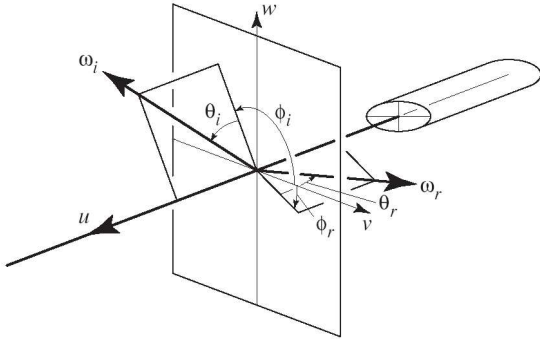


Fig. 23. Notation for scattering geometry

Because fibers are usually treated as one-dimensional entities, light reflection from fibers needs to be described somewhat differently from the more familiar surface reflection. Light scattering at a surface is conventionally described using the bidirectional reflectance distribution function (BRDF),  $f_r(\omega_i, \omega_r)$ . The BRDF gives the density with respect to the projected solid angle of the scattered flux that results from a narrow incident beam from the direction  $\omega_i$ . It is defined as the ratio of surface radiance (intensity per unit projected area) exiting the surface in direction  $\omega_r$  to surface irradiance (flux per unit area) falling on the surface from a differential solid angle in the direction  $\omega_i$ :

$$f_r(\omega_i, \omega_r) = \frac{dL_r(\omega_r)}{dE_i(\omega_i)}.$$

Under this definition, the radiance due to an incoming radiance distribution  $L_i(\omega_i)$  is

$$L_r(\omega_r) = \int_{H^2} f_r(\omega_i, \omega_r) L_i(\omega_i) \cos \theta_i d\omega_i$$

where  $H^2$  is the hemisphere of directions above the surface.

Light scattering from fibers is described similarly, but the units for measuring the incident and reflected light are different because the light is being reflected from a one-dimensional curve [62]. If we replace “surface” with “curve” and “area” with “length” in the definition above we obtain a definition of the scattering function  $f_s$  for a fiber: “the ratio of *curve radiance* (intensity per unit projected length) exiting the curve in direction  $\omega_r$  to *curve irradiance* (flux per unit length) falling on the curve from a differential solid angle in the direction  $\omega_i$ .” The curve radiance due to illumination from an incoming radiance distribution  $L_i$  is

$$L_r^c(\omega_r) = D \int_{H^2} f_s(\omega_i, \omega_r) L_i(\omega_i) \cos \theta_i d\omega_i$$

where  $D$  is the diameter of the hair as seen from the illumination direction.

This transformation motivated Marschner et al. [62] to introduce curve radiance and curve irradiance. Curve radiance is in some sense halfway between the concepts of radiance and intensity, and it describes the contribution of a thin fiber to an image independent of its width. Curve irradiance measures the radiant power intercepted per unit length of fiber and therefore

increases with the fiber’s width. Thus, given two fibers with identical properties but different widths, both will have the same scattering function but the wider fiber will produce a brighter curve in a rendered image because the wider fiber intercepts more incident light. This definition is consistent with the behavior of real fibers: very fine hairs do appear fainter when viewed in isolation.

Most of the hair scattering literature does not discuss radiometry, but the above definitions formalize the common practice, except that the diameter of the hair is normally omitted since it is just a constant factor. The factor of  $\cos \theta_i$  is often included in the model, as was common in early presentations of surface shading models.

3) *Reflection and Refraction in Cylinders*: For specular reflection, a hair can be modeled, to a first approximation, as a transparent (if lightly pigmented) or purely reflecting (if highly pigmented) dielectric cylinder. The light-scattering properties of cylinders have been extensively studied in order to inversely determine the properties of optical fibers by examining their scattering [63], [64], [65].

As first presented in graphics by Kajiya and Kay [59] (their scattering model is presented in Section IV-B.5), if we consider a bundle of parallel rays that illuminates a smooth cylinder, each ray will reflect across the local surface normal at the point where it strikes the surface. These surface normals are all perpendicular to the fiber axis—they lie in the normal plane. Because the direction of each reflected ray is symmetric to the incident direction across the local normal, all the reflected rays will make the same angle with the normal plane. This means that the reflected distribution from a parallel beam due to specular reflection from the surface lies in a cone at the same inclination as the incident beam.

For hairs that are not darkly pigmented, the component of light that is refracted and enters the interior of the hair is also important. As a consequence of Bravais’s Law [66], a corollary of Snell’s Law that is often used to describe refractions through crystals with a cylinder-like structure, the directions of the rays that are refracted through the cylinder surface also fall on a cone centered on the cylinder axis. The same holds for the refractions as the rays exit the cylinder. Therefore all specularly reflected light from a smooth cylinder will emit on the same cone as the surface reflection, no matter what sequence of refractions and internal reflections it may have taken.

4) *Measurements of Hair Scattering*: In cosmetics literature, some measurements of incidence-plane scattering from fibers have been published. Stamm et al. [61] made measurements of reflection from an array of parallel fibers. They observed several remarkable departures from the expected reflection into the specular cone: there are two specular peaks, one on either side of the specular direction, and there is a sharp true specular peak that emerges at grazing angles. The authors explained the presence of the two peaks using an incidence-plane analysis of light reflecting from the tilted scales that cover the fiber, with the surface reflection and the first-order internal reflection explaining the two specular peaks.

A later paper by Bustard and Smith [67] reported additional measurements of single fibers, including measuring the four

combinations of incident and scattered linear polarization states. They found that one of the specular peaks was mainly depolarized while the other preserved the polarization. This discovery provided additional evidence for the explanation of one lobe from the surface reflection and one from the internal reflection.

Bustard and Smith also discussed preliminary results of an azimuthal measurement, performed with illumination and viewing perpendicular to the fiber. They reported finding bright peaks in the azimuthal distribution and speculated that they were due to caustic formation, but they did not report any data.

Marschner *et al.* [62] reported measurements of single fibers in more general geometries. In addition to incidence plane measurements, they presented normal plane measurements that show in detail the peaks that Bustard and Smith discussed and how they evolve as a strand of hair is rotated around its axis. The authors referred to these peaks as “glints” and showed a simulation of scattering from an elliptical cylinder that predicts the evolution of the glints; this clearly confirmed that the glints are caused by caustic formation in internal reflection paths. They also reported some higher-dimensional measurements that show the evolution of the peaks with the angle of incidence, which showed the full scattered distribution for a particular angle of incidence.



Fig. 24. Comparison between Kajiyá's model (left), Marschner's model (middle) and real hair (right).

5) *Models for Hair Scattering*: The earliest and most widely used model for hair scattering is Kajiyá and Kay's model which was developed for rendering fur [59]. This model includes a diffuse component and a specular component:

$$S(\theta_i, \phi_i, \theta_r, \phi_r) = k_d + k_s \frac{\cos^p(\theta_r + \theta_i)}{\cos(\theta_i)}.$$

Kajiyá and Kay derived the diffuse component by integrating reflected radiance across the width of an opaque, diffuse cylinder. Their specular component is simply motivated by the argument from the preceding section that the ideal specular reflection from the surface will be confined to a cone and therefore the reflection from a non-ideal fiber should be a lobe concentrated near that cone. Note that neither the peak value nor the width of the specular lobe changes with  $\theta$  or  $\phi$ .

Banks [68] later re-explained the same model based on more minimal geometric arguments. For diffuse reflection, a differential piece of fiber is illuminated by a beam with a cross section proportional to  $\cos \theta_i$  and the diffusely reflected power

emits uniformly to all directions.<sup>1</sup> For specular reflection, Fermat's principle requires that the projection of the incident and reflected rays onto the fiber be the same.

In another paper on rendering fur, Goldman [69], among a number of other refinements to the aggregate shading model, proposed a refinement to introduce azimuthal dependence into the fiber scattering model. He multiplied both terms of the model by a factor  $f_{dir}$  that can be expressed in the current notation as:

$$f_{dir} = 1 + a \cos \Delta\phi.$$

Setting  $a > 0$  serves to bias the model toward backward scattering, while setting  $a < 0$  biases the model towards forward scattering.<sup>2</sup>

Tae-Yong Kim [70] proposed another model for azimuthal dependence, which accounts for surface reflection and transmission using two cosine lobes. The surface reflection lobe derives from the assumption of mirror reflection with constant reflectance (that is, ignoring the Fresnel factor), and the transmission lobe is designed empirically to give a forward-focused lobe. The model is built on Kajiyá-Kay in the same way Goldman's is, defining:

$$g(\phi) = \begin{cases} \cos \phi & -\frac{\pi}{2} < \phi < \frac{\pi}{2} \\ 0 & \text{otherwise} \end{cases}$$

This model is Kajiyá and Kay's model multiplied by:

$$f_{dir} = a g(\Delta\phi/2) + g(k(\Delta\phi - \pi))$$

where  $a$  is used to balance forward and backward scattering and  $k$  is a parameter to control how focused the forward scattering is. The first term is for backward (surface) scattering and the second term is for forward (transmitted) scattering.

Marschner *et al.* [62] proposed the most complete physically-based hair scattering model to date. Their model makes two improvements to Kajiyá and Kay's model: it predicts the azimuthal variation in scattered light based on the ray optics of a cylinder, and it accounts for the longitudinal separation of the highlight into surface-reflection, transmission, and internal-reflection components that emerge at different angles. The azimuthal component of the model is based on a ray analysis that accounts for focusing and dispersion of light, absorption in the interior, and Fresnel reflection at each interaction. The longitudinal component models the shifts of the first three orders of reflection empirically using lobes that are displaced from the specular cone by specific angles.

6) *Light Scattering on Wet Hair*: The way light scatters on hair is changed when hair becomes wet. Jensen *et al.* [71] noted that when objects become wet they typically appear darker and shinier; hair behaves the same way. Bruderlin [72] and Ward *et al.* [33] altered previous light scattering models to capture the effects of wet fur and wet hair, respectively.

As hair becomes wet, a thin film of water is formed around the fibers, forming a smooth, mirror-like surface on the hair.

<sup>1</sup>Banks does not discuss why uniform curve radiance is the appropriate sense in which the scattered light should be uniform.

<sup>2</sup>In Goldman's original notation  $a = (\rho_{reflect} - \rho_{transmit}) / (\rho_{reflect} + \rho_{transmit})$ . A factor of  $\frac{1}{2}(\rho_{reflect} + \rho_{transmit})$  can be absorbed into the diffuse and specular coefficients.



In contrast to the naturally rough, tiled surface of dry hair, this smoother surface creates a shinier appearance of the hair due to increased specular reflections. Furthermore, light rays are subject to total internal reflection inside the film of water around the hair strands, contributing to the darker appearance wet hair has over dry hair. Moreover, water is absorbed into the hair fiber, increasing the opacity value of each strand leading to more aggressive self-shadowing (see Section IV-C).

Bruderlin [72] and Ward *et al.* [33] modeled wet strands by increasing the amount of specular reflection. Furthermore, by increasing the opacity value of the hair, the fibers attain a darker and shinier look, resembling the appearance of wet hair (see Figure 10).

### C. Hair Self-Shadowing



Fig. 25. Importance of self-shadowing on hair appearance. (left) Shadows computed using Deep Shadow Maps [73] compared to (right) No shadows. Images courtesy of Pixar Animation Studios.

Hair fibers cast shadows onto each other, as well as receiving and casting shadows from and to other objects in the scene. Self-shadowing creates crucial visual patterns that distinguish one hairstyle from another, see Figure 25. Unlike solid objects, a dense volume of hair exhibits complex light propagation patterns. Each hair fiber transmits and scatters rather than fully blocks the incoming lights. The strong forward scattering properties as well as the complex underlying geometry make the shadow computation difficult.

One can ray trace hair geometry to compute shadow, whether hair is represented by implicit models [59] or explicit models [62]. For complex geometry, the cost of ray traversal can be expensive and many authors turn to caching schemes for efficiency. Two main techniques are generally used to cast self-shadows into volumetric objects: ray casting through volumetric densities and shadow maps.

1) *Ray-casting through a Volumetric Representation*: With implicit hair representations, one can directly ray trace volume density [20], or use two-pass shadowing schemes for volume density [59]; the first pass fills volume density with shadow information and the second pass renders the volume density.

2) *Shadow Maps*: LeBlanc [58] introduced the use of the shadow map, a depth image of hair rendered from the light’s point of view. In this technique, hair and other objects are rendered from the light’s point of view and the depth values are stored. Each point to be shadowed is projected onto the light’s camera and the point’s depth is checked against the depth in the shadow map. Kong and Nakijima [56] extended

the principle of shadow caching to the visible volume buffer, where shadow information is stored in a 3D grid.

In complex hair volumes, depths can vary radically over small changes in image space. The discrete nature of depth sampling limits shadow buffers in handling hair. Moreover, lights tend to gradually attenuate through hair fibers due to forward scattering. The binary decision in depth testing inherently precludes such light transmission phenomena. Thus, shadow buffers are unsuitable for volumetric hair.

The transmittance  $\tau(p)$  of a light to a point  $p$  can be written as:

$$\tau(p) = \exp(-\Omega), \text{ where } \Omega = \int_0^l \sigma_t(l') dl'.$$

$l$  is the length of a path from the light to  $p$ ,  $\sigma_t$  is the extinction (or density) function along the path.  $\Omega$  is the opacity thickness (or accumulated extinction function).

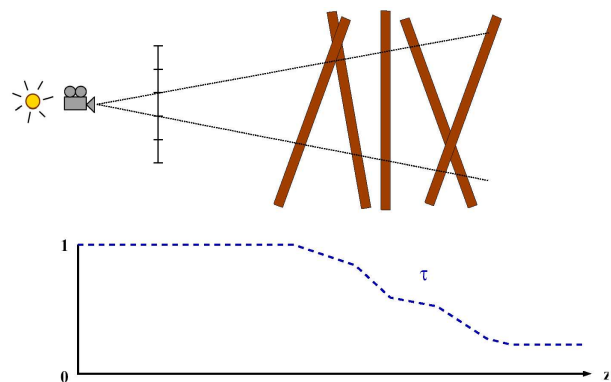


Fig. 26. Top: a beam of light starting at the shadow camera origin (*i.e.*, the light source) and passing through a single pixel of the deep shadow map. Bottom: the corresponding transmittance (or visibility) function  $\tau$ , stored as a piecewise linear function.

In the deep shadow maps technique [73], each pixel stores a piecewise linear approximation of the transmittance function instead of a single depth, yielding more precise shadow computations than shadow maps, see Figure 26 for an illustration. The transmittance function accounts for two important properties of hair.

**Fractional Visibility**: In the context of hair rendering, the transmittance function can be regarded as a fractional visibility function from the light’s point of view. If more hair fibers are seen along the path from the light, the light gets more attenuated (occluded), resulting in less illumination (shadow). As noted earlier, visibility can change drastically over the pixel’s extent. To handle this partial visibility problem, one should accurately compute the transmission function by correctly integrating and filtering all the contributions from the underlying geometry.

**Translucency**: A hair fiber not only absorbs, but also scatters and transmits the incoming light. Assuming that the hair fiber transmits the incoming light only in a forward direction, the translucency is also handled by the transmittance function.

Noting that the transmittance function typically varies radically over image space, but gradually along the light direction,

one can accurately approximate the transmittance function with a compact representation. Deep shadow maps [73] use a compressed piecewise linear function for each pixel, along with special handling for discontinuities in transmittance. Figure 25 shows a comparison of hair geometry with and without shadows using the deep shadow maps algorithm.

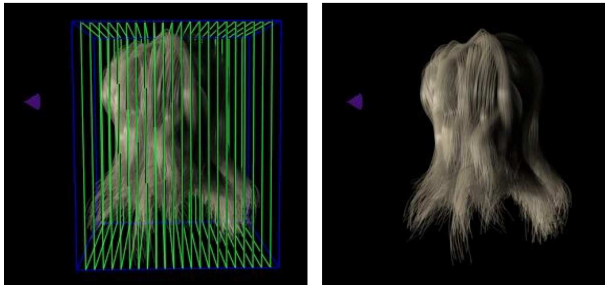


Fig. 27. Opacity Shadow Maps. Hair volume is uniformly sliced perpendicular to the light direction into a set of planar maps storing alpha values (top). The resulting shadowed hair (bottom).

Opacity shadow maps [74] further assume that such transmittance functions always vary smoothly, and can thus be approximated with a set of fixed image caches perpendicular to the lighting direction (see Figure 27). By approximating the transmittance function with discrete planar maps, opacity maps can be efficiently generated with graphics hardware (see Section IV-D.3). Linear interpolation from such maps facilitates fast approximation to hair self-shadows.

#### D. Rendering Acceleration Techniques

Accurately rendering complex hairstyles can take several minutes for one frame. Many applications, such as games or virtual reality, require real-time rendering of hair. These demands have initiated recent work to accelerate precise rendering algorithms by simplifying the geometric representation of hair, by developing fast volumetric rendering, or by utilizing recent advances in graphics hardware.

1) *Approximating Hair Geometry*: Section IV-B explained the structure of hair and showed that hair fibers are actually quite complex. Simplifying this geometry, using fewer vertices and rendering fewer strands, is one strategy for accelerating hair rendering. Removing large portions of hair strands can be distracting and unrealistic, therefore surfaces and strips have been used for approximating large numbers of hair strands [16], [47], [48], [75].

These two-dimensional representations resemble hair by texture mapping the surfaces with hair images and using alpha mapping to give the illusion of individual hair strands. Curly wisps can be generated by projecting the hair patch onto a cylindrical surface [75].

Level of detail (LOD) representations used by Ward *et al.* [53], [37] (see Section III-D.1) for accelerating the dynamic simulation of hair, also accelerates hair rendering. Using a coarse LOD to model hair that cannot be seen well by the viewer requires rendering fewer vertices with little loss in visual fidelity. As a result, the time required to calculate light scattering and shadowing effects is diminished by an order of magnitude.

2) *Interactive Volumetric Rendering*: Bando *et al.* [10] modeled hair as a set of connected particles, where particles represent hair volume density. Their rendering method was inspired by fast cloud rendering techniques [76] where each particle is rendered by splatting a textured billboard, both for self-shadowing computation and final rendering. This method runs interactively, but it does not cast very accurate shadows inside hair (see Figure 15).

Bertails *et al.* [77] use a light-oriented voxel grid to store hair density values, which enables them to efficiently compute accumulative transmittance inside the hair volume. Transmittance values are then filtered and combined with diffuse and specular components to calculate the final color of each hair segment. Though very simple, this method yields convincing interactive results for animated hair (see Figure 28). Moreover, it can easily be parallelized to increase performance.



Fig. 28. Interactive hair self-shadowing processed by accumulating transmittance values through a light-oriented voxel grid [77]. (left) Animated hair without self-shadows; (right) Animated hair with self-shadows.

3) *Graphics Hardware*: Many impressive advances have been made recently in programmable graphics hardware. Graphics processor units (GPUs) now allow programming of more and more complex operations through dedicated languages, such as Cg. For example, various shaders can directly be implemented on the hardware, which greatly improves performance. Currently, the major drawback of advanced GPU programming is that new features are neither easy to implement nor portable across different graphics cards.

Heidrich and Seidel [78] efficiently render anisotropic surfaces by using OpenGL texture mapping. Anisotropic reflections of individual hair fibers have also been implemented with this method for straightforward efficiency.

As for hair self-shadowing, some approaches have recently focused on the acceleration of the opacity shadow maps algorithm (presented in Section IV-C.2), by using the recent capabilities of GPUs. Koster *et al.* [75] exploited graphics hardware by storing all the opacity maps in a 3D texture, to have the hair self-shadow computation done purely in graphics hardware. Using textured strips to simplify hair geometry (as seen in Section IV-D.1), they achieve real-time performance. Mertens *et al.* [79] explored efficient hair density clustering schemes suited for graphics hardware, achieving interactive rates for high quality shadow generation in dynamically changing hair geometry. Finally, a real-time demonstration showing long hair moving in the sea was presented by NVidia in 2004 [80] to illustrate the new capabilities of their latest graphics cards (see Figure 29).



Fig. 29. Real-time rendering of long, moving hair using recent graphics hardware [80]. Image Courtesy of NVIDIA Corporation, 2004

## V. NEW CHALLENGES

As the need for hair modeling continues to grow in a wide spectrum of applications, the computational demands on hairstyling, animation, and rendering become even more challenging. Depending on the targeted application, the main focus may be put either on physically-based realism (for cosmetic prototyping), visual realism with a high user control (for feature films), or computations acceleration (for virtual environments and videogames). Some of these goals have been partially achieved, but many important issues still remain, especially in the field of hair animation.

### A. Hairstyling

One of the most difficult challenges to virtual hairstyling remains to be creating intricate styles with a high level of user control in a short amount of time. There is typically a tradeoff between the amount of user control and the amount of manual input time. An interesting future direction in hairstyling could be to combine different shaping techniques in a manner that keeps a high degree of user control while still accelerating the time for user input. Moreover, haptic techniques for 3D user input have shown to be quite effective for mimicking real-world human interactions and have only recently been explored for hairstyling [7]. Attaining input through haptic gloves rather than through traditional mouse and keyboard operations is a possibility that could allow a user to interact with hair in a manner similar to real-world human-hair interactions. Creating a braid, for example, could potentially be performed in just minutes with haptic feedback, similar to real-world hairstyling.

In addition to user input, interactive virtual hairstyling techniques can also benefit from accelerations in rendering and simulation. While most styling techniques are targeted towards

static hair, faster hair animation and rendering techniques would enable more realistic human-hair interaction. Styling of dynamic hair would be beneficial for cosmetic training and other interactive hairstyling functions. These high-performance applications demand the ability to interact accurately with hair via common activities, such as combing or brushing hair, in real time. But as explained in next Section, hair dynamic behavior as well as hair interactions are currently far from being satisfactorily simulated, especially in terms of accuracy.

### B. Animation

Individual hair strands and groups of strands (or wisps) have been animated in various ways. Some of these techniques yield a fair visual appearance of moving hair, but still, they lack a high degree of physical basis. Unlike some other mechanical systems, such as fluids, hair has not been deeply studied by physicists, and thus no macroscopic model describing the accurate dynamics of hair (individual and collective behavior) is currently available. As a result, computer scientists have built hair dynamics models that are often approximations of reality, and they seldom propose any rigorous validation of their models. Some recent work accounting for relevant structural and mechanical properties of hair starts to explore and to develop new mechanical models for simulating more closely the complex, nonlinear behavior of hair [6]. But for the moment, only static hairstyles are computed. Moreover, proposing a validated model for handling hair-hair interactions remains an open issue. Such investigations would benefit applications such as medical simulations and cosmetic prototyping.

While hair animation methods still lack physically-based grounds, many advances have been made in terms of performance through the use of hair strips (Section III-C.2.a), FFD (Section III-C.1.d), and multi-resolution techniques (Section III-D), but each of these methods have various limitations to overcome. Hair strips can be used for real-time animation of hair, though hairstyles and hair motions are limited to simple examples due to the flat surface representation of the hair. Multi-resolution techniques have been able to model some important features of hair behaviors, including dynamic grouping and separation of hair strands, and have successfully accelerated hair simulation while preserving visual fidelity to a certain extent. However, highly complex hairstyles with motion constraints are still not simulated in real-time with these multi-resolution methods. FFD methods have been used to attain real-time animation of various hairstyles; nevertheless such approaches are limited mainly to small deformations of hair. It would be interesting to explore the synthesis of one or more of these techniques by drawing on their strengths; for example, the use of an FFD approach that would allow for the hair volume to split into smaller groups for finer detail.

### C. Rendering

Whereas very little physical data is available for hair mechanical properties, especially the way a collection of hair fibers behave together during motion, the microscopic structure of hair is well-known (Section IV-B.1). Measurements of hair scattering have recently led researchers to propose



an accurate physically-based model for a single hair fiber, accounting for multiple highlights observable in real hair (Section IV-B.5). So far, this model is only valid for a single hair fiber. Other complex phenomena such as inter-reflection inside the hair volume should also be considered for capturing the typical hair lighting effects. Another important aspect of hair is self-shadowing. Many existing approaches already yield convincing results. The most challenging issue perhaps lies in simulating accurate models for both the scattering of individual hair fibers and the computations of self-shadows at *interactive* rates.

## VI. CONCLUSION

We presented a literature review on hair styling, simulation, and rendering. For hairstyling, the more flexible methods rely mostly on manual design from the user. Intuitive user interfaces and pseudo-physical algorithms contribute to simplifying the user's task, while recent approaches capturing hair geometry from photographs automatically generate existing hairstyles. Various methods for animating hair have also been described, such as through a continuous medium or disjoint groups of hairs. Existing hair simulation techniques typically require a tradeoff among visual quality, flexibility in representing styles and hair motion, and computational performance. We also showed how multi-resolution techniques can be used to automatically balance this tradeoff. Finally, we discussed the main issues in hair rendering. We explained the effects of light scattering on hair fibers, explored techniques for representing explicit and implicit hair geometry, and examined different shadowing methods for rendering hair.

Hair modeling remains an active area of research. Depending on the specific field—styling, animation or rendering—different levels of realism and efficiency have been made. While hair rendering is probably the most advanced field, styling and above all animation still raise numerous unsolved issues. Researchers have begun to explore techniques that will enable more authentic user experiences with hair.

## REFERENCES

- [1] N. Magnenat-Thalmann and S. Hadap, "State of the art in hair simulation," in *International Workshop on Human Modeling and Animation*, ser. Korea Computer Graphics Society, June 2000, pp. 3–9.
- [2] R. Rosenblum, W. Carlson, and E. Tripp, "Simulating the structure and dynamics of human hair: Modeling, rendering, and animation," *The Journal of Visualization and Computer Animation*, vol. 2, no. 4, pp. 141–148, 1991.
- [3] K. Anjyo, Y. Usami, and T. Kurihara, "A simple method for extracting the natural beauty of hair," in *Proceedings of ACM SIGGRAPH 1992*, ser. Computer Graphics Proceedings, Annual Conference Series, August 1992, pp. 111–120.
- [4] C. R. Robbins, *Chemical and Physical Behavior of Human Hair*, 3rd ed. Springer-Verlag, New York, 1994.
- [5] L'Oréal, "Hair science," 2005, <http://www.hair-science.com>.
- [6] F. Bertails, B. Audoly, B. Querleux, F. Leroy, J.-L. Lévêque, and M.-P. Cani, "Predicting natural hair shapes by solving the statics of flexible rods," in *Eurographics (short papers)*, J. Dingliana and F. Ganovelli, Eds. Eurographics, August 2005. [Online]. Available: <http://www-evasion.imag.fr/Publications/2005/BAQLLC05>
- [7] K. Ward, N. Galoppo, and M. Lin, "A simulation-based vr system for interactive hairstyling," in *IEEE Virtual Reality - Application and Research Sketches (to appear)*, 2006. [Online]. Available: <http://gamma.cs.unc.edu/iSalon>
- [8] Y. Yu, "Modeling realistic virtual hairstyles," in *Proceedings of Pacific Graphics'01*, Oct. 2001, pp. 295–304.
- [9] T.-Y. Kim and U. Neumann, "Interactive multiresolution hair modeling and editing," *ACM Transactions on Graphics*, vol. 21, no. 3, pp. 620–629, July 2002, proceedings of ACM SIGGRAPH 2002.
- [10] Y. Bando, B.-Y. Chen, and T. Nishita, "Animating hair with loosely connected particles," *Computer Graphics Forum*, vol. 22, no. 3, pp. 411–418, 2003, proceedings of Eurographics'03.
- [11] D. Patrick and S. Bangay, "A lightwave 3d plug-in for modeling long hair on virtual humans," in *Proceedings of the 2nd international conference on Computer graphics, virtual Reality, visualisation and interaction in Africa*. ACM Press, 2003, pp. 161–187.
- [12] E. Plante, M.-P. Cani, and P. Poulin, "A layered wisp model for simulating interactions inside long hair," in *Computer Animation and Simulation 2001Proceeding*, ser. Computer Science, N. M.-T. Marie-Paule Cani, Daniel Thalmann, Ed., EUROGRAPHICS. Springer, sep 2001, proceedings of the EG workshop of Animation and Simulation. [Online]. Available: <http://www-imagis.imag.fr/Publications/2001/PCP01>
- [13] B. Choe and H.-S. Ko, "A statistical wisp model and pseudophysical approaches for interactive hairstyle generation," *IEEE Transactions on Visualization and Computer Graphics*, vol. 11, no. 2, March 2005.
- [14] L. Chen, S. Saeyor, H. Dohi, and M. Ishizuka, "A system of 3d hairstyle synthesis based on the wisp model," *The Visual Computer*, vol. 15, no. 4, pp. 159–170, 1999.
- [15] B. Hernandez and I. Rudomin, "Hair paint," in *Computer Graphics International (CGI)*, June 2004, pp. 578–581.
- [16] C. Koh and Z. Huang, "Real-time animation of human hair modeled in strips," in *Computer Animation and Simulation'00*, Sept. 2000, pp. 101–112.
- [17] W. Liang and Z. Huang, "An enhanced framework for real-time hair animation," in *Pacific Graphics Conference on Computer Graphics and Applications*, October 2003.
- [18] P. Noble and W. Tang, "Modelling and animating cartoon hair with nurbs surfaces," in *Computer Graphics International (CGI)*, June 2004, pp. 60–67.
- [19] T.-Y. Kim and U. Neumann, "A thin shell volume for modeling human hair," in *Computer Animation 2000*, ser. IEEE Computer Society, 2000, pp. 121–128.
- [20] X. D. Yang, Z. Xu, T. Wang, and J. Yang, "The cluster hair model," *Graphics Models and Image Processing*, vol. 62, no. 2, pp. 85–103, Mar. 2000.
- [21] Z. Xu and X. D. Yang, "V-hairstudio: an interactive tool for hair design," *IEEE Computer Graphics & Applications*, vol. 21, no. 3, pp. 36–42, May / June 2001.
- [22] D. Patrick, S. Bangay, and A. Lobb, "Modelling and rendering techniques for african hairstyles," in *Proceedings of the 3rd international conference on Computer graphics, virtual reality, visualisation and interaction in Africa*. ACM Press, 2004, pp. 115–124.
- [23] T. Wang and X. D. Yang, "Hair design based on the hierarchical cluster hair model," *Geometric modeling: techniques, applications, systems and tools*, pp. 330–359, 2004.
- [24] S. Hadap and N. Magnenat-Thalmann, "Interactive hair styler based on fluid fbw," in *Computer Animation and Simulation '00*, Aug. 2000, pp. 87–100.
- [25] J. Stam, "Multi-scale stochastic modelling of complex natural phenomena," Ph.D. dissertation, University of Toronto, 1995.
- [26] W. Kong, H. Takahashi, and M. Nakajima, "Generation of 3d hair model from multiple pictures," in *Proceedings of Multimedia Modeling*, 1997, pp. 183–196.
- [27] S. Grabli, F. Sillion, S. R. Marschner, and J. E. Lengyel, "Image-based hair capture by inverse lighting," in *Proc. Graphics Interface*, May 2002, pp. 51–58. [Online]. Available: <http://artis.imag.fr/Publications/2002/GSML02>
- [28] S. Paris, H. Briceño, and F. Sillion, "Capture of hair geometry from multiple images," *ACM Transactions on Graphics (Proceedings of the SIGGRAPH conference)*, 2004. [Online]. Available: <http://artis.imag.fr/Publications/2004/PBS04/>
- [29] Y. Wei, E. Ofek, L. Quan, and H.-Y. Shum, "Modeling hair from multiple views," in *Proceedings of ACM SIGGRAPH'05*, 2005.
- [30] X. Mao, S. Isobe, K. Anjyo, and A. Imamiya, "Sketchy hairstyles," in *Proceedings of Computer Graphics International*, 2005.
- [31] D.-W. Lee and H.-S. Ko, "Natural hairstyle modeling and animation," *Graphical Models*, vol. 63, no. 2, pp. 67–85, March 2001.
- [32] J. T. Chang, J. Jin, and Y. Yu, "A practical model for hair mutual interactions," in *ACM SIGGRAPH Symposium on Computer Animation*, July 2002, pp. 73–80.



- [33] K. Ward, N. Galoppo, and M. C. Lin, "Modeling hair influenced by water and styling products," in *International Conference on Computer Animation and Social Agents (CASA)*, May 2004, pp. 207–214. [Online]. Available: <http://gamma.cs.unc.edu/HairWS>
- [34] C. Bouillon and J. Wilkinson, *The Science of Hair Care, second edition*. Taylor & Francis, 2005.
- [35] C. Zviak, *The Science of Hair Care*. Marcel Dekker, 1986.
- [36] D. Baraff and A. Witkin, "Large steps in cloth simulation," *Proc. of ACM SIGGRAPH*, pp. 43–54, 1998.
- [37] K. Ward and M. C. Lin, "Adaptive grouping and subdivision for simulating hair dynamics," in *Pacific Graphics Conference on Computer Graphics and Applications*, October 2003, pp. 234–243. [Online]. Available: <http://www.cs.unc.edu/geom/HAIR/>
- [38] B. Choe, M. Choi, and H.-S. Ko, "Simulating complex hair with robust collision handling," in *SCA '05: Proceedings of the 2005 ACM SIGGRAPH/Eurographics symposium on Computer animation*. New York, NY, USA: ACM Press, 2005, pp. 153–160.
- [39] F. Bertails, T.-Y. Kim, M.-P. Cani, and U. Neumann, "Adaptive wisp tree - a multiresolution control structure for simulating dynamic clustering in hair motion," in *ACM SIGGRAPH Symposium on Computer Animation*, July 2003, pp. 207–213. [Online]. Available: <http://www-imagis.imag.fr/Publications/2003/BKCN03>
- [40] S. Hadap and N. Magnenat-Thalmann, "Modeling dynamic hair as a continuum," *Computer Graphics Forum*, vol. 20, no. 3, pp. 329–338, 2001, proceedings of Eurographics'01.
- [41] R. Featherstone, *Robot Dynamics Algorithms*. Kluwer Academic Publishers, 1987.
- [42] S. Hadap, "Hair simulation," Ph.D. dissertation, University of Geneva, 2003.
- [43] D. Baraff, "Linear-time dynamics using lagrange multipliers," in *SIGGRAPH '96: Proceedings of the 23rd annual conference on Computer graphics and interactive techniques*. New York, NY, USA: ACM Press, 1996, pp. 137–146.
- [44] A. Daldegan, N. M. Thalmann, T. Kurihara, and D. Thalmann, "An integrated system for modeling, animating and rendering hair," *Computer Graphics Forum*, vol. 12, no. 3, pp. 211–221, 1993.
- [45] T. Kurihara, K. Anjyo, and D. Thalmann, "Hair animation with collision detection," in *Proceedings of Computer Animation '93*. Springer, 1993, pp. 128–138.
- [46] P. Volino and N. Magnenat-Thalmann, "Animating complex hairstyles in real-time," in *ACM Symposium on Virtual Reality Software and Technology*, 2004.
- [47] C. Koh and Z. Huang, "A simple physics model to animate human hair modeled in 2D strips in real time," in *Computer Animation and Simulation '01*, Sept. 2001, pp. 127–138.
- [48] Y. Guang and H. Zhiyong, "A method of human short hair modeling and real time animation," in *Pacific Graphics*, Sept. 2002.
- [49] E. Sugisaki, Y. Yu, K. Anjyo, and S. Morishima, "Simulation-based cartoon hair animation," in *Proceedings of the 13th Conference in Central Europe on Computer Graphics, Visualization and Computer Vision*, 2005.
- [50] H. D. Taskiran and U. Gudukbay, "Physically-based simulation of hair strips in real-time," in *Proceedings of the 13th Conference in Central Europe on Computer Graphics, Visualization and Computer Vision*, 2005.
- [51] Y. Watanabe and Y. Suenaga, "A trigonal prism-based method for hair image generation," *IEEE Computer Graphics and Applications*, vol. 12, no. 1, pp. 47–53, Jan 1992.
- [52] E. Plante, M.-P. Cani, and P. Poulin, "Capturing the complexity of hair motion," *Graphical Models (Academic press)*, vol. 64, no. 1, pp. 40–58, January 2002, submitted Nov. 2001, accepted, June 2002. [Online]. Available: <http://www-imagis.imag.fr/Publications/2002/PCP02>
- [53] K. Ward, M. C. Lin, J. Lee, S. Fisher, and D. Macri, "Modeling hair using level-of-detail representations," in *International Conference on Computer Animation and Social Agents*, May 2003, pp. 41–47. [Online]. Available: <http://gamma.cs.unc.edu/HSL0D>
- [54] 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.
- [55] I. Neulander and M. van de Panne, "Rendering generalized cylinders with paintstrokes," in *Graphics Interface*, 1998.
- [56] W. Kong and M. Nakajima, "Visible volume buffer for efficient hair expression and shadow generation," in *Computer Animation*. IEEE, 1999, pp. 58–65.
- [57] D. P. Mitchell, "Consequences of stratified sampling in graphics," *ACM SIGGRAPH*, 1996.
- [58] A. M. LeBlanc, R. Turner, and D. Thalmann, "Rendering hair using pixel blending and shadow buffers," *The Journal of Visualization and Computer Animation*, vol. 2, no. 3, pp. 92–97, – 1991. [Online]. Available: [citeseer.ist.psu.edu/leblanc91rendering.html](http://citeseer.ist.psu.edu/leblanc91rendering.html)
- [59] J. Kajiya and T. Kay, "Rendering fur with three dimensional textures," in *Proceedings of ACM SIGGRAPH 89*, ser. Computer Graphics Proceedings, Annual Conference Series, 1989, pp. 271–280.
- [60] F. Neyret, "Modeling animating and rendering complex scenes using volumetric textures," *IEEE Transaction on Visualization and Computer Graphics*, vol. 4(1), Jan-Mar 1998.
- [61] R. F. Stamm, M. L. Garcia, and J. J. Fuchs, "The optical properties of human hair i. fundamental considerations and goniophotometer curves," *J. Soc. Cosmet. Chem.*, no. 28, pp. 571–600, 1977.
- [62] S. Marschner, H. W. Jensen, M. Cammarano, S. Worley, and P. Hanrahan, "Light scattering from human hair fibers," *ACM Transactions on Graphics*, vol. 22, no. 3, pp. 780–791, July 2003, proceedings of ACM SIGGRAPH 2003.
- [63] C. L. Adler, J. A. Lock, and B. R. Stone, "Rainbow scattering by a cylinder with a nearly elliptical cross section," *Applied Optics*, vol. 37, no. 9, pp. 1540–1550, 1998.
- [64] D. Marcuse, "Light scattering from elliptical fibers," *Applied Optics*, vol. 13, pp. 1903–1905, 1974.
- [65] C. M. Mount, D. B. Thiessen, and P. L. Marston, "Scattering observations for tilted transparent fibers," *Applied Optics*, vol. 37, no. 9, pp. 1534–1539, 1998.
- [66] R. A. R. Tricker, *Introduction to Meteorological Optics*. Mills & Boon, London, 1970.
- [67] H. Bustard and R. Smith, "Investigation into the scattering of light by human hair," *Applied Optics*, vol. 24, no. 30, pp. 3485–3491, 1991.
- [68] D. C. Banks, "Illumination in diverse codimensions," *Proc. of ACM SIGGRAPH*, 1994.
- [69] D. Goldman, "Fake fur rendering," in *Proceedings of ACM SIGGRAPH '97*, ser. Computer Graphics Proceedings, Annual Conference Series, 1997, pp. 127–134.
- [70] T.-Y. Kim, "Modeling, rendering, and animating human hair," Ph.D. dissertation, University of Southern California, 2002.
- [71] H. W. Jensen, J. Legakis, and J. Dorsey, "Rendering of wet material," *Rendering Techniques*, pp. 273–282, 1999. [Online]. Available: [citeseer.nj.nec.com/257481.html](http://citeseer.nj.nec.com/257481.html)
- [72] A. Bruderlin, "A method to generate wet and broken-up animal fur," in *Computer Graphics and Applications, 1999. Proceedings. Seventh Pacific Conference*, October 1999, pp. 242–249.
- [73] T. Lokovic and E. Veach, "Deep shadow maps," in *Proceedings of the 27th annual conference on Computer graphics and interactive techniques*. ACM Press/Addison-Wesley Publishing Co., 2000, pp. 385–392.
- [74] T.-Y. Kim and U. Neumann, "Opacity shadow maps," in *Rendering Techniques 2001*, ser. Springer, July 2001, pp. 177–182.
- [75] M. Koster, J. Haber, and H.-P. Seidel, "Real-time rendering of human hair using programmable graphics hardware," in *Computer Graphics International (CGI)*, June 2004, pp. 248–256.
- [76] Y. Dobashi, K. Kaneda, H. Yamashita, T. Okita, and T. Nishita, "A simple efficient method for realistic animation of clouds," in *SIGGRAPH '00: Proceedings of the 27th annual conference on Computer graphics and interactive techniques*. ACM Press/Addison-Wesley Publishing Co., 2000, pp. 19–28.
- [77] F. Bertails, C. M  nier, and M.-P. Cani, "A practical self-shadowing algorithm for interactive hair animation," in *Graphics Interface*, May 2005, graphics Interface'05. [Online]. Available: <http://www-evasion.imag.fr/Publications/2005/BMC05a>
- [78] 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.
- [79] T. Mertens, J. Kautz, P. Bekaert, and F. V. Reeth, "A self-shadow algorithm for dynamic hair using density clustering," in *Proceedings of Eurographics Symposium on Rendering*, 2004.
- [80] C. Zeller, R. Fernando, M. Wloka, and M. Harris, "Programming graphics hardware," in *Eurographics - Tutorials*, September 2004.

# State of the Art in Hair Simulation

Nadia Magnenat-Thalmann

Sunil Hadap

Prem Kalra \*

MIRALab, CUI, University of Geneva  
24, rue du General Dufour, CH-1211 Geneva, Switzerland  
email: {thalmann,sunil}@cui.unige.ch

## Abstract

In this paper we summarize the technological advances in hair simulation for computer graphics. There are mainly three tasks in hair simulation - Hair Shape Modeling, Hair Dynamics and Hair Rendering. Various models developed for these tasks, fall mainly in the categories of *particle systems*, *explicit hair models*, *cluster hair models* and models based on *volumetric textures*. We discuss advantages and disadvantages of each of these approaches. We also introduce a new hair shape modeling paradigm based on fluid flow. The proposed method provides a sound basis for modeling hair-body and hair-hair interaction.

**Keywords:** hair shape modeling, hair animation, hair rendering, hypertexture

## 1 Introduction

One of the many challenges in simulating believable virtual humans has been to produce realistic looking hair. The virtual humans, two decades ago, were given polygonal hair structure. Today, this is not acceptable. Realistic visual depiction of virtual humans has improved over the years. Attention has been given to all the details necessary for producing visually convincing virtual humans and many improvements have been done to this effect.

On a scalp, human hair are typically 100,000 to 150,000 in number. Geometrically they are long thin curved cylinders having varying thickness. The strands of hair can have any degree of waviness from straight to curly. The hair color can change from white to grey, red to brown, due to the pigmentation, and have shininess. Thus, difficulties of simulating hair stem from the huge number and geometric intricacies of individual hair, complex interaction of light and shadow among the hairs, the small scale of thickness of one hair compared to the rendered image and intriguing hair to hair interaction while in motion. One can conceive three main aspects in hair simulation - hair shape modeling, hair dynamics or animation, and hair rendering. Often these aspects are interconnected while processing hairs. Hair shape modeling deals with exact or fake creation of thousands of individual hair - their geometry, density, distribution, and orienta-

tion. Dynamics of hair addresses hair movement, their collision with other objects particularly relevant for long hair, and self-collision of hair. The rendering of hair involves dealing with hair color, shadow, specular highlights, varying degree of transparency and anti-aliasing. Each of the aspects is a topic of research.

Many research efforts have been done in hair simulation research, some dealing only with one of the aspects of simulation -shape modeling, dynamics or rendering. Several research efforts were inspired by the general problem of simulation of natural phenomena such as grass, and trees. These addressed a more limited problem of simulating of fur or short hair. We divide hair simulation models into four categories depending upon the underlying technique involved: *particle systems*, *volumetric textures*, *explicit hair models* and *cluster hair model*. We discuss models presented by researchers in each of these model categories and state their contribution to the three aspects of hair simulation, i.e. hair shape modeling, hair dynamics and hair rendering. We also introduce a new hair shape modeling paradigm based on fluid flow.

The paper is organized as follows. First we give the state of the art in hair shape modeling. The hair shape modeling research in each category of the simulation models is presented. Models for hair dynamics are briefly described in Section 3. Section 4 presents the problem of hair rendering and the various solutions proposed by different people. Finally, we summarize the effectiveness and limitations of models in the four categories related to each aspect of hair simulation in the form of a table. Some future avenues for research in hair simulation are also outlined.

## 2 Hair Shape Modeling

Intricate hairstyle is indeed a consequence of physical properties of an individual hair and complex hair-hair and hair-body interactions. As we will see in the next section, modeling complex hair dynamics, that too at interactive speeds, is currently impractical. For the reasons, it would be worthwhile to treat *hair shape modeling* as a separate problem and use some heuristic approach.

Early attempts of styling long hair were based on *explicit hair models*. In the explicit hair model, each hair strand is considered for the shape and the dynamics. Daldegan

\*Visiting from Department of Computer Science and Engineering, Indian Institute of Technology, Delhi, India. pkalra@cse.iitd.ernet.in



Figure 1: Hairstyling by defining a few curves in 3D

*et al* [5] proposed that the user could interactively define a few characteristic hair strands in 3D and then populate the hair style based on them. The user is provided with a flexible graphical user interface to sketch a curve in 3D around the scalp. A few parameters such as density, spread, jitter and orientation control the process that duplicates the characteristic hairs to form a hair style. Figure 1 illustrates the method of defining few characteristic curves and resulting hairstyles from the method. Similarly, even for the fur modeling, Daldegan *et al* [4], Gelder *et al* [8] and Bruderlin *et al* [1] took similar explicit hair modeling approach. Figure 12 illustrates a furry coat modeled by the explicit hair model.



Figure 2: Cluster Hair Model, by Yan *et al*

The explicit hair models are very intuitive and close to reality. Unfortunately, they are tedious for hairstyling. Typically, it takes 5-10 hours to model a complex hair style, as in figure 1, using the method in [5]. They are also numerically expensive for hair dynamics. These difficulties are partially overcome by considering a bunch of hair instead of

individual hair in the case of the wisp/cluster models. This assumption is quite valid as in reality. Due to effects of adhesive/cohesive forces, hairs tend to form clumps. Watanabe introduced the wisp modeled in [24, 25]. Yan *et al* [26] modeled the wisps as *generalized cylinders*, see figure 2. One of the contributions of the work was also in rendering of hair using the blend of ray-tracing generalised cylinders and the *volumetric textures*. The wisp model is also evident in [2]. Surprisingly, till now, the wisp models are only limited to static hair shape modeling and we feel that it offers an interesting research possibility of modeling hair dynamics, efficiently. It would be interesting to model, how hair leave one wisp and join the other under dynamics.

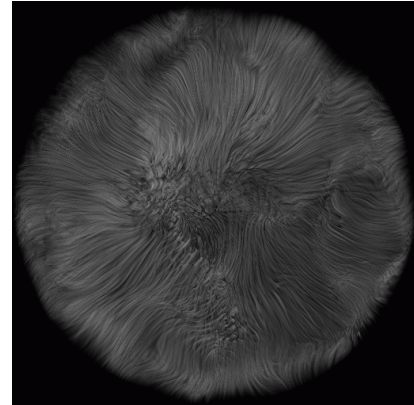


Figure 3: Fur as a Volumetric Texture, by Perlin *et al*

Nature exhibits some interesting fuzzy objects such as clouds, fire, eroded rocks and fur for which it is hard to have explicit geometric definition. Using the volumetric texture approach, fur can be modeled as a volumetric density function. Perlin *et al* [18] introduced *hypertextures*, which can model fur, see figure 3. Here, fur is modeled as intricate density variations in a 3D space, which gives an illusion of the fur like medium without defining geometry of each and every fiber. The model is essentially an extension to procedural solid texture synthesis evaluated through out the region, instead of only in the surface. They demonstrated that, combinations of simple analytical functions could define furry ball or furry donut. They further used 3D vector valued noise and turbulence to perturb the 3D texture space. This gave the natural looks to the otherwise even fur defined by the hypertexture. A good discussion on the procedural approach to modeling volumetric texture and fur in particular is in [7]. Hypertexture method by Perlin *et al* is only limited to geometries that can be analytically defined. Kajiya *et al* [12] extended this approach to have hypertextures tiled on to complex geometry. They demonstrated this by modeling a furry bear, see figure 10. They used a single solid texture tile namely texel and mapped it repeatedly on the bear's geometry. The texels automatically orient in the direction away from the surface and thus one has fuzzy volumetric density variation around the bear, which is the fur.

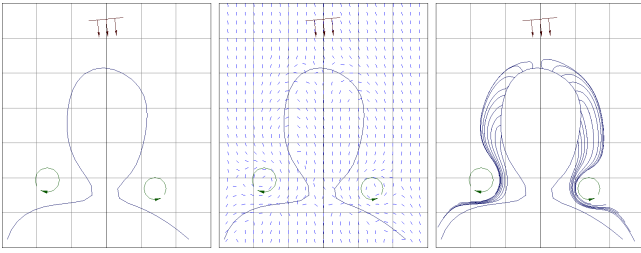


Figure 4: Hair as streamlines of a fluid flow

As evident from previous discussions, one of the strengths of the explicit hair models is their intuitiveness and ability to control the global shape of the hair. On the contrary, volumetric textures give a nice way of interpreting complexity in nature and they are rich in details. We notice that the fluid flow has both the characteristics, which we would like to exploit for hair shape modeling. We model hair shape as streamlines of a fluid flow. For complete details of the method, we refer to [10]. We choose the flow to be an ideal flow. User can setup few ideal flow elements around the body geometry to design a hairstyle, as shown in figure 2. The hair-body interaction is modeled using *source panel method* and hair-hair interaction is handled by the continuum property of fluid. Thus user can design complex hairstyles without worrying about hair-body and hair-hair interaction. Hairstyles in figure 5 and 6 are the examples of modeling hair as a fluid flow.



Figure 5: Hair as a fluid flow



Figure 6: Adding overall volumetric perturbations to the fluid flow

### 3 Hair Dynamics

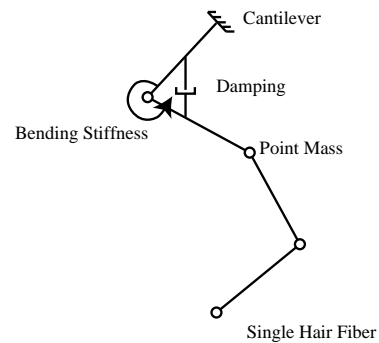


Figure 7: Simple mass-spring system for an individual hair dynamics

Anjyo *et al* [11], Rosenblum *et al* [22] and Kurihara *et al* [23] developed dynamic models that are essentially based on individual hair. An individual hair is modeled as connected rigid segments having bending stiffness at each joint. Then the individual hair is solved for the movement due to the inertial forces and the collision with the body. Though the cantilever dynamics and collision avoidance with the body of each hair is within the scope of current computing power, modeling complex hair-to-hair interaction is still a challenge. Figure 8 illustrates the effectiveness of the dynamic model even though no hair-hair interaction is considered.





Figure 8: Hair animation using the explicit model, by Kurihara *et al*

In the case of fur, which is mostly modeled as volumetric texture, one cannot take the explicit model approach for the animation. In this case, a time varying volume density function can facilitate animation of fur. One can simulate effects of turbulent air on the fur using stochastic space perturbation such as turbulence, noise, Brownian motion etc. Apart from Lewis [15] and Perlin [17, 18], work by Dischler [6] gave a generalized method for these animated shape perturbations.

## 4 Hair Rendering

In the field of virtual humans, hair presents one of the most challenging rendering problems. The difficulties arise from various reasons: large number of hair, detailed geometry of individual hair and complex interaction of light and shadow among the hairs and their small thickness. The rendering of hair often suffers from the aliasing problem due to many individual hairs reflecting light and casting shadows on each other contribute to the shading of each pixel. Further, concerning display of hairs, we see not only individual hairs but also a continuous image consisting of regions of hair color, shadow, specular highlights, varying degree of transparency and haloing under backlight conditions. The image, in spite of the structural complexity, shows a definite pattern and texture in its aggregate form.

In the last decade, the hair-rendering problem has been addressed by a number of researchers, in some cases with considerable success. However, most cases work well in particular conditions and offer limited (or none) capabilities in terms of dynamics or animation of hair. Much of the work refers to a more limited problem of rendering fur, which also has a lot in common with rendering natural phenomena such as grass and trees. As follows we give the related work in hair rendering focusing their salient features and limitations.

Particle systems introduced by Reeves *et al* [19], primarily meant to model class of fuzzy objects such as fire. Despite particles small size -smaller than even a pixel- the particle manifests itself by the way it reflects light, casts shadows, and occludes objects. Thus, the subpixel structure of the particle needs to be represented only by a model that can



Figure 9: Hair as Connected Particle System, “The End” by Alias—Wavefront

represent these properties. A particle system is rendered by painting each particle in succession onto the frame buffer, computing its contribution to the pixel and compositing it to get the final color at the pixel. The technique has been successfully used for rendering these fuzzy objects and integrated in many commercial animation systems. Figure 9 is an example of how one can use connected particle systems for the modeling of hair. However, the technique has some limitations for shadowing and self-shadowing. Much of it is due to the inherent modeling using particle systems: simple stochastic models are not adequate to represent the type of order and orientation of hair. Also, it requires appropriate lighting model to capture and control the hair length and orientation. The specular highlights in particular owing to the geometry of the individual strands are highly anisotropic.

Impressive results have been obtained for the more limited problem of rendering fur, which can be considered as very short hair. As we have already discussed in the case of hair shape modeling, Perlin *et al* [18] introduced hypertextures that can model fur like objects. Hypertexture approach remains limited to geometries that can be defined analytically. Kajiya and Kay extended this approach to use it on the complex geometries. They used a single solid texture tile namely texel. The idea of texels was inspired by the notion of volume density used in [18]. A texel is a 3D texture map where both the surface frame and lighting model parameters are embedded over a volume. Texels are a type of model intermediate between a texture and a geometry. A texel is however, not tied to the geometry of any particular surface and thus makes the rendering time independent of the geometric complexity of the surface that it extracts. The results are demonstrated by rendering a teddy bear (figure 10). Texels are rendered using ray casting, in a manner similar to that for volume densities using a suitable illumination model. Kajiya *et al* discusses more about the particular fur illumination model and a general rendering method for rendering volume densities. The rendering of volume densities are also covered in great detail in the book by Eber *et al* [7].



Figure 10: Volumetric Texture rendering by Kajiya *et al*

In another approach by Goldman [9], emphasis is given on rendering visual characteristics of fur in cases where the hair geometry is not visible at the final image resolution -object being far away from the camera. A probabilistic rendering algorithm, also referred to as fakefur algorithm is proposed. In this model, the reflected light from individual hairs and from the skin below is blended using the expectations of a ray striking a hair in that area as the opacity factor.

Though the volumetric textures are quite suitable for rendering furry objects or hair patches, rendering of long hair using this approach does not seem obvious.

A brute force method to render hair is to model each individual hair as curved cylinder and render each cylinder primitive. The sheer number of primitives modeling hair poses serious problem to this approach. However, the explicit modeling of hair has been used for different reasons employing different types of primitives.

An early effort by Csuri *et al* [3] generated fur-like volumes using polygons. Each hair was modeled as a single triangle laid out on a surface and rendered using a Z-buffer algorithm for hidden surface removal. Miller [16] produced better results by modeling hair as pyramids consisting of triangles. Oversampling was employed for anti-aliasing. These techniques however, impose serious problems considering reasonable number and size of hairs.

In an another approach, a hardware Z-buffer renderer was used with Gouraud shading for rendering hair modeled as connected segments of triangular prisms on a full human head. However, the illumination model used was quite simplistic and no effort was done to deal with the problem of aliasing. LeBlanc *et al* [14] proposed an approach of rendering hair using pixel blending and shadow buffers. This technique has been one of the most effective and practical hair rendering approach. Though it could be applied for the variety of hairy and furry objects, one of the primary intention of the approach was to be able to render realistic different styles of human hairs. Hair rendering is done by mix of ray tracing and drawing polyline primitives, with added module for



Figure 11: Rendering pipeline of the method-”Pixel Blending and Shadow Buffer”

the shadow buffer [20]. The rendering pipeline has the following steps: first the shadow of the scene is calculated for each light source. Then, hair shadow buffer is computed for each light source for the given hair style model; this is done by drawing each hair segment into a Z-buffer and extracting the depth map. The depth maps for the shadow buffers for the scene and hair are composed giving a single composite shadow buffer for each light source. The scene image with its Z-buffer is generated using scene model and composite shadow buffers. The hair segments are then drawn as illuminated polylines [27] into the scene using Z-buffer of scene for determining the visibility and the composite shadow buffers for finding the shadows. Figure 11 shows the process and Figure 12 gives final rendered image of a hairstyle of a synthetic actor with a fur coat.



Figure 12: Fur using Explicit Hair Model

Special effects like rendering wet hair require change in the shading model. Bruderlin [1] presented some simple ways to account for the wetness of hair -changing the specularly. That is, hairs on the side of a clump facing the light are brighter than hairs on a clump away from the light.

Kong and Nakajima *et al* [13] presented an approach of using visible volume buffer to reduce the rendering time. The volume buffer is a 3D cubical space defined by the user depending upon the available memory and the resolution re-

quired. They consider hair model as combination coarse background hair and detailed surface hair determined by the distance from the viewpoint or the opacity value. The technique reduces considerably the rendering time, however, the quality of results is not so impressive.



Figure 13: Braid rendered using generalized cylinders and volumetric texture, by Yan *et al*

Yan *et al* [26] combine volumetric texture inside the explicit geometry of hair cluster defined as a generalized cylinder. Ray tracing is employed to get the boundaries of the generalized cylinder and then the standard volume rendering is applied along the ray to capture the characteristics of the density function defined. This may be considered as a hybrid approach for hair rendering.

## 5 Conclusion

	Hair Modeling	Hair Animation	Hair Rendering
Explicit Models	effective - tedious to model - not suitable for knots and braids	adequate - expensive due to size - inappropriate for hair-hair interaction	fast - inadequate for self-shadowing
Particle Systems	inappropriate	ad hoc - lacks physical basis - no hair-hair interaction	effective - lacks shadowing and self-shadowing
Volumetric Textures	effective - not suitable for long hair	limited - via Animated Shape Perturbation	effective - expensive
Cluster Model	effective - not suitable for simple smooth hair	not done - via Animated Shape Perturbation	effective
Hair as a Fluid	effective - not suitable for knots and braids	not done	not done

Figure 14: Comparison of the various hair models

In this paper we present the state of the art in hair simulation, one of the most challenging problem of virtual hu-

mans. We consider three aspects in hair simulation: hair shape modeling, hair rendering and hair dynamics. Different approaches have been proposed in the literature dealing with one or more aspects of hair simulation. We divide them into four categories based on the underlying technique: particle systems, volumetric textures, explicit hair models and cluster hair model. Some of these techniques are appropriate and effective only for one of the aspects in hair simulation. In figure 14, we summarize their role with their effectiveness and limitations for each aspect of the hair simulation. Notice that we have introduced a new hair modeling paradigm - “Hair as a Fluid”. We believe, this approach has good potential in terms of hair shape modeling and hair dynamics, as the methodology gives a basis for modeling complex hair-hair interactions.

No, doubt research in hair simulation despite the inherent difficulty of its size has been encouraging and shown remarkable improvements over the years. People in general are not ready to accept a bald digital actor or an animal without fur. Such realism to computer graphics characters is also becoming more widely available to the animators. Many of the commercial software provide suitable solutions and plug ins for creating hairy and furry characters. An article by Robertson [21] gives an overview of various techniques available for animators.

However, the quest of realism increases after noticing what one can already achieve. This asks to continue our research for better solutions. Hair dynamics for instance remains an area, where existing computing resources impose constraints. It is still very far to imagine real time hair blowing with full rendering and collisions. Hair dressing and styling also require flexible and convenient modeling paradigms. Fast and effective rendering methods for all hair styles -short or long, in all conditions -dry or wet, modeling all the optical properties of hair are still to be explored. So there is still long way to go.

## 6 Acknowledgements

This work is supported by the Swiss National Research Foundation (FNRS).

## References

- [1] BRUDERLIN, A. A method to generate wet and broken-up animal fur. *Pacific Graphics '99* (October 1999). Held in Seoul, Korea.
- [2] CHEN, L.-H., SAEYOR, S., DOHI, H., AND ISHIZUKA, M. A system of 3d hair style synthesis based on the wisp model. *The Visual Computer* 15, 4 (1999), 159–170. Springer-Verlag, ISSN 0178-2789.

- [3] CSURI, C., HAKATHORN, R., AND PARENT, R. Towards an interactive high visual complexity animation system. In *Computer Graphics* (1979).
- [4] DALDEGAN, A., AND MAGNENAT-THALMANN, N. Creating virtual fur and hair styles for synthetic actors. In *Communicating with Virtual Worlds* (1993), N. Magnenat-Thalmann and D. Thalmann, Eds., Springer-Verlag.
- [5] DALDEGAN, A., THALMANN, N. M., KURIHARA, T., AND THALMANN, D. An integrated system for modeling, animating and rendering hair. *Computer Graphics Forum (Eurographics '93)* 12, 3 (1993), 211–221. Held in Oxford, UK.
- [6] DISCHLER, J.-M. A general model of animated shape perturbation. *Graphics Interface '99* (June 1999), 140–147. ISBN 1-55860-632-7.
- [7] EBERT, D. S., MUSGRAVE, F. K., PEACHEY, D., PERLIN, K., AND WORLEY, S. *Texturing and Modeling*. Academic Press, 1998.
- [8] GELDER, A. V., AND WILHELMS, J. An interactive fur modeling technique. *Graphics Interface '97* (May 1997), 181–188. ISBN 0-9695338-6-1 ISSN 0713-5424.
- [9] GOLDMAN, D. B. Fake fur rendering. *Proceedings of SIGGRAPH 97* (August 1997), 127–134. ISBN 0-89791-896-7. Held in Los Angeles, California.
- [10] HADAP, S., AND MAGNENAT-THALMANN, N. Interactive hair styler based on fluid flow. In *Proceedings of Eurographics Workshop on Computer Animation and Simulation '2000* (2000). to appear.
- [11] ICHI ANJYO, K., USAMI, Y., AND KURIHARA, T. A simple method for extracting the natural beauty of hair. *Computer Graphics (Proceedings of SIGGRAPH 92)* 26, 2 (July 1992), 111–120. ISBN 0-201-51585-7. Held in Chicago, Illinois.
- [12] KAJIYA, J. T., AND KAY, T. L. Rendering fur with three dimensional textures. *Computer Graphics (Proceedings of SIGGRAPH 89)* 23, 3 (July 1989), 271–280. Held in Boston, Massachusetts.
- [13] KONG, W., AND NAKAJIMA, M. Visible volume buffer for efficient hair expression and shadow generation. *Computer Animation '99, IEEE Computer Society* (May 1999). IEEE Press, Held in Geneva, Switzerland.
- [14] LEBLANC, A., TURNER, R., AND THALMANN, D. Rendering hair using pixel blending and shadow buffer. *Journal of Visualization and Computer Animation* 2 (1991), 92–97. John Wiley.
- [15] LEWIS, J.-P. Algorithms for solid noise synthesis. *Computer Graphics (Proceedings of SIGGRAPH 89)* 23, 3 (July 1989), 263–270. Held in Boston, Massachusetts.
- [16] MILLER, G. S. P. From wire-frames to furry animals. *Graphics Interface '88* (June 1988), 138–145.
- [17] PERLIN, K. An image synthesizer. *Computer Graphics (Proceedings of SIGGRAPH 85)* 19, 3 (July 1985), 287–296. Held in San Francisco, California.
- [18] PERLIN, K., AND HOFFERT, E. M. Hypertexture. *Computer Graphics (Proceedings of SIGGRAPH 89)* 23, 3 (July 1989), 253–262. Held in Boston, Massachusetts.
- [19] REEVES, W. T. Particle systems - a technique for modeling a class of fuzzy objects. *ACM Transactions on Graphics* 2, 2 (April 1983), 91–108. Held in USA.
- [20] REEVES, W. T., SALESIN, D. H., AND COOK, R. L. Rendering antialiased shadows with depth maps. *Computer Graphics (Proceedings of SIGGRAPH 87)* 21, 4 (July 1987), 283–291. Held in Anaheim, California.
- [21] ROBERTSON, B. Hair-raising effects. *Computer Graphics World, Magazine* (October 1995).
- [22] ROSENBLUM, R., CARLSON, W., AND TRIPP, E. Simulating the structure and dynamics of human hair: Modeling, rendering and animation. *Journal of Visualization and Computer Animation* 2 (June 1991), 141–148. John Wiley.
- [23] TSUNEYA KURIHARA, KEN-ICHI ANJYO, D. T. *Models and Techniques in Computer Animation*. Springer-Verlag, ch. Hair Animation with Collision Detection.
- [24] WATANABE, Y., AND SUENAGA, Y. Drawing human hair using wisp model. In *Proceedings of Computer Graphics International '89* (1989), Springer Verlag, pp. 691–700.
- [25] WATANABE, Y., AND SUENAGA, Y. A trigonal prism-based method for hair image generation. *IEEE Computer Graphics & Applications* 12, 1 (January 1992), 47–53.
- [26] YAN, X. D., XU, Z., YANG, J., AND WANG, T. The cluster hair model. *Journal of Graphics Models and Image Processing* (1999). Academic Press.
- [27] ZCKLER, M., STALLING, D., AND HEGE, H.-C. Interactive visualization of 3d-vector fields using illuminated streamlines. *IEEE Visualization '96* (October 1996), 107–114. ISBN 0-89791-864-9.



# Animating Complex Hairstyles in Real-Time

Pascal Volino  
MIRALab, Univ. of Geneva  
CH-1211, Geneva, Switzerland  
+41 22 379 10 76  
pascal@miralab.unige.ch

Nadia Magnenat-Thalmann  
MIRALab, Univ. of Geneva  
CH-1211, Geneva, Switzerland  
+41 22 379 77 69  
thalmann@miralab.unige.ch

## ABSTRACT

True real-time animation of complex hairstyles on animated characters is the goal of this work, and the challenge is to build a mechanical model of the hairstyle which is sufficiently fast for real-time performance while preserving the particular behavior of the hair medium and maintaining sufficient versatility for simulating any kind of complex hairstyles.

Rather than building a complex mechanical model directly related to the structure of the hair strands, we take advantage of a volume free-form deformation scheme. We detail the construction of an efficient lattice mechanical deformation model which represents the volume behavior of the hair strands. The lattice is deformed as a particle system using state-of-the-art numerical methods, and animates the hairs using quadratic B-Spline interpolation. The hairstyle reacts to the body skin through collisions with a metaball-based approximation. The model is highly scalable and allows hairstyles of any complexity to be simulated in any rendering context with the appropriate tradeoff between accuracy and computation speed, fitting the need of Level-of-Detail optimization schemes.

**Categories and Subject Descriptors:** I.3.7 [Three-Dimensional Graphics and Realism]: Animation, Virtual Reality; I.6.3 [Simulation and Modeling]: Applications. **General Terms:** Algorithms. **Keywords:** Real-time animation, mechanical simulation, hair modeling, virtual characters.

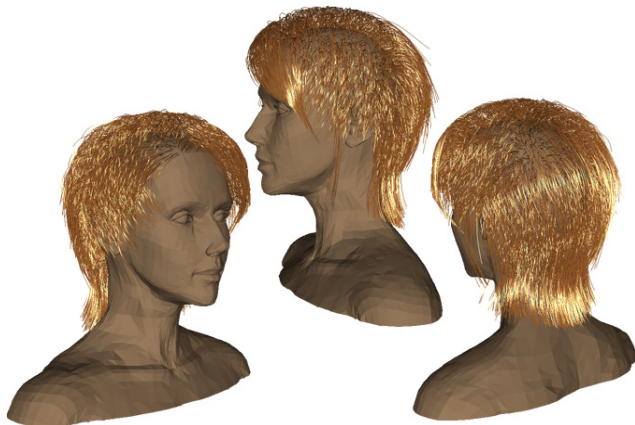


Figure 1: Our system animates hairstyles of large complexity.

## 1. INTRODUCTION

Modeling and rendering hair on virtual characters always remains a challenge. Its nature of intricate arrangement of a huge number of strands (often more than 100 000) provides to the medium a

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.

VRST'04, November 10-12, 2004, Hong Kong.  
Copyright 2004 ACM 1-58113-907-1/04/0011...\$5.00.

complex behavior that depends not only on the elastic properties of the strands, but also on the way they interact through friction. This behavior is highly dependent on the type of hairstyle (straight or curly, long or short) and condition (stiff or soft, dense or sparse, dry or wet...). Techniques may consider any compromises between simulating individually each strand and simulating a volume medium representing the complete hairstyle.

### 1.1. Overview of Hair Simulation Techniques

Hair modeling has started along the emergence of computer-generated virtual characters. The first models were explicit, and considered the modeling and the rendering of each individual strand. These are illustrated by the contributions of Daldegan et al [9] for modeling and rendering, Rosenblum et al [31], Kurihara and Anjyo et al [1] for strand-based animation.

However, the explicit model had serious limitations resulting from the large amount of hair strands to be considered, resulting in large computation times unsuitable for real-time applications. Reducing this complexity and the number of degrees of freedom of the model is essential for fast computation. The first idea is to consider that hairs of neighboring locations share similar shapes and mechanical behaviors. In many models, such hairs are grouped into *wisps*, sometimes also called *clusters*. This eases the work of the hair designer as well as the computation required for hair animation. This technique was defined by Watanabe et al [34], and has been frequently used since with many variations. Examples are the guide hairs with collisions and interpolation of Chang et al [6], the multilayer approach with cluster collisions of Plante et al [29], and some variations from Chen et al [7] and Yang et al [35]. Another evolution is to replace the hairs by approximate surfaces (*strips* or *patches*), as done by Koh et al [23] [24], or even volumes that can easily be modeled as polygonal meshes, such as the thin shell approach of Kim et al [21]. Animation is done by specific mechanical models associated to these representations, which also include specific methods for handling collisions efficiently as described by Lee et al [27].

Combining various approaches can lead to Level-of-Detail methods where the representation is chosen depending on the current requirement of accuracy and available computational resources. A good example developed by Bertails et al [4] is based on wisps tree, and a similar approach from Kim et al [22] is based on multiresolution clusters. Advanced Level-of-Detail methods also include combination of strands, clusters and strips modeled with subdivision schemes and animated using advanced collision techniques, as developed by Ward et al [32] [33].

In another kind of approach, volume hair models construct a representation of the hair based on a vector fields representing the orientation of the hair strands. These approaches have been exploited for hair design by Hadap et al [16], like a similar model from Yu [36]. A major difficulty of volume hair models is the difficulty to connect them to the actual mechanics of hair strands. An attempt based on explicit strand mechanics has been carried by Hadap et al [18], but the resulting computation times were incompatible with interactive applications.

In the specific context of real-time applications, cluster and strip models are good candidates. Fast strip or guide hair animation can be obtained through simplified articulated body mechanics, such

as the rigid body animation methods described by Featherstone [12]. However, they still suffer from various problems, such as the necessity of collision detection between hairs, and their inability to represent efficiently some mechanical properties, such as bending stiffness. Specific approaches are oriented toward real-time applications, such as the Loosely Connected Particles approach described by Bando et al [2] and the real-time short hair model from Guang et al [15], but they suffer from scalability problems and hairstyle design constraints. Among the major issues, the specific mechanical models they rely on are only suitable for specific hairstyles, which are typically long and soft straight hair (possibly deformed with some geometrical buckling). We intend to overcome these difficulties through the design of a new hair animation model based on volume deformations, with the aim of animating any complex hairstyle design in real-time. Our animation system should also be compatible with most approaches used for hairstyle representation and real-time rendering, and offer the designer the possibility of creating any kind of hairstyle that can robustly be simulated in any animation context.

## 1.2. Our Free-Form Deformation Approach

Our idea is to use *Free-Form Deformations* (FFD) to animate the hair in real-time. Free-Form deformations are widely used for shape design and deformation [13] and recent developments also include mechanical deformations [10].

### Principles of the method

Our method works as follows (Figure 2): During preprocessing, a lattice is defined around the head and the initial hairstyle. A mechanical model is then constructed to provide the lattice with a mechanical behavior related to the hair contained in it. This mechanical model is based on a particle system constructed on the lattice nodes. A particular kind of particle interaction has been developed to represent the mechanical behavior of a hair strand on the lattice. This model can be decimated at will to obtain the best compromise between accuracy and computation speed. Collisions act directly on the lattice nodes through repulsion forces from a simple metaball representation of the head and shoulders.

During computation, the lattice is animated using state-of-the-art simulation methods for integrating the mechanical behavior of particle systems. The current positions of the hair strands or any other rendered features are finally computed at rendering time using linear or quadratic B-Spline interpolation from the current position of the lattice nodes.

### Benefits

At the first glance, it seems pretty unnatural to map hairstyle deformation on a simple cubic lattice rather than using an interpolation scheme more fitted to the actual topology of the hairstyle (for instance, interpolating between a reduced set of animated guide hairs). However, this method allows to take advantage of several benefits:

- The cubic lattice defines a true volume around the skull, and any location inside this volume can be computed using very simple interpolation formula (fast interpolation of any feature during animation).

- Computing lattice coordinates of any location is also fast and straightforward (for use in interactive hairstyle editing).

Decoupling the topology of the animated interpolation structure from the actual hair topology also have good benefits:

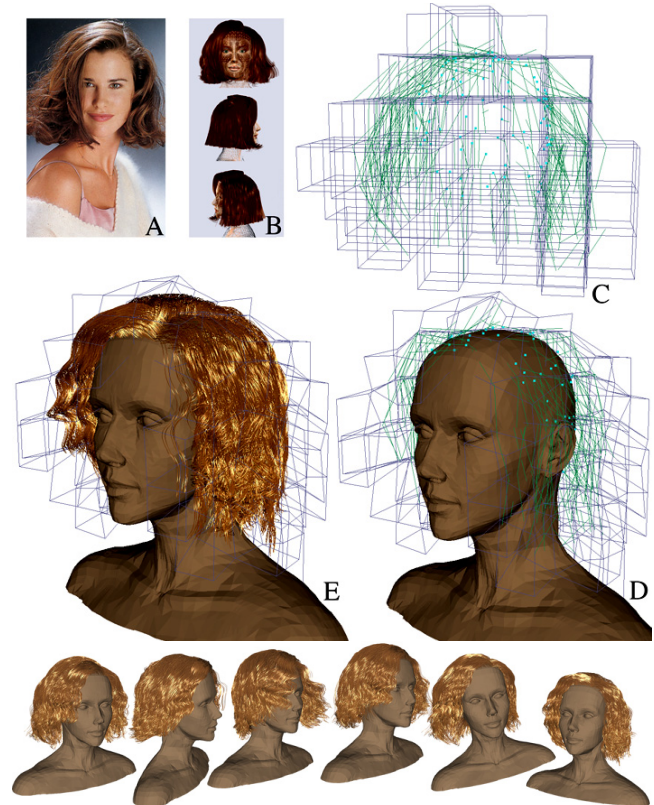
- *Limited reliance on the strand nature of the hair*, which is only used during preprocessing for constructing the mechanical model representing hair mechanics. The resulting Free-Form interpolation can be directly applied on strands as well as on clusters, strips and even volume representations of the hair, giving a highly versatile model suitable for all rendering contexts.

- *Total freedom for hairstyle design*, as hairs can be long or short, curled or straight, and even contain specific features such as ponytails without any particular adaptation of the model. This is

quite impossible with current wisp-based animation systems that require wisp shapes to be in a mechanically consistent state (usually straight) only influenced by collisions.

- *Good scalability*, by allowing complete control of the complexity of the mechanical model (lattice size and number of interactions) independently from the complexity of the initial hairstyle. Realism of the hair animation is maintained to an according extent by the mechanical interactions in the lattice that are constructed to model the mechanical behavior of hair strands.

Using our system, the hairstyle can be designed in its natural equilibrium state on the head using any hairstyle creation software, which is much easier to manage for the designer. Whatever the input, the lattice and its mechanical model are automatically constructed at any resolution suitable for the performance requirements.



**Figure 2: Workflow of the hair animation process: The hairstyle is designed (B) from a model (A). A lattice is then build from the hair strands with a simplified mechanical model (C). During animation, the lattice is mechanically deformed (D) and hair features are interpolated from the lattice (E) for real-time animation and rendering.**

Section 2 describes the principles of our simulation scheme, whereas Section 3 details the mechanical model and Section 4 the geometric Free-Form interpolation method. Results and further considerations are detailed in Section 5.

## 2. LATTICE-BASED FFD ANIMATION

In a typical real-time animation system, the body animation is usually carried out by animating a skeleton through various high-level methods, such as prerecorded animation, real-time tracking or smart autonomy algorithms. A transformation matrix is associated to each skeleton component, and expresses its current position in world coordinates. The body skin is deformed using these matrices. Interpolation methods may eventually deform the skin around the joints. We focus our attention on the motion of the head, described at any time by a transformation matrix  $\mathbf{R}$  and speed matrix  $\mathbf{R}'$  (element-wise derivative of  $\mathbf{R}$  against time).

## 2.1. The Lattice Model

We construct a 3D lattice around the head that includes all the hair. As shown in Figure 4, this lattice is defined by the initial position vector  $\mathbf{P}^0$  containing the positions  $\mathbf{p}_i^0$  of all its nodes. During animation, the lattice is moved according to the head motion. The current rigid-motion positions  $\mathbf{R}\mathbf{p}_i^0$  and velocities  $\mathbf{R}'\mathbf{p}_i^0$  of the lattice nodes define the rigid motion pattern of the hair, in which the hair follows the head motion without any deformation (Figure 3).

In our approach, the lattice is however deformed by mechanical computation. For this, we define the current deformed position vector  $\mathbf{P}$  and velocity vector  $\mathbf{P}'$  of the lattice containing the current positions  $\mathbf{p}_i$  and speeds  $\mathbf{p}_i'$  of the lattice nodes (Figure 4). Their evolution is ruled by mechanical computation iterations that use the current values of  $\mathbf{R}\mathbf{p}_i^0$  and  $\mathbf{R}'\mathbf{p}_i^0$  as equilibrium states. This mechanical model, aimed at providing the lattice with the volumic behavior of the hair contained in it, is constructed during preprocessing.

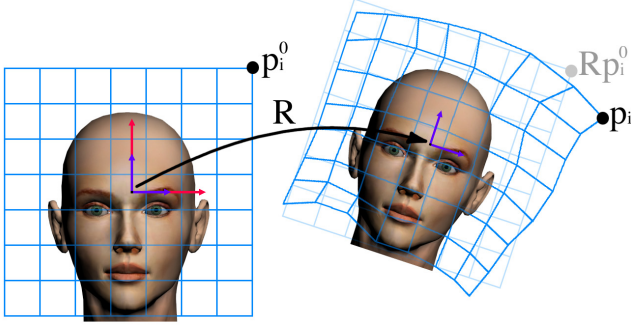


Figure 3: Rigid transformation and mechanical deformation of the lattice during animation of the head.

## 2.2. An Efficient Model using Lattice Stiffeners

Different approaches are possible for designing this mechanical lattice deformation model. However, spring-mass approaches still seem to be the best candidate for designing really fast mechanical models.

Restricting the model to springs directly linking the lattice nodes imposes force directions ruled by the lattice orientation. This is too inaccurate to precisely model the real effect of elastic hair segments that follow very precise arbitrary directions. The only way to allow a spring to represent accurately a hair segment would be to attach its extremities anywhere in the lattice volume, and not only on the lattice nodes.

A possible solution is to create additional intermediate nodes inside the lattice elements. Such approaches are usually combined to adaptive subdivision schemes. This would however complicate the FFD interpolation procedures, and the benefit of a reduced number of degrees of freedom would be lost as well.

### Lattice stiffeners

We have solved this problem by creating *lattice stiffeners*, a kind of mechanical interaction model that acts on a weighted sum of lattice nodes rather than on individual nodes only (Figure 4). They are defined by a behavior law  $\boldsymbol{\sigma}(\boldsymbol{\varepsilon}, \boldsymbol{\varepsilon}')$  that relates particle forces  $\mathbf{f}_i$  to particle positions  $\mathbf{p}_i$ , and velocities  $\mathbf{p}_i'$ , as well as a particle weights  $\mathbf{w}_i$  that relates the influence of the law on each lattice node, as follows:

$$\mathbf{f}_i = \mathbf{w}_i \boldsymbol{\sigma} \quad \text{with} \quad \boldsymbol{\varepsilon} = \sum_j \mathbf{w}_j \mathbf{p}_j \quad \text{and} \quad \boldsymbol{\varepsilon}' = \sum_j \mathbf{w}_j \mathbf{p}_j' \quad (1)$$

Lattice stiffeners are quite suited for being integrated with implicit methods, as their force derivative contribution to the Jacobian matrix of the system are simply computed as follows:

$$\frac{\partial \mathbf{f}_i}{\partial \mathbf{p}_j} = \mathbf{w}_i \mathbf{w}_j \frac{\partial \boldsymbol{\sigma}}{\partial \boldsymbol{\varepsilon}} \quad \text{and} \quad \frac{\partial \mathbf{f}_i}{\partial \mathbf{p}_j'} = \mathbf{w}_i \mathbf{w}_j \frac{\partial \boldsymbol{\sigma}}{\partial \boldsymbol{\varepsilon}'} \quad (2)$$

Weight vectors are typically constructed using the linear interpolation coefficients for defining the virtual locations on which the forces are exerted in the lattice.

Among them, we use *linear lattice springs* of which can be used to create a viscoelastic spring force (elasticity  $\mathbf{k}$  and viscosity  $\mathbf{q}$ ) between two points anywhere in the lattice volume, using a behavior law defined as follows:

$$\boldsymbol{\sigma} = (-\mathbf{k}(|\boldsymbol{\varepsilon}| - |\boldsymbol{\varepsilon}_0|) - \mathbf{q}(\bar{\boldsymbol{\varepsilon}}^T \boldsymbol{\varepsilon}')) \bar{\boldsymbol{\varepsilon}} \quad \text{with} \quad \bar{\boldsymbol{\varepsilon}} = |\boldsymbol{\varepsilon}|^{-1} \boldsymbol{\varepsilon} \quad (3)$$

$$\frac{\partial \boldsymbol{\sigma}}{\partial \boldsymbol{\varepsilon}} = -\mathbf{k} \bar{\boldsymbol{\varepsilon}} \bar{\boldsymbol{\varepsilon}}^T \quad \text{and} \quad \frac{\partial \boldsymbol{\sigma}}{\partial \boldsymbol{\varepsilon}'} = -\mathbf{q} \bar{\boldsymbol{\varepsilon}} \bar{\boldsymbol{\varepsilon}}^T$$

We also use *linear lattice attachments* that attach any point of the lattice volume to a particular position with a viscoelastic elastic force, defined as follows:

$$\boldsymbol{\sigma} = -\mathbf{k}(\boldsymbol{\varepsilon} - \boldsymbol{\varepsilon}_0) - \mathbf{q}(\boldsymbol{\varepsilon}) \quad (4)$$

$$\frac{\partial \boldsymbol{\sigma}}{\partial \boldsymbol{\varepsilon}} = -\mathbf{k} \mathbf{I} \quad \text{and} \quad \frac{\partial \boldsymbol{\sigma}}{\partial \boldsymbol{\varepsilon}'} = -\mathbf{q} \mathbf{I}$$

The rest position  $\boldsymbol{\varepsilon}_0$  of any stiffener is precomputed from the initial positions of the lattice nodes  $\mathbf{p}_i^0$  and the current skeleton transformation matrix  $\mathbf{R}$  (which has constant scale) as follows:

$$\boldsymbol{\varepsilon}_0 = \mathbf{R} \sum_j \mathbf{w}_j \mathbf{p}_j^0 \quad (5)$$

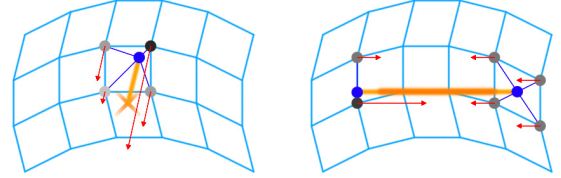


Figure 4: Lattice stiffeners: A lattice attachment (left) and a lattice spring (right), linking arbitrary locations in the lattice (blue dots) through forces acting on the lattice nodes (red arrows).

### Mechanical animation of the lattice

The nodes of the lattice are subject to the forces generated by all lattice stiffeners of the mechanical model, and the whole lattice is handled as a particle system. We integrate this system using implicit integration based on the Inverse Euler steps. Widely used in efficient mechanical cloth simulation methods [3] [19], this method allows the robust resolution of any mechanical situations with large constant time steps, including stiff models for almost rigid hairs.

In our implementation, a dynamic matrix assembly process is embedded in the Conjugate Gradient iterations [30], a solver well suited for handling iteratively large sparse systems. This allows more accuracy in the simulation by suppressing the need of approximative linearization of the model [8], and brings better simulation accuracy for large deformations along with huge quality improvements in the simulated animations.

## 3. BUILDING THE HAIR MECHANICS

Using lattice stiffeners as building blocks, the mechanical representation of the hair is a sum of several components:

- The *hair mechanical model*, which provides the lattice with a mechanical behavior similar to the hair it contains, and models the attachment of the hair to the skull as well. It is detailed in Section 3.1.

- The *ether*, which defines the rest position of the lattice through weak forces and ensures stability of the model, and may also



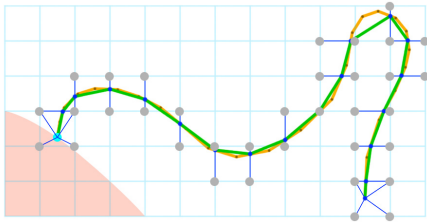
model additional external forces such as gravity and aerodynamic effects. This is detailed in Section 3.2.

- The *Collision effects* against the body, which prevents hair penetrating through the body skin. Their representation is detailed in Section 3.3.

### 3.1. The Mechanical Model of the Hair

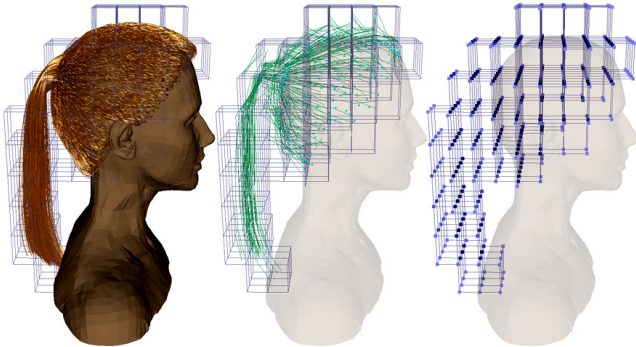
The mechanical model of the hair is defined as a sum of linear viscoelastic lattice springs relating the elasticity of each individual hair strand segment in the lattice model. Their viscoelastic parameters correspond to those of the modeled hair. The attachments of the hair extremities to the skull are modeled by stiff viscoelastic lattice attachments, which are positioned exactly at the end of each hair. The resulting mechanical model is shown in Figure 6.

In order to reduce the complexity of the model, we resample each hair during the model construction as segments defined by the intersection points of the hair line on the lattice boundaries for which the crossing angle is above  $45^\circ$ , as shown in Figure 5. This limits the number of created lattice springs whatever the discretization of the initial hair curve, as well as the number of particles involved in each lattice springs.



**Figure 5: Construction of the lattice springs (green lines) of a hair strand (yellow line) with the proposed rediscretization (blue dots) on the lattice. The lattice nodes along the line (gray dots) are affected.**

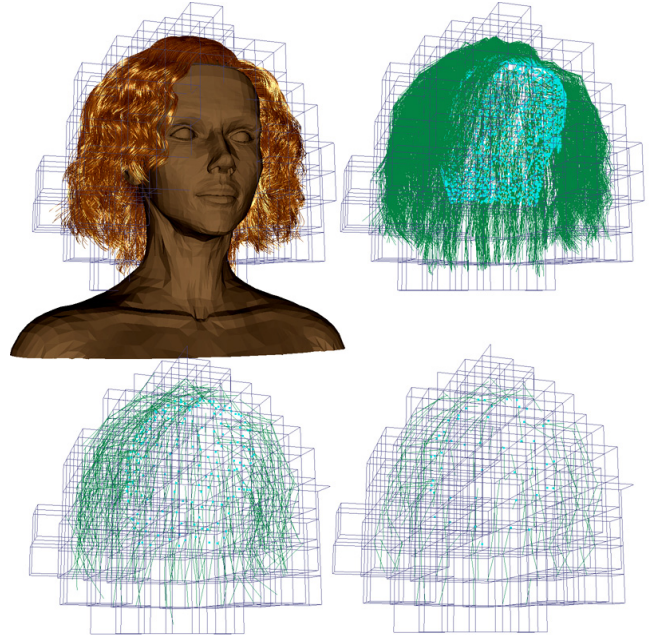
Each hair also contributes to the mass of the lattice nodes, as shown in Figure 6. The mass of a hair segment is shared on the corresponding lattice nodes according to the linear interpolation coefficients of its extremities. Nodes not involved in any hair segment are trimmed away from the simulation.



**Figure 6: From a hairstyle (left), the corresponding mechanical model is built using lattice stiffeners (center), and the mass of the hair strands is distributed on the lattice nodes (right) (darker is heavier).**

While this model makes a good approximation of the mechanical behavior of the hair on the lattice model, its computation time is still much too high because of the high number of hairs. We therefore carry out a simplification of the model by decimating the redundant lattice stiffeners (hair springs as well as skull attachments), as shown in Figure 7. This simplification is carried out by evaluating, for all couples of stiffeners sharing common vertices, the "mechanical error" resulting from merging one of the stiffeners to the other. Error and merging computations are based

dot product operations on the weight vectors, which evaluate the "synergy" between stiffeners (their "parallelism" in the weight space). The merging operations that minimize the error are carried out until the expected number of stiffeners remain. We avoid quadratic decimation search times by constructing the adequate temporary data structures.



**Figure 7: The decimation process: From the strands of the hairstyle (up-left) is constructed the initial model (up-right) containing roughly 14200 springs and 3600 attachments. The model can then be decimated at will, for example 800 springs and 200 attachments (down-left), or 200 springs and 50 attachments (down-right).**

### 3.2. The Ether Model

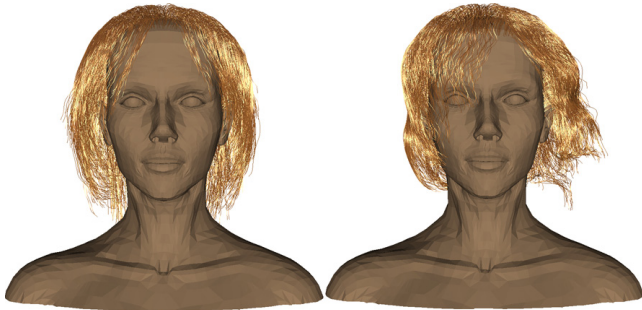
While this properly gives an adequate behavior of the lattice nodes inside the hairs, nodes surrounded by lattices empty of any hair are quite underconstrained and may exhibit erratic behavior. To limit this, we handle the whole lattice as an "ether medium" that has its own stable behavior. Besides adding to all lattice nodes an ether mass, we add viscoelastic ether forces relating each node to its rest position defined by rigid head motion. They are modeled as one-particle linear viscoelastic lattice attachments. They have a weak constant elasticity parameter which pull back each node to its equilibrium position, as well as a constant viscosity parameter for limiting oscillatory behavior.

Besides limiting erratic behavior of the lattice nodes, adjusting parameters of ether forces are a good way to control the global stiffness of the hair design (for example simulating designs with hair gel), as well as roughly simulating aerodynamic damping effects. Ether forces are also the key factors for ensuring robustness of a hair model which should return close to its initial posture after whatever mechanical or not-so-mechanical body motion.

The ether also supports additional external forces, such as aerodynamic effects. Air drag and wind is simulated by our system by assigning to each lattice node a viscous reactivity parameter representing the drag created by the hair surrounding this node. These factors are actually tensors which are precomputed by distributing the anisotropic drag contributions of each hair segment to the neighboring lattice nodes (similarly to how the hair mass is distributed). Using a global viscosity parameter  $\mathbf{q}$  (air drag per unit hair length), the contribution tensor  $\mathbf{Q}$  of a hair segment  $\mathbf{\epsilon}$ , which only creates drag perpendicularly to the segment direction, is computed as follows:

$$\mathbf{Q} = \mathbf{q} |\boldsymbol{\epsilon}| (\mathbf{I} - \bar{\boldsymbol{\epsilon}} \bar{\boldsymbol{\epsilon}}^T) \quad \text{with} \quad \bar{\boldsymbol{\epsilon}} = |\boldsymbol{\epsilon}|^{-1} \boldsymbol{\epsilon} \quad (6)$$

This aerodynamic model can successfully model the drag created by airflow with very minor impact on the computation time. For high wind situations, some additional wind speed perturbations are welcome for simulating turbulence (Figure 8).



**Figure 8: Turbulent air drag on the hairstyle.**

### 3.3. Handling Collisions

Collision detection is usually one of the most tedious computational tasks, particularly for time-critical applications such as real-time simulations. In the case of hair animation, this complexity results from the huge hair geometry as well as the complex shape of the body surface.

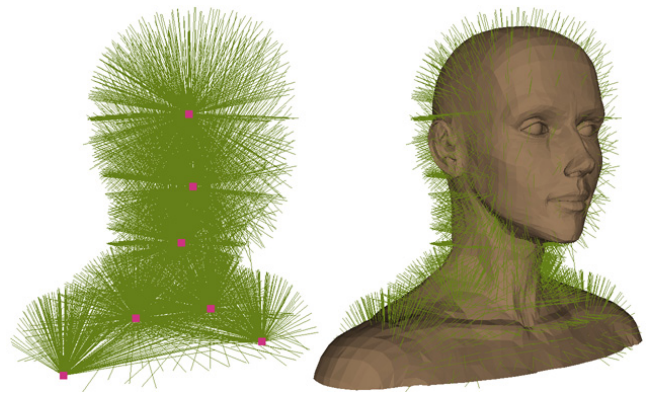
Collisions between individual hair strands are not really an issue when using volume deformations, as volume deformation continuity of FFD preserves the local relative positions between the deformed hair strands. The only issues may concern self-intersection of the FFD lattice in case of huge deformations, which in practice does not occur so as to produce disturbing animation effects.

The real issue is actually the detection and response of collisions between the hair and the body surface, which should prevent hair from entering the skin surface of the body.

Traditional collision approaches would consider computing the intersection between the hair geometry and the polygonal mesh that describes the body surface, possibly using optimizations based on bounding volumes, subdivision and incremental evaluation, which in any case remains a tedious task. We overcome this difficulty by replacing the body surface description by an approximate model which defines a repulsion force exerted on the lattice nodes from an analytically-defined volume energy potential roughly describing the body volume. Unlike surface-based approaches, such a scheme ensures robust collision handling by pushing out even deeply penetrating points unambiguously.

We have chosen 6th-order polynomial metaballs to perform this modeling. Metaballs are widely-used primitives for implicit surface modeling [5]. They have a well-defined finite radius of influence and are good additive primitives for modeling full volumes with adequate continuity properties, while evaluation of their repulsion forces (gradient of their energy potential) can be computed efficiently. 6th-order also offer null derivative (Jacobian) on the metaball limit, offering suitable continuity properties for use with implicit integration methods.

Yet, for real-time applications, we cannot afford to compute the influence of more than a few metaballs for each lattice node. However, an accurate model of the head and shoulders of a body may require several tens of metaballs depending on the required accuracy. Rather than working with a huge number of metaballs, we have chosen to customize the parameters (radius and base potential) of a reduced number of metaballs for each lattice node. Hence, a given node "sees" its own custom modeling of the skin with a suitable accuracy relatively to the relative position of the node along the body skin (Figure 9).

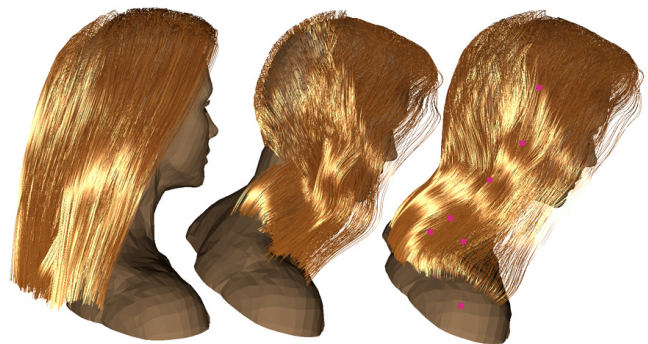


**Figure 9: Metaball centers and customized radii.**

The customization scheme works as follows:

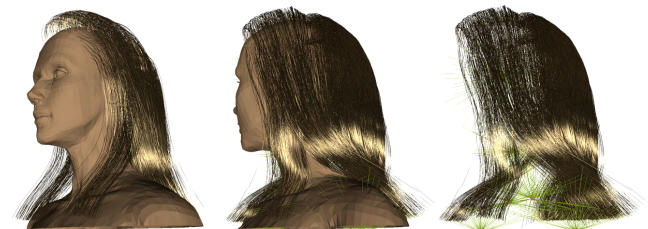
- If a lattice node is initially positioned outside the mesh, the metaball radius is chosen so as to prevent it from coming nearer to the mesh than a given "hair thickness" distance.
- If a lattice node is inside the mesh, the metaball radius is chosen so as to prevent it from going further inside the mesh. We prefer this option rather than completely blocking the node, in order to allow better tangential sliding of the hair on the skin surface.

Experimentally, this approach has shown to prevent quite successfully collisions from occurring against the body skin using a very small number of metaballs. For instance, a typical skull can be represented using one or two metaballs, whereas six to ten metaballs would be required when the hair may touch the shoulders and chest as well (Figure 10), depending on the wanted collision accuracy. Its robustness also allows this method to cope with any physical or not-so-physical head motions.



**Figure 10: Using an initial hairstyle (left), a tilting head would produce penetrating hairs (center) that is avoided through collisions using metaballs centered at the red dots (right).**

Metaball models do also nicely deform with simple displacement of their centers. Metaballs representing the shoulders and upper trunk are attached directly to the skeleton of these body parts rather than to the head, and our method is therefore applicable for deformable characters (Figure 11).



**Figure 11: When turning the head, hairs react to the shoulders properly.**



## 4. LATTICE FFD FOR HAIR ANIMATION

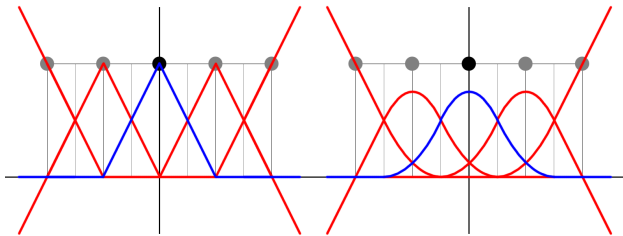
As the lattice is deformed during animation, another issue is to recompute the current position of each hair features for each frame. Depending on the selected rendering techniques and optimizations, these features may either be individual hair segments, or larger primitives such as hair strips or groups.

Different kinds of interpolations allow this to be performed with various degrees of continuity. They define how the weight coefficients should be computed from the initial position of the feature relatively to the undeformed lattice. A good review of interpolation curves and surfaces is described in [11].

The most continuous interpolations are based on Bezier curves. However, these are impractical as the weight vector is not sparse (the interpolated point position depends on the positions of all the nodes of the lattice), and therefore very inefficient to compute. For the best compromise between continuity and computation speed, we have selected quadratic B-Spline curves, which offer second-order interpolation continuity. For 3D interpolation, each interpolated point is a linear combination of the 27 nearest nodes of the lattice. In our adaptation, we use linear extrapolation to handle border nodes so as to decrease deformations for points located outside the lattice (Figure 12).

In applications where interpolation is really time-critical, linear interpolation still remains a good candidate. The resulting first-order continuity still looks acceptable if the lattice deformations are not too large, and the computation is roughly three times faster as each interpolated point is only a linear combination of the 8 nearest lattice nodes.

In our hair interpolation scheme, we precompute and store for each hair feature that needs to be rendered the sparse vector containing the weights corresponding to all lattice nodes. Then, during rendering, the current interpolation is simply computed by a weighted sum of the current lattice node positions.

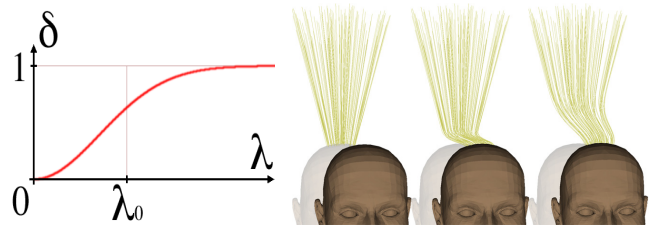


**Figure 12: Shape functions for linear interpolation (left) and quadratic B-Spline interpolation (right) with linear extrapolation, displayed for five lattice nodes (1D).**

We can take advantage of the interpolation to enhance the attachment of the hair on the skull through rigid motion. For each interpolated feature, a deformation coefficient  $\delta$  is defined which is a blending coefficient between the motion defined by the rigid head motion ( $\delta = 0$ ) and the motion defined by the interpolated lattice position ( $\delta = 1$ ). In our implementation, we vary progressively the deformation coefficient from 0 to 1 from the root to the extremity of each hair according to the following expression:

$$\delta = 1 - \exp(-\lambda^2 / \lambda_0^2) \quad (7)$$

Where  $\lambda$  is the distance of a hair point from the root along the hair curve.  $\lambda_0$ , which is the "typical" hair length from the root that does not deform significantly, acts as a parameter for defining the bending stiffness of hair near its root (Figure 13). This also removes the artifacts resulting from the imperfections of the mechanical attachment of the hair roots to the moving skull.



**Figure 13: (left) The deformation coefficient plotted against the distance along the hair from the root using Expression (7), and (right) resulting hair deformation corresponding to a head translation for various values of  $\lambda_0$  (From left to right: No correction (hairs rooted at initial position), low length, high length (moving roots to actual position)).**

There are many other ways to alter the deformation coefficient on different regions of the hair, for example for differentiating stiff hairs from soft ones, or introducing some randomness in the motion of various hair fibers.

## 5. RESULTS

We have implemented the computational algorithms of our system in standard C++ with standard floating-point math. The system was tested on a 3GHz Pentium4 PC with a nVIDIA Quadro4 980 XGL graphics card and 512MB of memory running Microsoft Windows 2000.

Hair rendering is also critical issue for achieving realistic display in real-time. The beauty of the hair mainly lies in the way it interacts with the light. A good example of lightning model is studied by Marschner et al [28]. Our implementation is mainly based on real-time rendering of strands and textured strips [33] using the anisotropic lightning model described by Kajiva et al [20]. This model was implemented by bypassing the standard OpenGL Texture & Lightning pipeline with our own illumination model implemented using nVIDIA's Vertex Shaders. We integrate this rendering in a Level-of-Detail scheme by adapting dynamically the number of rendered strands and strips according to the required visual accuracy.

### 5.1. Performances

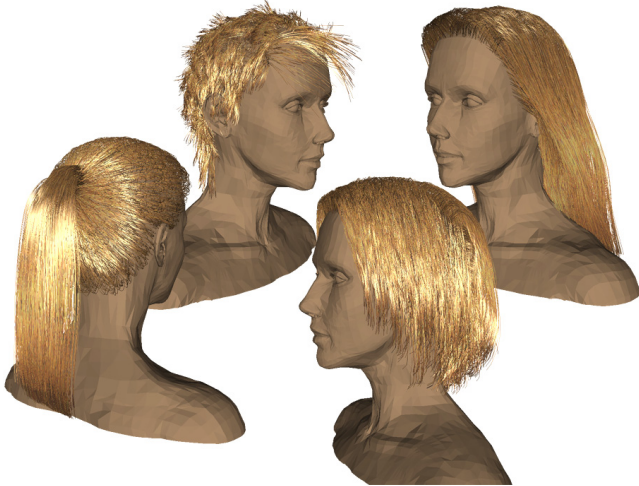
We have tested the hair model shown in Figure 8 with various complexities of the mechanical representation. The high-end mechanical model, which represents a very good simulation accuracy, contains 200 attachments and 800 springs built on a grid containing 1000 lattice nodes. The low-end model, still fairly accurate, only contains 50 attachments and 200 springs built on 125 lattice nodes. Both models react to collisions with the head and the shoulder using 7 metaballs. Models with no attachments and springs and no metaballs and relying only on ether stiffness, which may still be used for certain hair designs (short or stiff hair) or low accuracy requirements, are also tested for comparison.

The following table gives the time necessary for the mechanical computation of one frame (1/25 s) of the animation. The resolution of the implicit mechanical model uses 16 iterations of the Conjugate Gradient method. More iterations may be necessary for longer frame times or particular stiff hair models.

**Table 1: Computation time (milliseconds) of one animation frame using different mechanical model complexities and lattice sizes.**

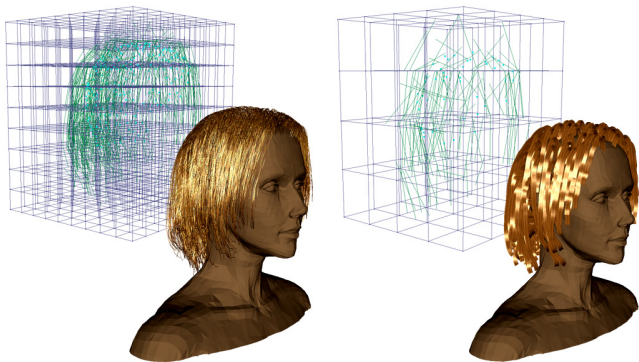
Mech.Mod ----- Lat.Size	0 Att. 0 Spr. 0 Meta.	0 Att. 0 Spr. 7 Meta.	50 Att. 200 Spr. 7 Meta.	100 Att. 400 Spr. 7 Meta.	200 Att. 800 Spr. 7 Meta.
10x10x10	1.1 ms	5.2 ms	10.1 ms	13.8 ms	21.4 ms
5x5x5	0.1 ms	0.2 ms	2.1 ms	3.8 ms	7.6 ms

Our implementation also needs approximately 2.5 ms for interpolating 10 000 features using quadratic B-Spline FFD, and only 1.0 ms using linear FFD. Adequate parametrization allows most hairstyles to be mechanically simulated within 10 milliseconds for one frame, which is 1/4 of the 40 milliseconds available (25 frames per second). This leaves 30 milliseconds for carrying out the animation of the face, body and other features of the scene, as well as rendering.



**Figure 14: Various hairstyles animated by our real-time simulation system.**

For most models, rendering is now the most time-consuming task of the whole real-time process. While the best rendering qualities are obtained with our system through explicit rendering of the hair strands with lighting and texture, this consumes between 50 and 100 milliseconds for about 10 000 strands. This is roughly the performance obtained for the rendered hairstyles shown in Figure 14. Better performance is obtained by reducing the number of rendered features and using textured strips instead of hairs. Rendering 500 textured strips roughly consumes 20 milliseconds in our implementation.



**Figure 15: Level-of-Detail: 200 ms for animating and rendering a frame with high-accuracy (left), and 40 ms with low-accuracy (right).**

We have taken advantage of the huge performance difference between models of various accuracies and rendering methods for implementing a Level-of-Detail animates scheme that simulates a given hairstyle by switching between several mechanical models depending on the required accuracy (related to on-screen head size and character motion speed). This allows total frame computation times between roughly 200 milliseconds (10x10x10 lattice, model containing 1000 springs and 250 attachments, 10 000 rendered textured hair strands with quadratic interpolation, textured body animation) and 40 milliseconds (5x5x5 lattice, 200 springs and 50 attachments, rendering 200 textured hair strips

with linear interpolation, low-quality body animation) (Figure 15). Faster times may still be obtained is the hair is approximated as rigid (lowest detail level).

## 5.2. Perspectives

The major benefit of this approach is its scalability, and its versatility. First, the approach allows a clear distinction between the mechanical model that animates the lattice and the actual objects that are deformed by the lattice during the rendering. This greatly eases the task of combining efficient simulation with complex hair representations, as well as designing level-of-detail schemes that can act independently on the mechanical simulation aspect and on the rendering aspect. This high versatility also allows the simulation of any hairstyle directly from the output of hairstyle design system without any specific handling. Hence, long flowing hair can be animated along with short and stiff hair using the same models. It is quite easy to extend the model to support extra original features (weighted hair knots, threads, and even cloth) through the simple addition of the corresponding mechanical behaviors in the particle system constituted by the lattice nodes.

This method still has plenty of room for evolution. The algorithmic simplicity of the model also turns it into a good candidate for hardware implementations. We could also benefit from the low number of mechanical degrees of freedom to replacing mechanical simulation by fast animate-by-example methods such as the one described by Grzeczuk et al [14]. We also plan to take advantage of the versatility to create real-time hairstyling systems based on mechanical simulation. We are still looking forward for many new developments to the exciting field of animated virtual characters.



**Figure 16: Animating a large diversity of hairstyles.**

## Acknowledgements

We are grateful to all participants to this research for their contributions and advices, as well as to Nedjma Cadi and Cristiane Luible for their high-quality creative work. This project is funded by the Swiss Fonds National pour la Recherche Scientifique.



**Figure 17: Easy simulation of real-world hairstyles.**

## REFERENCES

- [1] ANJYO K., USAMI Y., KURIHARA T., 1992, A Simple Method for Extracting the Natural Beauty of Hair, *Computer Graphics*, Proceedings of ACM SIGGRAPH 92, 26(2), p.111-120.
- [2] BANDO Y., CHEN B.Y., NISHITA T., 2003, Animating Hair with Loosely Connected Particles, *Computer Graphics Forum*, Proceedings of Eurographics 2003, Blackwell, 22(3) p.411-418.

- [3] BARAFF D., WITKIN A., 1998, Large Steps in Cloth Simulation, *Computer Graphics*, Proceedings of ACM SIGGRAPH 98, 32, p.106-117.
- [4] BERTAILS F., KIM T.Y., CANI M.P., NEUMANN U., 2003, Adaptive Wisp-Tree: A Multiresolution Control Structure for Simulating Dynamic Clustering in Hair Motion, *SIGGRAPH Symposium of Computer Animation*.
- [5] BLOOMENTHAL J, BAJAJ C., BLINN J., CANI M.P., ROCKWOOD A., WYVILL B., WYVILL G., 1997, *Introduction to Implicit Surfaces*, Morgan Kaufmann Publishers.
- [6] CHANG J.T., JIN J., YU Y., 2002, A Practical Model for Hair Mutual Interactions, *SIGGRAPH Symposium on Computer Animation*, p.73-80.
- [7] CHEN L.H., SAEYON S., DOHI H., HISHIZUKA M., 1999, A System of 3D hair Style Synthesis based on the Wisp Model, *The Visual Computer*, Springer Verlag, 15(4), p.159-170.
- [8] CHOI K.J., KO H.S., 2002, Stable but Responsive Cloth, *Computer Graphics*, Proceedings of ACM SIGGRAPH 02.
- [9] DALDEGAN A., MAGNENAT-THALMANN N., KURIHARA T., THALMANN D., 1993, An Integrated System for Modeling, Animating and Rendering Hair, *Computer Graphics Forum*, Proceedings of Eurographics 1993, Blackwell, 12(3), p.211-221.
- [10] FALOUTSOS P., VANDEPANNE M., TERZOPOULOS D., 1997, Dynamic Free-Form Deformations for Animation Synthesis, *IEEE Transactions on Visualization and Computer Graphics*, IEEE Computer Press, 3(3), p.201-214.
- [11] FARIN G.E., 1997, *Curves and Surfaces for Computer-Aided Design: A Practical Guide*, Morgan-Kaufmann.
- [12] FEATHERSTONE R., 1987, *Robot Dynamic Algorithms*, Kluwer Academic Publishers.
- [13] FENG J., HENG P., 1998, Accurate B-spline Free-Form Deformation of Polygonal Objects, *Journal of Graphics Tools*, 3(3), p.11-27.
- [14] GRZEZCZUK R., TERZOPOULOS D., HINTON G., 1998, Neuroanimator: Fast Neural Network Emulation and Control of Physics-Based Models, *Computer Graphics*, Proceedings of ACM SIGGRAPH 98, p.9-20.
- [15] GUANG Y., ZHIYONG H., 2002, A Method for Human Short Hair Modeling and Real-Time Animation, Proceedings of Pacific Conference on Computer Graphics and Applications, IEEE Computer Press, p.435-438.
- [16] HADAP S., MAGNENAT-THALMANN N., 2000, Interactive Hair Styler based on Fluid Flow, *EuroGraphics Workshop on Computer Animation and Simulation*, p.87-100.
- [17] HADAP S., MAGNENAT-THALMANN N., 2000, State-of Art in Hair Simulation, *International Workshop on Human Modeling and Animation*, Korea Computer Graphics Society, p.3-9.
- [18] HADAP S., MAGNENAT-THALMANN N., 2001, Modeling Dynamic Hair as a Continuum, *Computer Graphics Forum*, Proceedings of Eurographics 2001, Blackwell, 20(3), p.329-338.
- [19] HAUTH M., ETZMUSS O., 2001, A High Performance Solver for the Animation of Deformable Objects using Advanced Numerical Methods, Proceedings of Eurographics 2001, Blackwell.
- [20] KAJIVA J.T., KAY T.L., 1989, Rendering Fur with Three Dimensional Textures, *Computer Graphics*, Proceedings of ACM SIGGRAPH 89, p.271-280.
- [21] KIM T.Y., NEUMANN U., 2000, A Thin Shell Volume for Modeling Human Hair, Proceedings of Computer Animation 2000, IEEE Computer Society Press, p.121-128.
- [22] KIM T.Y., NEUMANN U., 2002, Interactive Multiresolution Hair Modeling and Editing, *Computer Graphics*, Proceedings of ACM SIGGRAPH 02, p.620-629.
- [23] KOH C.K., HUANG Z., 2000, Real-Time Human Animation of Hair Modeled in Strips, *Eurographics Workshop on Computer Animation and Simulation*, Springer-Verlag.
- [24] KOH C.K., HUANG Z., 2001, A Simple Physics Model to Animate Human Hair Modeled in 2D Strips in Real-Time, *Eurographics Workshop on Computer Animation and Simulation*, Springer-Verlag, pp127-138.
- [25] KOH C.K., HUANG Z., 2000, Real-Time Human Hair Modeling and Animation, *SIGGRAPH 2000 Sketches and Applications*.
- [26] KURIHARA T., ANIYO K., THALMANN D., 1993, Hair Animation with Collision Detection, Proceedings of Computer Animation 1993, IEEE Computer Society Press, p.128-138.
- [27] LEE D.W., KO H.S., 2001, Natural Hairstyle Modeling and animation, *Graphical Models*, 63(2), p.67-85.
- [28] MARSCHNER S.R., JENSEN H.W., CAMMARANO M, WORLEY S., HANRAHAN P., 2003, Light Scattering from Human Hair Fibers, *Computer Graphics*, Proceedings of ACM SIGGRAPH 03.
- [29] PLANTE E., CANI M.P., POULIN P, 2001, A Layered Wisp Model for Simulating Interactions Inside Long Hair, *Eurographics Workshop on Computer Animation and Simulation*, Springer-Verlag, p.139-148.
- [30] PRESS W.H., VETTERLING W.T., TEUKOLSKY S.A., FLANNERY B.P., 1992, *Numerical Recipes in C*, Cambridge University Press.
- [31] ROSENBLUM R., CARLSON W., TRIPP E., 1991, Simulating the Structure and Dynamics of Human Hair: Modeling, Rendering and Animation, *The Journal of Visualization and Computer Animation*, J. Wiley, 2(4), p.141-148.
- [32] WARD K., LIN M.C., 2003, Adaptive Grouping and Subdivision for Simulating Hair Dynamics, Pacific Graphics proceedings, IEEE Computer Society Press.
- [33] WARD K., LIN M.C., LEE J., FISHER S., MACRI D., 2003, Modeling Hair using Level-of-Detail Representations, *International Conference on Computer Animation and Social Agents*, Proceedings of Computer Animation 2003, IEEE Computer Society Press.
- [34] WATANABE Y., SUENAGA Y., 1989, Drawing Human Hair using the Wisp Model, Proceedings of Computer Graphics International 1989, Springer-Verlag, p.691-700.
- [35] YANG X.D., XU Z., YANG J., WANG T., 2000, The Cluster Hair Model, *Graphical Models*, Elsevier, 62(2), p.85-103.
- [36] YU Y., 2001, Modeling Realistic Virtual Hairstyles, Proceedings of Pacific Graphics, IEEE Computer Society Press, p.295-304.





# Modeling Hair Using Level-of-Detail Representations

Kelly Ward   Ming C. Lin   Joohee Lee   Susan Fisher   Dean Macri  
University of North Carolina at Chapel Hill   Alias|Wavefront   Pixar Studio   Intel Corporation  
<http://gamma.cs.unc.edu/HSLOD/>

**Abstract:** We present a novel approach for modeling hair using level-of-detail representations. The set of representations include individual strands, hair clusters, and hair strips. They are represented using subdivision curves or surfaces, and have the same underlying base skeleton to maintain consistent high-level physical behavior when a transition between different levels-of-detail occurs. This framework supports automatic simplification of dynamic simulation, collision detection, and graphical rendering of animated hair. It also offers flexibility to balance between the overall performance and visual quality, and can be used to model and render different hairstyles. We have used these level-of-detail representations to animate various hairstyles and obtained noticeable performance improvement, with little loss in visual quality.

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

## 1 Introduction

The ability to model human features has become an essential aspect of 3D graphics for modeling avatars in virtual environments, virtual humans in computer games, and human characters in animated films. A human head can have over 100,000 individual strands of hair. Animating hair in real-time is a challenging problem due to the high number of primitives required to model hair accurately and realistically. It is also difficult to achieve stable and robust simulation of hair dynamics as well as interactions among the hair and between the hair and the body for different types of hairstyles.

The current commercial renderers, such as Pixar's RenderMan and other proprietary software used in movies like *Monsters, Inc.* and *Final Fantasy*, can generate beautiful and realistic appearances for hair and fur. However, to the best of our knowledge, these systems do not offer interactive performance for either animation or rendering of hair. Modeling, styling, simulating and animating hair remains a slow, tedious and often painful process for animators. A few existing real-time algorithms [Koh and Huang 2001] or hardware accelerated rendering and shading [NVIDIA 2001], on the other hand, often do not yield realistic hair renderings or are limited to certain hairstyles (e.g. short, straight hair). One of the major bottlenecks in achieving real-time simulation of moving hair is collision detection among hairs and between the hair and the body. This is often the dominating factor in terms of the overall computational cost [Plante et al. 2001].

**Main Contribution:** In this paper, we present a novel, unified framework for modeling hair based on level-of-detail (LOD) representations. We use a series of subdivision surface patches [Schröder and Zorin 1998] for modeling the least significant layers of hair, hair clusters represented as subdivision surfaces of variable thickness for modeling the intermediate layers, and subdivision curves for simulating the most visible and highest-resolution individual hair strands. Two hairstyles modeled and simulated using a combination of these representations are shown in Fig. 1 and Fig. 2. Our algorithm combines this set of LOD representations 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 subdivision framework and the "base skeleton" representation;



Figure 1: Long, curly red hair blowing in the wind.



Figure 2: Short, wavy brown hair.

- **Automatic simplification** of both geometry and physics for hair animation and rendering;
- **Computational efficiency** in the overall dynamic simulation, collision detection, and graphical rendering;
- **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 these level-of-detail representations for modeling hair, we observed noticeable overall 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 subdivision framework and the base skeleton. The dynamics model and our collision detection algorithm using the LOD representations are described in Section 4 and Section 5 respectively. In Section 6, we describe techniques to render hair using the proposed level-of-detail representations. The criteria for automatically switching and selecting LOD representations are outlined in Section 7. Section 8 highlights the

results of our implementation, and analyzes its performance. Finally, we suggest several areas for future work.

## 2 Related Work

Our work is built upon a large body of knowledge and concepts from several different areas, including hair modeling, rendering and animation, multiresolution representations, and simulation acceleration techniques. We synthesize together many key ideas from different areas, improve upon several known algorithms for simulating and rendering hair, and propose a new approach for hair modeling based on the subdivision framework and the base skeleton.

### 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 [Magenat-Thalmann et al. 2000; Hadap and Magenat-Thalmann 2001; Yu 2001; Kim and Neumann 2002]. Some fundamental techniques were presented to model the motion of individual hair strands in [Anjyo et al. 1992; Kurihara et al. 1993; Daldegan et al. 1993], 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* [Kurihara et al. 1993]. 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 [Plante et al. 2001]. A similar approach is used for interactive hairstyling [Chen et al. 1999; Xu and Yang 2001]. Adaptive guide hairs were used in [Chang et al. 2002] to add more detail to overly interpolated regions. Using guide strands involves animating a few strands and the dynamics of the remaining strands are interpolated from these guides.

None of these techniques, though, can perform hair animation or rendering in real-time. Recently, a thin shell volume [Kim and Neumann 2000] and 2D strips [Koh and Huang 2000; Koh and Huang 2001] 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 and hair that exploit graphics hardware were presented in [Lengyel 2000; Lengyel et al. 2001; NVIDIA 2001]. However, these techniques do not work well or are not applicable 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 [Schröder and Zorin 1998] techniques, have been proposed to accelerate the rendering of complex geometric models. A recent survey on polygonal model simplification is presented in [Luebke 2001]. A generic framework for selecting and switching between different geometric levels-of-detail (LODs) to attain a nearly constant frame rate for interactive architectural walk-throughs was introduced in [Funkhouser and Séquin 1993].

### 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 [Carlson and Hodgins 1997]. In [Perbet and Cani 2001], 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 [O’Brien et al. 2001].

Other types of simulation acceleration techniques, such as

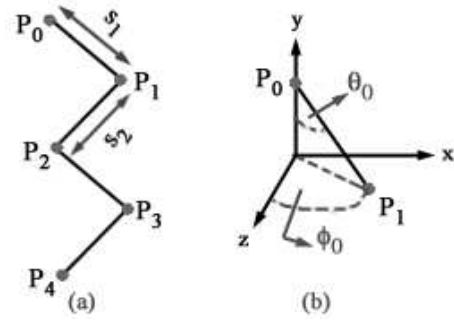


Figure 3: Basic Physics Models. (a) *The base skeleton model*; (b) *The parameters that define the style of hair*.

view-dependent dynamics culling [Chenney and Forsyth 1997] and Neuro-Animator [Grzeszczuk et al. 1998], have also been investigated to reduce the total computational costs for simulating a large, complex dynamical system.

## 3 Model Representations

Our approach uses three novel representations based on the subdivision framework and a base skeleton to create both discrete and continuous levels-of-detail for hair. They are *strips*, *clusters*, and *individual strands*.

### 3.1 Design Consideration

Although the set of proposed LOD representations may appear to be intuitive and simple, it is carefully designed and chosen. We have adapted the use of the subdivision framework. The subdivision framework can model different hairstyles as effectively as NURBS, quickly perform adaptive dynamic tessellation, and most of all can potentially take advantages of new graphics hardware for interactive rendering of curve primitives. (More detail will be given in section 6.)

The use of the base skeleton is intentionally selected to maintain a global, consistent, macroscopic physical behavior, as LOD switches take place. This choice helps to drastically simplify many transition difficulties typically present during LOD switching. It automatically reduces a fairly high degree-of-freedom dynamical system down to a lower degree-of-freedom dynamical system, without any extra expensive computations other than performing the LOD switching tests (to be described in section 7).

### 3.2 Subdivision Representations

Subdivision curves and surfaces have been chosen as the underlying geometric representation for all LODs in our hair modeling framework because of their scalability and uniformity of representation [Schröder and Zorin 1998]. The subdivision process creates smooth curves and surfaces through successively refining a curve or mesh of control points. Defining the levels of successive refinement can control the smoothness of the resulting surface or curve. This is used to generate adaptive, continuous LODs for rendering. A detailed discussion on the subdivision framework and techniques can be found in [Schröder and Zorin 1998].

Subdivision in 1D is used to create the curves that represent hairs as individual strands, to be discussed in Sec. 3.6. The *4pt* scheme in 1D is an efficient method for creating a smooth curve. For surface representations, we create a triangular base mesh and subdivide. The control mesh for each of the three representations is shown in Fig. 4(a)(c) and (e).

### 3.3 The Base Skeleton

Based on the idea for modeling each individual hair strand [Kurihara et al. 1993], we use a similar structure for the base skeleton, which forms the “core” of our proposed set of LOD representations. The base skeleton is comprised of  $n$  control points, or nodes. This value is decided automatically based on criteria such as the length of the hair, the waviness or curliness specified for the hair, and the desired smoothness. The skeleton is modeled as an open chain of line

segments that connect these nodes. Spring forces are used to control the angles between each node, while the distance between each node is fixed. Fig. 3(a) shows the basic setup of the skeleton. The  $n$  nodes ( $\mathbf{p}_0, \mathbf{p}_1, \dots, \mathbf{p}_{n-1}$ ) and  $n - 1$  segments ( $s_1, s_2, \dots, s_{n-1}$ ) define the skeleton. Specifying the shape of the skeleton model is discussed in Sec. 3.8.

### 3.4 Strips

The strip model in Fig. 4(a) and (b) uses a single skeleton model as its basis for motion. The structure for this model is inspired by the strips representation presented by [Koh and Huang 2000; Koh and Huang 2001]. The skeleton is the center of the strip and for each node in the skeleton there are two control points that are used to define the strip. These two strip control points and the skeleton node point are collinear. A skeleton with  $n$  nodes will result in a subdivision surface created from a control polygon consisting of  $2n$  control points.

A strip is typically used to represent the inner most layers of hair or parts of hair that are *not visible* to the viewer and, therefore, are often not rendered. It is the coarsest (lowest) level-of-detail used for modeling hair. It is mainly used to maintain the global physical behavior and the volume of the hair during the simulation.

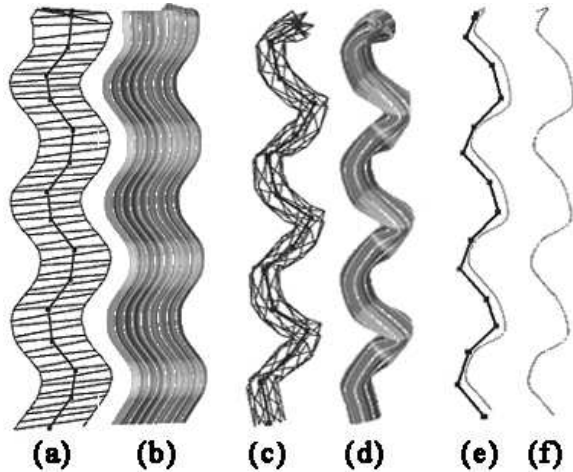


Figure 4: Level-of-Detail Representations for Hair Modeling. (a) Subdivision representation of strip with skeleton; (b) Rendered strip; (c) Subdivision representation of cluster with skeleton; (d) Rendered cluster; (e) Subdivision representation of a strand with skeleton; (f) Rendered individual strand.

### 3.5 Clusters

The clusters are represented as generalized cylinders created with texture-mapped subdivision surfaces, as shown in Fig. 4(c) and (d). Each cluster is based on one skeleton that is located at the center of the cluster. A radius is specified at 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, made up of  $m$  control points, is created based on the radius value at that node. Thus, a skeleton made up of  $n$  points will create a cluster of  $mn$  control points. Typically having  $m=4$  is enough detail to define the cross-section.

A cluster is used to model the intermediate layers of hair and often makes up the majority of the body of semi-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.

### 3.6 Individual Strands

Each individual strand is modeled as a subdivision curve using 1D subdivision with  $n$  control points, as shown in Fig. 4(e) and (f). Most human heads have a few individual strands that are separate from the body of their hair. These types of small imperfections are usually only noticeable when viewing the hair closely. The ability

to see these individual strands is what makes this representation a finer detailed model of hair than the clusters.

### 3.7 Generating LODs

In the model simplification literature, typically either static LODs are generated offline for online switching, or dynamic LODs are computed on the fly. In our current framework, a combination of both static and dynamic LOD representations is used.

Continuous LOD representations are generated by the subdivision of curves and surfaces for fast rendering. Given the three basic discrete LOD representations, a small number of individual strands are grouped into clusters and a few hair clusters are combined to form a hair strip. In our current implementation, for the ease of validating the effectiveness of this framework, a predefined number is used in the simulation. (Due to page limitation, more implementation detail is given at the project website.) However, this can be modified to generate dynamic groupings of strands and clusters on the fly. This is a non-trivial computation to perform in real-time, considering the number of strands or clusters on a human head. It is thus beyond the scope of this paper and we plan to investigate the feasibility of this option in our future work.

We also use a hybrid representation, where only a single skeleton model is used to simulate physical behavior for a group of individual strands, while individual strands are rendered. This is a fairly popular technique used to generate high-quality animation. This is used in some of our simulations to help further accelerate the overall performance, while maintaining the overall visual quality.

### 3.8 Hairstyling

The skeleton controls the motion and the shape of each hair section and is responsible for the overall style of the hair. Various shapes or styles of hair can be specified by stipulating the rest angles  $\theta_0$  and  $\phi_0$  (see Fig. 3) of each node  $i$  of the skeleton. Straight hair can be created by assigning  $\theta_{i0}$  to 0 and  $\phi_{i0}$  to 0 for each node  $\mathbf{p}_i$  of the skeleton. In addition, we can create a wavy hairstyle by zig-zagging the position of the nodes down the length of the skeleton. A zigzag or wavy skeleton is created by assigning each  $\theta_{i0}$  to a certain angle between 0 and 90 degrees and then the values of  $\phi_{i0}$  alternate by 180 degrees. Ringlet or spiral curls can also be created using the skeleton by specifying an angle value between 0 and 90 degrees for  $\theta_{i0}$  and then, to achieve the spiral effect, each  $\phi_{i0}$  value increments by 90 degrees down the length of the skeleton.

These two processes for stipulating waves or curls can be altered to create varying styles. The segment size and the values for  $\phi_{i0}$  can be changed to create non-uniform curls and waves according to the desires of the user.

Both the strands and the clusters are able to accurately depict the shape defined by the user. While the strip 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. Strips are only used when the viewer cannot observe fine detail, such as when the hair is at distances far from the viewer, or when the hair is not in sight. Thus, while the strip cannot depict all hairstyles as accurately as the other two LODs, it is not usually apparent to the viewer. Criteria for choosing an LOD is discussed in further detail in Sec. 7.

## 4 Dynamic Simulation

The use of the same underlying skeleton model for each hair representation, i.e. strips, clusters, and individual strands, provides a simple yet effective 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 that 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 [Anjyo et al. 1992; Kurihara et al. 1993]. The forces that are applied to each node point govern the motion of the skeleton. The force measured from the angular springs of the skeleton model,  $F_{spring}$ , helps hold the specified hairstyle during the simu-

lation.  $F_{i_{spring}}$ , force for the  $i$ th node of the skeleton, is calculated by combining the forces for  $F_{i_\theta}$  and  $F_{i_\phi}$ .

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

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

where  $k_\theta$  and  $k_\phi$  are angular spring constants and  $\theta_{i0}$  and  $\phi_{i0}$  are initial angles for node  $i$ .

Other forces that act on the hair are gravity,  $F_{gravity}$ , external forces,  $F_{ext}$ , such as wind, and forces due to collision. We will ignore the influence of collision forces on a node until Sec. 5. Summing the remaining forces together, we obtain a magnitude and direction for the simulation force  $F_{sim}$  applied to each skeleton node  $\mathbf{p}_i$ , to be:

$$F_{sim} = F_{ext} + F_{spring} + F_{gravity}.$$

## 4.2 Transitioning between LODs

One of the most crucial aspects of using LODs in a simulation is the ability to switch between different representations smoothly. 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 with little visual artifacts. When a switch is made, the skeleton of the new level-of-detail inherits the dynamics state of the skeleton of the previous level.

For example, if the current LOD is a strip, the algorithm refines the hair model and makes a transition to multiple clusters. The section of hair transitions from a model with one skeleton to one with  $c$  skeletons, where  $c$  is the number of clusters represented by one strip. Each cluster skeleton linearly interpolates position and motion values from the strip skeleton based on the position of the cluster skeleton’s root, or the first skeleton node,  $\mathbf{p}_0$ , in relation to the root of the strip skeleton.

The transitions are even more straightforward going in the reverse direction. As we move from multiple clusters back to a single strip, the skeleton of the strip simply inherits the average position and motion values of the cluster skeleton that the strip represents. 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, simplifying the dynamics of the hair as needed. Criterion for switching between LODs is discussed in Sec. 7. A strip can model the largest portion of hair and its simulation requires as little computation as a single strand or a cluster of hair. When appropriate, strips 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

Collision detection is a vital part of hair simulation since hair is in constant contact with the scalp of the head and other hairs. Due to the high complexity of hair, this can be a costly computation and it is crucial that the collision detection is performed efficiently.

### 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. To perform collision detection using BVHs, two objects are tested by recursively traversing their BVHs [Larsen et al. 2000].

### 5.2 Swept Sphere Volumes

To perform collision detection on the different representations of hair, we adapt the family of “swept sphere volumes” (SSV) [Larsen et al. 2000] 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. More precisely, let  $C$  be the core skeleton and  $S$  be a sphere of radius  $r$ . Each SSV,  $B$ , can be defined as:

$$B = C \oplus S = \{c + r \mid c \in C, r \in S\}$$

SSVs are chosen as the bounding volumes in our framework, because the shape of our LOD representations shares close resemblance to those of SSVs. Different SSVs provide varying tightness. For clusters and strands, line swept spheres (LSS) are the best candidates for each segment, while rectangle swept spheres provide the better fit for strips, and the point swept sphere for the head at the top level. To detect a collision between a pair of SSVs we simply perform a distance computation on the corresponding pair of core skeletons and then subtract the appropriate offset (radius of each SSV).

### 5.3 Constructing SSVs for Hair Representations

For each rigid segment of the skeleton model, that is, each line segment between two nodes, we pre-compute an SSV BV. Since each representation of the hair can be tightly encapsulated using a single SSV, a BVH is not necessary for the hair. For a skeleton with  $n$  nodes, there are  $n - 1$  segments, and thus  $n - 1$  single SSVs. The variable thickness of each segment defines the radius of the SSV along its length.

In order to compute a BV for a strip, we let the four control points of the strip that outline a skeletal segment define the area for a BV to enclose. This is performed for each of the  $n - 1$  segments along the skeleton. The geometry of the strip is different from the other two representations in that the strip is a surface while the clusters and a collection of strands are volumes. In order to allow the transition from a strip into multiple clusters remain faithful to the volume of hair being depicted we create a BV for a strip section by surrounding it with a box of certain thickness. Each strip is given a thickness equal to that of its cluster and strand grouping counterparts. While the strip is rendered as a surface, it acts physically as a volume. Thus, when a transition from a strip into clusters occurs, the *volume* of hair being represented remains constant throughout this process.

For the cluster representation, we create a BV around the  $2m$  control points that define a segment ( $m$  control points, as defined in section 3.5, from the cross-section at the top of the segment and  $m$  control points at the bottom of the segment).

For individual strands, we perform collision detection for each strand or group of strands, depending on implementation, in a manner similar to that of the clusters. We compute an LSS around the skeleton that defines each segment with a radius defining the thickness. The radius of each BV is varied based on the thickness of a group of strands.

Once the BVs are computed for the hair we construct a BVH of SSVs using a top-down hierarchy construction for the other objects in the scene, head, body, etc., as a pre-computation. During the runtime simulation, the single SSVs and the BVHs are used to perform collision queries, and are lazily updated on the fly. To detect a collision between a segment of hair and an outside object, the hair’s SSV is tested against the BVH of the object. (Please refer to [Ward et al. 2003] for more detail.)

### 5.4 Collision Response

Since the same skeleton model is used for each representation, whenever a collision is detected we calculate the response using the same method for all of the representations. Once a collision between the hair and the head, or other object, has been detected our algorithm determines how much the segment of hair is penetrating the head, the velocity of that segment of hair in the direction of the head is set to zero, and the hair segment is moved so that it is outside of the head. A frictional force in the direction tangent to the head is also applied to each skeleton node when a segment of hair collides with the head or body.

During the simulation, segments of hair are in constant contact with other segments of hair. Since we do not want to perform a collision detection test on a hair segment against all of the remaining segments of hair, the area around the head is spatially decomposed or broken into three-dimensional grids. Each segment of hair is inserted into the grids and only hair segments that fall into the same grid are tested against each other. Segments of hair are tested against each other for collision by performing an intersection test on their SSVs.

If two hair segments intersect with each other, they need to be moved apart. The SSV overlap test determines the distance the skeleton segments are from each other in addition to testing for collisions. We then use this calculated distance as the amount to move the intersecting segments apart.

Further detail on the hair collision detection and collision response algorithms, including exact methods for determining response magnitudes and directions for hair-object and hair-hair collisions and calculating frictional forces, can be found in [Ward et al. 2003].

## 6 Rendering

In our system, a mixed model of discrete and continuous geometric LODs is used to render the hair representations as described in section 3. We adapted the shading model suggested by [Kajiya and Kay 1989], to capture the anisotropic nature of hair. An efficient implementation of anisotropic surface rendering is performed using an OpenGL texture matrix, as appears in [Heidrich and Seidel 1998]. The rendering of the subdivision surfaces that are used for clusters and strips are assigned two additional textures. The first texture contains the hair color information. It is created as a pre-process from the same color range that assigns color values to the strands. The next texture used on the surfaces contains alpha values that define the transparency of the surfaces [Koh00]. Alpha mapping is used to create the illusion that there are individual strands being rendered. The anisotropic lighting, the hair color, and the alpha textures for the surfaces are rendered in a single pass using multi-texturing.

Self-shadowing is a vital factor to increase the volumetric cue of the hair. We use opacity shadow maps [Kim and Neumann 2001], which are a fast approximation of deep shadow maps.

Aliasing is an innate problem of hair rendering, because each hair strand is too thin to occupy a single pixel. Fortunately it is not as major of a problem for high LOD representations like the clusters and strips, as it is for strands. We rely on graphics hardware for antialiasing. NVIDIA GeForce family provides hardware implemented high resolution antialiasing through multi-sampling [NVIDIA 2002]. We select 4-sample 9-tab multi-sample mode – the highest quality available to us, because various multi-sampling modes provided by the graphics hardware does not affect overall performance of our system. In order to render the simulated hairs with motion blur, we adapt the technique presented by [Haerberli and Akeley 1990]. The current image is rendered into the accumulation buffer so that it can be integrated with previous images. Using the accumulation buffer also reduces the aliasing of individual strands.

Switching between different LODs can introduce visual disturbances. To address this problem, we combine several methods. By using the alpha channel in the textures for clusters and strips we add the visual illusion of strands, making the LOD transition less noticeable. The density values necessary for the opacity shadow map computations for the clusters and strips are based on the number of strands they represent. This helps to keep the brightness of the shadows constant. We also blend the images of previous and current LODs to make transitions smoother.

## 7 Choosing Hair Representations

Given the three representations for 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. The viewer cannot see hair 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 representation is normally simulated using  $g$  number of skeletons but is occluded by other objects, we simulate that section of hair using one larger strip, and therefore, one skeleton. 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, therefore we continue to simulate it. In addition, when a hair section is occluded, it 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 strands as strips that use fewer skeletons and these sections are not rendered.

In our current implementation, we perform a simple occlusion test that involves fitting a sphere to the head so that it is slightly smaller than the head. If there are other objects in the scene, such as a body, a similar method is applied. We then test the SSV bounding volumes of the hair to see if they are visible from the camera or if the occluders occlude them. It is possible to use more sophisticated occlusion culling algorithms, such as [Zhang et al. 1997], or special features on new GPUs (e.g. NVidia NV30) to perform these tests more efficiently.

### 7.2 Viewing Distance

Hair that is far from the viewer cannot be seen in great detail. We can estimate the amount of detail that will be seen by the viewer by computing the screen space area that the hair covers. As the distance from the viewer to the hair increases, the amount of pixels covered by the hair gets smaller and less detail is viewable. We can calculate the amount of pixels covered by the hair to choose the appropriate LOD.

A single strip, a group of  $c$  clusters, or a group of  $g$  strands represent a given portion of hair. Each is designed to cover a similar amount of world space, thus we can use the control skeleton of the strip as an estimate to the amount of screen space area a given hair section occupies. To determine the screen coverage of this hair, we create a line from the first skeleton node to the last skeleton node and project this line into 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. The pre-determined maximum allowable size for each LOD is decided experimentally based on the viewer's preference.

### 7.3 Hair Motion

If the hair is not moving at all, then a large amount of computation is not needed to animate it and we can use a lower level-of-detail. When the avatar makes sudden movements, e.g. shaking his or her head, or a large gust of wind blows through the hair, 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.

We choose the particular LOD based on hair motion by first determining the skeleton node 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 can be modeled as a strip.

### 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 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 in direct 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 strip 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 higher detail. 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 [Larsen et al. 2000], 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 or wavy hair automatically by setting 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 BV 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 on various scenarios. Please visit our project website:

<http://gamma.cs.unc.edu/HSLOD>

for MPEGs of these simulation runs and for a sequence of snapshots taken from a hair simulation using our LOD representations.

We also compared the performance for the overall dynamic simulation (not including collision detection) and collision detection using different representations on various simulation scenarios. Table 1 gives a detailed comparison of the *average* running times using a combination of LOD representations (indicated as LODs) against the use of only one of the three discrete LOD representations (Strands, Clusters, and Strips). Fig. 5 shows the runtime comparison of the simulation performance over the entire duration, as the camera zooms out, increasing the distance to the viewer. The rendering performance is similar to that of the simulation. The basis for our comparisons uses the average timings for the strand simulation as the value 1 on the graph.

For this benchmark, we used 8045 individual strands, which were represented using only 173 strips or only 519 clusters. In these benchmarks, a combination of all three discrete LOD representations was automatically determined by our framework at any given time during the simulations. Timings were taken on a PC equipped with an Intel(R) Pentium(R) 4 2-GHz processor, 1 GB main memory and GeForce(R) 4 graphics card.

Strips provide the best overall performance in simulation time, since it is the coarsest (lowest) LOD of hair. But, a combination of three discrete LOD representations using our framework offers significant performance advantages over the use of individual strands alone. Note there are also implicitly continuous LOD representations used in our system with the subdivision framework. While it renders images of almost equal visual quality as that of individual strands, our LOD implementation gives much better timing performances than modeling with individual strands in simulation, collision detection, as well as rendering.

Breakdown	LODs	Strands	Clusters	Strips
Dyn Sim	0.0175	0.5834	0.0298	0.00592
Col Detect	0.0567	2.1934	0.1297	0.01896
Total	0.0742	2.7768	0.1595	0.02488

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

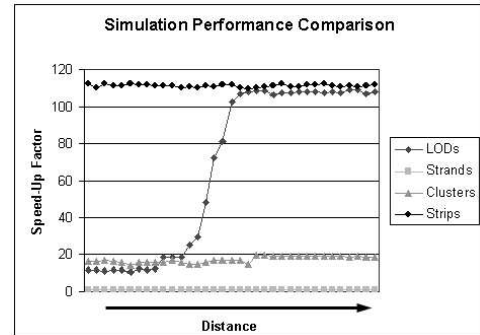


Figure 5: Simulation Performance Comparison. *We show the factors of speed-up among LOD, strips, and clusters over the strands alone, which is the baseline for comparison. The simulation speed of our system consistently outperforms the individual strands. It quickly outperforms the use of clusters alone, as the camera starts to zoom out. Then, soon after a certain distance threshold, it performs comparably to the use of strips alone.*

### 8.3 Analysis and Discussion

The impetus of this research is to explore the use of level-of-detail 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 comparison on the visual quality of rendered images.

Using side-by-side comparison (see the project website), we notice little degradation in the visual quality of the rendered image using LODs. While they offer the best computational performance, the images of hair simulated by strips appear sharp and angular, lacking a realistic appearance. The performance of our framework varies depending on the scenarios. In general, its overall performance in simulation and rendering compares favorably against the use of strands, clusters or strips alone. 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.

**Limitations:** One possible limitation of our algorithm is the slight popping that can occur if aggressive LOD transitions are used. However, we have alleviated this visual artifact with motion blurring and image blending during the LOD transition in our system.

Occlusion culling is an active and challenging area of research. To perform occlusion culling for hair rendering presents many more new challenges, as the hair can self occlude, but each strand, cluster or strip is rather small in size yet in aggregation its capability to occlude the rest of the hair can be significant. Our current implementation only has simple object-hair occlusion tests to validate the basic idea. However, we have already observed some performance gain using occlusion culling. We plan to investigate more sophisticated techniques, including the use of new graphics hardware, in the near future.

One of the foremost difficulties in hair simulation is to achieve stability for all types of hairstyles, lengths and interactions. We have implemented Runge-Kutter of order 4 for numerical integration. We are currently investigating the use of implicit methods.

In order to achieve high rendering rates, we are only performing two-pass refinement in our current implementation of opacity shadow maps. For better rendered images, more passes are required

[Kim and Neumann 2001]. We plan to develop a nicer interface to allow the users to trade off speed for higher-quality rendering and vice versa.

There are other application dependent transition criteria, such as collision, that we have not examined closely. These factors can further contribute to improving the overall system performance.

#### 8.4 Comparisons Against Other Approaches

A multiresolution technique for hairstyling is presented in [Kim and Neumann 2002]. They only use groups of strands of different sizes. They have not applied this approach to hair simulation. The switching is directly controlled by the user, while ours offers the capability of automatic switching. To apply this technique to hair simulation requires dynamic grouping on the fly. This is very difficult to perform given the number of strands.

Techniques that use interpolation from guide strands, such as [Chang et al. 2002], can be limiting in the types of hairstyles that can be modeled as well as interaction capabilities. Furthermore, their simulations run at slower rates than ours [Yu 2002].

Our approach compares favorably to those using only cluster-like representations (e.g. wisps or generalized cylinders) [Chen et al. 1999; Kurihara et al. 1993; Xu and Yang 2001; Plante et al. 2001], as we can achieve similar visual quality with faster rendering rates.

The real-time animation techniques [Koh and Huang 2000; Koh and Huang 2001; Lengyel 2000; Lengyel et al. 2001; NVIDIA 2001] and those commonly found in video games are either limited to certain hairstyles (mostly short, straight hair) or the resulting image quality is inferior to ours.

#### 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 costs. In addition to potential areas of improvements mentioned in the earlier sections, 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.;
- Dynamically change the hairstyle, as the user combs or brushes the hair with a 3D user interface;
- Automatically generate desired simulation outcomes, given high-level user guidance.

#### Acknowledgements

This research is supported in part by Army Research Office, Intel Corporation, National Science Foundation, and Office of Naval Research. We would like to thank Ben Lok for the body model and help in capturing head motion.

#### References

- ANJYO, K., USAMI, Y., AND KURIHARA, T. 1992. A simple method for extracting the natural beauty of hair. *Computer Graphics* 26, 2, 111–120.
- CARLSON, D., AND HODGINS, J. 1997. Simulation levels of detail for real-time animation. In *Proc. of Graphics Interface 1997*.
- CHANG, J., JIN, J., AND YU, Y. 2002. A practical model for hair mutual interactions. *Proc. of ACM Symposium on Computer Animation*.
- CHEN, L. H., SAEYOR, S., DOHI, H., AND ISHIZUKA, M. 1999. A system of 3d hair style synthesis based on the wisp model. *Visual Computer* 15, 4, 159–170.
- CHENNEY, S., AND FORSYTH, D. 1997. View-dependent culling of dynamic systems in virtual environments. In *Proc. of ACM Symposium on Interactive 3D Graphics*.
- DALDEGAN, A., KURIHARA, T., MAGNENAT-THALMANN, N., AND THALMANN, D. 1993. An integrated system for modeling, animating and rendering hair. *Computer Graphics Forum (Proc. of Eurographics 12)*, 3, 211–221.
- FUNKHOUSER, T. A., AND SÉQUIN, C. H. 1993. Adaptive display algorithm for interactive frame rates during visualization of complex virtual environments. In *Proc. of ACM SIGGRAPH*, J. T. Kajiya, Ed., 247–254.
- GRZESZCZUK, R., TERZOPOULOS, D., AND HINTON, G. 1998. Neuroanimator: Fast neural network emulation and control of physics-based models. In *Proc. of ACM SIGGRAPH*, 9–20.
- HADAP, S., AND MAGNENAT-THALMANN, N. 2001. Modeling dynamic hair as a continuum. *Computer Graphics Forum (Proc. of Eurographics 2001)* 20, 3.
- HAEBERLI, P. E., AND AKELEY, K. 1990. The accumulation buffer: Hardware support for high-quality rendering. In *Proc. of ACM SIGGRAPH*, F. Baskett, Ed., 309–318.
- HEIDRICH, W., AND SEIDEL, H.-P. 1998. Efficient rendering of anisotropic surfaces using computer graphics hardware. *Proc. of Image and Multi-dimensional Digital Signal Processing Workshop (IMDSP)*.
- KAJIYA, J. T., AND KAY, T. L. 1989. Rendering fur with three dimensional textures. In *Proc. of ACM SIGGRAPH*, J. Lane, Ed., 271–280.
- KIM, T.-Y., AND NEUMANN, U. 2000. A thin shell volume for modeling human hair. *Computer Animation*.
- KIM, T.-Y., AND NEUMANN, U. 2001. Opacity shadow maps. *Proc. of Eurographics Rendering Workshop*.
- KIM, T.-Y., AND NEUMANN, U. 2002. Interactive multiresolution hair modeling and editing. In *Proc. of ACM SIGGRAPH*, 620–629.
- KOH, C. K., AND HUANG, Z. 2000. Real-time animation of human hair modeled in strip. *Eurographics CAS Workshop*, 101–112.
- KOH, C. K., AND HUANG, Z. 2001. A simple physics model to animate human hair modeled in 2d strips in real time. *Proc. of Eurographics Workshop on Animation and Simulation*.
- KURIHARA, T., ANJYO, K., AND THALMANN, D. 1993. Hair animation with collision detection. In *Models and Techniques in Computer Animation*, Springer-Verlag, 128–38.
- LARSEN, E., GOTTSCHALK, S., LIN, M., AND MANOCHA, D. 2000. Distance queries with rectangular swept sphere volumes. *Proc. of IEEE Int. Conference on Robotics and Automation*.
- LEBLANC, A. M., TURNER, R., AND THALMANN, D. 1991. Rendering hair using pixel blending and shadow buffers. *The Journal of Visualization and Computer Animation*.
- LENGYEL, J. 2000. Real-time fur. *Proc. of Eurographics Workshop on Rendering*.
- LENGYEL, J., PRAUN, E., FINKELSTEIN, A., AND HOPPE, H. 2001. Real-time fur over arbitrary surfaces. *Proc. of ACM Symp. on Interactive 3D Graphics*.
- LUEBKE, D. 2001. A developer's survey of polygon simplification algorithms. *IEEE CG & A* (May), 24–35.
- MAGNENAT-THALMANN, N., HADAP, S., AND KALRA, P. 2000. State of the art in hair simulation. *Int. Workshop on Human Modeling and Animation*, 3–9.
- NVIDIA. 2001. Final fantasy technology demo 2001. <http://www.nvidia.com>.
- NVIDIA. 2002. <http://developer.nvidia.com/docs/lo/1451/SUPP/accuview/final.pdf>.
- O'BRIEN, D., FISHER, S., AND LIN, M. 2001. Simulation level of detail for automatic simplification of particle system dynamics. *Proc. of Computer Animation*, 210–219.
- PERBET, F., AND CANI, M. 2001. Animating prairies in real-time. *Proc. of ACM Symposium on Interactive 3D Graphics*.
- PLANTE, E., CANI, M., AND POULIN, P. 2001. A layered wisp model for simulating interactions inside long hair. *Proc. of Eurographics Workshop on Animation and Simulation*.
- SCHRÖDER, P., AND ZORIN, D. 1998. Subdivision for modeling and animation. In *ACM SIGGRAPH Course Notes*.
- WARD, K., LIN, M. C. AND MACRI, D. Collision Detection for Animating Hair Using Multiresolution Respresentations. Technical Report, University of North Carolina at Chapel Hill. January 2003.
- XU, Z., AND YANG, X. D. 2001. V-hairstudio: An interactive tool for hair design. *IEEE Computer Graphics and Applications* 21, 3, 36–43.
- YU, Y. 2001. Modeling realistic virtual hairstyles. *Pacific Graphics*.
- YU, Y. 2002. Personal Communication.
- ZHANG, H., MANOCHA, D., HUDSON, T., AND HOFF, K. 1997. Visibility culling using hierarchical occlusion maps. In *Proc. of ACM SIGGRAPH*, 77–88

# Adaptive Grouping and Subdivision for Simulating Hair Dynamics

Kelly Ward      Ming C. Lin  
Department of Computer Science  
University of North Carolina at Chapel Hill  
{wardk,lin}@cs.unc.edu  
<http://gamma.cs.unc.edu/HAIR>

## Abstract

We present a novel approach for adaptively grouping and subdividing hair using discrete level-of-detail (LOD) representations. The set of discrete LODs include hair strands, clusters and strips. Their dynamic behavior is controlled by a base skeleton. The base skeletons are subdivided and grouped to form clustering hierarchies using a quad-tree data structure during the precomputation. At run time, our algorithm traverses the hierarchy to create continuous LODs on the fly and chooses both the appropriate discrete and continuous hair LOD representations based on the motion, the visibility, and the viewing distance of the hair from the viewer. Our collision detection for hair represented by the proposed LODs relies on a family of “swept sphere volumes” for fast and accurate intersection computations. We also use an implicit integration method to achieve simulation stability while allowing us to take large time steps. Together, these approaches for hair simulation and collision detection offer the flexibility to balance between the overall performance and visual quality of the animated hair. Furthermore, our approach is capable of modeling various styles, lengths, and motion of hair.

## 1 Introduction

Modeling and animating human characters present some of the most difficult problems in computer graphics. Simulating natural hair movement is crucial to generating realistic appearances of animated characters. Real-time applications, such as virtual environments and game development, pose additional computational challenges for modeling the dynamics and mutual interactions of human hair at interactive rates. Such challenges primarily stem from the high number of hair strands required to model a human character and the resulting complexity of mutual interactions among many nonrigid hair strands. In addition to modeling virtual humans, the techniques used to simulate complex interactions among many deformable strands can also be ap-



**Figure 1. Dynamic Simulation of Long Hair Using Adaptive Grouping and Subdivision of Hair Strands, Hair Clusters, and Hair Strips.**

plied to modeling the motion of long animal hair and manes, brushes, bristles, strings, and other synthetic fibers.

**Main Contribution:** Herein we present a novel approach to adaptively subdivide and group hair modeled using discrete and continuous level-of-detail (LOD) representations. The set of discrete LOD representations include hair strands, clusters, and strips represented by subdivision curves, subdivision swept volumes, and subdivision patches, respectively [24]. Each representation has a base skeleton used to simulate the hair dynamics, which aids in switching between different LOD representations during the simulation [24]. To accelerate the simulation performance even more while achieving higher visual quality, we precompute a clustering hierarchy using the quad-tree data structure, which generates continuous LODs on the fly, based on the root location of each skeleton. At runtime, our algorithm selects both the appropriate continuous and discrete LOD representations based on the viewing distance, the motion and the visibility of the hair. Complementing the continuous LOD representations generated by the dynamic grouping and subdivision of hair, we use a collision detection al-

gorithm based on a family of swept sphere volumes and an implicit integration method to achieve greater stability while allowing the simulation to take larger time steps. The resulting method has the following characteristics:

- It can smoothly and dynamically switch between any hair LOD representations, as opposed to transitioning between discrete hair representations [24].
- It can incorporate any switching criteria (e.g. the viewing distance, the visibility, the motion of hair) to automatically group and subdivide hair representations and seamlessly control continuous LOD generation.
- It can balance between the visual quality and the performance of the overall dynamic simulation.
- It can be applied to simulate mutual hair interactions for various hairstyles of different thickness, weight, and strength.

We demonstrate our algorithm on several simulation scenarios, including hair braiding, hair shaking abruptly, hair brushing against a complex wrinkled torus, and hair blowing in the wind. We observed noticeable performance improvement with higher visual quality, as compared to earlier techniques, such as [24].

**Organization:** The rest of the paper is organized as follows. Sec. 2 sets forth a synopsis of related work. Sec. 3 describes the three discrete representations of hair upon which this work is based. We present our method on constructing clustering hierarchies for hair strands, hair clusters, and hair patches in Sec. 4. The collision detection method and the dynamics model for simulating hair movement and interactions are described in Sec. 5. The criteria for performing automatic subdivision and grouping of hair representations on the fly are outlined in Sec. 6. Sec. 7 describes the results of our prototype implementation and compares its performance against earlier techniques.

## 2 Related Work

In this section, we briefly survey prior research in hair modeling 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 the problem it presents [7, 11, 19, 26]. Some fundamental techniques have been presented to model the motion of individual hair strands in [1, 6, 14] in which each hair strand was represented as a series of connected line segments and the shape of the hair was determined by specifying the desired angles between segments. 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* [14]. 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 [23]. A similar approach has been used for interactive hairstyling [5, 25]. Another approach, as evidenced in [4], has been to use adaptive guide hairs to add more detail to overly interpolated regions.

None of these techniques however, can perform hair animation or rendering in real-time. Recently, a thin shell volume [9] and 2D strips [12, 13] 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 and hair that exploit graphics hardware were presented in [16, 17, 20]. The foregoing techniques, however, do not work well or are not applicable for rendering long, wavy or curly hair.

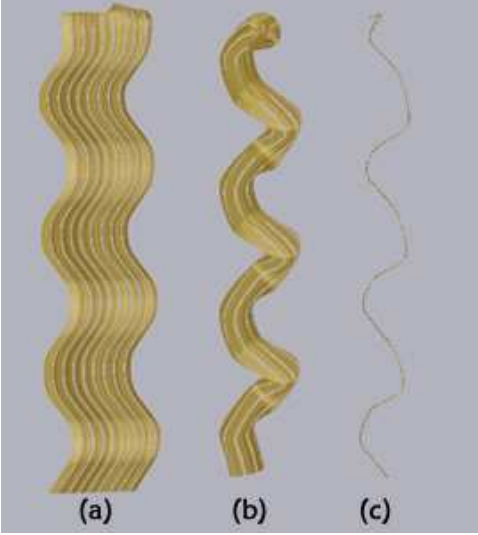
### 2.2 Simulation Levels of Detail

Model simplification has been an active research area and many algorithms have been proposed to accelerate graphical rendering of complex environments. Similar approaches have also been suggested to accelerate dynamic simulations for complex systems. A survey on geometric and simulation levels of detail can be found in [18] and [24], respectively. Our work bears some resemblance to the approach proposed by [21] for simplifying dynamics of particle systems using clustering. However, our approach for generating continuous LODs of hair modeled using a combination of hair strands, clusters and strips is quite different and the switching between different LODs is far more complex. Recently, the work of [3] has used the notion of continuous LODs for hair simulation in order to achieve faster simulation results. While our work employs similar techniques as [3], we use different splitting and grouping methods and couple the continuous LODs with discrete LODs to achieve faster rendering and simulation results.

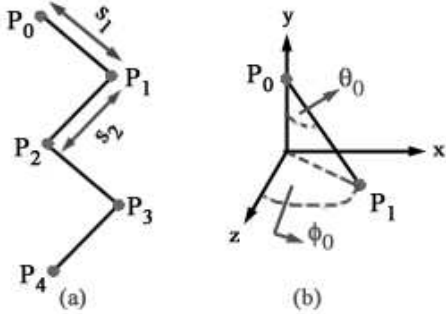
## 3 Preliminaries

Our approach is built upon the use of three discrete LOD representations for hair that utilize the subdivision framework and a base skeleton [24]. The three LOD representations are texture-mapped individual strands, clusters and strips. They are represented by subdivision curves, subdivision swept volume and subdivision patches, respectively. They are tessellated on the fly to further generate adaptive, continuous LODs for rendering only. Fig. 2 shows the rendered images of each LOD representation.

We model each skeleton as a series of rigid line segments connected by node points. Fig. 3 shows the layout of a skeleton. A skeleton contains  $n$  node points,



**Figure 2. Three Discrete LODs for Hair.** (a) Hair Strip (b) Hair Cluster (c) Hair Strand.



**Figure 3.** (a) The base skeleton model; (b) The parameters that define the style of hair.

$(p_0, p_1, \dots, p_{n-1})$  and  $n - 1$  rigid line segments between the points  $(s_1, s_2, \dots, s_{n-1})$ . Springs are used to control the angles between each line segment. The resting style of a skeleton is specified by assigning the desired rest angle positions to  $\theta_0$  and  $\phi_0$ . In this paper, we have extended the foregoing approach [24] by adaptively subdividing and grouping hair strands, clusters and strips, to generate continuous LODs for simulating complex hair motion and mutual interactions.

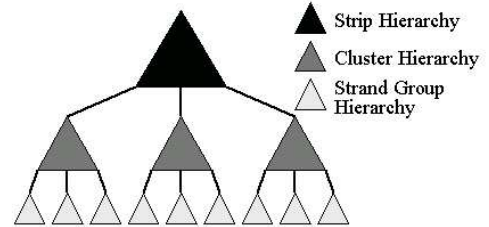
## 4 Construction of Hair Hierarchy

Our LOD algorithm uses the continual subdivision of strand groups, clusters, and strips in order to attain varying detail based on the current simulation state. The subdivision is performed as a pre-process and the information is stored for retrieval during runtime. As the subdivision of a hair group is performed, more skeletons are added to the system, creating a more detailed simulation. Our strand group hierarchy is built in a top-down manner creating smaller groups of strands until we reach the limit of a single strand in a

group. Likewise, the hierarchies of clusters are built from top to bottom, as are the strip hierarchies.

We couple the strand group hierarchies with the cluster and strip hierarchies to attain both discrete and continuous LOD representations. This hair representation hierarchy is illustrated in Fig. 4. Assumed to be given at the initial setup, the root strip in the strip hierarchy is the coarsest representation for an assemblage of hair. In order to gain more detail we traverse down the strip tree until we reach its leaves. For more detail, a finest LOD strip representation transforms into the coarsest LOD cluster representation, or the root cluster in the cluster hierarchy. Similarly, to attain even more detail we traverse down the cluster hierarchy until we reach the leaves of the cluster tree. At this point, the cluster representations are replaced by the top-level strand groups of the strand group hierarchies. To attain the finest detailed simulation possible, we traverse to the leaves of the strand group trees, which contain individual strands.

The next sections explain our subdivision process and hierarchy building mechanisms starting with the creation of strip and cluster hierarchies.



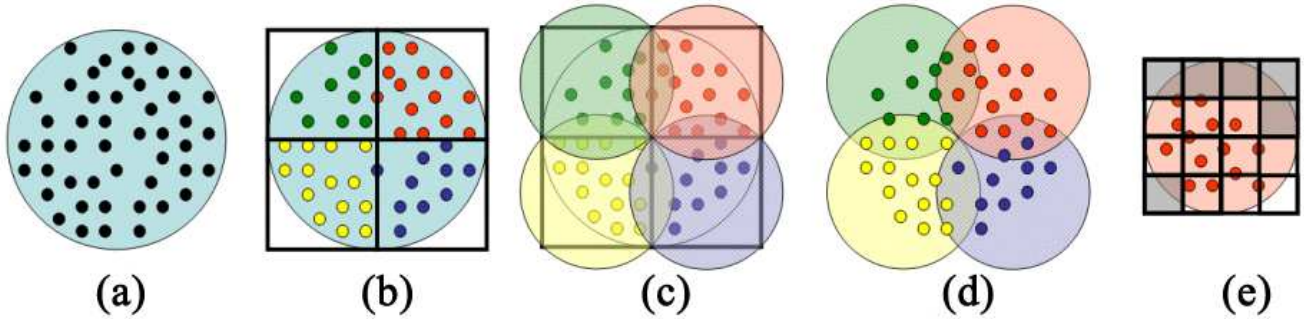
**Figure 4. Hair Hierarchy.** One hair hierarchy consists of a single strip hierarchy, multiple cluster hierarchies and multiple strand group hierarchies. The coarsest hair representations are located in the strip hierarchy at the top of the overall hair hierarchy. Note: The number of hierarchies, or children per node, within a hair hierarchy fluctuates.

### 4.1 Strip and Cluster Subdivision

Before we can build a hierarchy of strips or clusters, we must first create the initial top-level strip. A top-level strip is created by choosing a location on the scalp for the origin of the skeleton (the first node point of the skeleton). Next, a user-defined width is specified controlling the thickness of the strip. Hairstyle specific information is then declared, defining the length of the hair, the number of control points of the skeleton, and the desired curls or waves down the length of the skeleton.

Because the strip is a two-dimensional surface, we restrict its subdivision such that it may only be split into two equal parts. Strip subdivision is simply the degenerate case to cluster or strand group subdivision, using a degenerate quad-tree, or a binary tree, instead of the quad-tree data structure that is used for cluster and strand group hierar-





**Figure 5. Strand group subdivision.** *The subdivision process of a strand group into multiple strand groups. (a) The cross-section of a single strand group. (b) Strand group is divided into 4 equal quadrants and the strands are separated by the quadrant in which they lie (designated by different shades). (c) Circular cross-section is fit around each quadrant, or child, of original strand grouping. (d) Four new strand groups are created which are children of the original strand group. (e) Continual subdivision process is repeated on each child. Tinted squares show empty quadrants that contain no strands, these quadrants are set to null.*

chies. The subdivision ends once the width of the current strip is below a user-defined threshold.

For cluster subdivision, we start with a circular cross-section that defines the cluster. This circular cross-section is then split into four equal parts. The four sub-clusters have the same radius value but represent four different quadrants of the original cluster. The subdivision of a cluster always results in four children, so its information is held in a quad-tree. Clusters stop subdividing once their radius is below a user-defined threshold value. At this point, further detail is created in the strand group hierarchies.

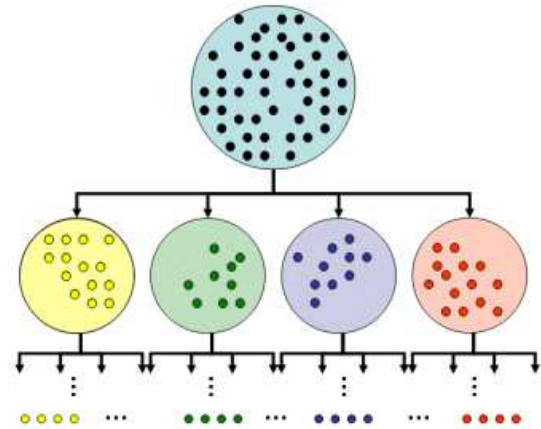
## 4.2 Strand Group Subdivision

A strand group is defined by a single skeleton, a radius to define the circular cross-section of the group, and the individual strands of hair for rendering purposes. A strand group cross-section is illustrated in Fig. 5a. The individual hair strands are randomly placed within the group and follow the dynamics of the skeleton. The circular shape of the strand groups is used for its simplicity in collision detection, explained in Sec. 5.

We use a quad-tree data structure to contain the hierarchy information. It follows therefore, that each strand group is split into four equal sections, as shown in Fig. 5b. The subdivision of a strand group into four sections creates the tightest fitting circular cross-section possible for each subgroup, as in Fig. 5c and Fig. 5d.

Once we have divided the strand group, we then calculate the number of strands in each quadrant. If a quadrant has no strands within its boundaries then the child associated with that quadrant is set to null (see Fig. 5e). A strand group will have between zero and four children. A strand group that contains only one strand will have zero children and becomes a leaf in the tree.

The final strand hierarchy is depicted in Fig. 6. Each node in the hierarchy contains a strand group, which in-



**Figure 6. Strand group hierarchy.** *Subdivision process creates a quad-tree containing strand group information. Strand group hierarchy can extend to individual strands.*

cludes its skeleton and the hair geometry used for the final rendering stage. In addition, each strand group, as well as each cluster and strip, holds its  $n - 1$  bounding volumes used for collision detection, where  $n$  is the number of nodes in the skeleton. The creation of these bounding volumes is described in Sec. 5.

Each skeleton contains the same number of control points as its parent hair group, which aids in dynamically switching between different levels, described in Sec. 6.

## 5 Collision Detection and Response

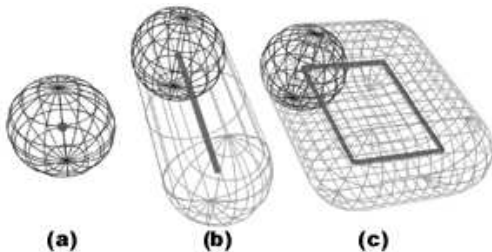
Collision detection and response is usually the most time consuming process for the overall simulation. We have separated our collision algorithm into two parts: *object-hair* collision detection, which occurs when hair interacts with an outside object like the head, and *hair-hair* collision detection, which involves hair mutual interactions.

## 5.1 Swept Sphere Volumes

Many techniques have been introduced for collision detection. Common practices have used bounding volumes (BVs) as a method to encapsulate a complex object within a simpler approximation of said object.

We have chosen the family of “swept sphere volumes” (SSVs) [15] to surround the hair. SSVs comprise a family of bounding volumes defined by a core skeleton grown outward by some offset. The set of core skeletons may include a point, line, or ngon. Fig. 7 shows examples of some SSVs. To calculate an SSV, let  $C$  denote the core skeleton and  $S$  be a sphere of radius  $r$ , the resulting SSV is defined as:

$$B = C \oplus S = \{c + r \mid c \in C, r \in S\}$$



**Figure 7. 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 shown as a bold line or point.

To detect an intersection between a pair of arbitrary SSVs we simply perform a distance test between their corresponding core skeletons and then subtract the appropriate offsets, i.e. the radius of each SSV.

## 5.2 Swept Sphere Volumes for Hair

We utilize the family of SSVs to encapsulate the hair because the shape of the SSVs closely matches the geometry of our hair representations. The SSVs that correspond to our three geometric representations for hair are line swept spheres (LSSs) for the strands and cluster levels, and rectangular swept spheres (RSSs) for the strip level. These SSVs can be used in combination to detect collisions between different representations of hair.

A skeleton containing  $n$  control points will contain  $n - 1$  SSVs. These SSVs correspond to the  $n - 1$  rigid line segments contained in the skeleton, see Fig. 3. The rigid line segment is used as the core skeleton of the LSS and the radius of the SSV is determined from the thickness of the hair representation.

## 5.3 Hair-Hair Interactions

Because hair is in constant contact with surrounding hair, interactions among hair are important to capture. The typical human head has thousands of hairs. Consequently, test-

ing the  $n - 1$  sections of each hair group against the remaining sections of hair would be too overwhelming for the simulation. Instead, we spatially decompose the area around the hair into three-dimensional grids and insert each SSV of the hair into the grids. Only SSVs that fall into the same grid are tested against each other. The average length of the rigid line segments of the skeletons is used as the height, width, and depth of each grid cell.

For each pair of SSVs that falls into the same grid cell, we determine the distance between their corresponding core skeletons,  $s1$  and  $s2$ . This distance,  $d$ , is subtracted from the sum of the radii of the two SSVs,  $r1$  and  $r2$ , to determine if there is an intersection. Let

$$overlap = d - (r1 + r2)$$

If *overlap* is positive then the sections of hair do not overlap and no response is calculated.

If there is an intersection, then we compute the cross product between the core skeletons,  $s1$  and  $s2$ , to determine the orientation of the skeletons in relation to each other. If  $s1$  and  $s2$  are near parallel, we set their corresponding velocities to the average of their initial velocities.

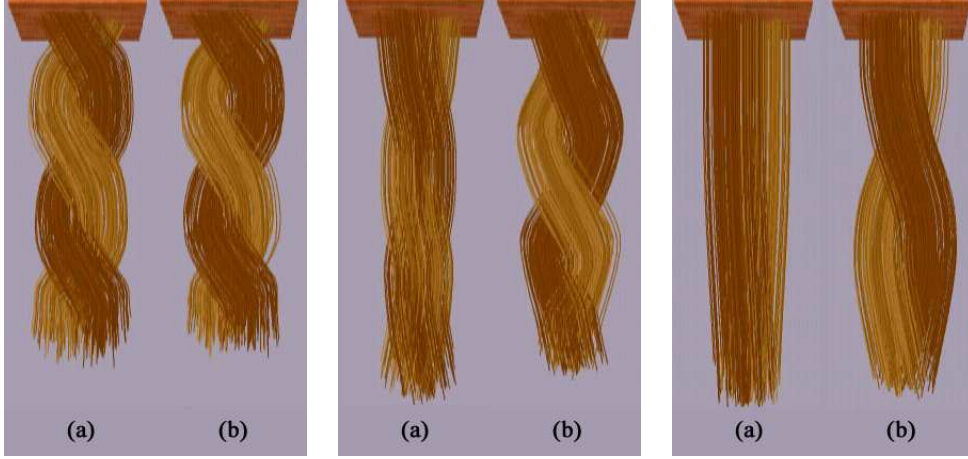
Intersecting hair sections that are not of similar orientation are pushed apart based on their amount of overlap. The direction to move each hair section is determined by calculating a vector from the closest point on  $s1$  to the closest point on  $s2$ . Each section is moved by half the overlap value and in opposite directions along the vector from  $s1$  to  $s2$ . Fig. 8 shows the effects of hair-hair interactions.

## 5.4 Hair-Object Interactions

Hair can interact with any object in the scene, such as the head or body of the character, where the object is a solid body that allows no penetration. Our hair-object collision detection algorithm begins by encapsulating the object with a bounding volume hierarchy (BVH) of SSVs that is pre-computed offline. A collision is detected between a section of hair and the object by recursively traversing the BVH testing the hair’s SSV against the bounding volumes in the hierarchy. If the hair is colliding with the object, the BVH will return the triangles in direct contact with the hair.

If a section of hair is colliding with the object, we adjust the position of the hair section so that it is outside of the object. We determine the amount by which to push the hair section by calculating the amount of penetration of the hair section into the object. We then push the skeleton in the direction normal to the object in the amount of the penetration. The section of hair is now no longer colliding with the object. In addition, the velocity of the section of hair interacting with the object is set to zero so that the hair is restricted to only move tangential to and away from, the object.

In the next time step, we know that the hair is still in



**Figure 8. Effects of Hair-Hair Collision Detection.** Side-by-side comparison with (RIGHT) and without (LEFT) hair-hair collision detection in a sequence of simulation snapshots.

close proximity to the object. If there is no intersection between the object and the hair we determine whether the hair is still within a certain distance threshold. If it is within this threshold, then the hair is still restricted so that its velocity in the direction of the object is zero. If it is not within this threshold, then the hair can move about freely.

When hair interacts with an object, a frictional force must be applied. We calculate a friction force by projecting the acceleration of the hair onto the plane tangential to the object at the point of contact. The result is the acceleration component that is tangent to the object. We apply the friction force in the opposite direction to oppose the motion of the hair. The magnitude of this force is based on the acceleration of the hair and the frictional coefficient,  $\mu_f$ , which is dependent upon the surface of the object, where  $0 < \mu_f < 1$ .

### 5.5 Overall Collision Checking Algorithm

During a single time step, a section of hair can have many interactions, some with other hairs and some with outside objects. The order in which we process these interactions is important as certain interactions must allow no penetration. Therefore, we process the hair-hair interactions first. The avoidance of hair-hair intersections is a soft constraint. Where possible, hair-hair intersections will be avoided, especially in the case of intersecting hair sections of different orientations. In contrast, the avoidance of hair-object intersections is a hard constraint. At the end of the time step, there will be no intersections between the hair and an outside object.

### 5.6 Implicit Integration for Dynamic Simulation

After our system computes appropriate collision responses, the dynamic simulation proceeds. We follow the basic dynamics model for simulating hair that was first pro-

posed by [1, 14]. We extend this method by using an implicit integration technique, in order to achieve greater stability while allowing us to take larger time steps throughout the simulation. This approach is similar to cloth simulations that use implicit integration for greater stability [2].

In this approach, each control point of a hair skeleton is governed by the set of ordinary differential equations:

$$I_i \frac{d^2 \theta_i}{dt^2} + \gamma_i \frac{d\theta_i}{dt} = M_{\theta_i},$$

$$I_i \frac{d^2 \phi_i}{dt^2} + \gamma_i \frac{d\phi_i}{dt} = M_{\phi_i}.$$

where  $I_i$  is the moment of inertia for the  $i$ th control point of the skeleton,  $\gamma_i$  is the damping coefficient, and  $M_{\theta}$  and  $M_{\phi}$  are the  $\theta$  and  $\phi$  torque components, respectively.  $M_{\theta}$  and  $M_{\phi}$  are computed from the spring forces controlling the style of the hair and external forces such as wind. The resultant  $M_{\theta}$  and  $M_{\phi}$  become:

$$M_{\theta} = M_{\theta_{spring}} + M_{\theta_{external}},$$

$$M_{\phi} = M_{\phi_{spring}} + M_{\phi_{external}}.$$

The torques due to the spring forces are calculated by:

$$M_{\theta} = -k_{\theta}(\theta_i - \theta_{i0}),$$

$$M_{\phi} = -k_{\phi}(\phi_i - \phi_{i0}),$$

where  $k_{\theta}$  and  $k_{\phi}$  are the spring constants for  $\theta$  and  $\phi$ , respectively. Furthermore,  $\theta_0$  and  $\phi_0$  are the specified rest angles and  $\theta$  and  $\phi$  are the current angle values. Although explicit methods such as Euler or fourth-order Runge-Kutter can be used for this integration, we choose implicit integration for greater stability of simulations. Appendix A shows the derivation of our implicit integration equations using polar coordinates. Because we are working with polar coordinates, we will use angular positions,  $\theta$  and  $\phi$ , and angular velocities,  $\omega_{\theta}$  and  $\omega_{\phi}$ .

The change in angular velocity for the  $\theta$ -component of a skeleton node point,  $\Delta\omega_\theta$ , becomes

$$\Delta\omega_\theta = \frac{-hk_\theta(\theta - \theta_0) - h^2k_\theta\omega_{\theta 0}}{1 + h^2k_\theta}$$

where  $h$  is the time step, and  $\omega_{\theta 0} = \omega_\theta(t_0)$  is the angular velocity at time  $t_0$ . Here,  $\Delta\omega_\theta = \omega_\theta(t_0+h) - \omega_\theta(t_0)$ . Once we have calculated  $\Delta\omega_\theta$ , we calculate the change in angular position  $\Delta\theta$  from  $\Delta\theta = h(\omega_0 + \Delta\omega)$ . The same process can be applied to the  $\phi$ -component of the angular position and angular velocity for each control point of a skeleton.

Implicit integration allows us to use stiffer springs when warranted, for example, when simulating the bristles of a brush which have different spring constants than the hair on a human head. Using stiff springs with explicit integration on the other hand, requires much smaller time steps to ensure a stable simulation.

## 6 Runtime Selection

Our hair hierarchies allow us to choose appropriate discrete and continuous LOD representations for the hair dynamically during the simulation. We simply traverse the hierarchy selecting the desired hair assemblage. As we move to a different level in the hair hierarchy we are either dividing a hair group into multiple groups or combining several groups into one larger group of hair. The base skeleton makes these transitions smooth and straightforward. Because each hair representation uses the same dynamic skeleton, we generalize the transitioning algorithm so that it can be applied at any location in the hierarchy.

### 6.1 Criteria for Grouping and Subdividing

The current assemblage of hair is determined by the viewing distance from the viewer to the hair, the motion of the hair, and the visibility of the hair. We use a technique similar to that described by [24]. This method first tests the hair’s visibility by determining if it is outside of the field of view of the camera or if it is occluded by the body. If a section of hair is not visible, it is simulated using the coarsest representation, the root strip of the hierarchy, and it is not rendered.

The next criterion for choosing the hair’s representation is the viewing distance. As the distance from the viewer to the hair increases we move up the hair hierarchy, simulating and rendering the hair with less detail. Meanwhile, as the velocity of the hair increases, we move down the hierarchy, simulating and rendering the hair with finer detail. Each level in the hierarchy has predetermined intervals for its appropriate viewing distances and velocities. These predetermined values are set by defining the intervals for the coarsest LOD in the hierarchy and for the finest LOD in the hierarchy. The remaining values for the rest of the levels are

linearly interpolated from the start and end values to create a smooth progression of distance and velocity thresholds.

If a section of hair is not occluded, then its distance and velocity are compared against the current level’s thresholds. The current level is chosen based on whichever test requires more detail. In a given time step, a section of hair only moves one level in the hierarchy in order to avoid visual distractions, unless a transition is triggered by occlusion.

A transition caused by an occlusion permits the hair representation to transform into the coarsest strip representation regardless of its current location in the hair hierarchy. When the hair is no longer occluded, it transforms into the LOD that is appropriate given the hair’s current velocity and distance values.

### 6.2 Adaptive Subdivision

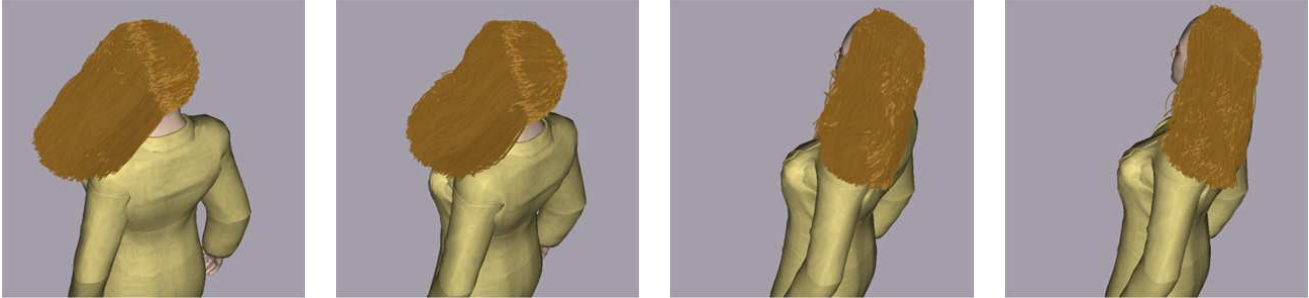
Using our precomputed hierarchy, we divide a group of hair into multiple groups by moving a level down the hierarchy. This becomes a simple process through the use of the base skeleton. As explained in Sec. 4, each hair group’s skeleton has the same number of control points as its parent skeleton. Furthermore, all of the style properties are the same from parent to child. Accordingly, when a transition to a hair group’s children occurs, the child skeletons inherit the dynamic state of their parent skeleton. Each control point in a child skeleton corresponds to a control point in its parent skeleton. When the child groups are created from the parent group, the offset of each child from the parent is stored. When we switch to the children, these offsets are used to position the children accordingly.

Fig. 9 shows two skeletons dynamically subdivide into multiple skeletons as a gust of wind blows through the hair.

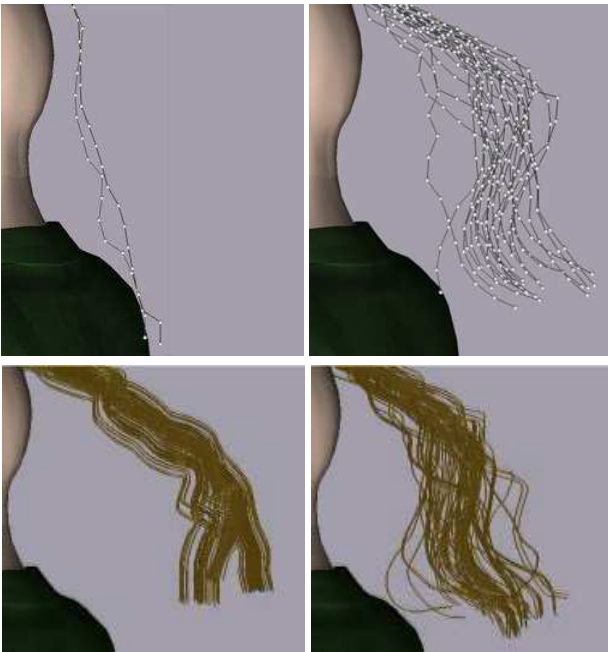
### 6.3 Adaptive Merging

Merging multiple child skeletons back into their parent skeleton is, again, rather straightforward. Our method averages the dynamic states of the children, including position and velocity values, and assigns this average to the parent skeleton.

In order to alleviate visual artifacts that can appear by merging children into a parent skeleton, a transition may only occur if all of the children are ready to transition back into the parent; that is, the criteria explained in Sec. 6.1 for switching levels are satisfied for all of the children. Furthermore, in order to avoid a sudden jump in the position of the hair, we impose a positional constraint on the children. After we have averaged the control point positions of the child skeletons, we determine the distance of the child control points from their corresponding parent control point. If this distance is greater than a certain threshold, the transition will not occur. If the distance is less than the threshold but not in exact position, a spring force is used to subtly pull the children into place so a smooth transition may occur.



**Figure 10. Dynamic Simulation of Hair Using LODs.** A sequence of snapshots (from left to right).



**Figure 9.** TOP: Two skeletons (LEFT) are dynamically subdivided into multiple (RIGHT). BOTTOM: The rendered images without (LEFT) and with (RIGHT) adaptive subdivision.

## 7 Results and Comparisons

We have implemented our LOD hair simulation algorithm in C++. We modified and extended the publicly available proximity query package (PQP) [15], to perform collision detections. The simulation results are displayed using OpenGL.

### 7.1 Rendering

Our rendering system follows that described in [24] for LOD hair representations. This approach uses the shading model suggested by [8] and opacity shadow maps introduced by [10]. The use of motion blurring and image blending helped to alleviate the visual artifacts associated with LOD transitions.

### 7.2 Performance Comparisons

We have tested our implementation on various scenarios. Please visit our project website:

<http://gamma.cs.unc.edu/HAIR>

for MPEGs of these simulation runs and for snapshots taken from hair simulations using our LOD representations.

We also compared the performance for the overall dynamic simulation (not including collision detection) and collision detection using different representations on various simulation scenarios. Table 1 gives a detailed comparison of the *average* running times using a combination of discrete and continuous LOD representations (indicated as LODs) against (i) the use of the finest hair representations, or the lowest possible level in the hair hierarchies (indicated as Fine Strands in the table), and (ii) the coarsest strand representations, or highest level within the strand hierarchies (indicated as Coarse Strands in the table). This simulation entails wind blowing through the hair as the camera remains stationary. The camera is positioned close to the figure, so the viewer can see fine detail, and primarily shows the effects of the continuous LODs used within the strand hierarchy. Our method allows us to simulate strands with visual detail comparable to that of the finest strand representation, whereas the timings for the simulation is comparable to that of the coarsest strand representation.

Table 2 shows results of the same simulation as the camera zooms away from the figure. With this simulation the influence of the discrete LODs is obvious. The use of clusters and strips with adaptive grouping and subdivision increases the performance of the simulation with little visual loss. The table shows comparisons of the same simulation with the finest detailed strands, versus the coarsest detailed clusters and coarsest detailed strips.

For this benchmark, we used 9,350 individual strands. At the finest detail in the hierarchy, these strands were simulated with 3,570 skeletons, averaging 2.6 strands per skeleton. The algorithm does not allow all of the hierarchies to extend to each individual strand due to the overwhelming computational cost entailed. Rather, some hierarchies extend to the individual strand level, while oth-



ers contain a minimum of four or five strands at the lowest level. This combination, automatically generated by our approach, enables the simulation to distribute the computational resources where they are needed the most. Hair strands that originate at the top of the head, near the part of the hair, are more viewable and will be allowed to extend to the individual strand level, whereas hairs located at the base of the neck are typically not as viewable and do not require a hierarchy reaching as far. These 9,350 strands are then represented with 110 strips, at the coarsest level, or 330 clusters at the coarsest level in the cluster hierarchy. The skeletons comprising the hairstyle consist of 6 control points on average. Timings were taken on a PC equipped with an Intel(R) Pentium(R) 4 2-GHz processor, 1 GB main memory and GeForce(R) 4 graphics card.

Breakdown	Fine Strands	LODs	Coarse Strands
Dyn Sim	0.107636	0.041624	0.038271
Col Detect	7.642328	0.411793	0.338298
Total	7.749964	0.453417	0.376569

**Table 1. Performance Comparison.** *Simulation for a stationary camera. The average performance numbers are measured in seconds per frame.*

Breakdown	LODs	Strands	Clusters	Strips
Dyn Sim	0.026142	0.107636	0.015374	0.003242
Col Detect	0.239489	7.642328	0.171781	0.020142
Total	0.265631	7.749964	0.187155	0.023384

**Table 2. Performance Comparison.** *Simulation for a camera zooming out. The average performance numbers are measured in seconds per frame.*

### 7.3 Analysis and Discussion

The impetus of this research is to explore the use of dynamic grouping and subdivision of hair to automatically generate continuous LODs for hair simulation. This approach enables us to further increase the visual fidelity while maintaining an interactive dynamic simulation. It is difficult to meaningfully quantify the computational errors introduced by the use of simplified representations for modeling hair. Notwithstanding the foregoing, we can subjectively evaluate the resulting simulation by performing comparisons on the visual quality of the simulated results. Using side-by-side comparisons as shown in the supplementary video, we notice higher visual fidelity of the simulated hair using continuous LODs. The performance of our framework varies depending on the scenarios. In general, its overall performance in simulation and rendering compares favorably against the use of discrete LOD representations [24].

**Limitations:** As with most LOD algorithms that generate

hierarchical representations offline, our approach necessitates considerable memory requirements.

There are other application-dependent transition criteria, such as collision, that we have not examined closely but which can improve the system’s overall performance.

### 7.4 Comparisons Against Earlier Approaches

This research is built upon the discrete LOD representations introduced in our recent work [24]. In [24], we discussed the benefit of using discrete LOD representations that compared favorably against earlier approaches [4, 5, 11, 12, 13, 14, 16, 17, 20, 23, 25], as the use of LODs can achieve both high visual quality and interactive dynamic simulation at the same time.

Compared to [24], our current approach allows for higher quality visual appearances while maintaining similar or better runtime performances. Using continuous and discrete LODs for hair simulation enable the user to maintain complete control over the visual and performance results of the system.

## 8 Summary and Future Work

In this paper, we present an approach to adaptively split and group collections of hair to generate continuous LODs for accelerating the dynamics computation of hair while achieving higher visual fidelity. In addition to potential areas of improvements mentioned in the earlier sections, there are several possible directions to extend this research:

- Dynamically change the hairstyle, as the user combs or brushes the hair with a 3D user (e.g. haptic) interface;
- Interactively model the dynamics of the hair in the presence of other substances, such as styling gel, hair spray, water, etc.;
- Automatically generate desired simulation outcomes, given high-level user control.

### Acknowledgements

This research is supported in part by Army Research Office, Intel Corporation, National Science Foundation, and Office of Naval Research.

### References

- [1] K. Anjyo, Y. Usami, and T. Kurihara. A simple method for extracting the natural beauty of hair. *Computer Graphics*, 26(2):111–120, 1992.
- [2] D. Baraff and A. Witkin. Large steps in cloth simulation. *Proc. of ACM SIGGRAPH*, pages 43–54, 1998.
- [3] F. Bertails, T.-Y. Kim, M.-P. Cani, and U. Neumann. Adaptive Wisp Tree - a multiresolution control structure for simulating dynamic clustering in hair motion. *Proc. of ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, 2003.

- [4] J. Chang, J. Jin, and Y. Yu. A practical model for hair mutual interactions. *Proc. of ACM Symposium on Computer Animation*, 2002.
- [5] 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.
- [6] 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.
- [7] S. Hadap and N. Magnenat-Thalmann. Modeling dynamic hair as a continuum. *Computer Graphics Forum (Proc. of Eurographics 2001)*, 20(3), 2001.
- [8] J. T. Kajiya and T. L. Kay. Rendering fur with three dimensional textures. In J. Lane, editor, *Computer Graphics (SIGGRAPH '89 Proceedings)*, volume 23, pages 271–280, July 1989.
- [9] T.-Y. Kim and U. Neumann. A thin shell volume for modeling human hair. *Computer Animation*, 2000.
- [10] T.-Y. Kim and U. Neumann. Opacity shadow maps. *Proc. of Eurographics Rendering Workshop*, 2001.
- [11] T.-Y. Kim and U. Neumann. Interactive multiresolution hair modeling and editing. *Proc. of SIGGRAPH*, 2002.
- [12] C. K. Koh and Z. Huang. Real-time animation of human hair modeled in strip. *Eurographics CAS Workshop*, pages 101–112, 2000.
- [13] 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.
- [14] 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.
- [15] 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.
- [16] J. Lengyel. Real-time fur. *Proc. of Eurographics Workshop on Rendering*, 2000.
- [17] J. Lengyel, E. Praun, A. Finkelstein, and H. Hoppe. Real-time fur over arbitrary surfaces. *Proc. of ACM Symp. on Interactive 3D Graphics*, 2001.
- [18] D. Luebke, M. Reddy, J. Cohen, A. Varshney, B. Watson, and R. Huebner. *Level of Detail for 3D Graphics*. Morgan-Kaufmann, 2002.
- [19] N. Magnenat-Thalmann, S. Hadap, and P. Kalra. State of the art in hair simulation. *Int. Workshop on Human Modeling and Animation*, pages 3–9, 2000.
- [20] NVIDIA. Final fantasy technology demo 2001. <http://www.nvidia.com>, 2001.
- [21] 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.
- [22] E. Plante, M. Cani, and P. Poulin. Capturing the complexity of hair motion. *GMOD number 1 volume 64*, January 2002.
- [23] 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.
- [24] K. Ward, M. Lin, J. Lee, S. Fisher, and D. Macri. Modeling hair using level-of-detail representations. *Proc. of Computer Animation and Social Agents*, 2003.
- [25] Z. Xu and X. D. Yang. V-hairstudio: An interactive tool for hair design. *IEEE Computer Graphics and Applications*, 21(3):36–43, 2001.
- [26] Y. Yu. Modeling realistic virtual hairstyles. *Pacific Graphics*, 2001.

## Appendix A: Implicit Integration

We use implicit integration for the dynamic simulation of hair, as explained in Section 5.6. Here we show the derivation of equations for our implicit integration formulation. We will first show how this works with the  $\theta$ -component. Since we are working with polar coordinates, we will denote the angular position  $\theta$ , angular velocity  $\omega$ , and angular acceleration  $\alpha$ .

We start with the second-order differential equation:

$$\ddot{\theta}(t) = f(\theta(t), \dot{\theta}(t)) = -k_{\theta}(\theta_i - \theta_{i0}).$$

We can rewrite this as a first-order differential equation by substituting the variables  $\alpha = \dot{\theta}$  and  $\omega = \theta$ . The resulting set of first-order differential equations is

$$\frac{d}{dt} \begin{pmatrix} \theta \\ \dot{\theta} \end{pmatrix} = \frac{d}{dt} \begin{pmatrix} \theta \\ \omega \end{pmatrix} = \begin{pmatrix} \omega \\ f(\theta, \omega) \end{pmatrix}$$

We get the following formulations for  $\Delta\theta$  and  $\Delta\omega$  when using the explicit forward Euler method, where  $\Delta\theta = \theta(t_0 + h) - \theta(t_0)$  and  $\Delta\omega = \omega(t_0 + h) - \omega(t_0)$  and  $h$  is the time step value

$$\begin{pmatrix} \Delta\theta \\ \Delta\omega \end{pmatrix} = h \begin{pmatrix} \omega_0 \\ -k_{\theta}(\theta - \theta_0) \end{pmatrix}$$

Instead, we are going to take an implicit step which is often thought of as taking a backwards Euler step since we are evaluating  $f(\theta, \omega)$  at the point we are aiming for rather than at the point we were just at. In this case, the set of differential equations changes to the form

$$\begin{pmatrix} \Delta\theta \\ \Delta\omega \end{pmatrix} = h \begin{pmatrix} \omega_0 + \Delta\omega \\ f(\theta_0 + \Delta\theta, \omega_0 + \Delta\omega) \end{pmatrix}$$

A Taylor series expansion is applied to  $f$  to obtain the first-order approximation,

$$f(\theta_0 + \Delta\theta, \omega_0 + \Delta\omega) \approx f_0 + \frac{\partial f}{\partial \theta} \Delta\theta + \frac{\partial f}{\partial \omega} \Delta\omega$$

$$\approx -k_{\theta}(\theta - \theta_0) - k_{\theta} \Delta\theta + 0(\Delta\omega) \approx -k_{\theta}(\theta - \theta_0) - k_{\theta} \Delta\theta$$

After we substitute the approximation of  $f$  back into the differential equation we get

$$\begin{pmatrix} \Delta\theta \\ \Delta\omega \end{pmatrix} = h \begin{pmatrix} \omega_0 + \Delta\omega \\ -k_{\theta}(\theta - \theta_0) - k_{\theta} \Delta\theta \end{pmatrix}$$

We can focus on the angular velocity  $\Delta\omega$  alone and substitute  $\Delta\theta = h(\omega_0 + \Delta\omega)$ . We get

$$\Delta\omega = h(-k_{\theta}(\theta - \theta_0) - k_{\theta}h(\omega_0 + \Delta\omega))$$

Rearranging this equation gives us

$$(1 + k_{\theta}h^2)\Delta\omega = -hk_{\theta}(\theta - \theta_0) - k_{\theta}h^2\omega_0$$

$$\Delta\omega = \frac{-hk_{\theta}(\theta - \theta_0) - h^2k_{\theta}\omega_0}{1 + h^2k_{\theta}}$$

Once we have calculated the change in angular acceleration,  $\Delta\omega$ , we can calculate the change in angular position  $\Delta\theta$  trivially from  $\Delta\theta = h(\omega_0 + \Delta\omega)$ . The same process can be applied to the  $\phi$ -component of the angular position and angular velocity for each control point of a strand.

# Adaptive Wisp Tree - a multiresolution control structure for simulating dynamic clustering in hair motion

F. Bertails<sup>1</sup>, T-Y. Kim<sup>2†</sup>, M-P. Cani<sup>1</sup> and U. Neumann<sup>2</sup>

<sup>1</sup> GRAVIR-IMAG, joint lab of CNRS, INRIA, INPG and UJF

<sup>2</sup> Integrated Media System Center, University of Southern California

---

## Abstract

*Realistic animation of long human hair is difficult due to the number of hair strands and to the complexity of their interactions. Existing methods remain limited to smooth, uniform, and relatively simple hair motion. We present a powerful adaptive approach to modeling dynamic clustering behavior that characterizes complex long-hair motion. The Adaptive Wisp Tree (AWT) is a novel control structure that approximates the large-scale coherent motion of hair clusters as well as small-scaled variation of individual hair strands. The AWT also aids computation efficiency by identifying regions where visible hair motions are likely to occur. The AWT is coupled with a multiresolution geometry used to define the initial hair model. This combined system produces stable animations that exhibit the natural effects of clustering and mutual hair interaction. Our results show that the method is applicable to a wide variety of hair styles.*

Categories and Subject Descriptors (according to ACM CCS): I.3.7 [Computer Graphics]: Animation

---



**Figure 1:** Comparisons between real hair and simulation results.

## 1. Introduction

Despite recent advances in the animation of virtual humans, simulating the complex motion of long hair remains an area of active research. In contrast to short hair and fur that has

been successfully used in feature films such as *Stuart Little*<sup>2</sup>, *Monsters Inc.*<sup>9</sup>, simulating reasonably long, human-like hair poses additional challenges since both hair interaction with the character's body and mutual hair interaction need to be modeled. Without these interactions, the resulting hair animation would lose realism.

This paper describes an effective adaptive method that models the dynamic clustering behavior often observed in long hair motion (see Figure 1). To model such splitting and merging behavior, we employ multiresolution representations for both modeling (geometric level of detail) and animation (control level of detail), where the latter is extracted from the former. The adaptive wisp tree (AWT), a control structure built upon the multiresolution model, approximates the large-scale coherent motion of clusters as well as small-scale variations in hair motion. During animation, the AWT automatically adapts itself based on local wisp fusion and separation. It also helps to focus computations where visible hair motions are likely to occur. The time-varying AWT segments control the motion of underlying multiresolution hair model, including the individual hair strands used at the rendering stage. The resulting animations are efficiently computed and stable.

---

<sup>†</sup> Currently working at Rhythm & Hues Studio.

## 1.1. Previous work

This section focuses on animation methods that can be used for animating long hair<sup>†</sup>. Hair modeling and rendering are not addressed. A complete state of the art can be found elsewhere<sup>20, 23, 4, 17</sup>.

Most methods for animating hair rely on a physically-based dynamics model. A brute-force dynamics simulation would consume unacceptable computational resources, due to the large number of hair strands ( $\sim 100,000$ ) and the complexity of their interactions. Early work on hair animation does not consider the problem of mutual hair interaction and mostly focuses on the motion of individual hair strand<sup>24, 1, 26</sup>. For efficient processing of mutual hair interactions, an auxiliary mechanism is often employed. Examples include the cloth-like patches used for the production of Final Fantasy<sup>14</sup> and a layered shell to add volume around the head model<sup>19</sup>, but these are limited to modeling smooth lateral interactions (as if hairs were layers of patches), rather than the realistic complex motions and interactions we seek to model.

Hadap and Magnenat-Thalmann propose a novel paradigm for handling mutual hair interaction by considering hair as a continuum and coupling a fluid dynamics model with an elaborate model for stiff hair dynamics<sup>12</sup>. The smoothed particle approach is used to model both hair-hair and hair-air interactions. The method gives high quality results for straight, smooth hair motions and interactions. However, it is computationally expensive since every hair strand needs to be processed, and results do not illustrate the dynamic clustering effects observed in real hair motion. Our goal is to mimic such complex dynamic clustering behavior, which we find essential when dealing with most hair styles.

For efficient computation of hair motion, neighboring hairs are often grouped and approximated by compacter representations. Among the two main approaches to hair grouping, the first approach uses *guide hair* or *key hair* to model a sparse set of hair strands, from which dense hair is interpolated at the rendering stage<sup>6, 4</sup>. These methods are useful for modeling and animating smooth or short hairstyles and animal fur<sup>2, 9</sup>. The second approach uses *wisps* to group nearby hairs into representative geometric primitives (hair wisps), inside which individual hair strands are rendered. This approach is employed as a few large wisps (e.g., in Shrek<sup>8</sup>), or surface wisps<sup>18</sup>, or volumetric wisps<sup>27, 23</sup>, that are deformable or rigid.

In the efforts to expedite hair animation by reducing the number of animated primitives, processing mutual hair interaction remains a challenge. Chang et al.<sup>4</sup> models the hair medium as a continuum, approximated by polygonal strips connecting the guide hairs. Limiting the interactions to only those between the polygons and guide hairs greatly reduces the computation needed for modeling mutual hair interac-

tion. Additional guide hairs are adaptively inserted to ensure that guide hairs remain evenly distributed during hair motion. However, the underlying continuum assumption limits the method to relatively simple and smooth motion, since general hair motion does not maintain the strand-neighbor relationships assumed by guide hair approaches.

The geometry and control provided in the wisp approaches can model the complex hair geometry and clustering effects of long, thick hair. Plante et al.<sup>23</sup> models hair interaction by allowing individual wisp to deform. While each wisp models the coherent motion of neighboring hair strands, the interaction between wisps are handled with an anisotropic collision model- i.e., damped collisions occur when two wisps of different orientations collide, while wisps with similar orientation are allowed to interpenetrate with a high friction coefficient. The method proves useful for simulating complex clustering effects in hair motion, but remains inefficient (3 hours for a few seconds of animation) due to the complex shape of hair wisps and the number of penetration tests. More importantly, this approach can be unstable at the rest state due to the high number of mutual interactions between wisps. Our mechanism for processing interactions is related to this method, but our adaptive approach improves upon both efficiency and stability of the animations.

The use of adaptive level of detail is a well known way to improve efficiency during simulations. Previous adaptive deformable models such as<sup>10</sup> were devoted to the simulation of continuum medium. Chang et al.'s approach is also based on a continuum assumption as interpolation is used to add extra guide strands. Our claim is that hair *is not a continuous medium*. Strong discontinuities in geometry can be observed during motion, especially at the tip of hairs. A multiresolution hair representation was proposed to model such characteristics of hair geometry<sup>17</sup>, which proved effective for modeling a variety of complex hairstyles. Yet this work was not exploited for hair animation. Very recently, Ward et al.<sup>25</sup> proposed a novel approach for modeling hair using three geometric levels of detail : hair strips, hair clusters and individual hair strands. Transitions between these LODs are dynamically generated during motion, according to criteria such as camera position. This method yields an increased efficiency, especially at the rendering stage. Although promising, the results are shown only for gentle and smooth motions such as hair floating in the wind. In contrast, our approach focuses on the animation level of details, i.e. dynamically adapting the control structure for more complex hair motion. Our criteria for driving transitions between animation is different from the work of Ward et al. in that our method is driven by the local complexity of hair motion instead of the importance of the object on the screen for rendering<sup>25</sup>. Therefore, our approach is independent of any adaptation of such rendering LOD and could be coupled with Ward's method to increase the rendering efficiency as well.

<sup>†</sup> In the remainder of this paper, the term "hair" will be used for human-like hair as opposed to fur. We believe that such hair poses unsolved problems due to the complexity of hair interactions.

## 1.2. Overview

Our goal is to simulate the dynamic clustering behavior of hair motion and the mutual hair interactions. We strive for both efficiency and stability. The key idea is to dynamically adapt the local resolution of the control structure during animation, thereby concentrating computations where it is most needed, and reducing the number of interacting elements to maintain stability.

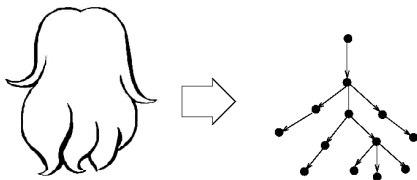
We employ a multiresolution hair wisp representation similar to that used for hair modeling by Kim and Neumann<sup>17</sup>. At each time step, interactions between active hair wisps are detected and processed, similarly to Plante et al.'s work<sup>23</sup>. The contributions of our method are :

- Adaptive wisp tree, a novel adaptive control structure for hair animation that simulates dynamic splitting and merging of hair clusters;
- Rules for detecting areas where refinement and merging should occur;
- Improving efficiency and stability of interaction processing in wisp-based approaches.

Sections 2 to 4 respectively describe these contributions. The last two sections discuss implementation and results before concluding.

## 2. Adaptive wisp tree

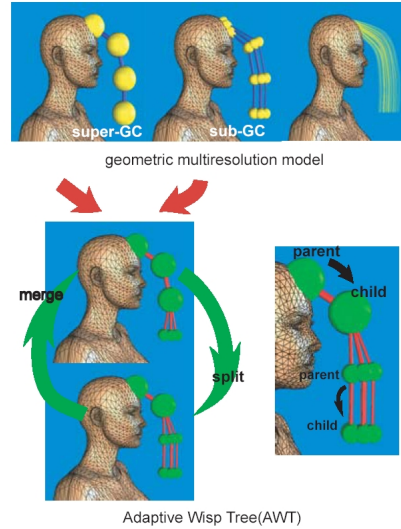
During real hair motion, clusters form and split due to frictions, static charges, and the features of the initial hairstyles. The number, location and size of the observed clusters vary over time. The adaptive wisp tree (AWT) is motivated by the fact that the clustering behaviors are observed moving from hair tips to roots, as if hair was progressively cut by a slider. We attribute the phenomena to the excessive collision between hairs near the scalp and to the fact that hair motions are constrained at the root, while hair is free to move apart at the end. Physical simulation of such behavior would require considerable amount of computation and pose stability issues in numerical integration. We rather aim at a simple data structure that visually mimics the clustering behavior, leading to a tree-like control structure that approximates hair cluster splitting and merging. See Figure 2.



**Figure 2:** The AWT is an acyclic oriented graph of wisp segments where nodes approximate the mass of nearby hairs and edges represent the control links between nodes.

The geometric complexity and motion complexity are not always correlated : for instance, curly hair modeled as a large number of spiral-looking small wisps may move as coherently as a single large hair cluster for slight head movement,

whereas seemingly smooth hair wisps often suddenly split into small clusters under drastic head motion. To model this behavior, we employ a multiresolution hair model to preserve the geometric detail, coupled with the AWT that controls the dynamic evolution of hair motion. The multiresolution geometric clusters are also used to guide potential splitting in a particular hairstyle during subsequent motion processing.



**Figure 3:** Relationship between geometric multiresolution model and the AWT

### 2.1. Multiresolution hair geometry

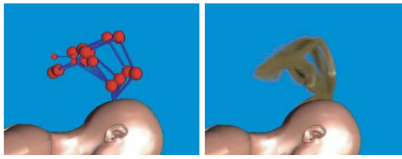
We represent a hair model with a hierarchy of generalized cylinders (GC), adapting from<sup>17</sup>. A hairstyle is progressively constructed by creating GCs and refining them into smaller ones. In the remainder of the paper, we use the terms *super-GC*, *sub-GC* when we refer to the modeling hierarchy. We reserve *parent* and *child* notations for denoting relationship between control links in the AWT (Figure 3).

The following modifications are made to<sup>17</sup> :

- The number of segments in sub-GCs should be multiples of that of the super-GC so that corresponding nodes can be easily identified.
- GCs have only circular cross-sections. Complex shapes emerge during animation as the result of splitting procedure. The condition greatly simplifies collision processing since a single radius per cross-section suffices, in contrast to complex deformable boundary used by<sup>23</sup>.
- The framework of<sup>17</sup> does not enforce that the boundary of a super-GC encloses all its sub-GCs. The boundary of a super-GC is modified to fit the shape of its sub-GCs, in a post-processing step. Given a set of sub-GCs, we first compute the center-of-mass curve, and then modify the GC contours around this curve - a mere radius change in our case.



## 2.2. Multiresolution mechanical structure



**Figure 4:** Left image illustrates the AWT. Active nodes are drawn as spheres and segments are drawn as cylinders. Right image shows the corresponding hair rendering.

The motion of each GC is approximated by a discrete set of nodes (Figure 4). A *node* corresponds to the center of the cross-section of the GC in the model hierarchy, the super-GC being refined at the end of the modeling stage, so that the number of cross-sections is the same at each hierarchy level. With each node, we store pointers to the corresponding nodes in the super-GC and sub-GC. We call such nodes *super-nodes* and *sub-nodes*, respectively. We also store the radius of the associated GC cross-section and the mass, computed from the number of hair strands that it simulates.

A set of *active* nodes partitions each GC so that each section of the finest level GC is governed by one and only one of them. The level of active nodes determines whether a GC section around the node is either animated as a coherent cluster or as a set of refined sub-GCs. We call *segment* the links that connect the set of active nodes, constructing a tree structure, the AWT (see Figure 5).

The AWT evolves during animation through nodes separation (a node splits into sub-nodes) or fusion (sub-nodes merge into their super-node). To ensure that the structure remains as a tree, merging is only allowed from root to tip, and splitting is allowed from tip to root<sup>‡</sup>.

Dynamic splitting and merging occur only in the control model (AWT), not in the geometric model. Before animation, each hair strand is attached to one of the finest GCs in the model hierarchy and thus controlled by the set of active segments at any time during animation. We thus ensure that the visual motion of geometric model remains consistent despite the discrete changes in the LOD of the control model.

## 2.3. Animation

In practice, an AWT is built for each top-level GC in the model hierarchy; the entire hair being partitioned as a forest of such AWTs. At the beginning of the animation, often at rest, each AWT is initialized with only the coarsest nodes active. During animation, the tree evolves over time based on the fusion and separation processes.

Segments in the AWT are represented by either rigid links or viscous springs, depending on the desired behavior. Since a curly wisp should elongate during animation, soft springs are preferred for curly hair, whereas straight hairs are realistically modeled with rigid links. Among all the available

simulation methods for this kind of mechanical models (see for instance <sup>12, 9, 23, 4</sup>), we have currently adopted a fast, approximate iterative method for simulating rigid links <sup>21</sup> as well as an implicit integration method for spring links <sup>9</sup>. Using Pai's strand model <sup>22</sup> may be a good alternate solution. External conditions such as head movement, collisions, wind force fields, gravity, etc. affect the motion of the AWTs, constrained by the dynamic model.

## 3. Adapting the AWT

The AWT adapts over time by dual processes - splitting and merging. The splitting process locally refines the hair motion when a single wisp segment is not sufficient for the motion and deformation. The merging process simplifies the AWT when the motion becomes coherent again. This adaptation greatly simplifies the interaction processing as detailed in section 4.

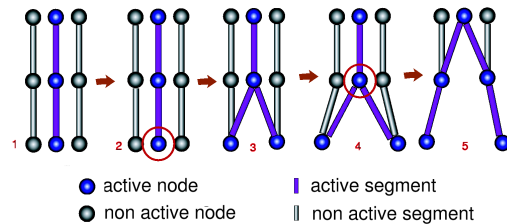
### 3.1. Splitting (separation)

Our splitting criteria is based on the observation that motion becomes more complex in regions where hair accelerates. A node of the AWT splits if :

1. Its acceleration times radius is over a threshold value; and
2. Its child node in the AWT has already been split.

The first condition models the observed behavior that a larger (radius) or faster (accelerating) wisp is more likely to split. The second condition enforces the rule that splitting occurs from tip to root. Figure 5 shows the splitting process. When a node splits into a set of sub-nodes :

- The position of each sub-node is updated with an offset vector computed initially and whenever merging occurs (see 3.3).
- The velocity of sub-nodes are set to the velocity of the split node.



**Figure 5:** The splitting process. 1. Initially, only coarse nodes are activated. 2. A node inside the circle meets the split criteria. 3. Its sub-nodes are activated 4. After a few time steps, the node inside the circle needs to split 5. Resulting AWT.

### 3.2. Merging (fusion)

Nodes sharing *the same parent* merge if their super-node is capable of approximating their motion. The merging conditions are :

1. The merged nodes are contained within the radius of the super-node.

<sup>‡</sup> We ignore rare cases where the tip of hair remains coherent while other parts move apart. In theory, this situation can be handled by turning the tree into a graph with cycles, at the cost of a more complex data-structure than our current implementation.

2. Their relative velocities are under a given threshold in magnitude, to avoid discontinuity in velocity when merging occurs.
3. The splitting condition 1 is not true (this test is added to prevent merging nodes from immediately splitting again).

Note that our merging test is local - it is only performed for the nodes with the same parent node. This way, the AWT eliminates any need to perform tests between arbitrary wisp segments. The tree structure of animated hair is preserved since merging takes place from the root to the tip of hair.

When nodes merge into a super-node :

- The super-node is positioned at the center of mass of the merging nodes. Its velocity is set to the mean velocity, weighted by masses of the merging nodes. Its radius is modified to tightly fit all the merging nodes.
- The offset vector for each merged node is computed and their positions are frozen with respect to the super-node thereafter.

### 3.3. Positions of Non-Active Nodes

To update the embedded multiresolution geometry, the position of non-active nodes should be updated during motion. Initially or when merging is performed, for each non-active node, an offset vector is computed that stores the node's relative position with respect to its super-node. At each time step, the position of a (non-active) sub-node is updated with the offset vector, rotated by a matrix that transforms the tangent vector of the active segment at the time of merging or initially, to the new tangent vector.

## 4. Hair interaction

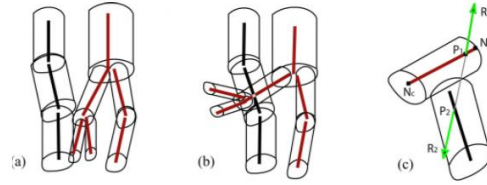
One of the key benefit of the AWT is that it implicitly models mutual hair interactions so that neighboring wisps with similar motions merge, mimicking the static friction in real hair. This avoids subsequent collision processing between these wisps, thus increasing efficiency as well as gaining stability from the reduced number of primitives. In addition, the splitting behavior models wisp deformation without the need of the complex deformable wisp geometry used in previous work<sup>23</sup>. For collision processing, active segments in the AWT are thus represented by cylinders, which greatly simplifies collision detection tests.

### 4.1. Collision detection

The character's body is surrounded by a few bounding spheres, and a space voxel grid is used to store the active wisp segments at each time step. Each time an AWT segment penetrates a bounding volume, or when two segments intersect the same voxel, the wisp segment geometry is used for collision detection. This geometry is simply defined as a cylinder that connects the associated nodes in the wisp tree (a parent  $N_p$  and a child node  $N_c$ ). The cylinder radius is set

to the child node's radius<sup>§</sup> since the parent node may belong to a coarser level of detail (see Figure 6). For instance, a collision between two wisp segments is detected if the distance  $d = d(P_1, P_2)$  between the closest points on their axes is smaller than the sum of their radii  $r = r_1 + r_2$ .

Note that collisions do not need to be detected when neighboring elements already have the same velocity. Testing for collisions in this case would even introduce artifacts, due to our split/merge process. When a merging occurs (e.g. when hair has come to rest), the coarser segment geometry only approximates its sub-segments geometry. Suddenly using this coarse geometry for collision detection would produce new, undesired motion towards a slightly different rest state. To avoid this problem, collision test is only performed between pairs of elements with relative motion differences.



**Figure 6:** Hair mutual interactions are detected efficiently since wisp segments are simple cylinders. (a) Friction forces are generated by the interaction of wisps segments of similar orientation; (b) soft response forces are added in other cases, (c) collision response being computed at closest points and then applied at mass nodes.

### 4.2. Response to mutual hair interaction

Our processing of interactions is inspired from Plante et al.'s anisotropic collision response model<sup>23</sup>. The generated response forces allow wisp segments of similar orientation to interpenetrate, but high viscous friction is produced in this case. Response forces also prevent penetration between colliding wisp segments of different orientation, thus modeling the fact that a hair strand cannot cross each other strand (see Figure 6 (a) and (b)).

This anisotropic behavior can easily be modeled by computing viscous friction forces  $V_1$  and  $V_2$  at the closest points  $P_1$  and  $P_2$  between the segment axes :

$$V_1 = k_f(\dot{P}_2 - \dot{P}_1); \quad V_2 = -V_1 \quad (1)$$

and by adding repulsion forces  $R_1$  and  $R_2$  when the angle between the two segments is over a threshold :

$$R_1 = k_r(r - d) \frac{(P_1 - P_2)}{\|P_1 - P_2\|}; \quad R_2 = -R_1 \quad (2)$$

The resulting force  $F = V + R$  at  $P$  (where  $P$  is  $P_1$  or  $P_2$ ) is then splitted into two forces  $F_p$  and  $F_c$  respectively, and added to the set of external actions applied at the segment

<sup>§</sup> Tapered cylinders computed from the parent and child nodes radii could be used instead. However, collision detection would be more intricate since closest points between two tapered cylinders do not necessarily correspond to the closest points between their axes. We did not observe any noticeable artifacts due to the use of standard cylinders.

nodes (see Figure 6, (c)) : if  $P = uN_p + (1 - u)N_c$ ,

$$F_p = uF; \quad F_c = (1 - u)F \quad (3)$$

The friction coefficient  $k_f$  used in equation (1) should vary depending on the hierarchy level of the wisp segment. Friction forces should reduce the relative speed between nodes, but never change its orientation, which may occur if mass is too small with respect to the applied force. To maintain  $k_f$  at a reasonable value while ensuring the same friction behavior for all resolution levels, we use :  $k_f = \alpha m$ , where  $m$  is the mass of the child node  $N_c$ . We also check that the integration step is small enough to prevent the friction from reversing the direction of relative motion ( $\alpha dt < 1$ ).

### 4.3. Hair interaction with obstacles

Hair is much lighter than the other objects in the scene, including the character's body. Hair position and velocity are modified to model response to collisions with obstacles as in <sup>23</sup>. An active wisp segment that intersects an obstacle is moved to remain in contact with it, by moving the child node  $N_c$ . The node's velocity is set to obstacle's velocity to model static friction, which we find more realistic than a bouncing behavior in the case of hair.

## 5. Results and discussion

In the examples shown, the mass of each coarse node was set so that the mass of the whole hair reaches between 500g and 2kg (that is, around 1g per coarse node), knowing that splitting an merging mechanisms obviously ensure conservation of mass.

In our interactive interface, the splitting threshold can vary from 0.1 to  $0.5N.m.kg^{-1}$ . The value set was chosen according to the desirable importance of splitting during motion; for curly hair, we chose a high value (less splitting) so that hair remains grouped in wisps, whereas for smooth hair we chose a low value (more splitting) so that hair can visibly expand during motion. Only 2 levels of detail thus turned out to be sufficient for animating curly hair; for the simulation of smooth hair, we used 3 levels of detail to get a good compromise between realism and computational costs. More levels of detail could be used to increase realism.

### 5.1. Qualitative results

Snapshots taken from animation examples are shown in Figure 7 (see color plate Section). A variety of hairstyles were animated with varying character motions such as jumping, leaning, running, etc. Thanks to the merging process, our model stabilizes the hair motion even at high speed.

The animations shown in Figure 7 were rendered with about 10,000 individual hair strands. For rendering, we use a modified version of opacity shadow maps <sup>16</sup> for hair-hair shadow computation. Local hair reflectance is based on Kajiyama-Kay hair reflectance model <sup>13</sup>, implemented in nVidia's GC shader. In each frame, hair strands are drawn as OpenGL

polylines, sorted by the visibility ordering algorithm in <sup>17</sup> for antialiasing. Rendering takes about a fraction of second to several seconds, depending on the desired features listed above.

### 5.2. Performance

Hair model (N, L)	single-level time (sec)	AWT mean time (sec)
Short (5149, 2)	3.1	0.32
Short curly (4909, 2)	2.7	0.27
Long (6065, 3)	3.8	0.09
Long curly (9853, 3)	7.9	0.29

The above table compares our method with the performance of a single-level animation method where all the finest GC are simulated. With each hair model, we show the number of nodes (N) at the finest level and the number of geometric levels of detail (L). The single-level method was used only to measure how much efficiency we get with the adaptive method. The time for the AWT is a mean value, reflecting split / merge frequency during animation. The table shows time (in seconds) to compute each animation step for 4 different hair models with same (leaning) motion of the character body. These values were measured on a PC with an 1.7 GHz Pentium CPU. We used 4 integration steps per each frame ( $dt = 10ms$ ). The maximum time to compute 10 seconds-long animation was about 5 minutes.

### 5.3. Discussion

In terms of time performance, our method scales favorably; it runs orders of magnitude faster than the work by Plante <sup>23</sup> which similarly used volumetric wisps for animation but without exploiting multiresolution. The time measurements are comparable to those of Ward et al. <sup>25</sup>, obtained with a level of detail representation for hair.

The AWT is best suited for fast and drastic head motions, in contrast to other methods where only slow and smooth hair motions were illustrated. Our approach can yield strong clustered appearance since we do not allow hair wisps to deform after they are split. Such appearance is essential for many kinds of hair (e.g. curly hair, thick hair, etc.), but may not be always desirable (e.g. for very smooth hair).

Our current choices for wisp splitting criteria could be further extended. We are currently investigating into an unified set of criteria that would determine whether a wisp should split or not. Possible extensions could include the body action on the hair (including collision and friction), interaction between air and wisps, and interaction inside hair. Lastly, the recent work by Ward et al.'s <sup>25</sup> on geometric LODs could be combined with our animation LOD algorithms, for increased efficiency at the rendering stage.

## 6. Conclusion and Future Work

We presented an effective adaptive method for hair animation. Our adaptive wisp tree mimics the dynamic clustering behavior of hair motion often observed in long, thick hair as

well as in curly hair. The reduced number of wisp segments and their simple shapes result in efficient processing of complex mutual hair interaction, as well as increased stability of the simulation.

Our model is best suited for large-scale to mid-scale behavior of long, thick hair, such as splitting and merging of clusters. As hair wisp refines, collision detection takes place between finer and finer cylinders, necessitating smaller time steps to handle mutual hair interaction during fast motion. For such small-scale interactions, an alternative approach such as Hadap and Thalmann's smoothed particle model<sup>12</sup> could be better suited. To guarantee continuity of hair representation, it could be also interesting to investigate a hybrid approach that combines a very smooth hair representation (e.g. using the adaptive guide hair method<sup>4</sup>) with dynamic clustering effects provided by the adaptive wisp tree.

## References

1. K. Anjyo, Y. Usami, and T. Kurihara. A simple method for extracting the natural beauty of hair. In *Proceedings of ACM SIGGRAPH 1992*, Computer Graphics Proceedings, Annual Conference Series, pages 111–120, August 1992.
2. J. Berney and J. K. Redd. A tale of fur, costumes, performance, and integration. *SIGGRAPH Course 14*, 2000.
3. S. Capell, S. Green, B. Curless, T. Duchamp, and Z. Popović. A multiresolution framework for dynamic deformations. In *ACM SIGGRAPH Symposium on Computer Animation*, pages 41–48, July 2002.
4. J. T. Chang, J. Jin, and Y. Yu. A practical model for hair mutual interactions. In *ACM SIGGRAPH Symposium on Computer Animation*, pages 73–80, July 2002.
5. L. Chen, S. Saeyor, H. Dohi, and M. Ishizuka. A system of 3d hairstyle synthesis based on the wisp model. *The Visual Computer*, 15(4):159–170, 1999.
6. A. Daldegan, N. M. Thalmann, T. Kurihara, and D. Thalmann. An integrated system for modeling, animating and rendering hair. *Computer Graphics Forum*, 12(3):211–221, 1993.
7. G. Debunne, M. Desbrun, M-P. Cani, and A. H. Barr. Dynamic real-time deformations using space & time adaptive sampling. In *Proceedings of ACM SIGGRAPH 2001*, Computer Graphics Proceedings, Annual Conference Series, pages 31–36, August 2001.
8. R. Falk and L. R. Sande. Shrek: The story behind the screen. *SIGGRAPH Course Note 19*, 2001.
9. M. Fong. Animating monster fur. *SIGGRAPH Course 36: From Ivory Tower to Silver Screen*, 2001.
10. E. Grinspun, P. Krysl, and P. Schröder. Charms: A simple framework for adaptive simulation. *ACM Transactions on Graphics*, 21(3):281–290, July 2002. Proceedings of ACM SIGGRAPH 2002.
11. S. Hadap and N. Magnenat-Thalmann. Interactive hair styler based on fluid flow. In *Computer Animation and Simulation '00*, pages 87–100, August 2000.
12. S. Hadap and N. Magnenat-Thalmann. Modeling dynamic hair as a continuum. *Computer Graphics Forum*, 20(3):329–338, 2001. Proceedings of Eurographics'01.
13. J. Kajiya and T. Kay. Rendering fur with three dimensional textures. In *Proceedings of ACM SIGGRAPH 89*, Computer Graphics Proceedings, Annual Conference Series, pages 271–280, 1989.
14. S. Kent. *The Making of Final Fantasy. The Sprits Within*. Brady Publishing, Indiana, 2001.
15. T-Y. Kim and U. Neumann. A thin shell volume for modeling human hair. In *Computer Animation 2000*, IEEE Computer Society, pages 121–128, 2000.
16. T-Y. Kim and U. Neumann. Opacity shadow maps. In *Rendering Techniques 2001*, Springer, pages 177–182, July 2001.
17. T-Y. Kim and U. Neumann. Interactive multiresolution hair modeling and editing. *ACM Transactions on Graphics*, 21(3):620–629, July 2002. Proceedings of ACM SIGGRAPH 2002.
18. C. Koh and Z. Huang. A simple physics model to animate human hair modeled in 2D strips in real time. In *Computer Animation and Simulation '01*, pages 127–138, September 2001.
19. D-W. Lee and H-S. Ko. Natural hairstyle modeling and animation. *Graphical Models*, 63(2):67–85, March 2001.
20. N. Magnenat-Thalmann and S. Hadap. State of the art in hair simulation. In *International Workshop on Human Modeling and Animation*, Korea Computer Graphics Society, pages 3–9, 2000.
21. C. Van Overveld. An iterative approach to dynamic simulation of 3-D rigid-body motions for real-time interactive computer animation. *The Visual Computer*, 7:29–38, 1991.
22. D. Pai. Strands: Interactive simulation of thin solids using cosserat models. *Computer Graphics Forum*, 21(3):347–352, 2002. Proceedings of Eurographics'02.
23. E. Plante, M-P. Cani, and P. Poulin. Capturing the complexity of hair motion. *GMOD*, 64(1), january 2002.
24. R. Rosenblum, W. Carlson, and E. Tripp. Simulating the structure and dynamics of human hair: Modeling, rendering, and animation. *The Journal of Visualization and Computer Animation*, 2(4):141–148, 1991.
25. K. Ward, M. C. Lin, J. Lee, S. Fisher, and D. Macri. Modeling hair using level-of-detail representations. In *International Conference on Computer Animation and Social Agents*, May 2003.
26. Y. Watanabe and Y. Suenaga. A trigonal prism-based method for hair image generation. *IEEE Computer Graphics and Applications*, 12(1):47–53, 1992.
27. Z. Xu and X. D. Yang. V-hairstudio: an interactive tool for hair design. *IEEE Computer Graphics & Applications*, 21(3):36–42, May / June 2001.
28. X. D. Yang, Z. Xu, J. Yang, and T. Wang. The cluster hair model. *GMOD*, 62(2), 2000.



**Figure 7:** From top to bottom : short, smooth hair in running motion; long, thick hair in leaning motion; long, curly hair in leaning motion; short, curly hair in leaning motion



# A Practical Self-Shadowing Algorithm for Interactive Hair Animation

Florence Bertails

Clément Ménéier

Marie-Paule Cani

GRAVIR - IMAG/INRIA, Grenoble, France

{florence.bertails, clement.menier, marie-paule.cani}@imag.fr



Figure 1: A dynamic hair without self-shadowing (left) and shaded with our algorithm (right). The 3D light-oriented map storing hair density and transmittance (middle). The whole simulation (including animation and our rendering) is running interactively on a standard CPU.

## Abstract

This paper presents a new fast and accurate self-shadowing algorithm for animated hair. Our method is based on a 3D light-oriented density map, a novel structure that combines an optimized volumetric representation of hair with a light-oriented partition of space. Using this 3D map, accurate hair self-shadowing can be interactively processed (several frames per second for a full hairstyle) on a standard CPU. Beyond the fact that our application is independent of any graphics hardware (and thus portable), it can easily be parallelized for better performance. Our method is especially adapted to render animated hair since there is no geometry-based precomputation and since the density map can be used to optimize hair self-collisions. The approach has been validated on a dance motion sequence, for various hairstyles.

*Key words:* Hair self-shadowing, interactive rendering, hair simulation.

## 1 Introduction

Self-shadowing is of great relevance to the realistic appearance of hair as it contributes to the impression of volume (see Figure 1). Considering the high number of thin, translucent fibers composing a human hair, this phenomenon is difficult to reproduce both accurately and efficiently.

Our work was especially motivated by the need for a simple, fast and accurate technique to render animated

sequences involving dynamic hair. Recently, much effort has been made to achieve interactive frame rates in the simulation of dynamic hair [3, 22, 1]. But most of the time, these good performances only include the cost for animation while realistic hair rendering is done offline.

Approaches targeting interactive self-shadowing are very recent and mostly rely on advanced GPU's capabilities [17, 12]. Though successful, these methods are highly dependent on the hardware architecture, and remain difficult to implement. This paper investigates an alternative solution based on the CPU which turns out to be simpler to implement, more flexible, and which still yields interactive frame rates.

### 1.1 Previous Work

Realistic rendering of human hair requires the handling of both local and global hair properties. Local hair properties describe the way individual hair fibers are illuminated and then represented in the image space, whereas global properties define how the hair fibers interact together. Global hair properties especially include hair self-shadowing which plays a crucial role in volumetric hair appearance and which is the main focus of this paper.

#### Local Illumination

To render an individual hair strand, Kajiya and Kay earlier proposed a reflectance model [9] that has been widely used subsequently. Their model is composed of a lambertian diffuse component and an anisotropic specular com-

ponent. Many later approaches have subsequently proposed further refinements to this model [13, 2, 7, 10]. Recently, Marschner *et al.* [16] measured the scattering from real individual hair fibers and proposed a physical-based scattering model accounting for subtle scattering effects (such as multiple specular highlights) observed in their experiments. In our approach, we use Kajya-Kay’s reflectance model but our self-shadowing technique could be combined with any other local illumination model.

### Self-Shadowing

Two main techniques are generally used to cast self-shadows into volumetric objects<sup>1</sup>: shadow maps and ray casting through volumetric densities.

In basic depth-based shadow maps, the scene is rendered from the light point of view, and the depth of every visible surface is stored into a 2D *shadow map*. A point is shadowed if the distance between the point and its projection to the light’s camera is greater than the depth stored in the shadow map. This algorithm is not adapted to render semi-transparent objects such as hair because it only stores a single depth per pixel. To handle the self-shadowing of semi-transparent objects, Lokovic *et al.* proposed an extension to the traditional shadow maps: the *deep shadow map* [15]. For each pixel of the map, the method stores a *transmittance* function (also called *visibility function*) that gives the fraction of light penetrating at every sampled depth along a ray casted from the pixel.

Kim and Neumann proposed a practical implementation of this approach, called the *opacity shadow maps* [11], and applied it to hair rendering. In their method, the hair volume is uniformly sliced along the light rays and each hair volume comprised between two consecutive slices is rendered from the light’s point of view into the alpha buffer, leading to an opacity shadow map. Final rendering is done by interpolating the different opacity shadow maps. This method has been used together with hair representations at different LODs for the interactive animation of hair [23].

The opacity shadow maps technique was recently exploited by other authors [17, 12] to get a fast rendering by using recent GPU’s capabilities. Koster *et al.* achieved real-time results by accelerating the implementation of the opacity shadow maps and by making some assumptions about the geometry of hair. Mertens *et al.* used an adaptive clustering of hair fragments instead of an uniform slicing, which enabled them to interactively build a more accurate transmittance function.

Volume rendering is a common approach for visualizing datasets that come on 3D grids [8]. To render

semi-transparent volumetric objects with shadows, a first step usually consists of casting rays from the light source into the volume, and storing the light attenuation function (in voxels for instance). Then, actual rendering is done by ray tracing from the viewpoint. Such methods can accurately render both volumetric data such as clouds or smoke [8] as well as data with fine geometry such as hair [9]. Ray-tracing-based approaches often yield good quality results, but they are usually very expensive in terms of time and memory. More recently, splatting approaches have been used in order to achieve interactive shadowing and rendering of volumetric dataset [18, 25]. Billboard splatting has been successfully applied to the rendering of clouds [6]. In the case of hair, this method is still efficient but it does not seem to be really adapted to render the fine geometry of hair [1].

Ray tracing-based methods can often be very prohibitive in terms of rendering time, as they require the calculation and the sorting of multiple intersections between the rays and the objects that need to be shadowed. Conversely, the key benefit of the shadow map-based approaches is the light-oriented sampling of geometry, which makes the computation of accumulative transmittance straightforward. Actually, our method is inspired by both. Combining a volumetric representation of density with a light-oriented sampling allows us to define a practical and interactive self-shadowing algorithm.

### 1.2 Overview

Our goal is to provide an easy, accurate and efficient way of casting shadows inside hair. Our method has to be flexible enough to handle and accelerate simulations that involve animated hair.

Our main contribution is to propose a new algorithmic structure called *3D light-oriented shadow map* that is inspired by both traditional 3D density volumes and more recent 2D shadow maps as it combines an optimized volumetric representation of hair with a light-oriented partition of space. This voxel structure stores the light attenuation through the hair volume, and it is used to compute the final color attributed to each hair drawing primitive (hair segment for instance).

The main advantages of our method are the following:

- Our application is portable, simple to implement and can render a whole hairstyle composed of thousands of hair strands interactively on a standard CPU. Furthermore, it can easily be parallelized to increase performance.
- The approach is especially adapted to animated hair since the algorithmic structures that we used are efficiently updated at each time step. Moreover we

---

<sup>1</sup>Please refer to [24] for a complete survey on shadowing methods.

show that our data structures provide an inexpensive way of processing hair self-collisions.

- Our technique does not make any assumption about the geometry of hair, and thus can be applied to render any hairstyle. It has been validated on various hairstyles, either static or animated with different kinds of motion.

Section 2 describes our 3D light-oriented shadow map structure. Section 3 explains how the self-shadowing process can be efficiently done by using this new structure. Section 4 deals with the two extensions of the method: on the one hand, we show that our 3D map is very helpful to process hair self-collisions efficiently; on the other hand we provide a parallelized version of our algorithm that improves the global performance of the simulation. The last two sections discuss results before concluding.

## 2 3D Light-Oriented Shadow Map

Our 3D shadow map is a uniform cubic voxel grid that associates to each voxel (or cell) a *density* value and a *transmittance* value.

The different hair models that we want to render are composed of a set of segments, but our algorithm could also apply to other kinds of geometry such as polygonal surfaces for example.

### 2.1 A Light-Oriented Local Frame

In our method, the light rays are assumed to be parallel (ie. coming from an infinitely distant source), which is a reasonable assumption for handling common lighting conditions like sun light. This point will be discussed in conclusion.

Instead of having a fixed-oriented structure like in previous approaches, our map is always aligned with the light direction. More precisely, the map is placed in a local frame  $\mathcal{R} = (O, \mathbf{X}_{map}, \mathbf{Y}_{map}, \mathbf{Z}_{map})$  where  $\mathbf{X}_{map}$  coincides with the normalized light vector  $\mathbf{L}$  and  $O$  is the origin of the map (see Figure 3).

As we shall see in Section 3.2, this configuration is very helpful for computing the accumulated transparencies efficiently. Note that for non-animated data requiring “dynamic” lighting (ie. a moving light), this choice would not be appropriate since the material geometry is to be recomputed each time the light moves. But in our case, the geometry of hair needs to be updated at each time step, so the moving light case does not yield extra cost for us.

### 2.2 Object Space to Map Space

To occupy a limited memory, our data structure exploits the fact that during animation, the hair volume is always located inside a bounding box of constant spatial dimension. Indeed hair always remains attached to a scalp, and

hair strands are assumed to be inextensible. Storing hair elements can thus be done inside a bounded structure, provided we build a mapping function from the 3D object space to this 3D bounding space.

The spatial dimension of the map is thus fixed and only depends on the maximal length  $l_{max}$  of a hair strand. If the dimension of the map is superior or equal to  $2 \times l_{max} + h_{max}$ , where  $h_{max}$  is the maximal dimension of the head, it is ensured that the grid will always represent a bounding volume for the hair at any time step. Of course, the best choice for the dimension of the map is the minimal number satisfying the constraint above.

The size (or resolution) of the map (ie. the number of cells it contains) depends on the desired accuracy of self-shadowing. Some tests have been made in Section 5 to compare results using different map resolutions.

In the remainder of the paper,  $NCELLS$  will denote the number of cells in each direction  $\mathbf{X}_{map}$ ,  $\mathbf{Y}_{map}$  and  $\mathbf{Z}_{map}$  of the map frame  $\mathcal{R}$ , and  $ds$  will represent the step of the map, ie. the spatial dimension of a cell (see Figure 2).

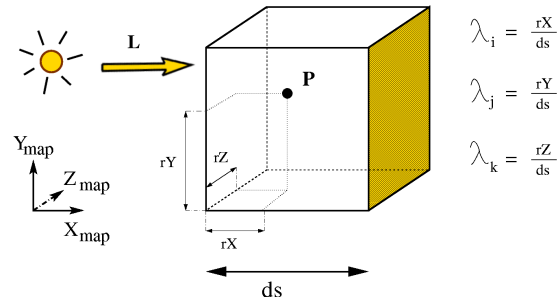


Figure 2: One cell of the map containing a point  $P$ . The  $\lambda_i$  parameters give the location of  $P$  inside the cell, and will be useful for the filtering process (see Section 3.3). By convention, each cell will store the quantity of light received by its back side (yellow side).

To find the index of the cell corresponding to a point  $P(x, y, z)$ , the coordinates of  $P$  are first expressed in the map frame  $\mathcal{R}$  as  $(x_{map}, y_{map}, z_{map})$ , and the following mapping function then applied:

$$\Psi : \quad \mathbb{R}^3 \quad \longrightarrow \quad [0..NCELLS]^3$$

$$\begin{bmatrix} x_{map} \\ y_{map} \\ z_{map} \end{bmatrix} \quad \longmapsto \quad \begin{bmatrix} \left\lfloor \frac{x_{map}}{ds} \right\rfloor \bmod NCELLS \\ \left\lfloor \frac{y_{map}}{ds} \right\rfloor \bmod NCELLS \\ \left\lfloor \frac{z_{map}}{ds} \right\rfloor \bmod NCELLS \end{bmatrix}$$

Figure 3 shows the mapping between the object space and the map space.

Thanks to the mapping function  $\Psi$ , access to elements of the map is done in constant time, which greatly contributes to the efficiency of the method.

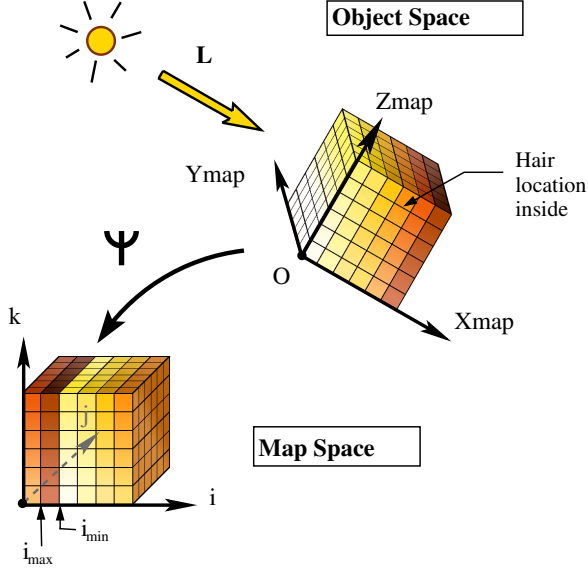


Figure 3: Correspondence between the object space and the map space. Because of the modulo operator in the mapping function  $\Psi$ , the first slice of the map (in light order) does not necessarily have the lowest index. The first slice and the last slice have consecutive indexes.

### 3 Self-Shading Algorithm

Our self-shadowing algorithm is composed of three main steps: hair density filling (1), transmittance computation (2), and filtering (3). Initially, each cell of the map has a null density (and we call it an *empty* cell).

The following figure summarizes the whole rendering pipeline<sup>2</sup>.

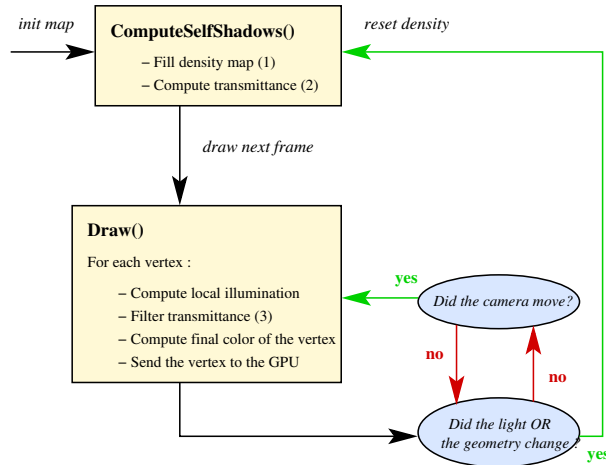


Figure 4: The rendering pipeline.

<sup>2</sup>In our case, each hair strand is drawn as an OpenGL line strip

### 3.1 Filling Hair Density into the Map

The first step of the algorithm consists of filling the map with hair density. This is simply done by traversing the geometry of hair and doing the following operations:

- Each hair strand  $s_i$  is sampled using a Catmull-Rom spline into  $n_{Smooth}$  points  $P_k^i$ ;
- For each point  $P_k^i$ , the density of the cell  $\Psi(P_k^i)$  is incremented.

Of course the resulting density value obtained for one cell only makes sense relative to values of the other cells. Indeed, each isolated density value is arbitrary, and especially depends on the number of sample points used for each strand. Assuming that hair sampling is uniform, which is a reasonable assumption, the relative density multiplied by a scaling factor  $f$  approximates the light fall off through the corresponding cell. This quantity is commonly called the *extinction* parameter [15].

In practice, our hair sampling is the same as the one that is used at the final drawing stage, in order to ensure that each drawn vertex belongs to a non-empty cell.

### 3.2 Computing Transmittance

The fraction of light that penetrates to a point  $P$  of space can be written as [15]:

$$\tau(p) = \exp\left(-\int_0^l \kappa(l') dl'\right) \quad (1)$$

where  $l$  is the length of the path from the light to the point, and  $\kappa$  is the extinction function along the path.

The function  $\tau$  is called the *transmittance* function. A sampled evaluation of  $\tau$  can be done by accumulating transparencies of sampled regions along the light direction.

In our case, we need to evaluate the transmittance function at each cell of the map. To do this, we compute the transparency of each cell  $(i, j, k)$  as:

$$t(i, j, k) = \exp(-\kappa_{i,j,k} ds) \quad (2)$$

where the extinction coefficient  $\kappa_{i,j,k}$  is computed using the density value of the cell  $(i, j, k)$ , as explained before in Section 3.1:  $\kappa_{i,j,k} = f \times d_{i,j,k}$  where  $d_{i,j,k}$  is the density of cell  $(i, j, k)$  and  $f$  is a scaling factor.

The transparencies are then composited together to get the final transmittance of each cell  $(i, j, k)$ :

$$Trans(i, j, k) = \prod_{i'=i_{min}}^i \exp(-d_{i',j,k} f ds) \quad (3)$$

where  $i_{min}$  is the index of the map slice that is the closest to the light (see Figure 3).

As we mentioned in the previous section, the novelty of our approach in comparison with previous algorithms using voxel grids is that cells are sorted along the light direction: accumulating transparencies then becomes straightforward:

- A transmittance parameter  $prevTrans$  is first initialized to 1 which is the proper value for a transparent and fully illuminated cell;
- The column  $(j, k)$  is traversed, starting from slice  $i_{min}$  (the closest slice to the light) until slice  $i_{max}$  (the furthest slice):
  - If cell  $(i, j, k)$  is non-empty, its transmittance is set to  $prevTrans \times \exp(-d_{i,j,k} f ds)$  (using Equation 3) and the parameter  $prevTrans$  is updated to this value.
  - Otherwise cell  $(i, j, k)$  is given the transmittance  $prevTrans$ .

Note that some empty cells might also be in shadow, since filling densities into the map does not necessary yield a connective set of non-empty cells. Even if only vertices belonging to non-empty cells will be drawn, giving a proper transmittance value to empty cells is important because such cells could be involved in the filtering process, if a non-empty cell has empty neighbors (see next section). The algorithm described above guarantees that every cell of the map has a proper transmittance value.

### 3.3 Filtering and Composing Colors

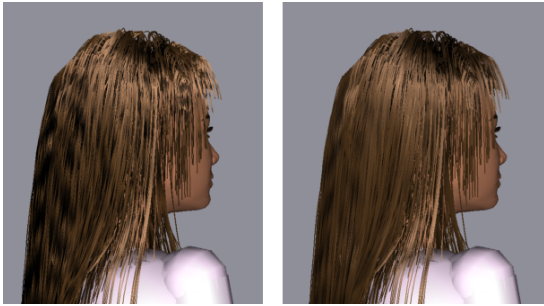


Figure 5: The effect of filtering the transmittance values. Self-shadows without filtering (left): regular patterns aligned with the map are visible. Self-shadows with filtering (right): artefacts have vanished, hair looks coherent.

Before drawing hair primitives, it is necessary to filter transmittance values, otherwise regular patterns aligned with the density map will be quite visible, as shown by Figure 5.

For each point  $P$  that has to be sent to the GPU for final drawing:

- We compute the relative position of  $P$   $(\lambda_i, \lambda_j, \lambda_k)$  with respect to its corresponding cell  $\Psi(P)$  (see Figure 2).
- We compute filtered transmittance at point  $P$  by applying a trilinear interpolation as:

$$Trans^f(P) = \sum_{\substack{i' \in \{i-1, \dots, i\} \\ j' \in \{j-1, \dots, j\} \\ k' \in \{k-1, \dots, k\}}} A_{i'} A_{j'} A_{k'} Trans(i', j', k')$$

$$\text{where } A_{i'} = \begin{cases} \lambda_i & \text{if } i' = i \\ (1 - \lambda_i) & \text{otherwise} \end{cases}$$

(similar for  $A_{j'}$  and  $A_{k'}$ )

- Finally, the color  $\Phi_P$  of vertex  $P$  is obtained by the following equation:

$$\Phi_P = \Phi_{Ambient} + Trans^f(P) \times (\Phi_{Diffuse} + \Phi_{Specular}(P))$$

## 4 Extensions

### 4.1 Handling Hair Self-Collisions

Because of the high number of hair strands composing a human hairstyle, hair self-collisions represent a difficult and computationally expensive issue in hair animation. In practice, it often takes more than 80% of the simulation time [21].

An acceptable approximation of hair self-interaction consists of considering that internal collisions mainly result into the hair volume [14]. Starting from this assumption, hair density information is very useful: if the density is local over a fixed threshold (corresponding to maximum quantity of hair that can be contained in a cell), the hair strands should undergo constraints that spread them out.

Hair is animated using an approach closed to hair guidance methods [5, 4]. In our case, hair is composed of approximately a hundred wisps where each hair wisp is simulated through three guide hair strands. Each guide hair strand is animated using a fast rigid links simulation [19]. Final rendered hair strands are simply interpolated from the guide hair strands within each wisp.

Using the density map at each time step, hair self-collisions are processed by applying repulsive forces from the center of each cell having a too high density. Although this method is extremely simple, it yields convincing results. Furthermore, this is a very cheap way to handle hair self-collisions (it only takes 2.5% of the whole processing time). Please visit our website and watch our videos at <http://www.evasion.imag.fr/Publications/2005/BMC05a/>.



## 4.2 Parallelization of the Algorithm

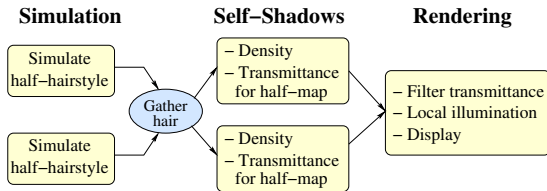


Figure 6: A parallel version of our algorithm.

One advantage of having a CPU-based algorithm is that parallelization can be considered in order to increase its efficiency. As a matter of fact, the described method is very well suited for such a technique. We present here the parallel implementation of the simulation and self-shadowing algorithms.

- **Simulation:** thanks to the use of the density-map for handling self-collisions, each hair wisp can be simulated independently. This allows for a straight-forward parallelization where each processor computes a part of the hair, gathering at the end their partial results.
- **Self-Shadowing:** here again a straight-forward parallelization can be applied thanks to the fact that the map is light-oriented. As described in Section 3.2, the calculations for each column  $(j, k)$  can be done independently.

We have tested this implementation on a standard PC cluster and were able, using 3 CPUs, to easily double the frame rate in comparison with the single processor results given in the next section.

When trying to use more CPUs, the network gathering and sending of the vertices to the GPU became the main bottleneck. Sending vertex arrays directly to the GPU should reduce this bottleneck.

## 5 Results and Discussion

Our algorithm has been applied both to static and dynamic hairstyles. In each case we compare it with existing methods in terms of quality and performance.

### 5.1 Rendering Static Hair

Figures 1 and 7 show that our self-shadowing algorithm produces good visual results for merely synthetic hairstyles as well as for hairstyles captured from real hair geometry. We can see in Figure 7 that self-shadows make volumetric wisps stand out, whereas no self-shadows flatten the hair.

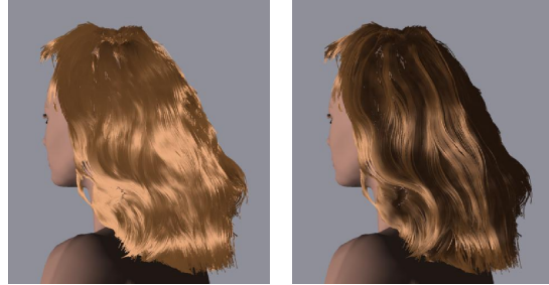


Figure 7: Applying our self-shadowing algorithm to a hairstyle captured from photographs by the method of Paris *et. al* [20]. The hairstyle is composed of 87,500 hair strands (1,123 K segments) and it took 2 seconds to render it.

Figure 8 shows results obtained on curly hair when using different map resolutions. We can notice that for fine resolutions ( $128 \times 128$  or  $256 \times 256$ ), curly wisps are properly shadowed and their shape is thus clearly visible, which is not the case for the coarsest resolutions. In practice, we found that a  $128 \times 128$  resolution was sufficient to account for small shape details of hair.

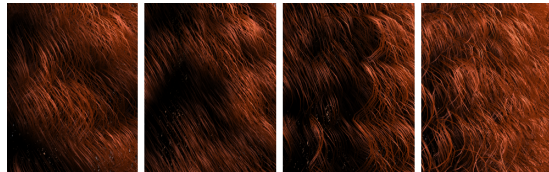


Figure 8: Evaluation of the quality of self-shadowing, using different map resolutions. From left to right:  $32 \times 32$  with  $ds = 0.5$ ;  $64 \times 64$  with  $ds = 0.2$ ,  $128 \times 128$  with  $ds = 0.1$  and  $256 \times 256$  with  $ds = 0.05$ .

	Map reset + density	Trans	Filter + draw	Total rendering
Smooth	0.038	0.015	0.037	0.09
Curly	0.062	0.015	0.053	0.13

Table 1: Detailed performance of the rendering process (computing density, transmittance, filtering and final drawing) of a smooth hairstyle composed of 100K segments and a curly hairstyle composed of 200K segments. The results are expressed in seconds per frame; they have been obtained using an Intel P4 CPU at 3GHz.

In comparison with [17] our self-shadowing algorithm runs at a higher frame rate (11 FPS instead of 6 FPS for 100K hair segments).

## 5.2 Rendering Dynamic Hair

Figure 9 shows two snapshots from our hair animations. Our self-shadowing algorithm captures the fine discontinuities observed in real hair during motion, as illustrated in Figure 10.



Figure 9: A smooth brown hairstyle (100 K segments) and a curly red hairstyle (200 K segments) animated with different dance motions and interactively rendered with our algorithm.

	Anim	Hair self-collisions	Rendering	Total simu
Smooth	0.067	0.003	0.09	0.16
Curly	0.254	0.003	0.13	0.557

Table 2: Detailed performance of the simulation (animation, rendering and hair self-collisions) of two hairstyles composed of 134 animated wisps: a smooth hair style (100K rendered segments) and a curly hair style (200 K rendered segments). The results are expressed in **seconds per frame**; they have been obtained using an Intel P4 CPU at 3GHz.

Table 2 gives the detailed performance of the whole simulation, including animation, hair self-collisions and rendering for both smooth and curly hairstyles. Note that the animation time is not the same for the two hairstyles, because it includes the update and the smoothing of the interpolated hair strands.

A hair composed of 3350 hair strands and 100K segments is thus completely simulated at an interactive frame rate of 6 FPS. For aesthetic results, we have implemented hair-body collisions using a standard method based on spheres approximation. Handling such collisions makes the performance fall down to 3.5 FPS for the smooth hairstyle, and 1.5 FPS for the curly hairstyle, but no optimization has been developed yet for that specific problem, considering it was beyond the scope of this paper.

In our approach, the hair volume is properly generated using a repulsive force field based on local densities, as explained in 4.1. However, this method does not account

for hair anisotropy nor wisps interpenetration. This could be done by adding more information to the map, such as hair orientation.



Figure 10: A real shadowed hair (left) and our model (right) with similar lighting conditions.

## 6 Conclusion and Future Work

We have presented a new hair self-shadowing algorithm based on a 3D-light oriented density map. Our approach can interactively render various hairstyles composed of thousands of hair strands, and yields convincing results. Our algorithm can easily be parallelized to improve the performance. Furthermore, we have shown that our density map is very helpful in accelerating the simulation process, as it can be used to handle self-collisions in an inexpensive way with good visual results. We are planning to use the hair density information again to optimize hair-body collisions.

For simplicity purposes, our approach makes the assumption of an infinitely distant source, which could be a limitation for rendering scenes illuminated by punctual sources. Yet, it seems that we could easily handle the case of punctual sources by only changing our mapping function  $\Psi$ . Instead of considering an uniform square space partition, the new mapping function  $\Psi'$  should account for an angular space partition starting from the source point, and then sampled normally to the light rays.

Our method could also handle several light sources by simply adding as many light-oriented maps as sources. The final transmittance of a point  $P$  would have to be interpolated between the transmittance values obtained from the different sources.

To get a better precision in our computations for a low cost, an interesting idea would be to follow the same approach as Mertens *et. al* [17] who build an adaptive slicing along a light ray and thus get a better approximation of the visibility function than approaches using a uniform slicing.

### Acknowledgements

This work was supported by L'Oréal Recherche. Thanks to Thanh Giang for proofreading the paper.

## References

- [1] Y. Bando, B-Y. Chen, and T. Nishita. Animating hair with loosely connected particles. *Computer Graphics Forum*, 22(3):411–418, 2003. Proceedings of Eurographics'03.
- [2] D. Banks. Illumination in diverse codimensions. In *Proceedings of ACM SIGGRAPH'94*, Computer Graphics Proceedings, Annual Conference Series, pages 327–334, 1994.
- [3] F. Bertails, T-Y. Kim, M-P. Cani, and U. Neumann. Adaptive wisp tree - a multiresolution control structure for simulating dynamic clustering in hair motion. In *ACM SIGGRAPH Symposium on Computer Animation*, pages 207–213, July 2003.
- [4] J. Chang, J. Jin, and Y. Yu. A practical model for hair mutual interactions. In *ACM SIGGRAPH Symposium on Computer Animation*, pages 73–80, July 2002.
- [5] A. Daldegan, N. M. Thalmann, T. Kurihara, and D. Thalmann. An integrated system for modeling, animating and rendering hair. *Computer Graphics Forum*, 12(3):211–221, 1993.
- [6] Y. Dobashi, K. Kaneda, H. Yamashita, T. Okita, and T. Nishita. A simple, efficient method for realistic animation of clouds. In *SIGGRAPH '00: Proceedings of the 27th annual conference on Computer graphics and interactive techniques*, pages 19–28. ACM Press/Addison-Wesley Publishing Co., 2000.
- [7] D. Goldman. Fake fur rendering. In *Proceedings of ACM SIGGRAPH'97*, pages 127–134, 1997.
- [8] J. Kajiya and B. Von Herzen. Ray tracing volume densities. In *Proceedings of the 11th annual conference on Computer graphics and interactive techniques*, pages 165–174. ACM Press, 1984.
- [9] J. Kajiya and T. Kay. Rendering fur with three dimensional textures. In *Proceedings of ACM SIGGRAPH'89*, Computer Graphics Proceedings, Annual Conference Series, pages 271–280, 1989.
- [10] T-Y. Kim. *Modeling, Rendering and Animating Human Hair*. PhD thesis, University of Southern California, 2002.
- [11] T-Y. Kim and U. Neumann. Opacity shadow maps. In *Rendering Techniques 2001*, Springer, pages 177–182, July 2001.
- [12] M. Koster and H-P. Seidel. Real-time rendering of human hair using programmable graphics hardware. In *Computer Graphics International (CGI)*, pages 248–256, June 2004.
- [13] A. M. LeBlanc, R. Turner, and D. Thalmann. Rendering hair using pixel blending and shadow buffers. *The Journal of Visualization and Computer Animation*, 2(3):92–97, – 1991.
- [14] D-W. Lee and H-S. Ko. Natural hairstyle modeling and animation. *Graphical Models*, 63(2):67–85, March 2001.
- [15] T. Lokovic and E. Veach. Deep shadow maps. In *Proceedings of the 27th annual conference on Computer graphics and interactive techniques*, pages 385–392. ACM Press/Addison-Wesley Publishing Co., 2000.
- [16] S. Marschner, H. Jensen, M. Cammarano, S. Worley, and P. Hanrahan. Light scattering from human hair fibers. *ACM Transactions on Graphics (Proceedings of the SIGGRAPH conference)*, 22(3):281–290, July 2003.
- [17] T. Mertens, J. Kautz, P. Bekaert, and F. Van Reeth. A self-shadow algorithm for dynamic hair using density clustering. In *Proceedings of Eurographics Symposium on Rendering*, 2004.
- [18] M. Nulkar and K. Mueller. Splatting with shadows. *Volume Graphics*, pages 35–50, 2001.
- [19] C. Van Overveld. An iterative approach to dynamic simulation of 3-D rigid-body motions for real-time interactive computer animation. *The Visual Computer*, 7:29–38, 1991.
- [20] S. Paris, H. Briceño, and F. Sillion. Capture of hair geometry from multiple images. *ACM Transactions on Graphics (Proceedings of the SIGGRAPH conference)*, 2004.
- [21] E. Plante, M-P. Cani, and P. Poulin. Capturing the complexity of hair motion. *Graphical Models (Academic press)*, 64(1):40–58, January 2002.
- [22] K. Ward and M. Lin. Adaptive grouping and subdivision for simulating hair dynamics. In *Proceedings of Pacific Graphics'03*, September 2003.
- [23] K. Ward, M. Lin, J. Lee, S. Fisher, and D. Macri. Modeling hair using level-of-detail representations. In *International Conference on Computer Animation and Social Agents (CASA)*, May 2003.
- [24] A. Woo, P. Poulin, and A. Fournier. A survey of shadow algorithms. *IEEE Computer Graphics and Applications*, 10(6):13–32, 1990.
- [25] C. Zhang and R. Crawfis. Shadows and soft shadows with participating media using splatting. *IEEE Transactions on Visualization and Computer Graphics*, 9(2):139–149, 2003.

# Modeling Hair Influenced by Water and Styling Products

Kelly Ward Nico Galoppo Ming C. Lin  
University of North Carolina at Chapel Hill  
{wardk,nico,lin}@cs.unc.edu

## Abstract

We present novel methods for capturing the key characteristics of hair influenced by water and styling products. Our approach includes a dynamics system that adaptively accounts for changing stiffness and weight of the hair, a geometric representation that can alter the physical depiction of hair based on the substance(s) present on it, and a rendering approach to account for the varying appearance of hair. Additionally, strands of hair can dynamically bond together due to the introduction of water or styling products. All of these properties can vary on the fly as water or styling products are applied to the hair.

## 1 Introduction

Realistically simulating the motion and appearance of hair is an important goal in modeling virtual humans for many applications. In the real world, external substances interact with hair, thereby changing its physical behavior and outward appearance. Existing hair modeling systems, however, have primarily focused on depicting the basic properties of hair, free of any external influences. By capturing the essential attributes of hair influenced by water and styling products, an enhanced hair modeling system can be used by beauticians, stylists, dermatologists and other physicians to “preview” the hair appearance and movement under different conditions.

**Main Results:** In this paper, we introduce several fundamental techniques for modeling hair

that capture the key features of hair appearance and behavior affected by external substances. The main results are:

- A dual-skeleton system that decouples the control of the local and global behavior of hair providing an efficient localized collision detection method;
- Automatic adjustment of dynamic properties of hair due to external substances;
- Design of flexible geometric structures that account for frequently changing hair volume;
- “Dynamic bonds” that model the adhesive forces introduced by styling chemicals to the hair;
- Approximation of lighting equations by parameterizing the key factors that affect the visual appearance of wet hair.

**Organization:** Related work on hair modeling is briefly reviewed in Section 2. Section 3 presents some basic properties of hair in the presence of water and styling products. The key elements of our improved hair modeling system are described in Section 4. The methods for achieving the desired characteristics of hair behavior and appearance due to the presence of water and styling products are explained in Sections 5 and 6, respectively. Discussion of our implementation and demonstration of the results thereof are provided in Section 7. Finally, we conclude with some future research directions.

## 2 Previous Work

### 2.1 Hair Modeling

Many techniques have been proposed to model hair dynamics. Individual strands can be represented as a series of open chains of line segments and hairstyles are specified by using the angles between them [1, 2, 3]. Groups of hair strands that are in close proximity of one another can be grouped together as “wisps”, each with a single skeleton to control motion [3, 4, 5, 6]. Similarly, guide strands can be used to interpolate the motion of the guide skeleton to the nearby strands within each hair cluster [7]. Fluid dynamics have been used in combination with individual strand dynamics to capture the complex interactions of hair [8]. The work of [9] produced hair-gel effects by retaining a deformed hairstyle shape where the product was applied.

Most recently, multiresolution representations, either using a hierarchy of hair clusters for styling [10], a combination of three discrete representations (strands, clusters, and strips) [11], or continuous, adaptive subdivision [12, 13], have also been proposed to further speed up the performance for hairstyling, modeling and simulation.

### 2.2 Hair Rendering

An anisotropic lighting model for hair was presented in [14], which was efficiently implemented with texture maps by [15]. Self-shadowing due to intra-hair occlusion is a vital feature. Opacity shadow maps [16] are less expensive than deep shadow maps [17] providing similar visual effects. Most recently, [18] introduced a new shading model that accounts for various light scattering effects, generating photorealistic appearances of hair.

## 3 Background

Before we describe our method for capturing the primary effects of water and styling products on hair, it is important to understand some physical properties of hair fibers and their interaction with these substances.

### 3.1 Hair and Water

Hairs of most mammals, including humans, mainly consist of the protein material  $\alpha$ -keratin [19]. Water acts as a plasticizer of the biopolymeric structure of keratin and is able to drastically modify many of the physical properties of  $\alpha$ -keratin fibers, including mechanical and electrical properties. As a plasticizer, water changes the longitudinal stiffness of fibers by as much as a factor of three as water is fully absorbed into the hair. Furthermore, hair fibers are highly permeable allowing hair to absorb 30 to 45% of its own weight in water causing the fibers to swell radially by about 16% as the hairs’ wetness increases [20, 19].

While individual hair fibers swell due to the absorption of water, wet strands in close proximity with each other group together due to the bonding nature of water. As a result, wet hair appears less voluminous in comparison to dry hair. Figure 1 shows side by side images of a real person with wet and dry hair. Note how much fuller the hair is when it is dry than when it is wet.

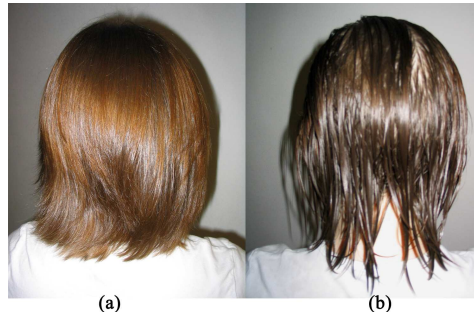


Figure 1: Real images of (a) dry and (b) wet hair.

### 3.2 Hair and Styling Products

The purpose of cosmetic styling, or fixative, products is to change the physical properties of hair. Given the myriad of fixative products on the market, we have chosen to look at the general effects of styling aids on hair. The functions of styling products are typically to hold a section of hair in place, alter the feel of hair, and/or increase the inter-fibril interactions of hair strands [19]. The application of fixative products prevents fibers from smoothly sliding over each other.

Fixative products usually cause a high degree of adhesiveness in hair, thereby causing



hair strands to cling together and move in large groups wherever the product is applied. Polymers constitute the primary active ingredient of most styling products and they increase the stiffness of the hair fibers, thus decreasing the general motion of the hair. The most observable effects of fixative products are typically stiff hair motion and the bundling of strands [19]. Figure 2 shows hair with and without styling products.



Figure 2: Real images of hair (a) without and (b) with styling products (hairspray).

## 4 Hair System

Here we introduce some of the key concepts for our hair modeling system.

### 4.1 Motivation for Dual-skeleton System

Hair motion is subject to changes in the global positioning of the strands, as well as localized styling changes such as the elongation of a curl under force. While the localized styling motion can be made very stiff through the application of a strong fixative product, the hair is still subject to a global motion when forces are applied to it.

Wisp-based hair modeling systems, such as [5, 12], have solved this problem by using a single skeleton curve modeled as a set of particles connected with rigid springs and hinges. Wavy hairs are produced by specifying the number of waves and amplitude of waves inside each wisp. As the wisp segment stretches, the amplitude and frequency of the waves are adjusted to show the wavy hair stretching straight.

The single skeleton dynamics can capture the deforming hairstyle, however there are no checks to ensure that the length of the hair is preserved over time or that the collision detection

remains both accurate and efficient throughout the simulation in light of the changing orientation and position of the hair inside of the wisp.

In order to capture both the global motion and localized styling motion of hair, to ensure hair length preservation at all time, and to maintain an accurate and efficient collision detection method throughout the simulation, we have created a *dual-skeleton* system for modeling hair. This dual-skeleton system provides a single skeleton to control the global motion of hair and a second one to provide a positioning guide and localized collision detection scheme for hair. We refer to these skeletons as the *global-skeleton* and the *local-skeleton*, respectively.

### 4.2 Dual-skeleton Setup

The global-skeleton is modeled as a series of line segments connected by node points,  $N_{g0}, N_{g1}, \dots, N_{g(n-1)}$ , where  $n$  is the number of node points. A hairstyle is defined by positioning the local-skeleton in the desired form in relation to the global-skeleton, see Figure 3. Let the line segment between the  $N_{gi}$  and  $N_{g(i-1)}$  global-skeleton node points be the  $i$ th global-skeleton segment,  $S_{gi}$ . The  $i$ th local-skeleton node,  $N_{li}$ , lies in the plane perpendicular to  $S_{gi}$  containing  $N_{gi}$ . Each local-skeleton node has a defined angular position to fix it around the global-skeleton segment.

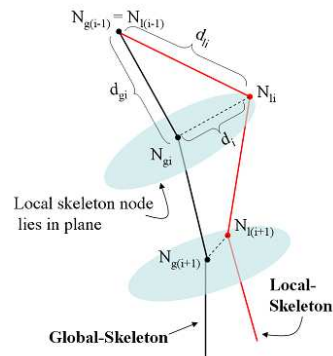


Figure 3: Positioning of the local-skeleton relative to the global-skeleton.

The rendered hair geometry follows the form of the local-skeleton. The hair strands are modeled as subdivision curves [11]. Strands in close proximity with each other are grouped together to follow the same dual-skeleton system. This grouping helps to capture the natural

clumping of strands due to external substances or electrostatic forces that can be found in nature. Once the dual-skeletons are created, circular cross-sections are defined at each node point of the local-skeleton. The strands of hair are placed randomly within the bounds of the cross-sections.

### 4.3 Dynamics Model

There are two types of motion controlling the global-skeleton. The first dictates the bending of the strands by maintaining a spring force to control the angular position of each node point in relation to its neighbors. Second, soft springs are used to control the elongation of the global-skeleton. Each global-skeleton section is prevented from stretching beyond the reach of its corresponding local-skeleton section since the length of the local-skeleton is preserved at all times. For straight hair the nodes of the global-skeleton are connected with rigid links and the position of the local-skeleton is equal to that of the global skeleton.

The only value to compute for updating the position of the local-skeleton node is the distance of the node point from its corresponding global-skeleton node point (see Figure 3). First, the current distance between  $N_{gi}$  and  $N_{g(i-1)}$  is calculated,  $d_{gi}$ . The root node,  $N_{g0}$ , of the local-skeleton is set equal to the root node of the global-skeleton. The distance between  $N_{li}$  and  $N_{l(i-1)}$  of the local-skeleton is fixed,  $d_{li}$ . Therefore,  $d_i$ , the distance of  $N_{li}$  from  $N_{gi}$ , is calculated by:

$$d_i = \sqrt{d_{li}^2 - d_{gi}^2}$$

Thus, as the global-skeleton elongates, increasing  $d_{gi}$ , the local-skeleton straightens and the node points of the local-skeleton move closer to the global-skeleton, decreasing  $d_i$ .

### 4.4 Localized Collision Detection

The local-skeleton is used for collision detection since the rendered strands adhere to its motion. We utilize the collision detection system proposed by [11, 13], which uses *swept sphere volumes* (SSVs) [21] to encapsulate the hair. The radii of the cross-sections at each local-skeleton node point define the offset for the SSVs.

When a collision is detected between the hair and body, the global-skeleton of the hair section is moved so that the local-skeleton (and hair geometry) are outside of the body. All positional changes are first made on the global-skeleton and the local-skeleton follows the changes as described in Section 4.2. It is important to note that if the global-skeleton collides with an object, but the SSVs of the hair, and the corresponding local-skeleton, do not collide with the object, then no action is taken.

Moreover, our method efficiently detects hair-hair collisions by decomposing the space encompassing the hair into three-dimensional grids. SSVs that fall in the same grid cell are tested against each other for overlap. If two sections of hair overlap, we use the response method of [5, 13] to push apart the hair sections based on their respective orientations.

This localized collision detection method allows the local-skeleton to be positioned farther away from the global-skeleton, increasing  $d_i$ , while maintaining accurate collision detection. This approach results in creating full, voluminous hair. Our dual-skeleton system is thus able to model more diverse types of hairstyles than most existing techniques.

## 5 Modifications of Physical Properties

We now account for various physical changes that occur when hair absorbs water or when fixative products are applied. The main characteristics we emphasize are:

- **Adjusting dynamics properties** to account for changing mass and spring stiffness of the hairs *on-the-fly*;
- **Flexible geometric structure** to allow the hair volume to change due to the presence of external substances;
- **Bonds between strands** that form and break *dynamically* reflecting the connection of strands due to water or styling products.

### 5.1 Adjustment of Dynamic Properties

The first two physical changes on hair we account for are the changes in mass and spring stiffness. As water is absorbed into hair fibers,

the mass of the hair increases up to 45% and the stiffness of the hair fibers increases by a factor of three. As styling products are applied to hair, the stiffness of the hair is increased and the mass of the hair is also increased due to the presence of additional substances.

External forces, such as wind and gravity, are applied to the node points following the standard equation of force:

$$F_i = m_i * a_i$$

$F_i$  is the force applied to the  $i$ th node,  $m_i$  and  $a_i$  are the mass and acceleration of the  $i$ th node, respectively. We vary the mass of the nodes to correspond to the length of the strand representing the nonuniform weight of strands from the root of the strand to its tip.

As the fraction of wetness of the hair,  $f_{wetness}$ , increases to 100%, the mass of the hair increases up to 45% of its initial dry weight. The mass of the  $i$ th node,  $m_i$ , is then calculated as:

$$m_i = m_{dryi} + (m_{dryi} * 45%)f_{wetness}$$

where  $m_{dryi}$  is the initial, dry mass of the  $i$ th node.

The internal forces acting on each node point  $i$  consist of angular torque,  $M_{\theta i}$  and  $M_{\phi i}$  for the  $\theta$  and  $\phi$  components, as well as the spring force controlling the length of each global-skeleton section,  $F_{leni}$ . All of these internal forces are spring forces containing separate spring constants,  $k_{\theta i}$ ,  $k_{\phi i}$ ,  $k_{leni}$ , respectively. The final spring force equations become:

$$\begin{aligned} M_{\theta i} &= -k_{\theta i}(\theta_i - \theta_{i0}), \\ M_{\phi i} &= -k_{\phi i}(\phi_i - \phi_{i0}), \\ F_{leni} &= -k_{leni}(d_{gi} - d_{gi0}), \end{aligned}$$

where  $\theta_i$ ,  $\phi_i$ ,  $\theta_{i0}$ ,  $\phi_{i0}$  are the current angle values and resting angles of the  $i$ th node in polar coordinates, respectively and  $d_{gi}$  and  $d_{gi0}$  are the current and resting lengths of the  $i$ th segment of the global-skeleton, respectively. With a high, or stiff,  $k_{len}$  value, the hair will be able to bend freely in the  $\theta$  and  $\phi$  directions, but will not stretch or compress as liberally.

As styling products are applied to the hair, the spring constant  $k_{len}$  is increased. We found that by increasing just the  $k_{len}$  value we obtained motion results similar to that of hairspray and other fixative products. The amount to increase the stiffness depends on the product that is being used. However, we found that by using an implicit integration technique [13], we were able to increase this spring constant by a factor of 10 and maintain a stable simulation.

## 5.2 Flexible Geometric Structure

As explained in Section 3.1, as hair gets wet it becomes less voluminous. To account for this property, when water is applied to the hair, the radii of the hair sections decrease accordingly. At 100% wetness, the thickness of a group of strands will be equal to the number of strands times the thickness of a strand:

$$\begin{aligned} AreaOfGroup &= (AreaOfStrand) * n; \\ WR_i &= \sqrt{n} * r \end{aligned}$$

where  $WR_i$  is the radius of the  $i$ th cross-section at 100% wetness,  $n$  is the number of strands in the current strand group, and  $r$  is the radius of a single strand.

We extend this radius contraction to variable amounts of wetness by creating a linear relationship based on the amount of wetness in the system:

$$CR_i = DR_i - (DR_i - WR_i)f_{wetness}$$

where  $CR_i$  is the current radius of the  $i$ th cross-section,  $DR_i$  is the dry radius of the  $i$ th cross-section, and  $f_{wetness}$  is the fraction of water absorbed at the given time.

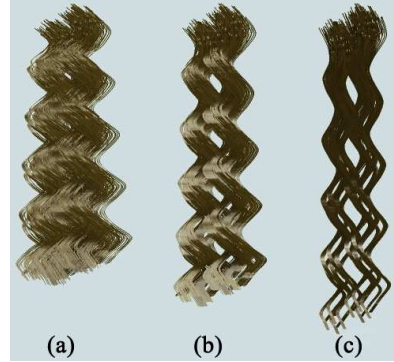


Figure 4: Sections of curly hair progressively getting wet: (a) 0% wetness (dry) (b) 50% wetness (c) 100% wetness.

Not all strands of hair are exactly the same length, so our system reflects this observation by varying the radius at each level of the strand grouping. These effects are illustrated in Figure 4, which shows a section of curly hair at 0%, 50%, and 100% wetness.

As the radii of the strand groups fluctuate, the offsets of the SSVs used for collision detection are automatically updated reflecting the change. This process is performed on-the-fly allowing the radii of the strand groups to change dynamically.

### 5.3 Dynamic Bonds between Strands

Due to the bonding effects of most fixative products, hair strands tend to adhere to each other where the product has been applied. We have extended the use of the *static links* exercised in [7], which were used as breakable connections between guide strands to enable hairstyle recovery. In [7], these links were selected and setup at the beginning of a hair simulation and were broken when they encountered excessive forces. In our system, “dynamic bonds” that model bonding forces between sections of hair are created on-the-fly when fixative products are applied. They can be created at any point in a simulation at any place along the dual-skeletons.

Our dynamic bonds are modeled as spring forces connecting two separate nodes of nearby global-skeletons. Our hair-hair collision detection method, described in Section 4.4, identifies which sections of hair are touching. When a styling product is applied to the hair, each section of hair maintains a list of the hair sections with which it is in contact. The dynamic bonds are then formed connecting the corresponding sections. A single section can have as many bonds as hair sections it is touching. The new bonding spring force,  $f_{bond}$ , between two global-skeleton nodes becomes:

$$f_{bond} = -k_{bond}(d_{current} - d_{initial})$$

where  $k_{bond}$  is the spring constant of the bond,  $d_{current}$  is the current distance between the nodes and  $d_{initial}$  is the distance between the two nodes when the fixative product is first applied.

Following methods similar to [7], these bonds are broken when a large force is applied to it. The force required to break the bond is directly related to the strength of the spring constant  $k_{bond}$ , which is determined based on the amount and strength of fixative product applied.

These dynamically forming and breaking bonds cause large sections of hair to group together and move in union, reflecting the clumping effect of real fixative products. Moreover, these bonds restrict the strands’ ability to move over or past each other, an effect exhibited in real hairs due to the increased frictional force caused by fixative products on hair [19].

## 6 Rendering

We model the hair strand reflections by light scattered from a cylindrical surface[14] and encode an anisotropic lighting equation in a texture map as in [15]. Self-shadows are generated by accumulating the opacity  $\alpha$  of the strands hit by the light rays along the light direction in the hardware framebuffer, a technique called *opacity shadow maps* [16].

### 6.1 Influence of Wetness

Wetness can make objects look darker, brighter, and/or more specular. As noted by [22], the differences in appearance are caused by a combination of the presence of liquid on the surface and inside the material. When hair becomes wet, a thin film of water is formed around the fibers. The rough, tiled air-fiber interface, observed by [18], changes to a smooth, mirror-like air-water interface. Obviously, this change gives the hair a shinier appearance due to specular reflections, typically modeled by an increasing fall-off exponent in the Phong illumination shading model.

Due to water absorbed inside and surrounding the hair fibers, the relative index of refraction decreases. The absorption of light inside the hair then increases due to a greater amount of total internal reflection. This phenomenon causes a darker appearance of wet hair in comparison to dry hair. The increased opacity value also leads to more aggressive self-shadowing.

Moreover, this relation implies that a smaller fraction of light that is radiated by the hair has traversed the fiber core. Hair fibers have a pigmented core at the center of the cylindrical volume, the source of its apparent color. Therefore, only the fraction of light rays that traversed the hair core are responsible for the color we observe [18]. Consequently, with increasing wetness, the hair radiates an increasing fraction of colorless light, originated at the air-water surface reflections. This effect contributes to the shinier appearance of wet hair over dry.

The three main parameters in our shading algorithm are:  $\alpha$ , which controls the opacity shadow map [16],  $s$  is the exponent for the specular reflection term, and  $f_a$  determines the contribution of (partly) non-colored anisotropic reflection. The rendering equation for each hair

strand can be expressed as:

$$I_o = (1 - f_a)k_a I_d + f_a(k_d \langle L, N' \rangle + k_s \langle V, R \rangle^s) I_i$$

where  $L$  is the light direction,  $N'$  is the projection of the light vector  $L$  onto the normal plane [15], and  $R$  is the reflected direction. The diffuse color  $k_a I_d$  is simply the strand color, while the incoming light from the scene is  $I_i$ .

We have captured the interactions of light with the wet strands by varying the rendering parameters based on the amount of water present on the hair. More specifically, we control the parameter vector  $\mathbf{V}_p = [\alpha, s, f_a]$ . Extreme values are empirically obtained, defined by  $\mathbf{V}_p^{min}$  and  $\mathbf{V}_p^{max}$ . Depending on the wetness percentage  $f_{wetness}$ , we then linearly interpolate according to the following formula:

$$\mathbf{V}_p = \mathbf{V}_p^{min} + f_{wetness}(\mathbf{V}_p^{max} - \mathbf{V}_p^{min})$$

As the wetness factor varies between 0% and 100%, the parameters vary accordingly, creating a damped or wet look for the hair strands.

## 7 Results and Analysis

We have implemented our hair modeling system in C++ and displayed the images using OpenGL.

### 7.1 Comparison

The results of our simulations are illustrated in Figure 5, showing, from left to right, the same hairstyle (red long, curly hair) blowing in the wind with different effects. Note that the wet hair is not as voluminous as the dry hair. Also, the hair with fixative products present retains tighter curls than the dry hair when carried by the wind, and moves together in larger bundles of hair. For further demonstrations and additional images, please visit our project website:

<http://gamma.cs.unc.edu/HairWS>

Our system requires no pre-computations to dynamically add water or styling products to the hair. As a result, our simulations run at approximately the same rate with or without external substances present. On average the results took 4.16 seconds per frame for simulation and 0.34 seconds per frame for rendering. Timings were taken on a PC with a 1.8 Ghz processor, 1 GB RAM, and a GeForce 4 graphics card, for the hair shown in Figure 5. This hairstyle has an average of 16 nodes per strand and a total of 9,680 rendered strands.

### 7.2 Limitations

Our current implementation applies a general wetness factor, as well as amount and type of fixative product, to the entire head of hair to illustrate the general effects of the substances on the hair. However, the ability to apply water and styling products to a specific section of hair is a feature that would be useful to an interactive virtual hairstyling system.

Furthermore, while a ray-tracing or photon-mapping implementation would give more accurate rendering results, we chose to use an approximate, faster rendering algorithm so that our work could be easily integrated with techniques using multiresolution representations [10, 11, 12, 13] for interactive styling.

## 8 Conclusion and Future Work

We presented several simple yet effective techniques to account for the influence of water and styling products on hair behavior and visual appearance. The methods we have presented may be used together or integrated separately into varying hair modeling schemes. We plan to integrate these techniques with multiresolution representations [10, 11, 12, 13] to further improve runtime performance and integrate the resulting system with a 3D interface for interactive virtual hairstyling.

## Acknowledgements

This project is supported in part by Intel Corporation, Army Research Office, National Science Foundation and Office of Naval Research.

## References

- [1] K. Anjyo, Y. Usami, and T. Kurihara. A simple method for extracting the natural beauty of hair. *Computer Graphics*, 26(2):111–120, 1992.
- [2] 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.
- [3] T. Kurihara, K. Anjyo, and D. Thalmann. Hair animation with collision detection, models and





Figure 5: Comparison images of long, curly, red hair blowing in the wind. From left to right: showing (1) hair at 100% wetness (2) clean, dry hair (3) hair with fixative products present.

- techniques. *Proc. of Computer Animation*, pages 128–138, 1993.
- [4] 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.
- [5] E. Plante, M. Cani, and P. Poulin. Capturing the complexity of hair motion. *GMOD*, 64(1), Jan 2002.
- [6] Z. Xu and X. D. Yang. V-hairstudio: An interactive tool for hair design. *IEEE Computer Graphics and Applications*, 21(3):36–43, 2001.
- [7] J. Chang, J. Jin, and Y. Yu. A practical model for hair mutual interactions. *Proc. of ACM Symposium on Computer Animation*, 2002.
- [8] S. Hadap and N. Magnenat-Thalmann. Modeling dynamic hair as a continuum. *Computer Graphics Forum (Proc. of Eurographics 2001)*, 20(3), 2001.
- [9] D.-W. Lee and H.-S. Ko. Natural hairstyle modeling and animation. *Graphical Models*, 63(2):67–85, March 2001.
- [10] T.-Y. Kim and U. Neumann. Interactive multiresolution hair modeling and editing. *Proc. of SIGGRAPH*, 2002.
- [11] K. Ward, M. Lin, J. Lee, S. Fisher, and D. Macri. Modeling hair using level of detail representations. *Proc. of Computer Animation and Social Agents*, 2003.
- [12] F. Bertails, T.-Y. Kim, M.-P. Cani, and U. Neumann. Adaptive wisp tree - a multiresolution control structure for simulating dynamic clustering in hair motion. *Proc. of ACM SIGGRAPH / Eurographics Symposium on Computer Animation*, 2003.
- [13] K. Ward and M. Lin. Adaptive grouping and subdivision for simulating hair dynamics. *Proc. of Pacific Graphics*, 2003.
- [14] J. T. Kajiya and T. L. Kay. Rendering fur with three dimensional textures. In Jeffrey Lane, editor, *Computer Graphics (SIGGRAPH '89 Proceedings)*, volume 23, pages 271–280, July 1989.
- [15] 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.
- [16] T.-Y. Kim and U. Neumann. Opacity shadow maps. *Proc. of Eurographics Rendering Workshop*, 2001.
- [17] T. Lokovic and E. Veach. Deep shadow maps. In *Proceedings of ACM SIGGRAPH 2000*, pages 385–392, 2000.
- [18] S. R. Marschner, H. W. Jensen, M. Cammarano, S. Worley, and P. Hanrahan. Light scattering from human hair fibers. *ACM Trans. Graph.*, 22(3):780–791, 2003.
- [19] D. Johnson. *Hair and Hair Care; Cosmetic Science and Technology Series*, volume 17. New York Marcel Dekker, Inc., 1997.
- [20] L'Oreal. <http://www.loreal.com>, 2004.
- [21] 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.
- [22] H. W. Jensen, J. Legakis, and J. Dorsey. Rendering of wet material. *Rendering Techniques*, pages 273–282, 1999.

# A Simulation-based VR System for Interactive Hairstyling\*

Kelly Ward<sup>†</sup>

Nico Galoppo

Ming C. Lin

The University of North Carolina at Chapel Hill  
{wardk,nico,lin}@cs.unc.edu

## ABSTRACT

We have developed a physically-based VR system that enables users to interactively style dynamic virtual hair by using multi-resolution simulation techniques and graphics hardware rendering acceleration for simulating and rendering hair in real time. With a 3D haptic interface, users can directly manipulate and position hair strands, as well as employ real-world styling applications (cutting, blow-drying, etc.) to create hairstyles more intuitively than previous techniques.

## 1 INTRODUCTION

Virtual environments created for interactive hairstyling can be used to understand and specify detailed hair properties for several applications, including cosmetic prototyping, education and entertainment, and cosmetologist training. Accurate virtual hairstyling requires both high performance simulation and realistic rendering to enable interactive manipulation and incorporation of fine details. The appearance of hairstyles results from physical properties of hair and hair mutual interactions. Therefore, hair dynamics should be incorporated to mimic the process of real-world hairstyle creation. However, due to the performance requirement, many interactive hair modeling algorithms tend to lack important, complex features of hair, including hair interactions, dynamic clustering of hair strands, and intricate self-shadowing effects.

An intuitive virtual hairstyling tool needs to take into account user interaction with dynamic hair. Until recently, the complexity of animating and rendering hair had been too computationally costly to accurately model hair's essential features at desired rates. As a result, many hairstyling methods ignore dynamic simulation and/or user interaction, which creates an unnatural styling process in comparison to what would be expected in practice.

**Main Results:** In this paper, we present a *physically-based* virtual reality system that mimics real-world hairstyling processes and requires no knowledge other than common hair manipulation techniques. By using multi-resolution simulation techniques and programmable graphics hardware, we developed a physically-based virtual hair salon system that animates and renders hair at accelerated rates, allowing users to interactively style virtual hair in a natural manner. With an intuitive 3D haptic interface, users can directly manipulate and position hair strands, as well as employ real-world styling applications (e.g. cutting, wetting, applying styling products) to create hairstyles as they would in the physical world. The main characteristics of our system are the following:

- **Direct 3D hair manipulation with a haptic interface:** We use a commercially available haptic device to provide an intu-

\*Supported in part by Army Research Office, National Science Foundation, Office of Naval Research, and Intel Corporation.

<sup>†</sup>now at Walt Disney Feature Animation



Figure 1: Hairstyle interactively created using our system.

itive 3D user interface, allowing the users to directly manipulate the hair in a manner similar to real-world hairstyling.

- **Visually plausible rendering:** By exploiting the capability of programmable graphics hardware and multi-resolution representations, we can render plausible hair appearance due to self-shadowing, wet surfaces, etc. in real time on current commodity desktop PCs.
- **Multi-resolution hairstyling:** We achieve interactive hair simulation by using level-of-detail representations, which accelerate dynamics computation and enable adaptive hair clustering and subdivision on the fly.
- **Physically-based interaction:** By modeling hair's properties and dynamic behavior in the presence of water and styling products, we introduce the ability to interactively apply hair-spray, wet, blow-dry, cut, and manipulate hair as in the physical world, like no other systems can at present.

Figure 1 illustrates a hairstyle created by a naive user using our virtual hairstyling system in less than 10 minutes.

**Organization:** The rest of this sketch is organized as follows. Related work is briefly reviewed in Section 2. Section 3 presents the user interface for the system. The dynamic simulation and rendering of hair are described in Section 4. Details of user interaction and application features are discussed in Section 5. Finally, we conclude with some results and possible future research directions in Section 6.

## 2 RELATED WORK

Hair modeling involves hair shape modeling, dynamic hair simulation, and hair rendering. An overview of work in these areas can be found in [5]. Our summary of related material is limited to the multi-resolution simulation and rendering methods this work extends.

We use the three LOD representations and adaptive grouping and splitting process introduced in [11, 10]. The representation and resolution of a volume of hair is controlled by a hair hierarchy that is constructed by the continual subdivision of hair strips, clusters, and strand groups. Throughout the simulation, the hair hierarchy is traversed on the fly to find the appropriate representation and resolution for a given section of hair adaptively.

In [11, 10], an appropriate LOD representation was chosen based on the hair’s importance to the application. Sections of hair that could be viewed well were simulated and rendered with high detail. Also, as a section of hair moves rapidly, a high LOD was used to capture the intricate detail, similarly performed by [1]. These criteria aided in accelerating simulation and rendering without losing much visual fidelity.

In this work, we couple the hair hierarchy with a simulation localization scheme (see Section 4.2) to achieve interactive hair animation. Moreover, our LOD selection criteria differ from that of [11, 10] to include areas of user interaction and an additional emphasis on the hair’s motion.

### 3 USER INTERFACE

Our prototype system uses a SensAble Technologies’ PHANToM as a 3D user input device. The real-time display of the PHANToM input is rendered using a commercial haptic toolkit called *GHOST*. The position and orientation of the device are updated and rendered at each frame.

A 2D menu is projected onto the 3D scene containing the avatar and hair model. Figure 2 illustrates a user operating the system. The user can interact with both the 3D scene and 2D menu using the PHANToM stylus in a seamless fashion. The user interactively positions the stylus over the 2D menu icons and pushes the stylus button to choose the desired application. The position, orientation, and range of influence of the application are depicted in the scene. As the user moves the stylus, the application’s area of influence interactively follows the position and orientation of the user’s hand in 3D. The camera is controlled through the mouse.



Figure 2: User Interface: PHANToM provides 3D user input and 2D menu buttons are labeled with icons to show applications.

### 4 INTERACTIVE HAIR SIMULATION AND RENDERING

Hairstyling in the natural world is performed by focusing on specific sections of hair and executing a desired task to a given area. This focus correlates naturally with the use of multi-resolution techniques to simulate hair. Moreover, accelerated hair rendering is required for user interaction.

#### 4.1 Dynamics and Collision Detection

Each LOD hair representation (strips, clusters, and strands) follows the same dynamics model for motion. We use the dual-skeleton

system introduced by [9] for controlling the hair’s dynamics. The dual-skeleton system is compelling in that it can capture the details of typical dry hair as well as wet hair and hair with hairspray, or some other styling products applied.

We have utilized the localized collision detection model of [9] that is based on the dual-skeleton setup and the family of *swept sphere volumes* (SSVs). A swept sphere volume is created by taking a core shape, such as a point, a line, or a rectangle, and growing outward by some offset. The SSVs encapsulate the hair geometry (of any type or resolution LOD) and are used as bounding volumes for collision detection.

Both hair-hair and hair-object collision detection is performed by checking for intersection between corresponding pairs of SSVs; this is done by calculating the distance between the core shapes and subtracting the appropriate offsets. Hair-object collisions are handled by moving the hair section outside of the body and that hair section is restricted to only move tangential to or away from the object, based on the object’s normal direction. Hair-hair collisions are processed by pushing the hair sections apart based on their respective orientations as explained in methods by [8, 10].

#### 4.2 Simulation Localization

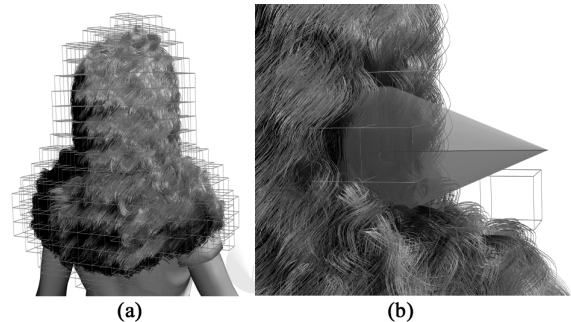


Figure 3: (a) Shows all of the grid cells that contain hair geometry (b) Highlights the cells that will be effected by the current application (applying water).

We use a spatial decomposition scheme to rapidly determine the high activity areas of the hair; these areas are then simulated with finer detail. We use a uniform grid consisting of axis-aligned cells that encompass the area around the hair and human avatar. Spatial decomposition schemes have been utilized previously for hair-hair interaction methods where sections of hair that are located in the same cell will be tested against each other for overlap. We have extended this process to all features of hair simulation, not just collision detection.

##### 4.2.1 Insertion into the Grid

The polygons of the avatar, or other objects, are placed into the grid to determine potential collisions with the hair. Object positions only need to be updated within the grid if the object is moving, otherwise the initial insertion is sufficient.

Every time a section of hair moves, or the skeleton for simulation is updated, its line swept spheres (LSSs) or rectangular swept spheres (RSSs) are inserted into the grid. An SSV is inserted into the grid by determining which cells first contain the core shape of the SSV (line or rectangle), then the offset of the SSVs are used to determine the remaining inhabited cells. Figure 3(a) shows the grid cells that contain hair geometry.

When the user employs an application (e.g. spraying water, grabbing the hair) the grid is used to indicate which portions of the hair are potentially affected by the user’s action. As the user moves the PHANToM stylus, its position and orientation are updated. Each

application has an *area of influence* that defines where in space its action will have an effect. This area is defined as a triangle for the cutting tool and a cone for the remaining tools. The cone of influence is defined by the application’s position, orientation, length, and cutoff angle. These properties define the cone’s position in the grid. Inserting the cone becomes similar to inserting an LSS, but the offset becomes a variable of radius along the core line (an SSV has a constant offset along its core shape). The triangle for cutting is defined by the space between the open blades of a pair of scissors.

#### 4.2.2 Retrieval from the Grid

Grid-cells that contain both impenetrable triangles (from the avatar or another object in the scene) and hair geometry are marked to be checked for hair-object collision. Only these cells contain a potentially colliding pair. Similarly, any grid cell containing more than one section of hair is marked to be checked for hair-hair collisions.

Likewise, the grid maintains a list of grid-cells where the user interaction cone or triangle has been inserted. Any of these grid cells that contain hair geometry are returned and the sections of hair within the cell are independently checked to see if they fall within the area of influence, see Figure 3.

#### 4.3 Multi-Resolution Simulation with the Grid

The grid aids us to localize our simulation on the areas of highest importance to the user. These areas are defined based on their distance from the viewer, visibility, motion, and the user’s interaction with the hair. We adopted the method for choosing a level of detail based on distance and visibility created previously by [11], but have used our grid-based system to expand on the motion criteria and to include the user’s interaction with the hair.

The motion of the hair is highly pertinent to the amount of detail needed to simulate the hair. Most applications performed on hair are localized to a small portion of the hair; the majority of the hair thus lies dormant. The sections of hair that are dormant are modeled with a lower LOD representation and resolution, determined by comparison against velocity thresholds, but we have gone a step further by effectively “turning-off” simulation for areas where there is no activity.

Each grid cell keeps track of the activity within the cell, tracking the hair sections that enter and exit the cell. When the action in a given cell has ceased and the hair sections in the cell have a zero velocity, there is no need to compute dynamic simulation due to gravity, spring forces, or collisions. The positions of the hair sections are thus frozen until they are re-activated. The cell is labeled as dormant and does not become active again until either the user interacts with the cell or until a new hair section enters the cell.

Rapid determination of the active cells and hair sections allows us to place the computational resources towards dynamic simulation for the hairs of highest interest to the user.

#### 4.4 Real-Time Rendering

In order to guarantee a convincing and interactive experience for the user, our rendering algorithm has to run in real time and produce realistic images at the same time.

We implemented two separate specular highlights, due to the multiple modes of scattering inside and on the surface of the hair fibers observed by Marschner et al. [6]. We compute both specular terms by shifting the hair tangent in Kayija’s original formulation [3] towards the hair root and towards the hair tip applying separate falloff exponents. The shifted tangents are used in the formulation proposed in [2]. All operations were performed in a fragment program for efficiency.

Realistic hair self-shadowing effects are hard to implement efficiently due to the large amount of dynamic geometry and the fineness of the hair strands. Our self-shadowing algorithm is based on opacity shadow maps created by [4] and recent GPU features [7]; it generates 16 opacity shadow maps in only one pass with multiple render targets, plus an extra pass for the actual rendering, without depth ordering required. As in [4], the opacity maps are placed at uniformly sampled distances from the eye, orthogonal to the view direction. Each of the four render targets holds four opacity maps, one in each 16-bit floating point component.

### 5 USER INTERACTION AND APPLICATIONS

Given our system for simulating and rendering hair described in the previous section, a user can now directly interact with hair through the 3D user interface and use operations commonly performed in hair salons. The operations supported in our system are described in this section.

#### 5.1 Hair Cutting

Cutting hair is crucial for changing a hair’s style, see Figure 4. Our cutting method models cuts performed with scissors. We model all the features of the cut, including capturing the hair that falls away or that is “cut-off”. The location for cutting is defined by a triangle formed by the space between the open blades of scissors. Hair skeletons that intersect this triangle are then cut. At the cutting location, the skeleton  $S$  is split into two separate skeletons,  $S_1$  and  $S_2$ ;  $S_1$  remains attached to the scalp, while  $S_2$  falls down.

At the intersection of skeleton  $S$  and the cutting triangle, two new control points are created. One control point becomes the last point of skeleton  $S_1$ , while the second becomes the first point of  $S_2$ . The geometry of the fallen hairs remains consistent with the geometry of the hair below the sever point before the cut is performed; curliness, wetness, hair distribution and other properties are maintained in the fallen hair segments.

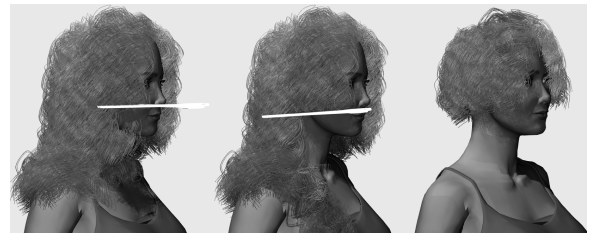


Figure 4: Example of haircutting, far right shows final style.

#### 5.2 Applying Water, Hairspray and Mousse

Wet hair is modeled using the technique described in [9]. When water is applied to the hair, the mass points of the global-skeleton become heavier with the mass of the water. The overall motion of the hair is limited due to the extra weight and if the hair is curly, the global-skeleton will stretch under the extra weight and the curls will lengthen as expected. The volume of the hair in the areas where water is applied is decreased by constricting the radius of the current hair representation (strand grouping, cluster, or strip); these hair segments are then rendered to show the wet appearance as described by [9].

Hairspray is simulated on the hair by increasing the spring constants of the global-skeleton where it is applied. Moreover, *dynamic bonds* [9] are added between sections of hair that are in contact when the hairspray is applied. Dynamic bonds are extra spring

forces that model the adhesive quality of hairspray to make the hair move as a group throughout subsequent motions.

We have chosen to model hair mousse that adds volume to hair. We inject volume into the hair model by growing the radii of the hair sections it affects. This process makes the hair fuller without adding more strands or skeletons.

### 5.3 Grabbing and Moving Hair

Typically when hair is clasped in the real world, a group of strands are selected at a local point along the strands. The user presses the stylus button and the control points that fall within the cone of influence are decided. Among these selected control points, only one control point per dual-skeleton is permitted to be *grabbed*. If multiple control points of a single dual-skeleton fall within the cone, the point that comes closest to the cone's center will be chosen.

In the grabbed state, as the user moves the stylus, the grabbed-point will follow the motion of the stylus. A grabbed-point cannot be pulled beyond its normal reach span (decided by its position in the dual-skeleton). The length of the hair is always maintained so that the lengths above the grabbed-point and below it are of consistent lengths while the point is moving. When the user wishes to release the grabbed-point(s), he or she releases the button of the stylus and the former grabbed-points will fall due to gravity.

### 5.4 Hairdryer

Hairdryers are one of the most common tools in a hair salon. When the stylus button is pressed, a strong constant force is applied in the direction of its orientation. Any control points that fall within the cone of influence receive this strong force. Moreover, if a wet control point (see Section 5.2) is influenced by the hairdryer, the control point will “dry”; the amount of water will decrease over the length of exposure dependent on the strength of the hairdryer force.

## 6 RESULTS AND DISCUSSION

Our virtual hairstyling system demonstrates the usefulness of our multi-resolution techniques for interactive hair modeling and was implemented in C++. The initial hairstyles are loaded as a pre-process.



Figure 5: Comparison between real (top) and virtual use of common hair salon activities (left) applying water (right) blow-drying hair.

We have been able to allow physically-based user interaction with dynamic hair while modeling several common hair salon applications. Figure 5 shows a comparison of real hair under the influence of common hairstyling applications with virtual hair styled

by our system under the same conditions. Level-of-detail representations coupled with our simulation localization scheme have accelerated the animation of hair so that a user can actually interact with it.

Dynamic simulation, including implicit integration, LOD selection, hair applications (wetting, cutting, etc.), and collision detection, to create the hair model shown in Figure 1 ran at an average of 0.092 sec/frame. This figure comprised between 37 to 296 skeleton models, determined on-the-fly throughout the simulation, with an average of 20 control points each. At the finest resolution, the model contained 8,128 rendered strands; throughout the simulation the rendering LOD contained between 6K and 1,311K rendered vertices. Lighting and shadow computations on the GPU were performed in 0.058 sec/frame on average. The benchmarks were measured on a desktop PC equipped with an Intel® Xeon™ 2.8Ghz processor with 2.0 GB RAM and an NVIDIA® GeForce™ 6800 graphics card.

## 7 SUMMARY

We presented a prototype VR system involving an intuitive 3D user interface and methods for animating and rendering hair that allows for a user to interactively manipulate hair through several common hair salon applications. This system provides a level of user interaction that has been previously too complex to achieve. We are interested to explore the use of two-handed haptic gloves with our system to provide higher fidelity force feedback to the user, while allowing for further interaction capability and the creation of even more complex hairstyles. User studies involving professional stylists and novices on using this VR salon system would help to determine the effectiveness of this work.

## REFERENCES

- [1] F. Bertails, T-Y. Kim, M-P. Cani, and U. Neumann. Adaptive wisptree - a multiresolution control structure for simulating dynamic clustering in hair motion. In *ACM SIGGRAPH Symposium on Computer Animation*, pages 207–213, July 2003.
- [2] 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.
- [3] J. T. Kajiya and T. L. Kay. Rendering fur with three dimensional textures. In *SIGGRAPH '89: Proceedings of the 16th annual conference on Computer graphics and interactive techniques*, pages 271–280. ACM Press, 1989.
- [4] Tae-Yong Kim and Ulrich Neumann. Opacity shadow maps. In *Proceedings of the 12th Eurographics Workshop on Rendering Techniques*, pages 177–182. Springer-Verlag, 2001.
- [5] N. Magnenat-Thalmann and S. Hadap. State of the art in hair simulation. In *International Workshop on Human Modeling and Animation*, Korea Computer Graphics Society, pages 3–9, June 2000.
- [6] Stephen R. Marschner, Henrik Wann Jensen, Mike Cammarano, Steve Worley, and Pat Hanrahan. Light scattering from human hair fibers. *ACM Trans. Graph.*, 22(3):780–791, 2003.
- [7] NVIDIA. Technical report, NVIDIA, 2005. <http://www.nvidia.com/>.
- [8] E. Plante, M-P. Cani, and P. Poulin. Capturing the complexity of hair motion. *Graphical Models (Academic press)*, 64(1):40–58, January 2002.
- [9] K. Ward, N. Galoppo, and M. C. Lin. Modeling hair influenced by water and styling products. In *International Conference on Computer Animation and Social Agents (CASA)*, pages 207–214, May 2004.
- [10] K. Ward and M. C. Lin. Adaptive grouping and subdivision for simulating hair dynamics. In *Pacific Graphics Conference on Computer Graphics and Applications*, pages 234–243, October 2003.
- [11] K. Ward, M. C. Lin, J. Lee, S. Fisher, and D. Macri. Modeling hair using level-of-detail representations. In *International Conference on Computer Animation and Social Agents*, pages 41–47, May 2003.



# Predicting Natural Hair Shapes by Solving the Statics of Flexible Rods

Florence Bertails<sup>1</sup>, Basile Audoly<sup>2</sup>, Bernard Querleux<sup>3</sup>, Frédéric Leroy<sup>3</sup>, Jean-Luc Lévêque<sup>3</sup> and Marie-Paule Cani<sup>1</sup>

<sup>1</sup> GRAVIR-IMAG/INRIA, Grenoble, France

<sup>2</sup> LMM-CNRS/Univ. Paris 6, Paris, France

<sup>3</sup> L'Oréal Recherche, France



Figure 1: Virtual versus real hair ringlets

## Abstract

*This paper presents a new physically-based method for predicting natural hairstyles in the presence of gravity and collisions. The method is based upon a mechanically accurate model for static elastic rods (Kirchhoff model), which accounts for the natural curliness of hair, as well as for hair ellipticity. The equilibrium shape is computed in a stable and easy way by energy minimization. This yields various typical hair configurations that can be observed in the real world, such as ringlets. As our results show, the method can generate different hair types with a very few input parameters, and perform virtual hairdressing operations such as wetting, cutting and drying hair.*

Categories and Subject Descriptors (according to ACM CCS): I.3.5 [Computer Graphics]: Physically based modeling

## 1. Introduction and Previous Work

Due to the relevance of hair in the overall appearance of a character, many recent works have focused on the challenging issue of virtual hairstyling. However, there is currently no hairstyling method able to account for the structural properties of the different hair types existing in the world - Asian, African and Caucasoid hair - nor to predict how a given hair would look as it grows or gets wet. Such a model would prove very useful for performing virtual hairdressing operations on any given hair type.

The geometry of human hair results from complex biological and mechanical processes. Most CG hairstyling methods reproduce this result using procedural techniques and manual editing [KN02, CK05]. This gives a high controllability, but also requires some significant input and modeling skills from the user. Several attempts have been made to use physically-based modelling in the design of hairstyles [AUK92, HMT00]. In these approaches, the finest geomet-

ric hair details, such as curls or waves, are still added in a procedural way. Image-based reconstruction techniques have recently proved useful in capturing the geometry of real hair [PBS04]. However, as they only model the visible part of hair, these methods can hardly capture the geometry of curly hair because of the occlusion issue.

Our method is physically-based, and relies on the Kirchhoff equations for static Cosserat rods. This accurate mechanical model, which accounts for the curvatures and twist deformation of elastic rods, was first introduced to the CG community by Pai [Pai02]. Our specific contributions are:

1. An improvement of Pai's model, namely a new formulation of the Kirchhoff equations for an elastic rod in the static case, based on energy minimization. This allows for handling external forces such as response to collisions, while still providing efficiency and robustness.
2. The extension of Pai's model to handle elliptic strands. This is mandatory for dealing with hair from different

ethnies, since both the eccentricity<sup>1</sup> and the natural curliness play a great role in the overall hair shape.

3. A very simple handling of basic hairstyling operations such as wetting, cutting and drying, since only a few intuitive physical parameters need to be changed (length, mean radius and Young modulus). This opens the way for the design of future virtual prototyping systems to be used by hairdressers.

## 2. Static Simulation of a Hair Strand

Building an accurate hair strand model first requires a good understanding of its geometric and mechanical properties. A hair strand is a very thin and light elliptic tube that deforms in an anisotropic way. Whereas it can easily bend and sometimes twist, it strongly resists stretching and shearing. A hair strand also has *elastic* properties as it tends to recover its original shape after the stress applied on it has been removed. Lastly, a real hair strand can be naturally straight, wavy or curly.

As suggested by Audoly and Pomeau [AP05], we represent a hair strand as a Cosserat rod obeying the Kirchhoff equations. These equations describe elastic rods subject to arbitrary external forces such as gravity.

### 2.1. Cosserat Model for Rods

We define a hair strand as an inextensible, unshearable rod, with a boundary condition on the position and the orientation at the root (imposed by the hair clamping on the scalp). In this section, the strand is submitted to the gravity field only.

In the Cosserat model, the configuration of a rod is described by its centerline, a space curve  $\mathbf{r}(s)$  (where  $s$  denotes the curvilinear abscissa of the rod), and a *material* frame  $\mathcal{F}(s) = (\mathbf{n}_1(s), \mathbf{n}_2(s), \mathbf{t}(s))$  attached to each point on this curve (see Figure 1, left). Usually, vector  $\mathbf{t}$  is the local tangent of the centerline  $\mathbf{r}(s)$ , and vectors  $\mathbf{n}_1$  and  $\mathbf{n}_2$  are lying in the plane of the local cross section of the rod. In our case,  $\mathbf{n}_1$  and  $\mathbf{n}_2$  are the principal axis of the elliptic cross section.

A Cosserat rod can locally bend in both directions  $\mathbf{n}_1$  and  $\mathbf{n}_2$ , or twist onto itself. The amount of bending around  $\mathbf{n}_1$  and  $\mathbf{n}_2$ , and the amount of twisting are respectively characterized by (material) curvatures  $\kappa_1$  and  $\kappa_2$ , and torsion  $\tau$ , through the following kinematics equations:

$$\frac{\partial \mathbf{u}_i}{\partial s} = \Omega \times \mathbf{u}_i \quad (1)$$

where  $\Omega = \kappa_1 \mathbf{n}_1 + \kappa_2 \mathbf{n}_2 + \tau \mathbf{t}$  is the rotation vector of the rod and  $\mathbf{u}_i$  is  $\mathbf{n}_1, \mathbf{n}_2, \mathbf{t}$  for  $i = 1, 2, 3$  respectively.

As previously mentioned, hair can be naturally curly. This intrinsic curliness can be expressed by natural curvatures and torsion  $\kappa_1^0, \kappa_2^0$  and  $\tau^0$ , which describe the state of the rod

<sup>1</sup> We use the mathematical definition of the eccentricity  $e$  of an ellipse. Thus,  $e = 0$  for a circle and  $e$  increases as the ellipse flattens.

with no applied force. In the case of hair, it is reasonable to assume that natural curvatures and torsion are roughly constant along the hair shaft. As a result, the configuration of a hair strand in the absence of gravity is a helix (this is easy to show, knowing that rotation vector  $\Omega$  is a constant vector).

### 2.2. Potential Energy

We are looking for stable equilibrium configurations of a strand in the gravity field. One solution would consist of solving the Kirchhoff's equations for static rods [Pai02]. However, the nonlinearities in these equations and the boundary conditions that need to be imposed at both the free and the clamped ends prevent the equations from being integrated in a single pass, and one is forced to iterate the integration. Instead, our new approach is based on energy minimization. It offers the advantage of being both robust and computationally cheaper, which is essential when handling a large number of hair strands. Our method can also account for collision forces, as demonstrated in section 3.1.

Finding static configurations of a physical system is equivalent to the search for its minimal potential energy. The potential energy of a rod of length  $L$  formulates as:

$$\mathcal{E}_{hair} = \mathcal{E}_g + \mathcal{E}_e \quad (2)$$

where  $\mathcal{E}_e$  is the internal elastic energy of the rod and  $\mathcal{E}_g$  the energy of the rod accounting for gravity. Assuming hair to be a rod of elliptic cross section and obeying Hooke's law for elasticity, the elastic energy  $\mathcal{E}_e$  can be written as:

$$\mathcal{E}_e = \int_0^L \left[ \frac{EI_1}{2} (\kappa_1(s) - \kappa_1^0)^2 + \frac{EI_2}{2} (\kappa_2(s) - \kappa_2^0)^2 + \frac{\mu J}{2} (\tau(s) - \tau^0)^2 \right] ds \quad (3)$$

where  $E$  is the Young modulus,  $\mu$  the Poisson ratio,  $I_1$  (resp.  $I_2$ ) the momentum of inertia of the rod's cross section with respect to  $\mathbf{n}_1$  (resp. to  $\mathbf{n}_2$ ) and  $J$  the axial momentum of inertia. Momenta of inertia depend on the principal radii of the cross section, and thus on hair's eccentricity  $e$ .

Potential gravitational energy  $\mathcal{E}_g$  can be written as:

$$\mathcal{E}_g = \rho S g \int_0^L z(s) ds \quad (4)$$

where  $\rho$  is the volumic mass of the rod,  $S$  the area of its cross section,  $g$  the gravity field value and  $z(s)$  the vertical coordinate of element  $ds$  at curvilinear abscissa  $s$ .

Note that the minimum for  $\mathcal{E}_{hair}$  results from a balance of two antagonistic effects: the tendency to recover a naturally helical shape, represented by  $\mathcal{E}_e$ , and the downward pull of gravity, represented by  $\mathcal{E}_g$ .

### 2.3. Numerical Solving

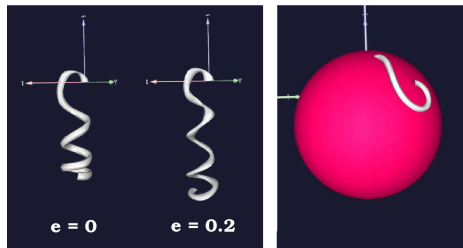
We first divide the rod into  $n$  slices  $s_i$  of equal length  $ds$ , for the sake of simplicity. Along each (small) slice  $s_i$ , curvatures and torsion are assumed to be constant. We note  $C_n$  the vector of size  $3 \times n$  composed of the  $n$  curvatures  $\kappa_1^i, \kappa_2^i$  and torsions  $\tau^i$ . Given the fact that the initial material frame  $\mathcal{F}$

is imposed by the clamping into the scalp, vector  $C_n$  defines a unique configuration for the rod.

Our aim is first to find the vector  $C_n$  that minimizes the discrete energy  $\mathcal{E}_{hair}$ , and then to compute the corresponding configuration of the rod. Our algorithm is as follows:

- We first initialize energy  $\mathcal{E}_{hair}$  and vector  $C_n$ ;
- Then, until energy  $\mathcal{E}_{hair}$  stops decreasing, we iteratively proceed the following steps:
  1. Compute the elastic energy  $\mathcal{E}_e$  using Equation (3);
  2. Calculate formally the configuration  $(\mathbf{r}(s), \mathcal{F}(s))$  of the rod, each slice being a helix;
  3. Compute the potential energy  $\mathcal{E}_g$ , which requires the integration of the kinematic relations (1), in order to determine the function  $z(s)$ . Indeed,  $z(s) = \langle \mathbf{r}(s), \mathbf{z} \rangle$  under the condition  $z(0) = 0$ , where  $\mathbf{z}$  is the vertical normalized axis. Using the relation  $\mathbf{t} = \frac{\partial \mathbf{r}}{\partial s}$ , we get the following expression for  $\mathcal{E}_g$ :
 
$$\mathcal{E}_g = \rho S g \int_0^L (L - s') \langle \mathbf{t}(s'), \mathbf{z} \rangle ds' \quad (5)$$
 which can be accurately evaluated in practice.
  4. Perform a minimization step for  $\mathcal{E}_{hair} = \mathcal{E}_g + \mathcal{E}_e$ , using a fast gradient descent approach [FP63].
- Knowing the vector  $C_n$  that minimizes energy  $\mathcal{E}$ , we compute the final geometric configuration  $(\mathbf{r}, \mathcal{F})$  of the rod.

Some typical strand configurations obtained by our method are shown in Figures 1 and 2 (left). Our method is fast enough to handle several hair strands (sampled into 15 points) in real-time, and a hundred strands within a few seconds. This enables to use the method within an interactive hairstyling system.



**Figure 2:** Left: simulating a curly hair strand with different eccentricity values; note that increasing the eccentricity increases the regularity of the curls along the strand. Right: a curly hair strand growing on a sphere.

### 3. Full Hair Modeling

Similarly to the approach of Choe *et al.* [CK05], we model hair as a set of wisps where each wisp is composed of a *master strand* and of numerous other strands that are procedurally generated within a generalized cylinder surrounding

the master strand. The shape of the master strand is physically computed by energy minimization, as explained above, whereas the shape of the other strands is generated using a stochastic process similar to the one used in [CK05].

### 3.1. Collisions

For creating realistic hairstyles, it is necessary to account for both hair-body collisions and hair self-collisions. The latter are essential for giving an adequate volume to hair.

In our model, each wisp is modelled by a *skeleton* composed of the sample points of the master strand, and a thickness  $r_w^i$  that is computed according to several factors such as the number of hair strands within the wisp and the level of hair curliness. The body is approximated by a set of spheres for collision detection and response. We compute the collision response using elastic penalty forces  $F_c$ . The advantage is that such forces derive from a potential energy  $\mathcal{E}_c$ . As our static model is based on energy minimization, accounting for this kind of force simply amounts to minimizing the new energy  $\mathcal{E}_{hair}$  defined by:

$$\mathcal{E}_{hair} = \mathcal{E}_g + \mathcal{E}_e + \mathcal{E}_c$$

In practice,  $\mathcal{E}_c$  is computed using the amount  $x$  of penetration:  $\mathcal{E}_c = \frac{1}{2} k x^2$ , where  $k$  is a stiffness parameter. As illustrated in Figure 2 (right), this method properly accounts for the contacts between hair and a sphere.

Hair self-collisions are computed using the multiple layer hulls method developed by Lee and Ko [LK01]. Results show that this method is satisfying in the static case, and properly accounts for the hair volume.

### 3.2. Hairstyling Editing Tools

This section briefly presents the virtual hairstyling tools that are provided to the user for creating natural hairstyles. The great advantage of our approach in comparison with existing ones is that usual hairstyling operations such as curling, wetting, cutting or drying hair become straightforward.

Hair wetting has a direct impact on the hair shape, as its stiffness coefficient is roughly divided by a factor 10 [Rob02]. Moreover, as a hair strand absorbs water, its radius increases of around 13%. Ward *et al.* have proposed a hair model accounting for these properties [WGL04]. But, as this model is not characterized by adequate physical parameters, they need to control multiple structures to apply the physical changes to the hair strands. In contrast, within our accurate physically-based modelling system, the wetting of any hair strand is simply effected by modifying two relevant parameters: the Young's modulus and the mean radius. Additionally, when hair is wetted, the hair volume is procedurally reduced through the hull layers. We simply process hair drying as the inverse process of hair wetting.

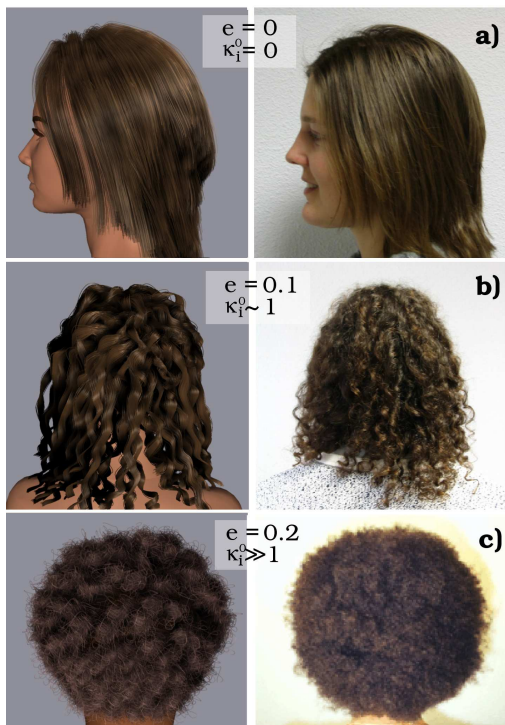
Haircutting consists of removing the hair part located under a given  $z$  position. This is simply done by changing each

hair length  $l_{prev}$  into  $l_{prev} - l_{cut}$  where  $l_{cut}$  is the length of the removed part. As a result, hair becomes lighter after cutting, and thus its overall shape is affected, as in the real world.

#### 4. Results and Concluding Remarks

Generated hairstyles are rendered using the accurate scattering model of Marschner et al. [MJC\*03]. In particular, this model accounts for the visual effect due to the elliptic cross section of a hair fiber. Thus, eccentricity  $e$  represents an unified parameter for both our mechanical model and rendering. This parameter varies according to the hair type: for Asian hair,  $e$  is nearly equal to 0, whereas for Caucasoid hair, it ranges from 0 to 0.1 and for African hair, it ranges from 0 to 0.2. Other hair physical parameters  $E$ ,  $\mu$  and  $\rho$  are taken equal to their actual value, given by physical measurements:  $E = 10 \text{ GPa}$ ,  $\mu = 0.3$ ,  $\rho = 1.3 \text{ g/cm}^3$  [Rob02].

Figure 1 and 3 show various hairstyles obtained with our method for different hair types and haircuts. Please also visit our website and watch our videos at <http://www-evasion.imag.fr/Publications/2005/BAQLLC05/>. Note that our method is able to capture realistic natural hair shapes.



**Figure 3:** Right: real pictures of (a) smooth, (b) curly and (c) African hair. Left: corresponding synthetic results, generated with adequate values for  $e$  and  $\kappa_i^0$ . Each synthetic hairstyle was created in less than half an hour.

The main limitation of our method is the computational time required to calculate the equilibrium configuration of hair (6 seconds in average for the hairstyles illustrated in

Figure 3). This prevents us from using more than a hundred master strands within our interactive hairstyling software. However, results are very satisfactory with this limited number of master strands, thanks to our procedural wisp model.

We are currently working on the extension of our physical model to the dynamic simulation of hair motion.

**Acknowledgments** We would like to thank Steve Marschner for letting us use his original rendering code.

#### References

- [AP05] AUDOLY B., POMEAU Y.: *Elasticity and Geometry: from hair curls to the nonlinear response of shells*. Oxford University Press, To Appear in 2005. 2
- [AUK92] ANJYO K., USAMI Y., KURIHARA T.: A simple method for extracting the natural beauty of hair. In *Proceedings of ACM SIGGRAPH'92* (August 1992), Computer Graphics Proceedings, Annual Conference Series, pp. 111–120. 1
- [CK05] CHOE B., KO H.-S.: A statistical wisp model and pseudophysical approaches for interactive hairstyle generation. *IEEE Transactions on Visualization and Computer Graphics* 11, 2 (March 2005). 1, 3
- [FP63] FLETCHER R., POWELL M. J. D.: A rapidly convergent descent method for minimization. *The Computer Journal* 6, 2 (1963), 163–168. 3
- [HMT00] HADAP S., MAGNENAT-THALMANN N.: Interactive hair styler based on fluid flow. In *Computer Animation and Simulation '00* (Aug. 2000), pp. 87–100. 1
- [KN02] KIM T.-Y., NEUMANN U.: Interactive multiresolution hair modeling and editing. *ACM Transactions on Graphics* 21, 3 (July 2002), 620–629. Proceedings of ACM SIGGRAPH'02. 1
- [LK01] LEE D.-W., KO H.-S.: Natural hairstyle modeling and animation. *Graphical Models* 63, 2 (March 2001), 67–85. 3
- [MJC\*03] MARSCHNER S., JENSEN H., CAMMARANO M., WORLEY S., HANRAHAN P.: Light scattering from human hair fibers. *ACM Transactions on Graphics (Proceedings of the SIGGRAPH conference)* 22, 3 (July 2003), 281–290. 4
- [Pai02] PAI D.: Strands: Interactive simulation of thin solids using cosserat models. *Computer Graphics Forum* 21, 3 (2002), 347–352. Proceedings of Eurographics'02. 1, 2
- [PBS04] PARIS S., BRICEÑO H., SILLION F.: Capture of hair geometry from multiple images. *ACM Transactions on Graphics (Proceedings of the SIGGRAPH conference)* (2004). 1
- [Rob02] ROBBINS C.: *Chemical and Physical Behavior of Human Hair*. 4th ed. Springer, 2002. 3, 4
- [WGL04] WARD K., GALOPPO N., LIN M. C.: Modeling hair influenced by water and styling products. In *International Conference on Computer Animation and Social Agents (CASA)* (May 2004). 3



## INTERACTIVE VIRTUAL HAIR-DRESSING ROOM

Nadia Magnenat-Thalmann, Melanie Montagnol, Rajeev Gupta, and Pascal Volino

MIRALab - University of Geneva  
[thalmann](mailto:thalmann@miralab.unige.ch), [montagnol](mailto:montagnol@miralab.unige.ch), [gupta](mailto:gupta@miralab.unige.ch), [volino](mailto:volino@miralab.unige.ch) @miralab.unige.ch

### ABSTRACT

Hair designing is one of crucial components of the hair simulation tasks. The efficiency of hair modeling is very much determined by the interactivity and the ease-to-use the designing tools within an application. This paper presents a unified framework that uses the various key techniques developed for specific tasks in hair simulation and to realize the ultimate goal of 'virtual hair-dressing room' that is simple to use but quite effective for generating fast hairstyles. Successful attempts have been made to handle the different challenging issues involved in simulation of hair at interactive rates. Effort has been put in developing methodologies for hair shape modeling, hair dynamics and hair rendering. A user friendly interface controlled by a haptic device facilitates designer's interactivity with the hairstyles. Furthermore, designer's visualization is enhanced by using real time animation and interactive rendering. Animation is done using a modified Free Form Deformation (FFD) technique that has been effectively adapted to various hairstyles. Hair Rendering is performed using an efficient scattering based technique, displaying hair with its various optical effects.

**Keywords:** Hair Simulation, Interactive hair Modeling, Force feedback Interaction, Volume deformation, Real Time Rendering

### 1. INTRODUCTION

Hair is essential for characterizing a virtual human and so is the need to design a suitable and realistic looking hairstyle for it quickly. Interactivity plays an important role in determining the efficiency of the design application. The process is further facilitated by an interface that provides user with simple but essential tools for performing interactions like for handling, cutting, and brushing hair displaying dynamic and optical behavior at interactive rates. Usually this a complex task and the techniques developed have to compromise between interactivity and realistic appearance.

The difficulties with human hair simulation arise due to the number, the structure and the interaction between hair. On a scalp, human hair are typically 100,000 to 120,000 in number. With this number, even simplified simulation takes vast amount of computing resources. Moreover, the complicated structure and interaction between hairs present challenges in every aspect of the simulation. Furthermore, addressing these issues to simulate hair accurately and realistically for real-time indeed makes virtual hair simulation technically a demanding process.

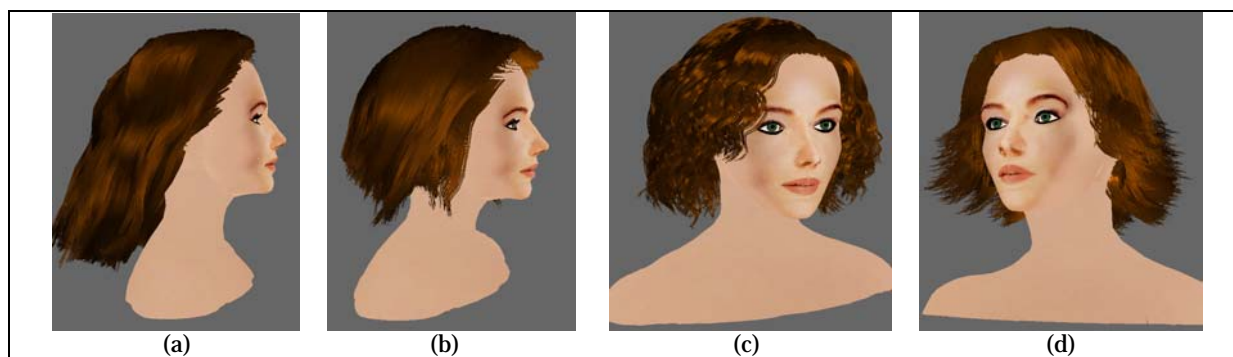


Figure 1: Various hairstyles created, animated and rendered using our interactive "virtual hair-dressing room" (a) is the initial Hairstyle, (b),(c), and (d) are the hairstyles created using different interactive tools (cutting, curling and brushing)



Geometrically hair strands are long and thin curved cylinders. The strands of hair can have any degree of waviness from straight to curly. Thus, a modeling method should provide the control over a large group of hair strands effectively, but still allow control over detailed variations. The structure (or style) of hairs as a group is often more meaningful than the properties of individual hair strands. The interactions between hair play an important role in generating these structures. Therefore, a modeling method should take into account the interactions as well as the individual properties. Furthermore, from a designer's aspect all these features should be available in an interactive system for fast prototyping of various hairstyles. The sense of immersion is also increased if he/she gets back force feedback while performing different interactions with the available tools. This requires integrating a haptic interface which provides user with increased degree of freedom as compared to a mouse and considerably increases the comfort, performance and productivity.

Real time simulation is another important feature that increases the sense of involvement during hair designing. It is obviously more realistic to see dynamic changes in the hairstyle as the user performs interactions. But, the complexity of hairstyles makes it a challenge for animating them on virtual characters with performances compatible with real-time. Each hair strand has a high degree of flexibility and easily gets bent upon external forces such as gravity and collisions, while it strongly resists linear stretches. The difficulty is actually to build a mechanical model of the hairstyle which is sufficiently fast for real-time performance while preserving the particular behavior of the hair medium and maintaining sufficient versatility for simulating any kind of complex hairstyles. The hair motion is affected by many subtle factors, such as wind forces, air drag, gravity, head motion and wetness. More interestingly, the static charge and friction between neighboring hairs often cause a group of hairs to form a cluster that moves coherently. Collisions between such clusters are important otherwise the hair interpenetration will be quite unnatural and distracting, and overall affect the realism of the simulation. This behavior is highly dependent on the type of hairstyle (straight or curly, long or short) and condition (stiff or soft, dense or sparse, dry or wet).

The hairstyle also plays an important role in how the light interacts with the hair which in turn is essential to be simulated because specifically self-shadow within hair gives important clue about the hair style itself. Rendering human hair often requires painful labors as well as demanding computational power. Optically, hair has many interesting phenomena such as anisotropic reflection, strong forward scattering, and translucency. A correct reflectance model is thus needed to compute reflectance off the individual hair geometry. While, a local shading model approximates coloring of an individual hair strand, interactions between hairs require efficient methods to compute such global illumination effects. In particular, the shadow between hairs provide essential visual cues for the structure of hair as a group. Hair is inherently volumetric and the internal structures can be correctly illustrated only with proper shadow. Furthermore, most of the current hair rendering simulations lack physical basis. All the opacity-based approaches consider a transmittance function that compute attenuation of light only towards the hair strand for which shadow is being computed. This is not physically correct as there is also an attenuation of light when it is scattered from the hair strands towards the viewer, and must be incorporated for the final hair color. Moreover, variations in shading due to animated hair also need to be considered for realistic results.

Our work is motivated by the need to have a unified model for facilitating hairstyling effectively utilizing various methodologies, like interactive modeling, simplified but efficient animation, and optimized yet realistic rendering giving a real-time performance. The specific contributions of this paper are:

- An interactive modeling system to allow the user with an easy and flexible modeling capability to design any hairstyle with added features of fine-tuning. The system should perform all the hair manipulation tasks (e.g. cutting, curling) at interactive rates.
- A mechanical model that efficiently simulates the motion of individual hair strand as well as the interactions between hairs themselves and the interaction between hairs and other objects.
- A scattering based illumination model that produces realistic rendered hair with various colors and showing optical effects like shadow, specular highlights.
- Importantly, all these simulations are performed in real-time.

The rest of this paper is organized as follows: in the next section we give a brief overview of various hair simulation techniques. In Section 3 we present various considerations and the ideas of our approach for styling, animating and rendering. In Section 4 we present the different interaction tools developed for designing hairstyles. Section 5 is dedicated to our lattice based mechanical model highlighting its key features and benefits. Section 6 gives the description of our scattering based illumination model discussing the various optimizations

made for rendering at interactive rates. We demonstrate the results and discuss the performance for the unified hair styling application in Section 7. We conclude with a look at avenues for future work in Section 8.

## **2. PREVIOUS WORK**

Researchers have devoted a lot of time and effort for producing visually convincing hair, giving more believability to virtual humans. Though implementing a unified framework for creating hairstyles using dynamic models along with realistic rendering has not been actively considered, a number of algorithms have been developed to resolve most of the challenges in the three main hair simulation tasks - hair shape modeling, hair animation and hair rendering. Specific advancements have been made in each of these tasks.

### **2.1 Interactive Hair Styling**

There exist different techniques proposed in the literature in order to provide tools for interactively creating new hairstyle. Most of these techniques involve designing an interactive 3D user interface giving user control to place hair on any part of the 3D model. Use of cluster or wisp for representing hairstyles is a common approach in these systems. Chen et al. [5] presents a model that represents each cluster with a trigonal prism and the hair strands within a cluster is modeled by the distribution map defined on the cross-sections of the prism. Similar to this approach, in [21] a wisp model is used but represents a set of a 2D patch projected on a cylinder as boundary. The developed tool allows user to choose number of patches, position them and then individually or collectively modify parameters such as length, stiffness, blending for the patches. Recently a more user-friendly interactive tool called Hair Paint [10] has been introduced. The system is based on a 2D paint program interface and color scale images to specify hair characteristics. Some of the researchers have also presented systems exploiting fluid flow [8] and vector fields [28] for explicit hair modeling. Other techniques [14][25] use hybrid models combining benefits of wisp model and strand model. In general, all these systems result in creating nice static hairstyles, but are too slow to be able to interact with hairstyles when dynamic behavior is added. Also utilizing force feedback devices for more interactive control of designing tools is still to be explored.

### **2.2 Hair Animation**

Usual simulation techniques may consider any compromises between simulating individually each strand and simulating a volume medium representing the complete hairstyle. In the context of efficient simulation, reducing this complexity and the number of degrees of freedom of the model is essential for fast computation. The first idea is to consider that hairs of neighboring locations share similar shapes and mechanical behaviors. In many models, such hairs are grouped into wisps, sometimes also called clusters. This technique was defined by Watanabe et al [26], and has been frequently used since with many variations in [4][5][20][27]. Another evolution is to replace the hairs by approximate surfaces (strips or patches), as done by Koh et al [15], or even volumes that can easily be modeled as polygonal meshes, such as the thin shell approach of Kim et al [12]. Combining various approaches can lead to Level-of-Detail methods where the representation is chosen depending on the current requirement of accuracy and available computational resources. A good example developed by Bertails et al [2] is based on wisps tree, and a similar approach from Kim et al [14] is based on multiresolution clusters. Advanced Level-of-Detail methods also include combination of strands, clusters and strips modeled with subdivision schemes and animated using advanced collision techniques, as developed by Ward et al [24]. In the specific context of fast real-time applications, cluster and strip models are good candidates. However, they still suffer from various problems, such as the necessity of collision detection between hairs, and their inability to represent efficiently some mechanical properties, such as bending stiffness. Furthermore, these simulation techniques have not yet been considered for efficient use during styling.

### **2.3 Hair Rendering**

Hair rendering involves immense problems ranging from large data to complex optical behavior and thus has been a challenging yet captivating research topic since quite some time. The research started with the limited problem of fur rendering by Kajiya et al. [11], who also derived a local anisotropic lighting model for hair that has been widely adopted and extended in later illumination models [1][6][17]. Recently the giant leap in performance of graphics card has made it a possibility to have faster and realistic results. In [16] intensive utilization of GPU is done for shadow map computation, though it doesn't take animation of the hair into consideration. In [19] the issue of shadow in dynamic hairs is efficiently dealt by computing a 3D density field and storing the samples of the hair density function in a compact way. The use of density clustering gives an advantage of not explicitly computing the entire transmittance function for time-varying hair data, resulting in better rendering speed and

quality. More recently [3] presented an animated hair rendering model based on a 3D light oriented density map. Unlike the previous interactive approaches, this algorithm computes accurate self-shadow on a standard CPU independent of any GPU at interactive rates. The paper combines a volume representation of density with a light oriented sampling for self-shadows.

### 3. CONSIDERATIONS AND APPROACH OVERVIEW

Hair styling scheme can be divided in two parts: firstly, defining a set of general characteristics of a hair style, and secondly, varying set of characteristics of an individual hair. We choose to model hair utilizing the data representation of animation model [23]. This hair modeling technique allows modifying the geometry of the hair directly. We develop these parameters in two phases. First, we propose a set of static parameters for hair styling (geometric modifications), and then we animate using these parameters (update simulation).

In order to increase the interactivity we have chosen to use a haptic interface. With this device it is now possible to see the virtual hair and other objects, to move and finally to touch them in the virtual environment. The using of haptic enables interaction with hair in real time. We present a system that provides the user with interactive and

easy-to-use tools for hairstyling. Even though we have utilized a mechanical model in our simulation, the system is fast and the user can make modifications at interactive rates. In addition, the dynamic behavior results in more accurate and realistic simulation as it include influence on the styles due to gravity and other external forces.

For animation, the system should be compatible with most approaches used for hairstyle representation and real-time rendering, and offer the designer the possibility of creating any kind of hairstyle that can robustly be simulated in any animation context. We intend to achieve this through the design of hair animation model based on volume deformations, with the aim of animating any complex hairstyle design in real-time. The idea of our method is the following: rather than building a complex mechanical model directly related to the structure of the hair strands, we take advantage of a volume free-form deformation scheme. We detail the construction of an efficient lattice mechanical deformation model which represents the volume behavior of the hair strands. The lattice is deformed as a particle system using state-of-the-art numerical methods, and animates the hairs using quadratic B-Spline interpolation. Another issue that needs to be considered during hair animation is collision. Since hair strands are deformed through FFD, there is no real need of detecting collisions between the hair strands, as they follow the volume of deformation of FFD which is rarely self-intersecting. However, we need to take into account collisions between the hairstyle and the skull. The hairstyle reacts to the body skin through collisions with a metaball-based approximation. The metaballs are represented as polynomial forces that are integrated in the mechanical model. The model is highly scalable and allows hairstyles of any complexity to be simulated in any rendering context with the appropriate tradeoff between accuracy and computation speed, fitting the need of Level-of-Detail optimization schemes.

We also aim to have an optimized technique that decreases the complexities involved in hair rendering and simulates optical effects while incorporating hair animations at interactive rates. Our rendering data representation for hair volume is quite similar to the animation system [23]. Utilizing a similar representation also gives us an advantage of having a fast access to lot of information from the underlying animation model for rendering computations. Based on these considerations we model our illumination model as follows: The local illumination of hair is defined by a Gaussian function adapted to our strip representation of hair. The self-shadow computation for hair is similar to the volume rendering as we perform voxelization of hairstyle into a hair volume for opacity computation. For the case of animated hair, the new vertex positions are calculated, and based on the displacement of the hair vertex a refinement map is computed. This takes care of the hair density variations within

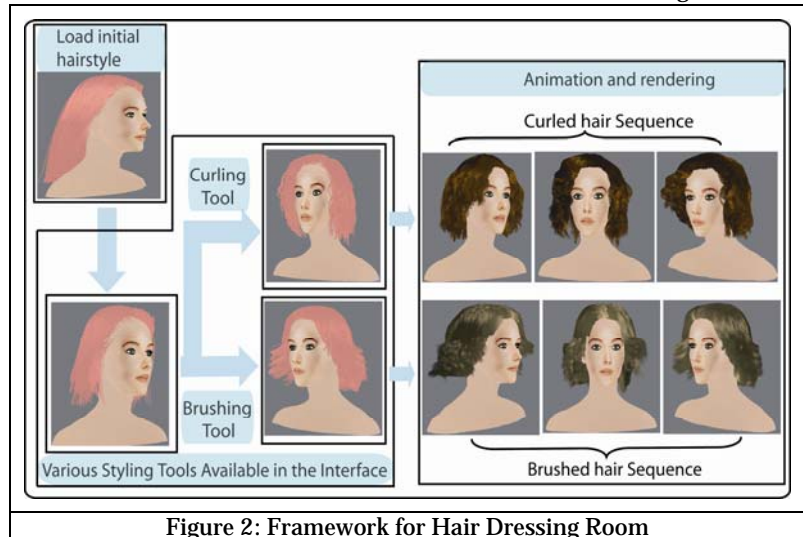


Figure 2: Framework for Hair Dressing Room

the rendering lattice and is then used for *'refining'* the shadow in animated hair. Our method's efficiency towards achieving fast updates is highly credited to the division of the pre-processing and the run-time tasks of the simulation. The model has been optimized to perform expensive computations, mainly involving physics, during pre-processing that contributes to the speedy performance of our illumination model without any significant degradation in quality.

#### 4. USER INTERACTIONS

We use PHANToM for the virtual manipulation of hair. Essentially it is force feedback device and we have efficiently used it like a simple tool allowing interactions with virtual 3D objects. This haptic device replaces and increases the functional of the mouse. The force-feedback is used to detect collision of hair with the head, as well as to interact with the cells in the lattice.

The system gives the user freedom to choose tools to modify hair collectively (using guide hairs) or individually. The user can choose hair via a 2D scalp map or make a more precise selection directly in 3D. Using the buttons provided with the PHANToM, the user can choose the tool to apply on the selection. While building this new hairstyle the user with his left can modify the camera position for better visualization, and with his right hand select and apply a set of tools developed to modify the hairstyle. Integrating these tools with our mechanical model produces realistic simulation that enhances the overall visualization during interaction. For efficiently realizing this effect, we create a lattice for the selected hair for animating them during interaction with tools using geometric and physics-based method.

##### 4.1 Selecting Hair

The first step in designing a hairstyle is to select a set of hair. There are different selection modes that are offered to the users. The interactivity is performed via the haptic device. We propose a point-based method for selection that involves implementation of 3 DOF force feedback. The selection is done by the position of the stylus of the haptic. If the stylus intersect a cell in the lattice, all the hair in that box are selected. This selection mode is linked to the discretization of the lattice, and does not allow to select a set hair between two cells. But generally, the hairdresser can select a set of hair as well as one by one via projection in the direction of the camera of the position of the stylus. After this selection step the user can "add" a set of hair using copy and paste tool. The user can change the position and the angles of rotation (via haptic) of the duplicate set of hair.

For hairstyling operators animation we have limited the animation to specific parts where modifications have to be made. Thus, we create a lattice just for the selected set of hair and only this set is later animated as shown in green in figure 3. Limiting the lattice to a specific section of hair decreases the computational time and allows efficiently using the geometrics methods. These geometrics modifications don't change the form of the lattice. This technique also avoids a computation of self-collision detection between strips and provides good result in simulation.

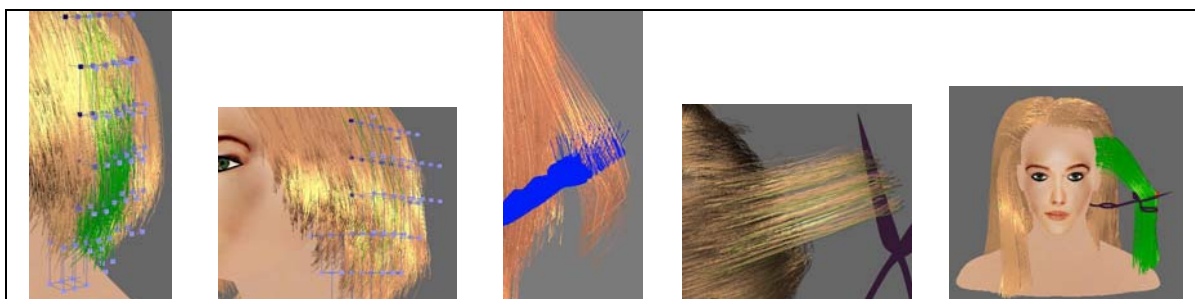


Figure 3: Selecting and Holding a group of Hair

##### 4.2 Virtual Scissor:

The most important tool for a hairdresser is probably the scissor. In our technique we first choose a cut plane for modifying the hair. Three planes are proposed; a horizontal plane, an inclined plane, and a curve plane like the function sine. The choice of these planes is linked to the fact that with these planes, all type of cut can be realized. A mass-spring system representation is used for all strands in order to simulate dropping the hair cut during the cutting process. The implementation of this modeling is easy, and allows obtaining good results in short time. In order to increase the interactivity, it is possible to modify the position and the orientation of this cut plane by a

line-based method (using 6DOF) via the haptic. The user can thus choose the cutting mode either by using a scissor (generally for a small set of hair) or by a cutting plane.

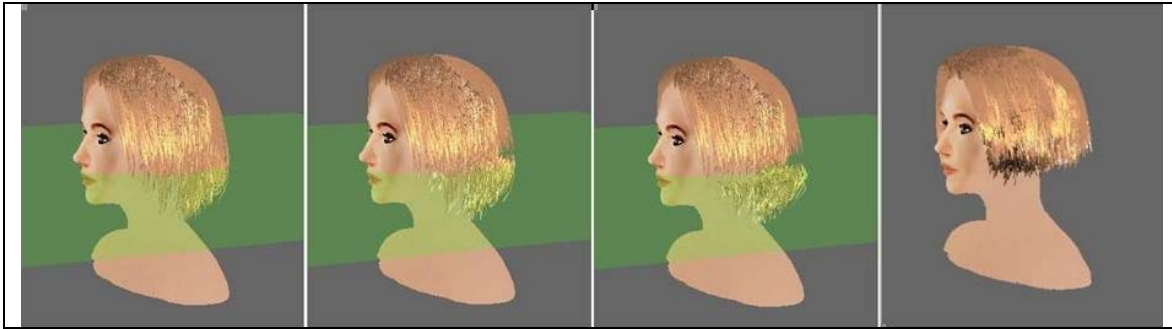


Figure 4: A sequence of Hair Cut in real time

### 4.3 Virtual Brushing and Curling

The virtual brushing and waviness tool involves a function of one parameter and allows modifying the “curve” of strands without using the position of points in the strand in 3D. We modify the hair beginning from the tip in the curving. Three parameters can be modified: the radius, the length of curve, and the direction of the brushing. Based on these parameters, a mathematical function is applied on the last node of all the selected hair resulting in curve hair. The choice of the mathematical function is based on the action to be performed or the tool required. When simulating waviness, the visual effect is similar to a compressed spring movement from top to bottom. When brushing, the movement of lattice node is away from the head and following the direction of the brush. The use of these techniques avoids us to compute collision between hair and an object (tools, comb for example). The visual realism is maintained because the simulation seems to be real.

## 5. LATTICE BASED HAIR ANIMATION

Realistic animation of hair requires advanced models that deal with the complex strand structure of human hairstyles. The challenge is to build a mechanical model of the hairstyle which is sufficiently fast for real-time performance while preserving the particular behavior of the hair medium and maintaining sufficient versatility for simulating any kind of complex hairstyles. We overcome the difficulties through the design of a hair animation [23] model based on volume deformations, with the aim of animating any complex hairstyle design in real-time.

### 5.1 The Lattice Model

We construct a 3D lattice around the head that includes all the hair. This lattice is defined by the initial positions of all its nodes. During animation, the lattice is moved according to the head motion. The resulting rigid-motion positions and velocities of the lattice nodes define the rigid motion pattern of the hair, in which the hair follows the head motion without any deformation.

In our approach however, the lattice is furthermore deformed by mechanical computation. The actual position and speed of the lattice nodes are ruled by mechanical computation iterations that use the rigid motion states as equilibrium states. This mechanical model, aimed at providing the lattice with the volumic behavior of the hair contained in it, is constructed during preprocessing.

### 5.2 The Mechanical Model

Different approaches are possible for designing this mechanical lattice deformation model. However, spring-mass approaches still seem to be the best candidate for designing really fast mechanical models. We have created a particular kind of interaction force that uses as attachment points any arbitrary point of the lattice, defined by its lattice coordinates. Hence, each attachment point corresponds to a weighted sum of a given number of lattice nodes. Hence, the mechanical model of the hair is defined as a sum of linear viscoelastic lattice springs relating the elasticity of each individual hair strand segment in the lattice model. Their viscoelastic parameters correspond to those of the modeled hair. The attachments of the hair extremities to the skull are modeled by stiff viscoelastic lattice attachments, which are positioned exactly at the end of each hair. In order to reduce the complexity of the model, we resample each hair during the model construction as segments defined by the intersection points of the



hair line on the lattice boundaries. We furthermore carry out a simplification of the model by decimating the redundant lattice stiffeners (hair springs as well as skull attachments) until the expected number of stiffeners remain (Figure 5).

### 5.3 Lattice Interpolation

As the lattice is deformed during animation, another issue is to recompute the current position of each hair features for each frame. Depending on the selected rendering techniques and optimizations, these features may either be individual hair segments, or larger primitives such as hair strips or groups. For the best compromise between continuity and computation speed, we have selected quadratic B-Spline curves as interpolating shape functions, which offer second-order interpolation continuity.

We can take advantage of the interpolation to enhance the attachment of the hair on the skull through rigid motion. For each interpolated feature, a deformation coefficient is defined which is a blending coefficient between the motion defined by the rigid head motion and the motion defined by the interpolated lattice position. Hence, progressively animating the root of the hair through rigid head motion rather than mechanics allows simulating bending stiffness of hair near its root, and also removes the artifacts resulting from the imperfections of the mechanical attachment of the hair roots to the moving skull.

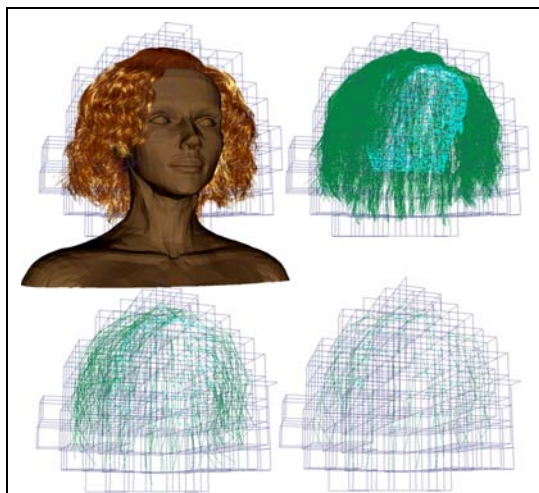


Figure 5: The decimation process: From the strands of the hairstyle (up-left) is constructed the initial model (up-right). The model can then be decimated at will (down-left) (down-right), for better performance

## 6. SCATTERING BASED HAIR RENDERING

Simulating fast and realistic complex light effect within hair is one of the most challenging problems in the field of virtual humans, both in terms of research and of development. Two main optical effects that contribute the most to realistic hair are the anisotropic reflection and self-shadows. We have developed a scattering-based fast and efficient algorithm that handles both the local specular highlights and global self-shadow in animated hair at interactive rates. We use a fast refinement technique [7] for incorporating illumination variations in animated hair from the encoded hair density changes considering the spatial coherency in hair data. Our method's efficiency towards achieving fast updates is highly credited to the division of the pre-processing and the run-time tasks of the simulation. At pre-processing we perform initializations that give us precise rendering information for static hair, and then perform an optimized update of optical variations in hair based on the dynamics of the guide hair, taking into account coherency between the hair data. The various optimizations implemented result in interactive simulation while maintaining the aesthetic visual appearance of the hair.

### 6.1 Scattering-based local Illumination

One of the features of our local illumination model is that it takes into the fresnel reflection and orientation of the cuticles on the hair which gives a control on defining the anisotropy of the hair. In addition, it also involves a gaussian function, which is not only important to provide physical basis to our model but modifications within the function also make it quite effective for our hair strips. We consider the gaussian function taking into account slope variations both along the tangent and the bi-normal of the strip vertices. The model takes care of both the diffuse and the specular reflectance. Various parameters control the width and the shift of the specular highlights as well as the sharpness and the anisotropy displayed by the strips.

### 6.2 Scattering-based self-shadow in hair

Our scattering-based approach considers two functions for shadow contribution. One function gives a measure of the absorption of light reaching a hair vertex and the second function considers the scattering from the hair. Since our shadowing model considers hair densities within the cells rather than explicit hair geometry, both the functions result in an overall reduced intensity of the hair vertex as visible to the viewer, which gives the

shadowing effect within the hair. The two terms (absorption and scattering) included in our shadow computation can be visualized in Figure 6.

Analytically, the absorption term generates self shadows due to its geometric and translucent nature while the scattering term is an additional component contributing to hair shading due to its material property. It is the collective effect of the two components computed for hair that gives light's intensity as perceived by the viewer and is correlated to the hair's shadow color, giving it a natural look.

### 6.3 Shadow Refinement in Animated Hair

During animation visual artifacts like irregular shadow regions can pop up as the hair vertex move from one cell to another if density variation within the cell is not properly dealt with. Our approach for incorporating variations in shadow in animated hair is based on the assumption that the hair vertices within a cell at initialization always stay within one cell after displacement. This assumption is valid not only from the physical aspect of the FFD-based animation model but also from real hair animation considerations where there is coherence among neighboring strands when hair is moving. Our idea is to encode the displacement of the particle during animation to give a measure of variation in hair density in the cells of the rendering lattice.

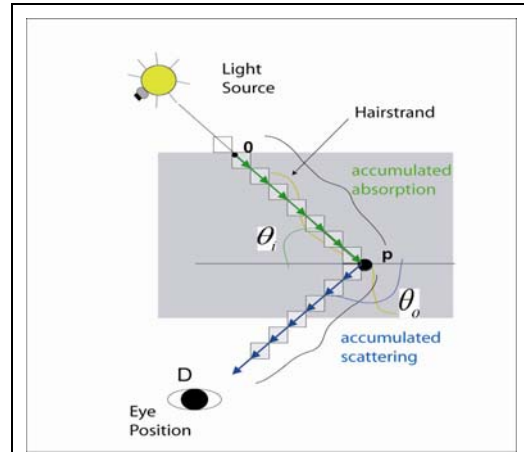


Figure 6: Variations in local specular highlights and the global self-shadows can be seen as the hair are animated and the light position is changed.

## 7. RESULTS AND DISCUSSION

We have tested our “virtual hair-dressing tool” for creating various hairstyles as shown in Figure 1 and Figure 8. The implementations are done on a workstation with Intel Pentium4 and 3.4 GHz processor with a 1 GB main memory and a GeForce 6800 graphics card. Our computation algorithm is written in standard C++ and rendering is done using the OpenGL API. The initial hairstyle has been created using our in home hairstyler based on fluid dynamics [8].

Once the initial hairstyle is loaded as a hybrid model, computation is performed to prepare the initial mechanical and rendering model. We ran our algorithm to create short, curly and brushed hairstyles using our unified framework. We have tested the hairstyles with various complexities of the mechanical representation and have found that a mechanical model containing 100 attachments and 400 springs built on 343 lattice nodes is fairly accurate and with fast simulation. The model reacts to collisions with the head and the shoulder using 7 metaballs. The computations for local illumination are done at run-time for each vertex using the graphics hardware. For shadow computation we found that for all the hairstyles a lattice size of  $128 \times 128 \times 32$  was optimal enough to provide nice result at an interactive speed. In the whole simulation process rendering is the most time-consuming task. The initial hairstyle in Figure 1 consists of around 40K segments. The rendering for static hair under a single dynamic light on GPU runs at around 14fps. Simulation of rendered animated hair runs at 9-10 fps.

While performing the user interactions, as the simulation is limited to only section of the hair to be modified, the average performance is better. As the interactions on the hairstyle are made the lattice for both animation and rendering are dynamically adapted to the modified data. During cutting, the cells containing the cut hair are deleted from the simulation thus further accelerating the process, although when brushing and curling the recomputation is done for calculating shadows based on the modified location of the hair segment which slows the process. For creating hairstyles in Figure 1, the average frame rate was around 10fps.

## 8. CONCLUSION AND FUTURE WORK

Our “interactive hair-dressing room” highlights the various developments made for providing an easily usable framework for hair styling. The use of a haptic interface is very advantageous during the designing process. It provides the user with more interactivity. This interface uses a 3D mouse for the aligning the head with the hairstyle and the haptic device for managing the set of tools for modifying the hairstyle. The overall performance

is also increased by the fact that the users can effectively utilize both their hands in order to create new hairstyles by applying special virtual tool like cut plane. Implementing techniques with accurate force feed back during all the hair styling operations (combing, curling, cutting, and effects by cosmetic styling products) will provide our hair dressing tool more realism. Though these devices are usually costly and also take time to adapting, but once a user gets used to them, it increases the comfortability and saves a lot of time.



Figure 7: With the ease-to-use the application, designer's can efficiently create hairstyles from images very quickly

Also, taking advantage of underlying FFD based mechanical model for dynamic simulation and the optimized scattering based rendering model for optical effect simulation, the interface provides an enhanced visualization of the hairstyling procedure. The FFD animation approach allows a clear distinction between the mechanical model that animates the lattice and the actual objects that are deformed by the lattice during the rendering. This makes the model highly scalable and versatile giving total freedom for hairstyle design, as hairs can be long or short, curled or straight, and even contain specific features such as ponytails, as well as facilitate designing level-of-detail schemes that can act independently on the mechanical simulation aspect and on the rendering aspect, without any particular adaptation of the model. Similarly, the rendering model is based on volume rendering and is independent of the geometry used. This makes it suitable for simulating self-shadows in any hairstyle. The various optimizations and the extensive use of graphics hardware, speeds up the rendering process allowing interactive hair manipulation. For handling variations in rendering due to animation we introduce a refinement technique, which is encoded based on the hair displacement vector. These models have been unified to successfully simulate some common user interactions with hair, like pick some hair, cut them and brush them and create realistic hairstyles.

With initial results of our unified framework being quite promising we look forward to adding more features for better interactivity and enhanced visualization. The use haptic rendering algorithm increases the virtual interactivity a lot. Currently, we implement just rigid interactions within the different cells of the lattice. But we are working on the interacting with deformable objects. The problem with these interactions is the computational time. We have yet decreased this time in modeling hair like a lattice but the feed-back forces aren't linked to the hair "touch" but the boxes. If the cells become deformable (likes our hair) the feed-back forces will become more linked to the reality. We are working on making our mechanical model more efficient and faster by making related computation on hardware especially for interpolation. We also plan to modify and simulate effects in hairstyles, both dynamically and optically, under influence of water and styling products.

#### **Acknowledgement**

This project is funded by the Swiss National Research Foundation.

## 9. REFERENCES

- [1] ANJYO K., USAMI Y., KURIHARA T.: A simple method for extracting the natural beauty of hair. In Proc. SIGGRAPH '92 (July 1992), vol. 26, pp. 111–120.
- [2] BERTAILS F., KIM T.-Y., CANI M.-P., NEUMANN U.: Adaptive wisp tree-a multiresolution control structure for simulating dynamic clustering in hair motion. In Symposium on Computer Animation'03 (Jul 2003), pp. 207-377.
- [3] BERTAILS F., MENIER C., CANI M. P.: A practical self-shadowing algorithm for interactive hair animation. In Graphics Interface (May 2005), pp. 71-78.
- [4] CHANG J., JIN J., YU Y.: A practical model for mutual hair interactions. In Proceedings of Symposium on Computer Animation 2002 (Jul 2002), pp. 51-58.
- [5] CHEN L., SAEYOR S., DOHI H., ISHIZUKA M.: A system of 3d hairstyle synthesis based on the wisp model. In The Visual Computer (1999), pp. 159-170.
- [6] DALDEGAN A., MAGNENAT-THALMANN N., KURIHARA T., THALMANN D.: An integrated system for modeling, animating and rendering hair. In Proc. of Eurographics '93 (July 1993), pp. 211–221.
- [7] GUPTA R., MAGNENAT-THALMANN N.: Scattering-Based Interactive Hair Rendering, In IEEE 9<sup>th</sup> International Conference on Computer Aided Design and Computer Graphics'05, (Dec 2005), pp. 489-494.
- [8] HADAP S., MAGNENAT-THALMANN N.: Interactive hair styler based on fluid flow. In Eurographics Workshop on Computer Animation and Simulation '00 (Aug. 2000), pp. 87–99.
- [9] HADAP S., MAGNENAT-THALMANN N.: Modeling dynamic hair as a continuum. In Proc. of EuroGraphics '01 (Sept. 2001), Vol. 20, Issue 3, pp. 329-338.
- [10] HERANANDEZ B., RUDOMIN I., Hair Paint, In IEEE Proc. of Computer Graphics International 2004 (CGI), Creta, Grecia, Junio 16 – 19, 2004, pp. 578-581.
- [11] KAJIYA J. T., KAY T. L.: Rendering fur with three dimensional textures. In Proc. SIGGRAPH '89 (July 1989), vol. 23, pp. 271–280.
- [12] KIM T., NEUMANN U.: A thin shell volume for modelling human hair. In IEEE Proc. of Computer Animation '00 (2000), pp. 104–111.
- [13] KIM T., NEUMANN U.: Opacity shadow maps. In Proc. of the Eurographics Rendering Workshop '01 (2001), pp. 177–182.
- [14] KIM T.-Y., NEUMANN U.: Interactive multiresolution hair modeling and editing. In Proc. of SIGGRAPH 2002 , pp. 287–294.
- [15] KOH C., HUANG Z.: A simple physics model to animate human hair modeled in 2d strips in real time. In Proc. of the Eurographics workshop on Computer animation and simulation (2001), pp. 127–138.
- [16] KOSTER M., HABER J., SEIDEL H. P.: Real-time rendering of human hair using programmable graphics hardware. In Proc. of Computer Graphics International (CGI'04) (2004), pp. 248–256.
- [17] LEBLANC A., TURNER R., THALMANN D.: Rendering hair using pixel blending and shadow buffers. In Journal of Visualization and Computer Animation '91 (1991), vol. 2, pp. 92–97.
- [18] MARSCHNER S. R., JENSEN H. W., CAMMARANO M., WORLEY S., HANRAHAN P.: Light scattering from human hair fibers. In Proc. SIGGRAPH '03 (July 2003), vol. 22, pp. 780–791.
- [19] MERTENS T., KAUTZ J., BEKAERT P., REETH F. V.: A self-shadow algorithm for dynamic hair using clustered densities. In Proc. of the Eurographics Symposium on Rendering '04 (June 2004), pp. 173–178.
- [20] PLANTE E., CANI M.-P., POULIN P.: A layered wisp model for simulating interactions inside long hair. In Proc. of EuroGraphics (sep 2001)pp. 139-148.
- [21] SCHMITH C., KOSTER M., HABER J., SEIDEL H-P., 2004, Modeling Hair using a Wisp Hair Model, Research Report MPI-I-2004-4-001, <http://domino.mpi-sb.mpg.de/internet/reports.nsf/0/d40109e7b2b4bd12c1256e45004ac335>
- [22] MAGNENAT-THALMANN N., HADAP S., KALRA P.: State of the art in hair simulation. In International Workshop on Human Modeling and Animation '00 (June 2000).
- [23] VOLINO P., MAGNENAT-THALMANN N.: Animating complex hairstyles in realtime. In Proc. of ACM Symposium on Virtual Reality Software and Technology (VRST'04) (2004), pp. 41-48.
- [24] WARD K., LIN M. C.: Adaptive grouping and subdivision for simulating hair dynamics. In Pacific Graphics Conference on Computer Graphics and Applications 2003 (2003), pp. 234–243.
- [25] WARD K., LIN M. C., LEE J., FISHER S., MACRI D.: Modeling hair using level-of-detail representations. In Proceedings of Computer Animation and Social Agents (CASA) 2003 (2003), pp. 41–47.
- [26] WATANABE Y., SUENAGA Y.: Drawing human hair using the wisp model. In Special issue on computer graphics international'89 (CGI'89) (May 1989), vol. 7, pp. 97 – 103.
- [27] YANG X.D., XU Z., YANG J., WANG T., 2000, The Cluster Hair Model, Graphical Models, Elsevier, 62(2), p.85-103.
- [28] YU Y.: Modeling realistic virtual hairstyles. In Proc. of Pacific Graphics 2001 (2001), pp. 295–304.

# **INTERACTIVE HAIR MODELING: *TECHNIQUES & CHALLENGES***

**Supplementary Course Notes**

**PART II – Presentation Slides**

**Ming C. Lin**

**University of North Carolina at Chapel Hill**

**Nadia Magnenat-Thalmann**

**University of Geneva**





1

# Animating Complex Hairstyles in Real-Time

Nadia Magnenat-Thalmann and Pascal Volino  
MIRALab, University of Geneva




MIRALab University of Geneva  
Where Research means Creativity www.miralab.ch Animating Complex Hairstyles in Real-Time

2

# Real-Time Animation of Hairstyles

- Managing Complexity of Hair Simulation
  - More than 100 000 hair strands
  - Mechanical behavior of individual strands
  - Collisions and friction on the skull
  - Collisions and friction between strands
  - Physical state (wetness, styling products)
  - Aerodynamics
- No chances to simulate an explicit mechanical model in real-time.



MIRALab University of Geneva  
Where Research means Creativity www.miralab.ch Animating Complex Hairstyles in Real-Time

3

# Real-Time Animation of Hairstyles

- Ideas for Speeding Up Hair Simulation
  - Simplification of the mechanical model
  - Reduction of the degrees of freedom
  - Macroscopic approximations
  - Interpolation
- Techniques
  - Wisp or Cluster models
  - Particle systems of variable topology
  - Volume hair models
  - Free-Form Deformations

MIRALab University of Geneva  
Where Research means Creativity www.miralab.ch Animating Complex Hairstyles in Real-Time

4

# Real-Time Animation of Hairstyles

- Wisp or Cluster Models
  - Defined by Watanabe et al [1989]
  - Interpolation on Guide Hairs by Chang et al [2001]
  - Multilayer colliding clusters by Plante et al [2001]
  - Hair strips by Koh et al [2000, 2001]
  - Thin shell approach by Lim et al [2000]
  - Advanced collision handling by Lee et al [2001]
- Level-of-Detail
  - Wisp-tree by Bertails et al [2003]
  - Multiresolution Clusters by Kim et al [2003]
  - Strand-Strip-Cluster adaptive by Ward et al [2003]

MIRALab University of Geneva  
Where Research means Creativity www.miralab.ch Animating Complex Hairstyles in Real-Time

## Real-Time Animation of Hairstyles

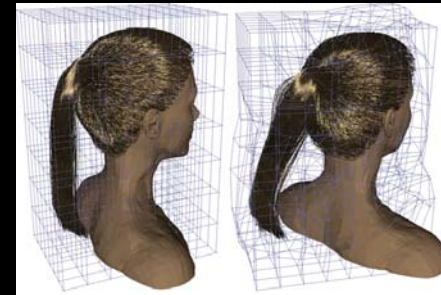
5

- Other Fast Simulation Models
  - Loosely Connected Particles by Bando et al [2003]
  - Short Hair Model from Guang et al [2002]
- Volumic Models
  - Vector Fields from Yu [2001]
  - Fluid Vector Fields from Hadap et al [2000, 2001]
  - Unsited for real-time simulation.
- Free-Form Deformation Models

## 1. Real-Time FFD Hair Animation

6

- The main idea: Deforming the complete hairstyle using a mechanically-animated Free-Form Deformation lattice.



## 1. Real-Time FFD Hair Animation

7

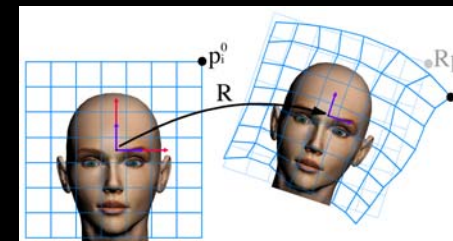
- Main advantages:
  - Drastic and controllable reduction of the number of degrees of freedom of the mechanical model
    - => Real-time mechanical simulation
  - Simple and fast interpolation scheme for computing the deformed hairstyle
    - => Real-time motion of any feature of the hairstyle
  - Any hairstyle can be animated
    - => Versatility and design simplicity



## 2. FFD Hairstyle Deformation

8

- Building the lattice around the hairstyle
  - The lattice is attached to the rigid-body motion  $R$  of the skull
  - The lattice is then deformed by mechanical simulation

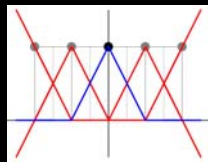


## 2. FFD Hairstyle Deformation

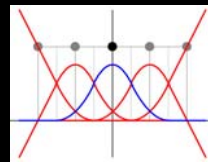
- Interpolating Hairstyle Features in the Lattice
  - Feature locations defined by weighted sums of lattice nodes

$$\mathbf{p} = \sum_i \mathbf{w}_i \mathbf{p}_i$$

- Different interpolation schemes for precomputing weights



Linear: Speed (3D: 8 nearest nodes)



Quadratic: Smoothness (3D: 27 nearest nodes)

## 2. FFD Hairstyle Deformation

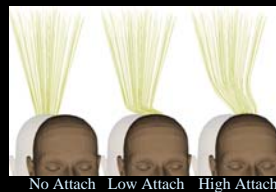
- Attaching Strand Roots on the Skull
  - Attachment of the hair strands on the skull should be ensured whatever the deformation
  - Strand roots should be defined by the rigid motion of the skull only
  - Continuous transition between rigid motion ( $\delta = 0$ ) and deformed lattice ( $\delta = 1$ ) along the hair

$$\mathbf{p} = \sum_i \mathbf{w}_i (\delta \mathbf{p}_i + (1 - \delta) \mathbf{R}\mathbf{p}_i^0)$$

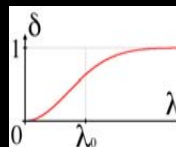
## 2. FFD Hairstyle Deformation

- Attaching Strand Roots on the Skull
  - Varying progressively rigid motion attachment from the strand root to its extremity
  - Typical attachment length  $\lambda_0$ 
    - Acts as a hair rigidity parameter

$$\delta = 1 - \exp(-\lambda^2 / \lambda_0^2)$$



No Attach Low Attach High Attach



## 3. Animating the FFD Lattice

- Mechanical Properties to be Simulated
  - Mechanical behavior of hair:
    - Density, Elasticity
    - Interactions between strands
  - External forces exerted on hair:
    - Gravity
    - Aerodynamic effects
  - Collisions between hairs and other objects:
    - Body parts: Skull, shoulders...
    - Other objects

### 3. Animating the FFD Lattice

13

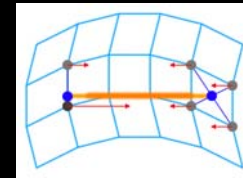
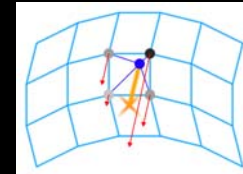
- Building Blocks: Lattice Stiffners
  - Mechanical interaction between lattice nodes
    - Weighted contributions of lattice nodes
  - Viscoelastic behavior law  $\sigma(\epsilon, \epsilon')$ 
    - Relating lattice forces  $f_i$  to positions  $p_j$  and speeds  $p'_j$

$$f_i = w_i \sigma \quad \text{with} \quad \epsilon = \sum_j w_j p_j \quad \text{and} \quad \epsilon' = \sum_j w_j p'_j$$

### 3. Animating the FFD Lattice

14

- Particular Lattice Stiffners
  - Lattice Attachment
    - Relates one location of the lattice to a given point in space
      - Weights: Linear interpolation coefficients of the location in the lattice
  - Lattice Spring
    - Relates two locations of the lattice
      - Weights: Linear interpolation coefficients, positive and negative



### 3. Animating the FFD Lattice

15

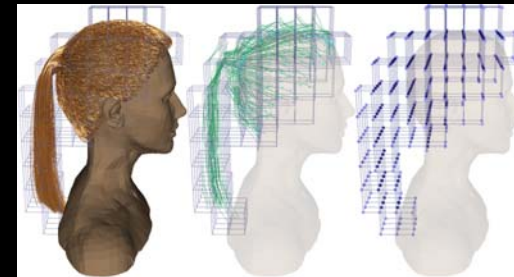
- Hair Strand Discretization
  - Creating lattice springs between grid edges ( $>45^\circ$ )
  - Creating lattice attachment on root



### 3. Animating the FFD Lattice

16

- Hair Strand Discretization
  - Mass of each strand distributed on the lattice nodes using weights of linear interpolation of strand segment extremities

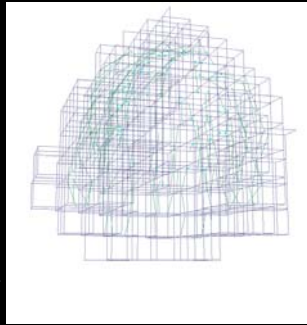




### 3. Animating the FFD Lattice

17

- Decimation of the Model
  - Number of lattice stiffners reduced at will
    - Similar stiffners are consolidated
    - Parallelism and projections in the space of lattice node weights
  - Optimal accuracy - speed tradeoff
    - Initial strands
    - Initial model: 12100 strands, 3500 attachments
    - Low decimation: 800 strands, 200 attachments
    - High decimation: 200 strands, 50 attachments



### 3. Animating the FFD Lattice

18

- The Ether Model
  - Weak viscoelastic forces attaching the lattice nodes to rest position (rigid motion of the head)
    - Modeled as lattice attachments
    - Stabilize and damp the motion of the hairs
    - Improve the robustness of the model for realistic or not-so-realistic head motions
    - Additional custom parameters for simulating hair stiffness (short hair, styling gel...)
    - May include additional physical effects (aerodynamics)



### 3. Animating the FFD Lattice

19

- Collision Effects
  - Between the hair and the body
    - Head
    - Shoulders
  - Metaball-based model of body parts
    - Approximate modeling with a low number of primitives
    - Easy animation of body deformations
    - Attached to the rigid head motion (skull) or the body skeleton (shoulders)

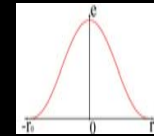
### 3. Animating the FFD Lattice

20

- Collisions with Metaballs
  - 6th-order polynomial metaballs ( $n=3$ ) modeled as lattice attachments

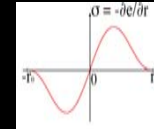
• Potential:

$$e = \begin{cases} \frac{e_0}{r_0^{2n}} s^n & \text{if } s \geq 0 \\ 0 & \text{elsewhere} \end{cases} \quad \text{with } s = r_0^2 - |\mathbf{e} - \mathbf{e}_0|^2$$



• Force:

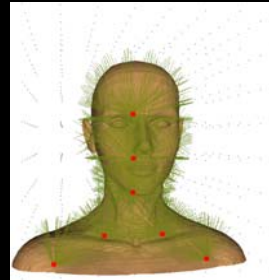
$$\boldsymbol{\sigma} = -\frac{\partial e}{\partial \mathbf{e}} = \begin{cases} \frac{e_0}{r_0^{2n}} (2n) s^{n-1} (\mathbf{e} - \mathbf{e}_0) & \text{if } s \geq 0 \\ 0 & \text{elsewhere} \end{cases}$$



### 3. Animating the FFD Lattice

21

- Customizing Metaballs on Lattice Nodes
  - Low number of metaballs
  - Their center are attached to the skeleton
  - Their radius is customized to each lattice node according to its position

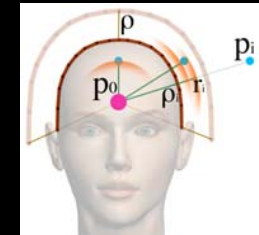
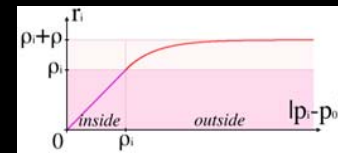


### 3. Animating the FFD Lattice

22

- Customizing Metaballs on Lattice Nodes
  - The metaball radius is chosen so as to:
    - Keep nodes outside the skull to a given distance from the surface
    - Prevent nodes inside the skull from moving deeper

$$r_i = \begin{cases} \rho_i + \rho \left( 1 - \exp\left(-\frac{\rho_i - |p_i - p_d|}{\rho}\right) \right) & \text{if } |p_i - p_d| \geq \rho_i \\ |p_i - p_d| & \text{elsewhere} \end{cases}$$



### 3. Animating the FFD Lattice

23

- Customizing Metaballs on Lattice Nodes
  - Normalizing the strength of the metaballs
    - The base potential of the metaball should be chosen so as to give same repulsion (acceleration) whatever:
      - The mass of the lattice node
      - The radius of the metaball

$$e_i = r_i^2 m_i \gamma$$

### 3. Animating the FFD Lattice

24

- Metaball-Based Collision Handling
  - Approximate collision response
  - Low computational overhead
  - Suited for animation



### 3. Animating the FFD Lattice

- Numerical Integration
  - Particle system integrated by using the implicit Inverse Euler method
    - Using the Conjugate Gradient as linear system solver
    - On-the-fly matrix assembly for capturing all the nonlinearities of the mechanical system
  - Good robustness with any mechanical system when using constant time steps

### 4. Scalability and Level-of-Detail

- Options for Adjusting the Accuracy Tradeoff
  - Number of lattice attachments
    - The more attachments, the more accurate the mechanical behavior of the hairs
  - Size of the free-form deformation grid
    - The finer the grid, the richer and diverse the deformation patterns of the hair
  - Number of metaballs
    - The more metaballs, the more accurate the collisions between the hair and the body

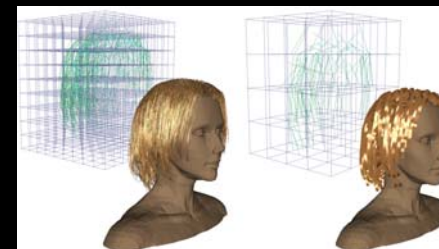
### 4. Scalability and Level-of-Detail

- Performance and Level-of-Detail
  - Computation times for a 1/25 s frame on a 3GHz Pentium4 PC computer

Mech.Mod	0 Att.	0 Att.	50 Att.	100 Att.	200 Att.
-----	0 Spr.	0 Spr.	200 Spr.	400 Spr.	800 Spr.
Lat.Size	0 Meta.	7 Meta.	7 Meta.	7 Meta.	7 Meta.
10x10x10	1.1 ms	5.2 ms	10.1 ms	13.8 ms	21.4 ms
5x5x5	0.1 ms	0.2 ms	2.1 ms	3.8 ms	7.6 ms

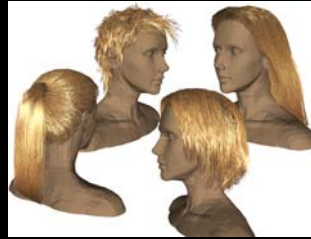
### 4. Scalability and Level-of-Detail

- Performance and Level-of-Detail
  - Level-of-Detail schemes have to be combined with adequate multiresolution rendering techniques



# 4. Scalability and Level-of-Detail

- Benefits of the FFD Model
  - Scalability, for efficient LOD implementations
  - Compatible with any hair rendering method
  - Weak reliance on the strand nature of hairs
  - Total freedom of hairstyle design using any standard design tool



## Modeling Hair using Levels of Detail

### Ming C. Lin and Kelly Ward

Joint work with S. Fisher, N. Galoppo, and J. Lee  
University of North Carolina at Chapel Hill  
[lin@cs.unc.edu](mailto:lin@cs.unc.edu) and [wardk@cs.unc.edu](mailto:wardk@cs.unc.edu)  
<http://www.cs.unc.edu/~lin> and <http://www.cs.unc.edu/~wardk>  
<http://gamma.cs.unc.edu/>

Interactive Hair Modeling – Ming C. Lin

## Motivation

- Hair simulation and rendering are complex
  - High number of primitives
  - Collisions among hairs and between hair & body
- Realistic appearance desirable for many interactive applications
  - Virtual Reality, Games, Entertainment



Modeling Hair & Cloth – Ming C. Lin

## Hair Modeling



Movie: Final Fantasy



Video Game: Lara Croft - Tomb Raider: Angel of Darkness

- Factors to consider
  - Speed vs. Appearance
  - Short vs. Long
  - Wavy vs. Straight
- Choose hair model based on desired outcome or intended applications

Modeling Hair & Cloth – Ming C. Lin

## Hair Simulation

- Physics of dynamic simulation
  - Control motion
  - Different styles
- Geometric representation of hair
  - Individual strands
  - Wisps
  - Strips
- Collision detection and response
  - Hair-Object Interactions
  - Hair-Hair Interactions

Modeling Hair & Cloth – Ming C. Lin



## Our Themes

- Discrete and continuous LODs for hair
  - Three representations to model hair
  - Hair hierarchy resulting in varying resolutions of hair
- LOD framework leads to:
  - Accelerated dynamics and rendering
  - Accurate and efficient collision detection method
  - Ability to model different hairstyles
  - Flexible dynamics model that can accommodate changing hair properties, e.g. hairspray

Modeling Hair & Cloth – Ming C. Lin

## Organization

- Overview
- Previous Work
- LOD Framework for Modeling Hair
  - *Discrete Representations*
  - *Hair Simulation*
  - *Hair Rendering*
  - *Runtime Selection*
- Hair Hierarchy
- Summary & Future Directions

Modeling Hair & Cloth – Ming C. Lin

## Organization

- **Overview**
- Previous Work
- LOD Framework for Modeling Hair
  - *Discrete Representations*
  - *Hair Simulation*
  - *Hair Rendering*
  - *Runtime Selection*
- Hair Hierarchy
- Summary & Future Directions

Modeling Hair & Cloth – Ming C. Lin

## Hair Statistics

- Diameter of single hair strand: 58 to 100 microns
- Number of strands on human head: 120,000 to 150,000
- Density: 200 to 300 hairs per cm<sup>2</sup> of scalp
- Weight of hair: 1.3grams per 1cm<sup>3</sup>

*Figures vary per person – Represent averages*  
[L'Oréal 2003]

Modeling Hair & Cloth – Ming C. Lin

## Hair Facts

- 120,000 to 150,000 of independent strands of hair on head
  - Do not always act independently
  - Electrostatic forces
  - Oils on hair
  - Hairstyles (Curly)
- Natural clumping tendency within hair
  - Strands group together and behave as if one

Modeling Hair &amp; Cloth – Ming C. Lin

## Overview

- Hair Modeling [Magnenat-Thalmann, et al. 2000]
  - Hair Simulation
    - Dynamic Simulation
    - Collision Detection
  - Hair Rendering
    - Hair color, shadows, lighting, transparency, and anti-aliasing
  - Hairstyling
    - Geometry of the hair – Shape specification
    - Density, distribution, and orientation of hair

Modeling Hair &amp; Cloth – Ming C. Lin

## Overview

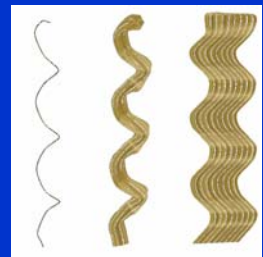
- Propose method to model hair using three discrete representations
- Create *hair hierarchy* from these representations creating continuous LODs
- Dynamically switch between different representations based on:
  - Visibility
  - Viewing Distance
  - Motion of Hair

Modeling Hair &amp; Cloth – Ming C. Lin

## Overview

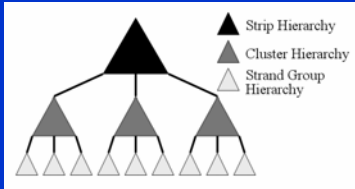
- Three Representations to model hair [Ward et al.; CASA 2003]

- **Individual Strands**
  - Modeled with subdivision curves
- **Clusters**
  - Generalized cylinders created with subdivision surfaces
- **Strips**
  - Modeled with subdivision surfaces



Modeling Hair &amp; Cloth – Ming C. Lin

## Overview



- Hair Hierarchy [Ward and Lin; Pacific Graphics 2003]
  - Create hierarchies of each representation
  - Generates continuous LODs

Modeling Hair & Cloth – Ming C. Lin

## Overview

- LOD Framework
  - Geometric hair representation
  - Dynamic simulation
    - Implicit integration
    - Collision detection
  - Rendering method
  - Hairstyling flexibility
    - Various hairstyles
    - Addition of external substances, e.g. hairspray

Modeling Hair & Cloth – Ming C. Lin

## Organization

- Overview
- **Previous Work**
- LOD Framework for Modeling Hair
  - *Discrete Representations*
  - *Hair Simulation*
  - *Hair Rendering*
  - *Runtime Selection*
- Hair Hierarchy
- Summary & Future Directions

Modeling Hair & Cloth – Ming C. Lin

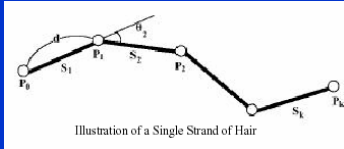
## Previous Work

- Hair Simulation
  - **Strands**
    - Model individual strands as a series of connected line segments  
[ Anjyo et al. 1992; Kurihara et al. 1993; Daldegan et al. 1993 ]
  - Wisps / Clusters
  - Strips

Modeling Hair & Cloth – Ming C. Lin

## Individual Hair Strands

- Shape represented by angles specified btw two segments
- Polar coordinates
  - Zenith  $\theta_i$
  - Azimuth  $\phi_i$
- Specify resting position for hair as  $\theta_0$  and  $\phi_0$



Modeling Hair & Cloth – Ming C. Lin

## Physics of Motion

$$M_{\theta \text{ spring}} = -k_{\theta}(\theta - \theta_0)$$

$$M_{\phi \text{ spring}} = -k_{\phi}(\phi - \phi_0),$$

where  $k_{\theta}$  and  $k_{\phi}$  are spring constants and  $\theta_0$  and  $\phi_0$  are initial angles

$$M_{\theta \text{ external}} = u F_{\theta}$$

$$M_{\phi \text{ external}} = v F_{\phi},$$

- $u$  is  $(1/2)d$ ,
- $d$  is the length of a segment of hair
- $v$  is the half length of the segment that is the projection of  $s_i$  onto the  $\Phi$  plane.

Modeling Hair & Cloth – Ming C. Lin

## Collision Response

- Collision Reaction
  - Use lookup table and bi-linear interpolation to find normal vectors for collision response direction
  - Reaction constraint method by Platt and Barr 1988 is used:

$$F_{\text{unconstrained}} = F_{\text{input}} - (F_{\text{input}} \cdot N) N,$$

$$F_{\text{constrained}} = -(k P T + c V \cdot N) N,$$

$N$  = normal vector at point  $T$

$V$  = velocity of point  $P$

$c$  = damping coefficient

$k$  = strength of the constraint

$F_{\text{input}}$  = applied force to node point  $P$

Modeling Hair & Cloth – Ming C. Lin

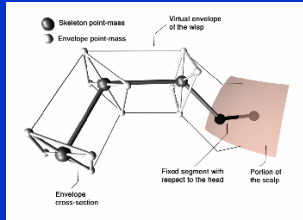
## Previous Work

- Hair Simulation
  - Strands
  - Wisps / Clusters
    - Individual strands are grouped together and simulated in bundles [ Kurihara et al. 1993; Plante et al. 2001; Chen et al. 1999; Xu Yang 2001 ]
    - Adaptive Wisp Trees [ Bertails et al. 2003 ]
  - Strips

Modeling Hair & Cloth – Ming C. Lin

## Layered Wisp Model

- **Skeleton Curve**
  - Defines global motion and deformations
- **Deformable Envelope**
  - Coats skeleton, defines deformations of wisp sections
- **Hair Strands**
  - Individual strands of hair for rendering



Modeling Hair & Cloth – Ming C. Lin

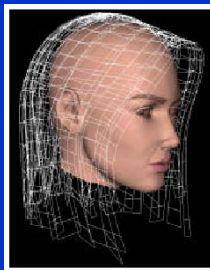
## Previous Work

- Hair Simulation
    - Strands
    - Wisps / Clusters
    - **Strips**
      - Thin shell volumes and 2D strips used to represent groups of hair
- [ Kim and Neumann 2000; Koh and Huang 2000; Koh and Huang 2001]

Modeling Hair & Cloth – Ming C. Lin

## Setup of Hair

- Hair strands are represented in layers of strips (NURBS) overlaying each other
- Dynamic equations are defined and solved for the control points of the surface
- Surfaces are texture mapped with hair images
- Alpha map is used to define transparency



Modeling Hair & Cloth – Ming C. Lin

## Setup of Hair



Modeling Hair & Cloth – Ming C. Lin



## Previous Work

- Hair Rendering
  - Anisotropic Lighting [ Kajiya and Kay 1989; Heidrich and Seidel 1998 ]
  - *Light Scattering from Human Hair Fibers* [ Marschner et al. 2003 ]
  - Shadows [ *Opacity Shadow Maps*: Kim and Neumann 2001 ]
- Hairstyling
  - Multi-resolution technique for styling hair [ Kim and Neumann, 2002 ]

Modeling Hair & Cloth – Ming C. Lin

## Previous Work

- Geometric Levels-of-Detail
  - Accelerates rendering of complex geometric models [ Luebke et al. 2002 ]
- Simulation Levels-of-Detail
  - Used to simplify or approximate dynamics in a scene [ Cheney and Forsyth 1997; Carlson and Hodgins, 1997; Perbet and Cani 2001; O'Brien et al. 2001 ]

Modeling Hair & Cloth – Ming C. Lin

## Organization

- Overview
- Previous Work
- **LOD Framework for Modeling Hair**
  - *Discrete Representations*
  - *Hair Simulation*
  - *Hair Rendering*
  - *Runtime Selection*
- Hair Hierarchy
- Summary & Future Directions

Modeling Hair & Cloth – Ming C. Lin

## Organization

- Overview
- Previous Work
- **LOD Framework for Modeling Hair**
  - *Discrete Representations*
  - *Hair Simulation*
  - *Hair Rendering*
  - *Runtime Selection*
- Hair Hierarchy
- Summary & Future Directions

Modeling Hair & Cloth – Ming C. Lin

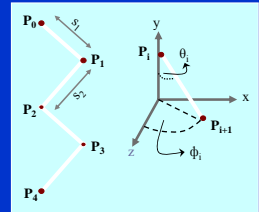
## Subdivision Framework

- Subdivision curves and surfaces used for geometric representations
- Ability to model each LOD shape and different hairstyles
- Adaptive dynamic tessellation
  - Continuous LODs for rendering

Modeling Hair &amp; Cloth – Ming C. Lin

## Base Skeleton

- All hair representations follow the same base skeleton model as basis of motion
  - Maintains global physical behavior of hair
  - Modeled as open chain of line segments
  - Spring forces used to control angles



Modeling Hair &amp; Cloth – Ming C. Lin

## Subdivision Framework

- Subdivision curves and surfaces used for geometric representations
- Ability to model each LOD shape and different hairstyles
- Adaptive dynamic tessellation
  - Continuous LODs for rendering

Modeling Hair &amp; Cloth – Ming C. Lin

## Strips


- Coarsest LOD
  - Used primarily when hair is not visible
- Creating Strips
  - 2 Control points per skeleton node define strip
    - Collinear with skeleton node
  - Subdivide control points
  - Texture map with hair image
  - Alpha map to give appearance of strands



Modeling Hair &amp; Cloth – Ming C. Lin

## Clusters


- Intermediate LOD
  - Generalized Cylinder
- Creating Clusters
  - Circular cross-section of control points at each skeleton node
  - Subdivide control points
  - Texture map with hair image
  - Alpha map to give appearance of strands



Modeling Hair & Cloth – Ming C. Lin

## Individual Strands

- Finest LOD
- Creating Strands
  - One control point per skeleton node
  - Subdivide control points
- Group multiple strands together
  - *Strand Group*



Modeling Hair & Cloth – Ming C. Lin

## Organization

- Overview
- Previous Work
- **LOD Framework for Modeling Hair**
  - *Discrete Representations*
  - **Hair Simulation**
  - *Hair Rendering*
  - *Runtime Selection*
- Hair Hierarchy
- Summary & Future Directions

Modeling Hair & Cloth – Ming C. Lin

## Hair Dynamics

- Base skeleton controls hair dynamics
  - Each control point is governed by a set of ordinary differential equations [Anjyo et al. 1992; Kurihara et al. 1993]

$$I_i \frac{d^2 \theta_i}{dt^2} + \gamma_i \frac{d\theta_i}{dt} = M_\theta \qquad I_i \frac{d^2 \phi_i}{dt^2} + \gamma_i \frac{d\phi_i}{dt} = M_\phi$$

$$M_\theta = M_{\theta_{spring}} + M_{\theta_{external}}$$

$$M_\phi = M_{\phi_{spring}} + M_{\phi_{external}}$$

- The torque due to spring force is calculated by:
 
$$M_{\theta_{spring}} = -k_\theta (\theta - \theta_j)$$

$$M_{\phi_{spring}} = -k_\phi (\phi - \phi_j)$$

Modeling Hair & Cloth – Ming C. Lin

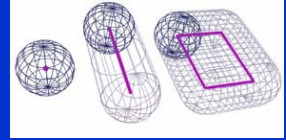
## Collision Detection & Response

- Typically most time-consuming process in simulation
  - Need efficient yet accurate method
- Exploit level-of-detail framework
  - Simplify collision detection and response algorithm
  - Still maintains accurate, consistent simulation state
- Two types of collisions
  - Object-Hair Interactions
  - Hair-Hair Interactions

Modeling Hair & Cloth – Ming C. Lin

## Collision Detection

- *Swept Sphere Volumes (SSVs)* [Larsen et al. 1999]
  - Bounding volumes to encapsulate hair
  - Shape closely matches each LOD geometry
- Simple collision detection algorithm:
  - Calculate distance between core shapes (line or rectangle)
  - Subtract appropriate offset (radius of each SSV)



Modeling Hair & Cloth – Ming C. Lin

## Hair-Hair Interactions

- Use spatial decomposition – Only test sections of hair near each other
- Compute overlap between two SSVs
- Use orientations of SSVs to determine response
  - **Near Parallel:** Average velocities
  - **Else:** Push apart SSVs based on overlap and average velocities

Modeling Hair & Cloth – Ming C. Lin

## Hair-Hair Interactions

Hair-Hair  
Collision Detection

Modeling Hair & Cloth – Ming C. Lin

## Hair-Object Interactions

- Occurs when hair interacts with body/head
  - Impenetrable object
- Encapsulate object with bounding volume hierarchy (BVH) of SSVs
  - Recursively test hair SSV against BVH
- When collision occurs
  - Move section of hair outside of object
  - Set velocity of hair in direction of object to zero

Modeling Hair & Cloth – Ming C. Lin

## Hair-Object Interactions

# Hair-Object Collision Detection

Modeling Hair & Cloth – Ming C. Lin

## Organization

- Overview
- Previous Work
- **LOD Framework for Modeling Hair**
  - *Discrete Representations*
  - *Hair Simulation*
  - **Hair Rendering**
  - *Runtime Selection*
- Hair Hierarchy
- Summary & Future Directions

Modeling Hair & Cloth – Ming C. Lin

## Hair Rendering

- Anisotropic Lighting [Kajiya and Kay 1989]
  - OpenGL texture matrix implementation [Heidrich and Seidel 1998]
- Self-Shadowing
  - Opacity shadow maps [Kim and Neumann 2001]
    - Fast approximations of deep shadow maps

Modeling Hair & Cloth – Ming C. Lin



## LOD Rendering

- Alleviate visual disturbances
- Alpha value in textures for clusters and strips
- Density values for shadows for clusters and strips based on number of strands represented
- Blend images of previous and current LODs to make transitions smoother

Modeling Hair & Cloth – Ming C. Lin

## Organization

- Overview
- Previous Work
- **LOD Framework for Modeling Hair**
  - *Discrete Representations*
  - *Hair Simulation*
  - *Hair Rendering*
  - **Runtime Selection**
- Hair Hierarchy
- Summary & Future Directions

Modeling Hair & Cloth – Ming C. Lin

## Runtime Selection

- Put more computational resources on hairs that are most important to viewer
  - Visibility
  - Viewing Distance
  - Hair Motion
- Entire head of hair is using combination of different hair resolutions at any given point
  - Add more detail or simplify at any time on any position on head

Modeling Hair & Cloth – Ming C. Lin

## Visibility

- Hair cannot be seen if:
  - Outside field of view of camera
  - Occluded by object in scene
- Perform simple occlusion tests on hair to determine visibility
- If hair is not visible
  - Simulate hair as strips
  - Do not render hair

Modeling Hair & Cloth – Ming C. Lin

## Visibility

### Occlusion Based Transitions

Modeling Hair & Cloth – Ming C. Lin

## Viewing Distance

- Hair that is far from viewer cannot be seen in great detail
- Determine amount of screen space hair covers
  - As distance from viewer grows, number of pixels covered by hair decreases
  - Calculate screen space covered by skeleton
    - Test value against maximum allowable size for each LOD

Modeling Hair & Cloth – Ming C. Lin

## Viewing Distance

### Distance Based Transitions

- Color-Coded
- Strands: Yellow
  - Clusters: Red
  - Strips: Blue

Modeling Hair & Cloth – Ming C. Lin

## Hair Motion

- Relatively still hair does not have much detail to view
  - Wind or large avatar motion causes more viewable detail
- Measure the largest velocity of each hair section
  - Compare velocity to thresholds for strands vs. clusters vs. strips

Modeling Hair & Cloth – Ming C. Lin

## Hair Motion

Color-Coded

- Strands: Yellow
- Clusters: Red
- Strips: Blue

### Motion Based Transitions


Modeling Hair & Cloth – Ming C. Lin

## Choosing Final Representation

```
if (occluded == true)
    Simulate as strip
    Do not render
else
    DistanceLOD = TestHairDistance()
    MotionLOD = TestHairMotion()
    // Use whichever test requires highest LOD
    if ( DistanceLOD > MotionLOD )
        CurrentLOD = DistanceLOD
    else
        CurrentLOD = MotionLOD
```


Modeling Hair & Cloth – Ming C. Lin

## Demonstrations



Modeling Hair & Cloth – Ming C. Lin

## Results



Modeling Hair & Cloth – Ming C. Lin

## Results



Modeling Hair & Cloth - Ming C. Lin

## Performance

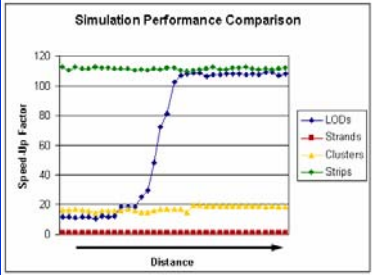
	LODs	Strands	Clusters	Strips
Dynamic Simulation	0.0175	0.5834	0.0298	0.00592
Collision Detection	0.0567	2.1934	0.1297	0.01896
<b>Total</b>	<b>0.0742</b>	<b>2.7768</b>	<b>0.1595</b>	<b>0.02488</b>

*Measured in Seconds/Frame*

- Benchmark:
  - 8045 Strands
  - 519 Clusters
  - 173 Strips
- Wind blowing through hair as camera moves away from figure

Modeling Hair & Cloth - Ming C. Lin

## Performance - Simulation

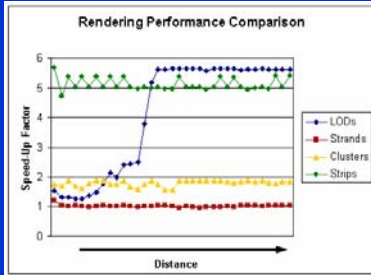


Simulation Performance Comparison

- Benchmark:
  - 8045 Strands
  - 519 Clusters
  - 173 Strips
- Up to 110 times speed-up

Modeling Hair & Cloth - Ming C. Lin

## Performance - Rendering



Rendering Performance Comparison

- Benchmark:
  - 8045 Strands
  - 519 Clusters
  - 173 Strips
- Up to almost 6 times speed-up

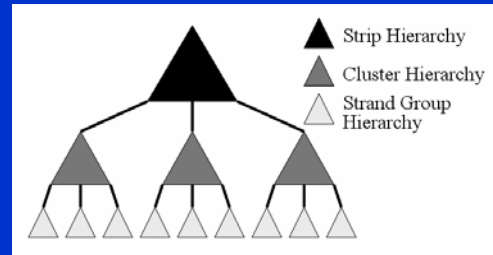
Modeling Hair & Cloth - Ming C. Lin

## Organization

- Overview
- Previous Work
- LOD Framework for Modeling Hair
  - *Discrete Representations*
  - *Hair Simulation*
  - *Hair Rendering*
  - *Runtime Selection*
- **Hair Hierarchy**
- Summary & Future Directions

Modeling Hair & Cloth – Ming C. Lin

## Hair Hierarchy



### Hierarchy of Hierarchies

- Contains both discrete and continuous LODs

Modeling Hair & Cloth – Ming C. Lin

## Benefits of Hair Hierarchy

- Gives more localized control over simulation
- Generates continuous LODs throughout simulation
- Transitions between LODs are less noticeable
  - Natural progression in detail
- Hierarchy can be traversed on the fly
- Results in higher visual quality simulation

Modeling Hair & Cloth – Ming C. Lin

## Construction of Hair Hierarchy

- Constructed offline as a pre-process
- Continual subdivision of strand groups, clusters, and strips
- Build hierarchies in top-down manner
- Gain more detail down the hierarchy

Modeling Hair & Cloth – Ming C. Lin

## Hierarchy Construction

- Strip Hierarchy
  - Process continues until user-defined width reached
- Clusters are continually subdivided until user-defined threshold reached

Modeling Hair & Cloth – Ming C. Lin

## Subdivision of Hair

- Cross-section of strand grouping
- Divide into four equal parts
  - LSS tightly fits for collision detection
- Placement of strands remains consistent

Modeling Hair & Cloth – Ming C. Lin

## Subdivision of Hair

- Cross-section of strand grouping
- Divide into four equal parts
  - LSS tightly fits for collision detection
- Placement of strands remains consistent
- Fit circular cross-section to four children

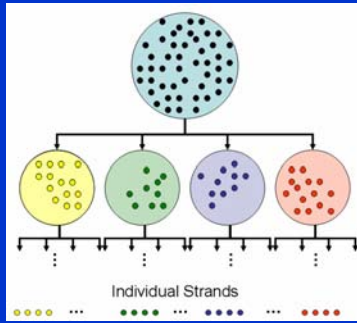
Modeling Hair & Cloth – Ming C. Lin

## Subdivision of Hair

Modeling Hair & Cloth – Ming C. Lin



## Subdivision of Hair



Modeling Hair & Cloth – Ming C. Lin

## Hierarchy Construction

- Strand group subdivision continues until nodes consist of single strand
  - Each node has 0 – 4 children
- Quad-tree data structure holds information for clusters and strand groups
  - Binary tree used for strip subdivision information

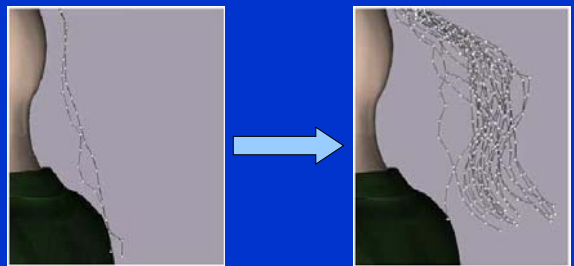
Modeling Hair & Cloth – Ming C. Lin

## Transitions Between LODs

- Hair hierarchy allows localized LOD control of hair
- Traverse hierarchy to select appropriate LOD throughout simulation
- Crucial to perform transitions smoothly to avoid visual artifacts
- During traversal either:
  - **Adding Detail:** One skeleton → Multiple skeletons
  - **Simplifying:** Multiple skeletons → One skeleton

Modeling Hair & Cloth – Ming C. Lin

## Adaptive Splitting



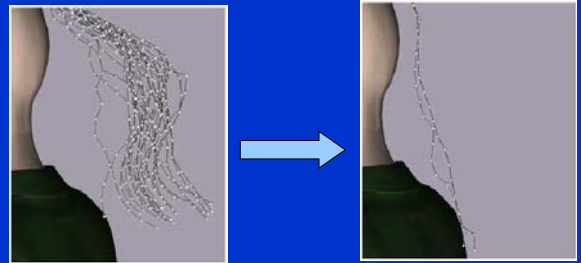
Modeling Hair & Cloth – Ming C. Lin

## Adaptive Splitting

- Move down one level in hierarchy
  - Add detail to simulation
- Simplified through use of base skeleton
- Child skeletons inherit dynamic state of parent skeleton
  - Velocity is copied from parent
  - Position is offset from parent

Modeling Hair & Cloth – Ming C. Lin

## Adaptive Grouping



Modeling Hair & Cloth – Ming C. Lin

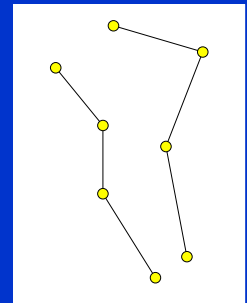
## Adaptive Grouping

- Multiple skeletons → One skeleton
  - Average velocity and position of children and assign to parent
- Constraints placed on merging:
  - All children must be ready to transition to parent
  - Positional constraint on children to avoid sudden jump

Modeling Hair & Cloth – Ming C. Lin

## Adaptive Grouping

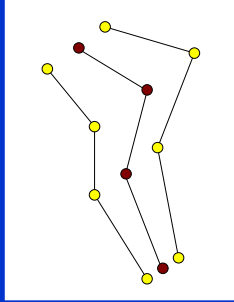
- Average positions of all children



Modeling Hair & Cloth – Ming C. Lin

## Adaptive Grouping

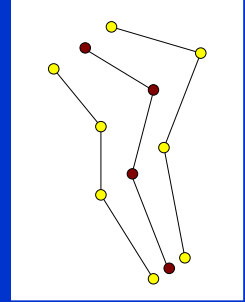
- Average positions of all children



Modeling Hair & Cloth – Ming C. Lin

## Adaptive Grouping

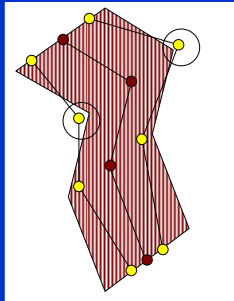
- Average positions of all children
- Measure distance of new nodes from child nodes



Modeling Hair & Cloth – Ming C. Lin

## Adaptive Grouping

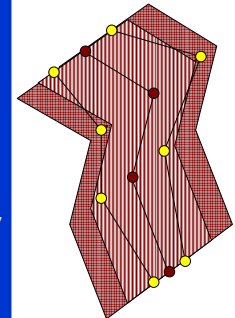
- Average positions of all children
- Measure distance of new nodes from child nodes
- Predefined threshold for “close enough”



Modeling Hair & Cloth – Ming C. Lin

## Adaptive Grouping

- Average positions of all children
- Measure distance of new nodes from child nodes
- Predefined threshold for “close enough”
- Spring force used to subtly pull nodes closer



Modeling Hair & Cloth – Ming C. Lin

## Implicit Integration

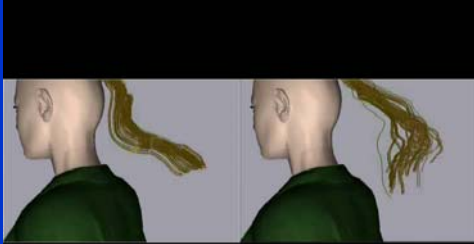
- Achieve greater stability while allowing us to take larger time steps [Ward and Lin 2003]

$$\Delta\omega_\theta = \frac{-hk_\theta(\theta - \theta_0) - h^2k_\theta\omega_{\theta_0}}{1 + h^2k_\theta}$$

- $\omega_{\theta_0} = \omega_\theta(t_0)$  is the angular velocity at time  $t_0$
- $h$  is the time step

Modeling Hair & Cloth – Ming C. Lin

## Results



The image shows two side-by-side views of a mannequin head with a ponytail. The left view, labeled 'Highest-Level Grouping', shows a single, thick, yellowish-brown ponytail. The right view, labeled 'Adaptive Grouping', shows a more detailed and voluminous ponytail with individual strands visible, also in a yellowish-brown color.

**Highest-Level Grouping**      **Adaptive Grouping**


Modeling Hair & Cloth – Ming C. Lin

## Results

Adaptive  
Hair Grouping

Modeling Hair & Cloth – Ming C. Lin

## Results



The image shows a 3D rendering of a woman's back and shoulders. She has long, flowing, reddish-brown hair that is rendered with a high level of detail and volume, demonstrating the results of adaptive hair grouping.

Modeling Hair & Cloth – Ming C. Lin

## Performance

	Fine Strands	LODs	Coarse Strands
Dynamic Simulation	0.1076	0.0416	0.0383
Collision Detection	7.6423	0.4118	0.3383
<b>Total</b>	<b>7.7499</b>	<b>0.4534</b>	<b>0.3766</b>

*Measured in Seconds/Frame*

- 9,350 individual strands (rendered)
- 3,570 skeletons at finest level
- Skeletons contain an average of 6 control points
- Wind Blowing through hair
- Camera remains stationary

Modeling Hair & Cloth – Ming C. Lin

## Performance

	LODs	Strands	Clusters	Strips
Dynamic Simulation	0.02614	0.1076	0.01537	0.00324
Collision Detection	0.23949	7.6432	0.17178	0.02014
<b>Total</b>	<b>0.26563</b>	<b>7.7508</b>	<b>0.18715</b>	<b>0.02338</b>


*Measured in Seconds/Frame*

- 9,350 individual strands (rendered)
- 3,570 skeletons at finest level
- 110 Strips at coarsest strip level
- 330 Clusters at coarsest cluster level
- Wind Blowing through hair
- Camera moves away from figure

Modeling Hair & Cloth – Ming C. Lin

## Hairstyling

- Defines Hair Properties:
  - Shape
  - Distribution
  - Orientation
  - Curl
  - Length
  - Density
- Long tedious process to define properties for all strands (or groups)



Modeling Hair & Cloth – Ming C. Lin

## Organization

- Overview
- Previous Work
- LOD Framework for Modeling Hair
  - Discrete Representations
  - Hair Simulation
  - Hair Rendering
  - Runtime Selection
- Hair Hierarchy
- Summary & Future Directions**

Modeling Hair & Cloth – Ming C. Lin

## Summary

- Simulation and rendering of *realistic* looking hair at interactive rates
- Flexibility to model various hairstyles (curly, straight, long, short)
- Accurately handles multiple simulation scenarios (wind blowing through hair, motion from avatar)
- Ability to allow user to easily decide tradeoff between visual quality and performance

**NOTE:** *Realistic* is a relative term that will be defined in comparison to previous hair simulation techniques that have similar performance

Modeling Hair & Cloth – Ming C. Lin

## Limitations

- Rendering is the bottleneck
- Very aggressive switch of LODs may lead to visual artifacts
- Implementation of occlusion culling is basic
- Other switching criteria have not been explored, e.g. collision

Modeling Hair & Cloth – Ming C. Lin

## External Substances



Real Hair Images

Modeling Hair & Cloth – Ming C. Lin

## Simulation with Substances

- Water
  - Larger mass on hair
  - Hair geometrically changes – Groups together more
- Hairspray
  - Stiffer motion
  - Moves in groups



Modeling Hair & Cloth – Ming C. Lin



## Possible Future Research

- Simulation and rendering with presence of hairspray and water [Ward et al.; CASA 2004]
- More intuitive hairstyling tool
- Accelerate rendering with graphics hardware
- Accelerated occlusion tests using graphics hardware

Modeling Hair & Cloth – Ming C. Lin

## References

- **Modeling Hair Using Level-of-Detail Representations.** Kelly Ward, Ming C. Lin, Joochi Lee, Susan Fisher, and Dean Macri. Proc. of Computer Animation and Social Agents, 2003. <http://gamma.cs.unc.edu/HSLOD/>
- **Adaptive Grouping and Subdivision for Simulating Hair Dynamics,** Kelly Ward and Ming C. Lin, Proc. of Pacific Graphics 2003. <http://gamma.cs.unc.edu/HAIR/>

Modeling Hair & Cloth – Ming C. Lin

## Questions and Comments?

Interactive Hair Modeling – Ming C. Lin

## Anisotropic Lighting

- Kajiya and Kay lighting model for hair

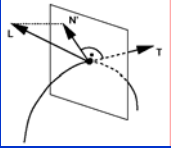
$$\Psi_{diffuse} = k_d \sin(t, l)$$

$$\Psi_{specular} = k_s ((t \cdot l)(t \cdot e) + \sin(t, l) \sin(t, e))^p$$

- $k_d$  – diffuse reflection coefficient
- $k_s$  – specular reflection coefficient
- $t$  – tangent vector
- $l$  – direction of light vector
- $e$  – eye vector
- $p$  – Phong exponent

Modeling Hair & Cloth – Ming C. Lin

## Anisotropic Lighting



- Heidrich and Seidel: OpenGL texture matrix implementation

$$I_o = I_{ambient} + I_{diffuse} + I_{specular}$$

$$I_o = k_d I_a + I_s (k_d (L \cdot N) + k_s (V \cdot R)^n)$$

$I_o$  depends on two dot products (L·T) and (V·T)

- Pre-compute texture containing function  $I_o(L \cdot T, V \cdot T)$
- Compute texture matrix using light direction and viewing direction
- Use tangent vector as texture coordinates

$$(L \cdot N) = \sin(N', T) = \sqrt{1 - (L \cdot T)^2}$$

$$(V \cdot R) = (\sqrt{1 - (L \cdot T)^2}) (\sqrt{1 - (V \cdot T)^2}) - (L \cdot T)(V \cdot T)$$

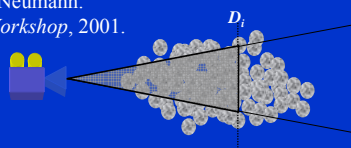

$$\frac{1}{2} \begin{bmatrix} t_x & t_y & t_z & 1 \\ v_x & v_y & v_z & 1 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 2 \end{bmatrix} \cdot \begin{bmatrix} t_x \\ t_y \\ t_z \\ 1 \end{bmatrix} = \begin{bmatrix} \frac{1}{2}(\langle L, T \rangle + 1) \\ \frac{1}{2}(\langle V, T \rangle + 1) \\ 0 \\ 1 \end{bmatrix}$$

Modeling Hair & Cloth - Ming C. Lin

## Opacity Shadow Maps

Tae-Yong Kim and Ulrich Neumann.  
*Eurographics Rendering Workshop, 2001.*

- Fast approximation of deep shadow maps

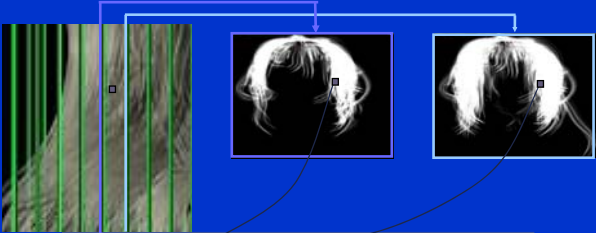
Modeling Hair & Cloth - Ming C. Lin

## Opacity Shadow Maps

- Use set of parallel *opacity maps* perpendicular to light's direction
  - Approximate transmittance function with discrete planar maps
  - Opacity map* can be efficiently generated with hardware
- On each opacity map, scene is rendered from light's POV to alpha buffer
  - Each primitive contributes associated alpha value
  - Each pixel in map stores alpha value that approximates opacity relative to light at pixel's position

Modeling Hair & Cloth - Ming C. Lin

## Opacity Shadow Maps

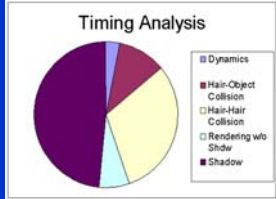


$$\Omega(p_k) = (1.0 - t)\Omega_{prev} + t\Omega_{current}; t = (Depth(p_k) - D_{i-1}) / (D_i - D_{i-1})$$

Opacity values of adjacent maps are sampled and linearly interpolated at position of each shadow computation

Modeling Hair & Cloth - Ming C. Lin

# Timing Breakdown



- Simulation 44.7 %
  - Dynamics 7.34 %
  - Hair-Object 24.61%
  - Hair-Hair 68.04%

- Typical hair simulation
  - Dynamics 3.29%
  - Hair-Object Collision 11.00%
  - Hair-Hair Collision 30.34%
  - Render (w/o Shadow) 6.77%
  - Shadows 48.50%
- Rendering 55.3 %
  - Render (w/o Shadow) 12.26%
  - Shadows 87.74%

# Wisp-tree Representations for Interactive Hair Animation

Marie-Paule Cani

GRAVIR lab,  
Grenoble, France

Real Hair goes from smooth...



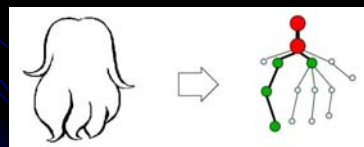
...To complex shapes with visible wisps



## The Wisp-tree Approach

Level of abstraction: a deformable *wisp*  
(group of neighboring strands)

- cohesion between neighboring strands
- strong discontinuities
- anisotropic interactions between wisps



## Wisp Model [Plante, Cani, Poulin02]

### 1. Skeleton curve

- global wisp movement



### 2. Deformable wisp envelope

- radial deformation
- two-way interaction with skeleton



### 3. Hair strands

- inscribed into wisp volume
- do not contribute to simulation



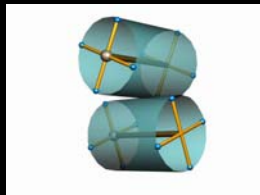
## Hair Strands

- Random position in circle defined in deformed cross-section 2D space
- Pseudo-random number generator reset at each frame for coherence

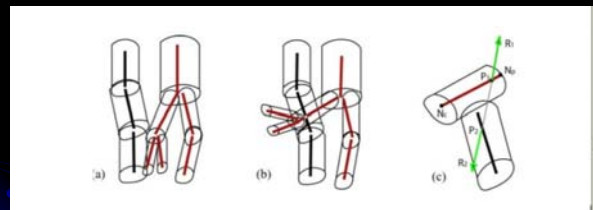


## Wisp Interaction Detection

- Accelerated by regular grid structure
- Test each pair once
- Bounding box test
- Point-in-volume test



## Anisotropic Response to Collision



- Interpenetration versus collision

## Collisions with Character

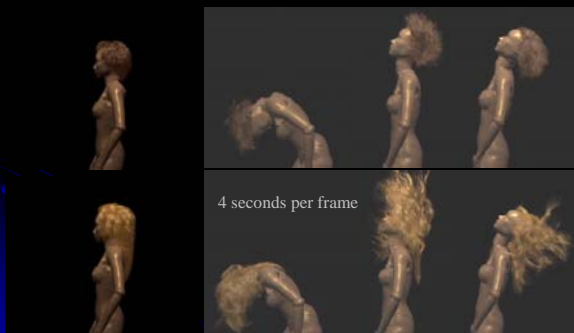
- Detection using regular 3D grid
- Collision is considered perfectly inelastic
- Collision avoidance
- Penetration detection from root to tip for plumb-line algorithm

## Algorithm for Animation

1. Compute the set of applied forces
2. Detect wisp and model interactions
3. Process velocities, apply velocity modifications
4. Process positions, apply position modifications

## Results

[Plante Cani Poulin, GMOD 02]



## Adaptive Wisp-tree

[Bertails, Kim, Cani, Neumann, SCA'03]

### Increase efficiency ?

- Animate hair only where needed !





## Our approach

### Adaptive clustering

- concentrate computation where needed
- interactive rates



© www.longs-cheveux.com

### Recent related work

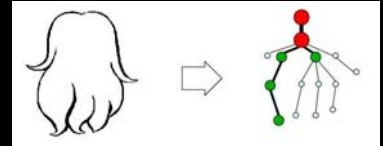
[Ward et al. CASA 03]

[Bando et al. EG 03]

[Ward, Lin PG 03]

## Adaptive Wisp Tree

- Tree-like structure



- Adaptive splitting and merging

## Animation

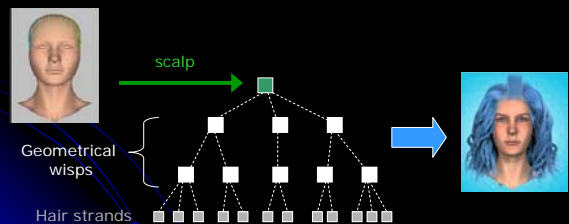
### At each time step

1. Adapt the wisp tree
  - Split where  $(\text{acceleration} \cdot \text{radius})$  is high
  - Merge where motion is uniform
2. Integrate motion of the skeleton
3. Process hair collisions (with body + inside hair)
  - Detection: grid + closest points on cylinder axes
  - Anisotropic response
4. Render hair strands

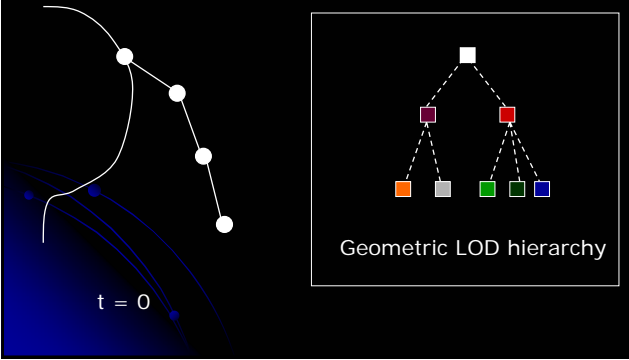
## Design of a hair hierarchy

[Kim-Neumann 02]

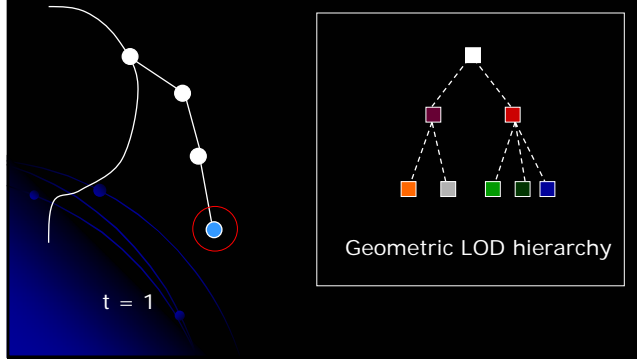
- The user designs a multiresolution hair geometry
- Adaptation is based on these LODs during motion



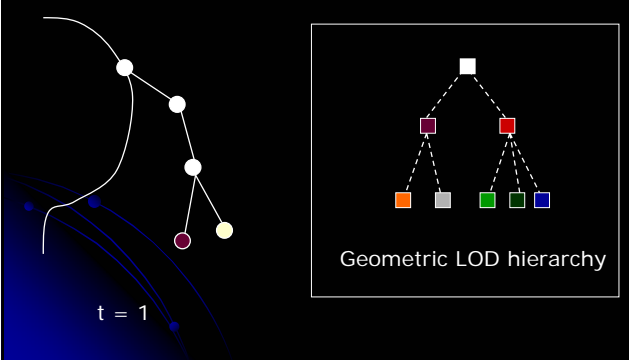
### Example (animation)



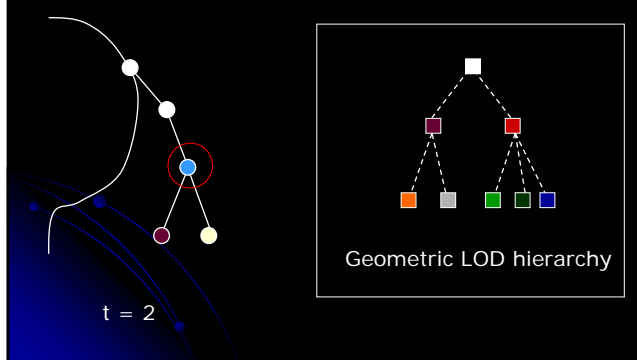
### Example



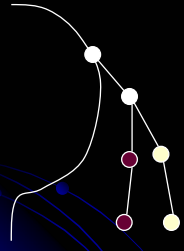
### Example



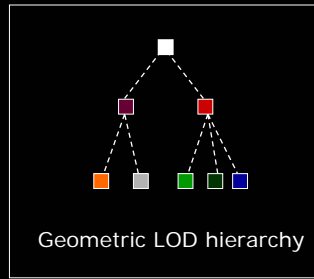
### Example



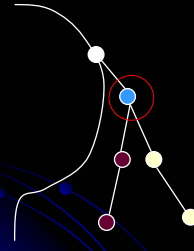
### Example



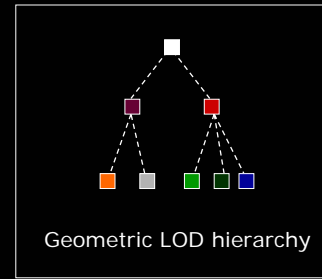
t = 2



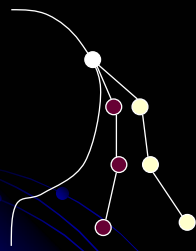
### Example



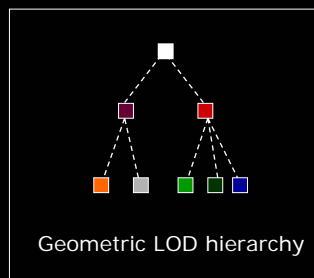
t = 3



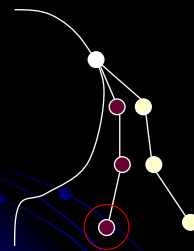
### Example



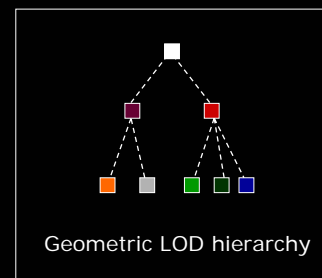
t = 3



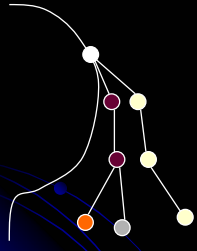
### Example



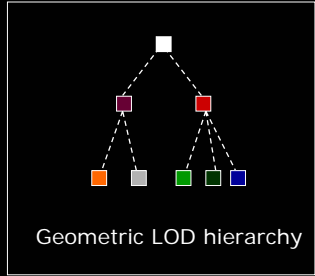
t = 4



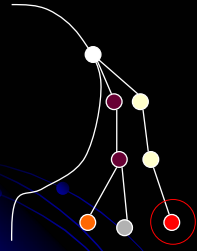
### Example



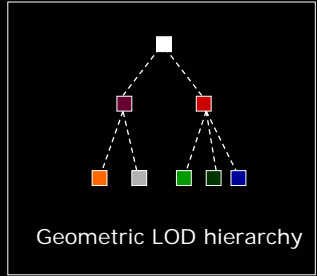
t = 4



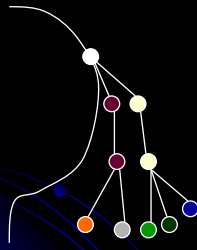
### Example



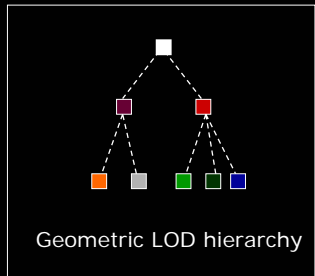
t = 5



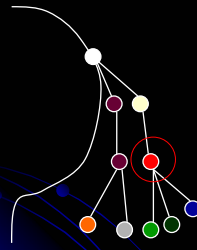
### Example



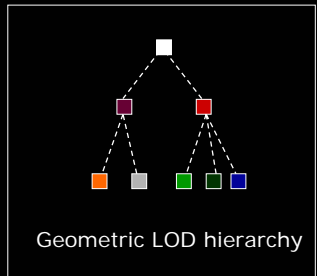
t = 5



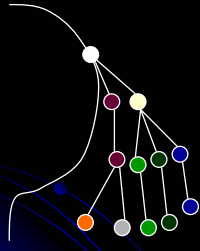
### Example



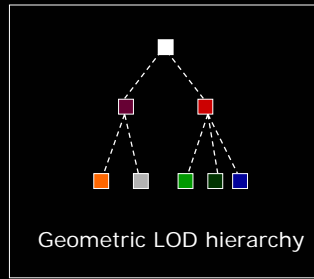
t = 6



### Example

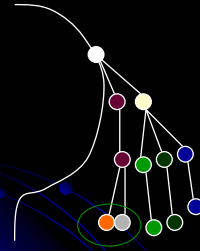


t = 6

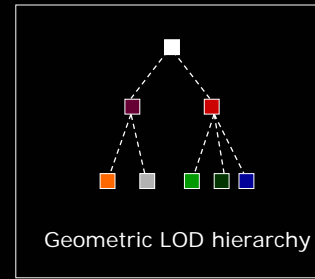


Geometric LOD hierarchy

### Example

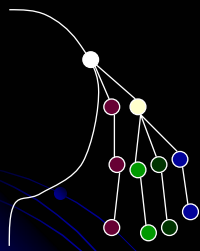


t = 7

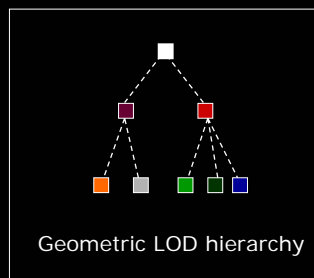


Geometric LOD hierarchy

### Example

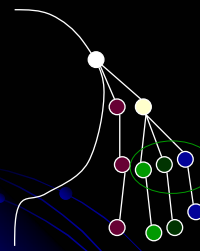


t = 7

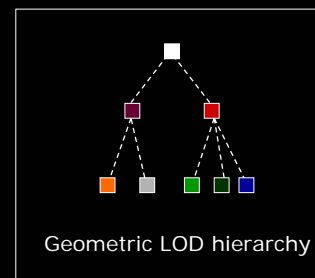


Geometric LOD hierarchy

### Example

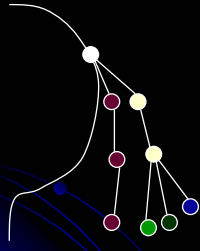


t = 8

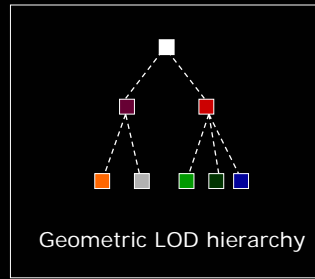


Geometric LOD hierarchy

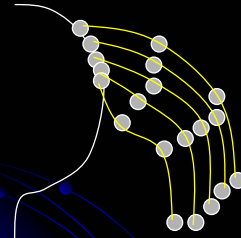
## Example



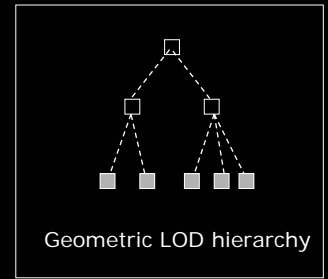
t = 8



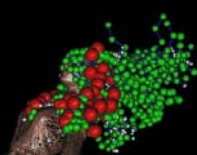
## Example (rendering)



Finest level of the hierarchy



## Results



## Video: Smooth hair

# Smooth hair

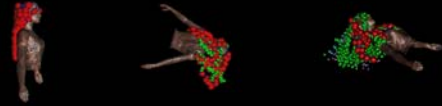


Video: Curly hair

# Curly hair

## Results

Adaptive behavior: qualitative validation



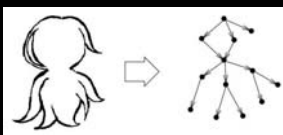
Performances (30 sec to 5 min per animation)

Hair model	Single-level hair	Adaptive Wisp Tree
Short (n=5149)	12.4	1.28
Long (n=6053)	15.2	0.36

Mean time in sec/frame (using a 1.7 GHz Pentium CPU)

## Conclusion

- Modeling of hair dynamic clustering
  - Adapted to many hair styles



- Interactive simulation rates
- Stable
  - avoids micro-collisions between small wisps at rest

## A Practical Self-Shadowing Algorithm for Interactive Hair Animation

Florence Bertails Clément Ménéier Marie-Paule Cani

GRAVIR Lab - IMAG/INRIA  
Grenoble, France



## Motivation: self-shadowing



- Depict the hair volume
- Improve the realism of hair animation



## Goals

- ❖ Interactive hair rendering
- ❖ Adapted to hair animation
- ❖ Good quality of rendering
- ❖ Easy and fast to implement
- ❖ Portable application



## Outline

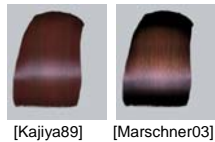
- I. Previous work on hair rendering
- II. 3D light-oriented shadow map
- III. Self-shadowing algorithm
- IV. Extensions
- V. Results and conclusion



## Hair rendering

### Local illumination

- Widely-used model [Kajiya89]
- The most realistic [Marschner03]



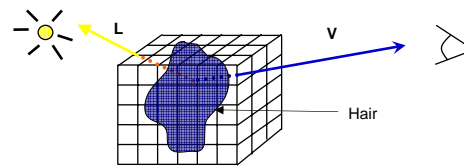
### Self-shadowing

⇒ The main focus of our paper



## Hair self-shadowing (1/2)

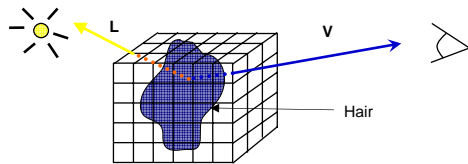
- Volume rendering
  - Rendering of fur [Kajiya89]



## Hair self-shadowing (1/2)

- Volume rendering
  - Rendering of fur [Kajiya89]

- ☺ Good quality of rendering
- ☹ Computationally expensive



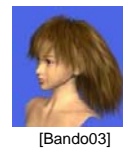
## Hair self-shadowing (1/2)

- Volume rendering
  - Rendering of fur [Kajiya89]

- ☺ Good quality of rendering
- ☹ Computationally expensive



- Billboard splatting [Bando03]



## Hair self-shadowing (1/2)

- Volume rendering
  - Rendering of fur [Kajiya89]



[Kajiya89]

- ☺ Good quality of rendering
- ☹ Computationally expensive

- Billboard splatting [Bando03]



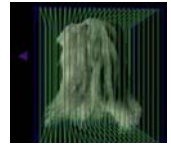
[Bando03]

- ☺ Well-suited for interactive rendering
- ☹ Not adapted for rendering the fine hair geometry

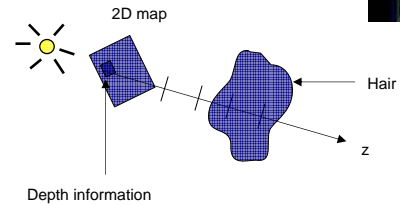


## Hair self-shadowing (2/2)

- Shadow maps
  - Deep shadow map [Lokovic00]
  - Opacity shadow map [Kim02]
  - GPU acceleration [Koster04, Mertens04]



[Kim02]



## Hair self-shadowing (2/2)

- Shadow maps
  - Deep shadow map [Lokovic00]
  - Opacity shadow map [Kim02]
  - GPU acceleration [Koster04, Mertens04]
    - ☺ Interactive rendering
    - ☹ Not easily portable (advanced GPU features)



## Our contribution

⇒ An interactive CPU-based method for hair self-shadowing

### Main idea

Combining :

- An optimized volumetric representation for hair
- A light-oriented partition of space

### Benefits of the approach

- Well-suited for rendering animated hair efficiently
- Provides an inexpensive way of processing hair self-collisions
- Can easily be parallelized



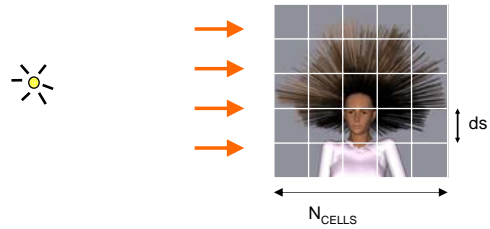
## Outline

- I. Previous work on hair rendering
- II. 3D light-oriented shadow map
- III. Self-shadowing algorithm
- IV. Extensions
- V. Results and conclusion

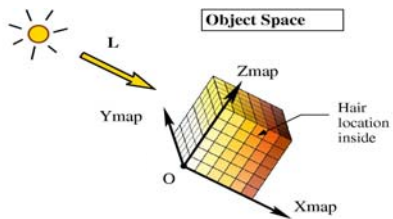


## Assumptions

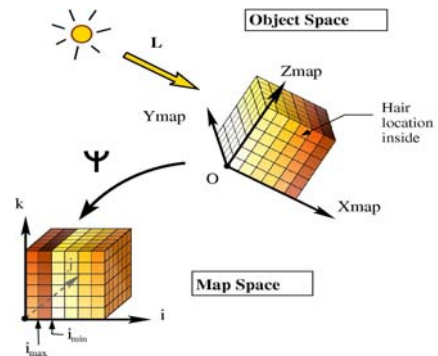
- Directional light
- Hair bounding box of fixed size  
⇒ uniform cubic grid of fixed size



## Light-oriented 3D map

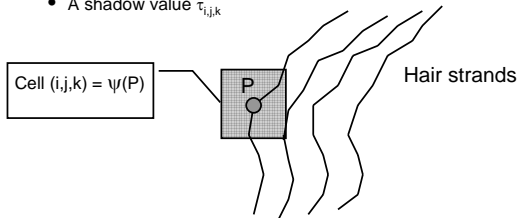


## Light-oriented 3D map



## Hair – map relationship

- A hair model composed of a set of hair segments
- Each hair 3D point can be associated to a map cell
- A map cell  $(i,j,k)$  contains:
  - A density value  $d_{i,j,k}$
  - A shadow value  $\tau_{i,j,k}$



## Outline

- I. Previous work on hair rendering
- II. 3D light-oriented shadow map
- III. Self-shadowing algorithm
- IV. Extensions
- V. Results and conclusion



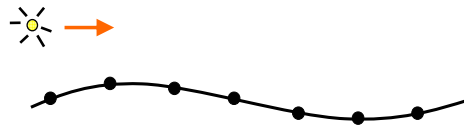
## Steps of the algorithm

- Filling hair density into the map
- Computing the transmittance
- Filtering and drawing



## Filling the hair density

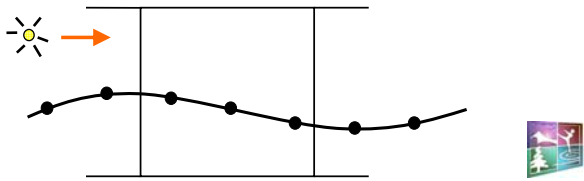
- Sampling each hair strand  $i$   
 $\Rightarrow$  Points  $P_k^i$





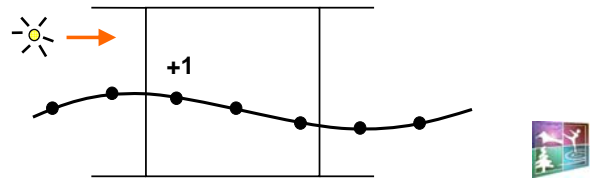
### Filling the hair density

- Sampling each hair strand  $i$   
⇒ Points  $P_k^i$
- Increasing the density of  $\psi(P_k^i)$



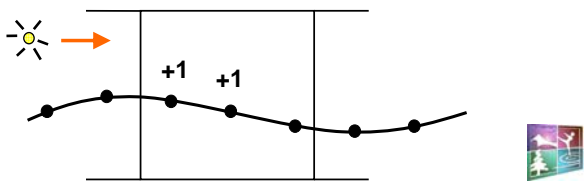
### Filling the hair density

- Sampling each hair strand  $i$   
⇒ Points  $P_k^i$
- Increasing the density of  $\psi(P_k^i)$



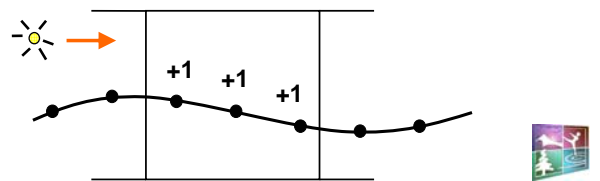
### Filling the hair density

- Sampling each hair strand  $i$   
⇒ Points  $P_k^i$
- Increasing the density of  $\psi(P_k^i)$



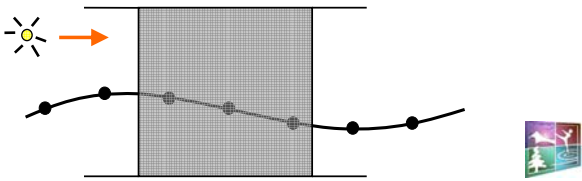
### Filling the hair density

- Sampling each hair strand  $i$   
⇒ Points  $P_k^i$
- Increasing the density of  $\psi(P_k^i)$



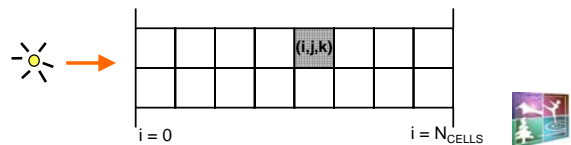
## Filling the hair density

- Sampling each hair strand  $i$   
 $\Rightarrow$  Points  $P_k^i$
- Increasing the density of  $\psi(P_k^i)$   
 $\Rightarrow f \cdot d_{i,j,k}$  is an approximation of the cell's opacity



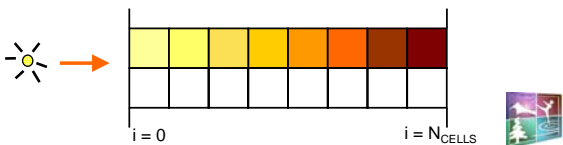
## Computing the transmittance

- Local transmittance of a cell  $i,j,k$ :  
 $\tau_{i,j,k}^{loc} = \exp(-f \cdot d_{i,j,k} \cdot ds)$



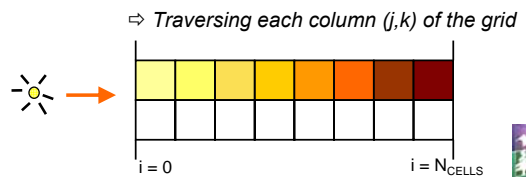
## Computing the transmittance

- Local transmittance of a cell  $i,j,k$ :  
 $\tau_{i,j,k}^{loc} = \exp(-f \cdot d_{i,j,k} \cdot ds)$
- Cumulated transmittance (shadow value):  
 $\tau_{i,j,k} = \prod_{i=0}^i \tau_{i,j,k}^{loc}$



## Computing the transmittance

- Local transmittance of a cell  $i,j,k$ :  
 $\tau_{i,j,k}^{loc} = \exp(-f \cdot d_{i,j,k} \cdot ds)$
- Cumulated transmittance (shadow value):  
 $\tau_{i,j,k} = \prod_{i=0}^i \tau_{i,j,k}^{loc}$



## Filtering and drawing



- Trilinear interpolation of the transmittance values
- Final color of a point P:  

$$\Phi_p = \Phi_{\text{Ambient}} + \tau^{\text{filter}}(\rho) \cdot (\Phi_{\text{Diffuse}} + \Phi_{\text{Specular}})$$
- Each hair strand drawn as an OpenGL polyline



## Outline

- I. Previous work on hair rendering
- II. 3D light-oriented shadow map
- III. Self-shadowing algorithm
- IV. Extensions
- V. Results and conclusion



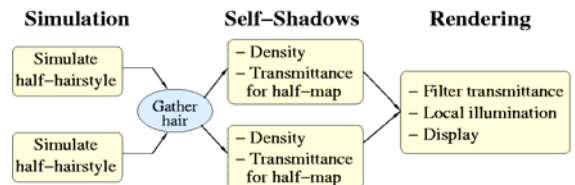
## Extensions (1/2)

- Hair self-collisions: pressure approach



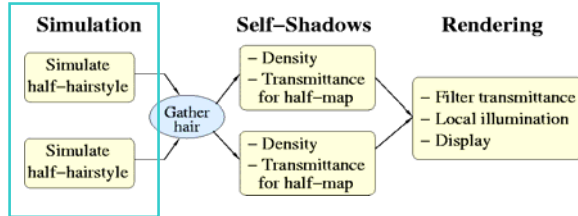
## Extensions (2/2)

- Parallelization of the algorithm



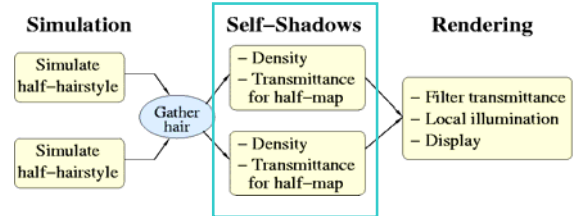
## Extensions (2/2)

- Parallelization of the algorithm



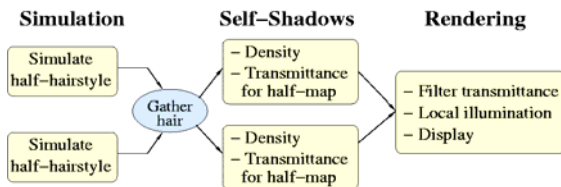
## Extensions (2/2)

- Parallelization of the algorithm



## Extensions (2/2)

- Parallelization of the algorithm



⇒ Easily double performance



## Outline

- I. Previous work on hair rendering
- II. 3D light-oriented shadow map
- III. Self-shadowing algorithm
- IV. Extensions
- V. Results and conclusion



## Results (1/3)

- Easily implemented
- Map resolution: 128 × 128 × 128
- Using a single P4 CPU at 3 GHz

- Rendering performance for 100 K segments

Map reset + density	Transmittance	Filter + draw	Total rendering
0.038 sec	0.015 sec	0.037 sec	0.09 sec

⇒ 11 FPS for rendering 100 K segments  
(instead of 6 FPS by [Mertens04])



## Results (2/3)

### Static Hair

- Captured Hairstyle by [Paris04]
- 1 G segments
- Kajiya's local model

⇒ Rendering time: ~2 sec



## Results (2/3)

### Static Hair

- Captured hairstyle by [Wei05]
- 1 G segments
- Combined with Marschner's model
- Alpha blending

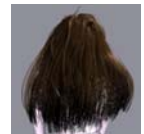
⇒ Rendering time: ~10 sec



## Results (3/3)

### Dynamic Hair

- Animation model:
  - Particles with constrained velocities and positions
  - Explicit Euler integration scheme  
(Animation of guide strands)
- For 100 K segments:
  - 6 FPS for animation + rendering + self-collisions
  - Self-collisions = 2% total time
  - 3.5 FPS when adding body collisions (not optimized)



## "A Practical Self-Shadowing Algorithm for Interactive Hair Animation"

F. Bertails, C. M nier, M-P. Cani

*GRAVIR - IMAG/INRIA  
Grenoble, France*

## Advantages

- Interactive (thousands of hair strands)
- Well-adapted for animation
  - Structures quickly updated
  - Inexpensive handling of hair self-collisions
- Software: no need for advanced GPU
- Well-adapted to parallelization



## Future improvements

- Handling punctual light sources
  - ⇒ Angular partition of space (change mapping function  $\psi$ )
- Using adaptive clustering
  - ⇒ Following [Mertens04]'s approach
- Improving collision handling
  - Improving stability at rest
    - ⇒ Using state-of-the-art fluids methods
  - Avoiding wisps interpenetration
    - ⇒ Using hair orientation



Thank You







## Modeling Hair Influenced by Water and Styling Products

Kelly Ward\* Nico Galoppo+ Ming C. Lin+

{wardk, nico, lin}@cs.unc.edu

\* Walt Disney Feature Animation

+ University of North Carolina at Chapel Hill

<http://gamma.cs.unc.edu/HairWS>



## Motivation

- Hair behavior and appearance influenced by substances on hair
  - Water – washing, swimming, perspiring, etc.
  - Styling products – hairspray, mousse, gel, etc.



Wet



Dry



Styling Products

The UNIVERSITY of NORTH CAROLINA at CHAPEL HILL



## Applications

- Modeling Avatars
- Entertainment
- Education & Training
- *Virtual Hairstyling Tools*
  - Preview hair under different dynamic conditions

Current hair modeling systems primarily focus on capturing basic hair behavior

The UNIVERSITY of NORTH CAROLINA at CHAPEL HILL



## Goals

- Fundamental techniques for capturing basic wet and stylized hair behavior
- On-the-fly adjustments of hair properties
- Easy integration with other hair modeling systems

The UNIVERSITY of NORTH CAROLINA at CHAPEL HILL



## Basic Approaches

- Modeling hair based on a *dual-skeleton* setup
  - Efficient captures intrinsic hair properties
  - Allows adjustment of physical behavior due to water or styling products
- Techniques to account for fundamental changes to hair
  - Physical behavior & properties
  - Geometric structure
  - Rendering

The UNIVERSITY of NORTH CAROLINA at CHAPEL HILL



## Key Features

- **Dynamic Behavior**
  - Change hair stiffness and weight
- **Flexible Geometric Structure**
  - Alter volume of hair on-the-fly
- **Localized Collision Detection**
  - Enable dynamic bonding of hair strands
- **Interactive Rendering**
  - Parameterize lighting and shadow calculations

The UNIVERSITY of NORTH CAROLINA at CHAPEL HILL



## Organization

- Previous Work
- Influence of Water and Styling Products
- Dual-Skeleton System
- Modifications of Physical Properties
- Modifications of Rendering Properties
- Implementation & Results

The UNIVERSITY of NORTH CAROLINA at CHAPEL HILL



## Organization

- **Previous Work**
- **Influence of Water and Styling Products**
- **Dual-Skeleton System**
- **Modifications of Physical Properties**
- **Modifications of Rendering Properties**
- **Implementation & Results**

The UNIVERSITY of NORTH CAROLINA at CHAPEL HILL



## Previous Work: Hair Simulation

- Individual strands as a series of connected line segments  
[Rosenblum et al. 1991; Anjyo et al. 1992]
- *Wisps* or *clusters*  
[Kurihara et al. 1993; Daldegan et al. 1993; Plante et al. 2001; Bertails et al. 2003]
- Fluid dynamics used with individual strand dynamics to handle hair interactions  
[Hadap and Magnenat-Thalmann 2001]

The UNIVERSITY of NORTH CAROLINA at CHAPEL HILL



## Previous Work: Styling Effects

- *Static links* – Used for hairstyle recovery by connecting nearby guide strands  
[Chang et al. 2002]
- Hair gel – Deform initial hairstyle where gel applied  
[Lee et al. 2001]
- Broken-up wet fur – Clumping of fur and increase amount of specular reflection  
[Bruderlin 1999]

The UNIVERSITY of NORTH CAROLINA at CHAPEL HILL



## Previous Work: Hair Rendering

- Anisotropic lighting  
[Kajiya and Kay 1989; Heidrich and Seidel 1998]
- Shadow computations
  - Deep Shadow Maps [Lokovic and Veach 2000]
  - Opacity Shadow Maps [Kim and Neumann 2001]
- Light scattering from hair fibers  
[Marschner et al. 2003]

The UNIVERSITY of NORTH CAROLINA at CHAPEL HILL



## Organization

- Previous Work
- **Influence of Water and Styling Products**
- Dual-Skeleton System
- Modifications of Physical Properties
- Modifications of Rendering Properties
- Implementation & Results

The UNIVERSITY of NORTH CAROLINA at CHAPEL HILL



## Hair and Water – Physical

- Hair absorbs up to 45% of its weight in water
- Stiffness of hair increases by factor of 3
- Wet hair less voluminous than dry hair



- L'Oréal: [www.loreal.com](http://www.loreal.com), 2004.
- D. Johnson. *Hair and Hair Care; Cosmetic Science and Technology Series*, Vol 17. NY Marcel Dekker, Inc., 1997.

The UNIVERSITY of NORTH CAROLINA at CHAPEL HILL



## Hair and Styling Products

- Purpose is to change physical behavior of hair
  - Exact effects vary by product [Johnson 1997]
- Common effects:
  - Stiffen hair motion
  - Hold section of hair in place
  - Strong adhesiveness between strand



The UNIVERSITY of NORTH CAROLINA at CHAPEL HILL



## Organization

- Previous Work
- Influence of Water and Styling Products
- **Dual-Skeleton System**
- Modifications of Physical Properties
- Modifications of Rendering Properties
- Implementation & Results

The UNIVERSITY of NORTH CAROLINA at CHAPEL HILL

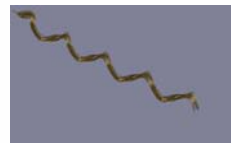


## Hair Motion Properties

- Hair motion subject two types of motion
  - **Global Motion** – Overall positional changes
  - **Local Styling Motion** – Curl elongation under force



Resting



Tight Curl



Loose Curl

The UNIVERSITY of NORTH CAROLINA at CHAPEL HILL



## Single Skeleton Systems

- Stretching and compression of skeleton captures deforming hairstyle
- Require intricate collision detection steps
  - Recalculate bounding volume each time step, or
  - Bounding volume remains inefficient or inaccurate with changing hair



The UNIVERSITY of NORTH CAROLINA at CHAPEL HILL



## Dual-Skeleton

- Use two skeletons
  - **Global-Skeleton** – Controls overall global motion of hair
  - **Local-Skeleton** – Provides position guide and localized collision detection scheme

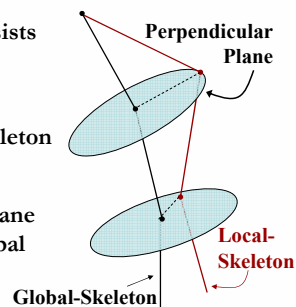


The UNIVERSITY of NORTH CAROLINA at CHAPEL HILL



## Dual-Skeleton Setup

- Global-skeleton consists of  $n$  nodes
- Hairstyle defined by positioning local-skeleton around global
- Local-points lie in plane perpendicular to global section



The UNIVERSITY of NORTH CAROLINA at CHAPEL HILL



## Dual-Skeleton Setup

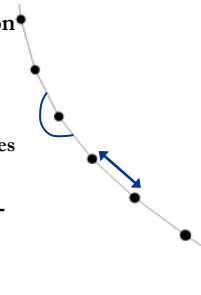
- Rendered hair follows local-skeleton
- Hairs in close proximity grouped together
  - Natural clumping of hair
- Strands modeled as subdivision curves

The UNIVERSITY of NORTH CAROLINA at CHAPEL HILL



## Dynamics

- Global-skeleton controls motion of hair
  - Bending** – Angular spring force
  - Stretching** – Linear spring force
    - Between two consecutive nodes
- All forces are applied to global-skeleton
  - Wind, gravity

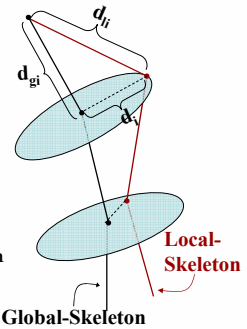


The UNIVERSITY of NORTH CAROLINA at CHAPEL HILL



## Position Update

- Global-skeleton moves from spring forces
- Update local-skeleton
  - $d_{li}$  = Length of local section is fixed
  - $d_{gi}$  = Length of global section
  - Calculate  $d_i$  = distance between global and local points

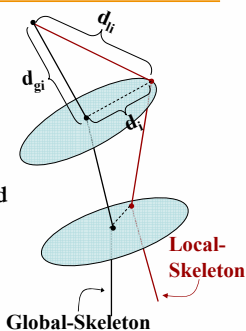


The UNIVERSITY of NORTH CAROLINA at CHAPEL HILL



## Position Update

- As length of global-section increases length of  $d_i$  decreases
  - Curl elongates
- Length of local-skeleton fixed
  - Hair length preserved
  - Global skeleton section not permitted out of reach of local point (length is clamped)

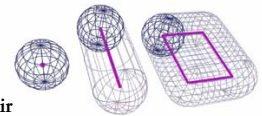


The UNIVERSITY of NORTH CAROLINA at CHAPEL HILL



## Collision Detection

- Swept Sphere Volumes (SSVs)** [Larsen et al. 1999]
  - Bounding volumes to encapsulate hair
  - Shape closely matches hair geometry
- Simple collision detection algorithm:
  - Calculate distance between core shapes (line or rectangle)
  - Subtract appropriate offset (radius of each SSV)



The UNIVERSITY of NORTH CAROLINA at CHAPEL HILL





## Localized Collision Detection

- SSV around each section of *local-skeleton*
  - Skeleton section is core line of *Line Swept Sphere*
  - Provides tighter fit than around global-skeleton section
  - Do not need to re-compute shape SSV at each time step

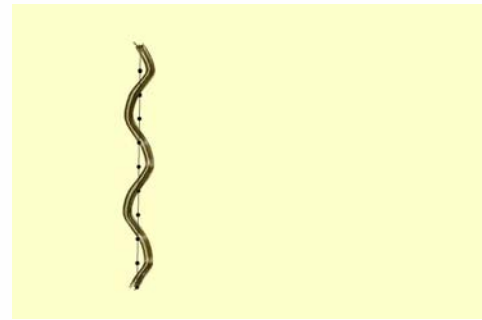


Ours Others

The UNIVERSITY of NORTH CAROLINA at CHAPEL HILL



## Localized Collision Detection



The UNIVERSITY of NORTH CAROLINA at CHAPEL HILL



## Benefits of Dual-Skeleton

- Captures both global and local motion of hair
- Ensures hair length preservation
- Provides accurate and efficient collision detection at all times

The UNIVERSITY of NORTH CAROLINA at CHAPEL HILL



## Organization

- Previous Work
- Influence of Water and Styling Products
- Dual-Skeleton System
- **Modifications of Physical Properties**
- Modifications of Rendering Properties
- Implementation & Results

The UNIVERSITY of NORTH CAROLINA at CHAPEL HILL



## Physical Changes to Hair

- Adjusting dynamics properties
  - Changing mass and spring stiffness of the hairs
- Flexible geometric structure
  - Hair volume changes due to external substances
- Bonds between strands
  - Form and break *dynamically*
  - Captures connection of strands

The UNIVERSITY of NORTH CAROLINA at CHAPEL HILL



## Mass Adjustment

- Forces applied to global nodes using formula:

$$F_i = m_i * a_i$$

$F_i$  (force),  $m_i$  (mass),  $a_i$  (acceleration) of  $i$ th node

- Hair absorbs up to 45% of mass in water

$$m_i = m_{dryi} + (m_{dryi} * 45\%) f_{wetness}$$

$m_{dryi}$  = dry mass of  $i$ th node

$f_{wetness}$  = fraction of water absorbed in hair  
(0% → 100%)

The UNIVERSITY of NORTH CAROLINA at CHAPEL HILL



## Spring Stiffness

- Angular torque controls bending of strands

$$M_{\theta i} = -k_{\theta i}(\theta_i - \theta_{i0})$$

$$M_{\phi i} = -k_{\phi i}(\phi_i - \phi_{i0})$$

Spring constants → Resting angles → Current angles

- Spring force controls elongation of global-skeleton

$$F_{leni} = -k_{leni}(d_{gi} - d_{gi0})$$

Spring constant → Resting length of global section → Current length of global section

The UNIVERSITY of NORTH CAROLINA at CHAPEL HILL



## Spring Stiffness – Adjustments

- Styling products *vary widely*
  - Exact measurements differ
  - Larger length spring constant duplicates styling products
    - Implicit integration
  - Angular spring constant increases slightly



Tight Curl – Higher  $k_{leni}$



Loose Curl – Lower  $k_{leni}$

The UNIVERSITY of NORTH CAROLINA at CHAPEL HILL



## Flexible Geometric Structure

- As hair gets wet it becomes less voluminous
  - Styling products decrease/ increase volume of hair
- Vary thickness of strand group
  - Radii of hair section increase / decrease
  - Update offset of SSV around local-skeleton

The UNIVERSITY of NORTH CAROLINA at CHAPEL HILL



## Radius Contraction

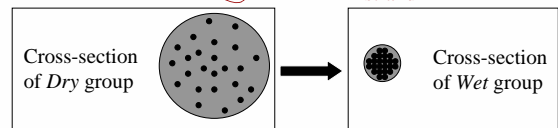
- Hair at 100% wet
  - Smallest contraction is all strands in group directly next to each other

$$AreaOfGroup = (AreaOfStrand) * n$$

Number of strands in group

$$Wet\ radius\ WR_i = \sqrt{n} * r$$

Radius of single strand



The UNIVERSITY of NORTH CAROLINA at CHAPEL HILL



## Radius Contraction

- Extend to variable amounts of wetness
  - Linear relationship based on fraction of water
- Thickness varies down length of group



$$CR_i = DR_i - (DR_i - WR_i) f_{wetness}$$

Current Radius
Dry Radius
Wet Radius

The UNIVERSITY of NORTH CAROLINA at CHAPEL HILL



## Gradually Wet



The UNIVERSITY of NORTH CAROLINA at CHAPEL HILL



## Dynamic Bonds

- Adhesiveness of styling products
  - Strands move in larger groups
  - Increased friction between strands
    - Strands cannot easily pass over each other
- **Dynamic Bonds** form between touching strand groups
  - Formed on-the-fly where product applied
  - Bonds broken under excessive force

The UNIVERSITY of NORTH CAROLINA at CHAPEL HILL



## Dynamic Bonds

- Determine touching hair sections
  - Hair-hair collision detection returns touching hair sections
  - Each hair section maintains list of touching hair sections
- Spring force created between corresponding global-skeleton nodes
  - Strength of force based on strength of styling product

The UNIVERSITY of NORTH CAROLINA at CHAPEL HILL



## Dynamic Bonds



The UNIVERSITY of NORTH CAROLINA at CHAPEL HILL



## Organization

- Previous Work
- Influence of Water and Styling Products
- Dual-Skeleton System
- Modifications of Physical Properties
- **Modifications of Rendering Properties**
- Implementation & Results

The UNIVERSITY of NORTH CAROLINA at CHAPEL HILL



## Rendering Hair

- Anisotropic Lighting
  - Texture map [Heidrich and Seidel 1998]
- Self-Shadow Computation
  - Opacity Shadow Maps [Kim and Neumann 2001]
  - Accumulate opacity  $\alpha$  of strands hit by light rays



The UNIVERSITY of NORTH CAROLINA at CHAPEL HILL



## Hair and Substances – Rendering

- Wet objects [Jensen et al. 1999]
  - Darker
  - Shinier
  - More specular
- Styling products on hair
  - Results vary widely based on product
  - Largely of water and alcohol
  - Rendering similar to wet hair



The UNIVERSITY of NORTH CAROLINA at CHAPEL HILL



## Influence of Water

- Thin film of water surrounds strand
  - Smooths surface of fiber
  - Creates shinier appearance due to specular reflections
- Water absorbed into strand
  - Higher opacity  $\alpha$ : More aggressive self-shadowing
  - Index of refraction increases: *total internal reflection*
    - Less light traverses core of fiber
    - Hair appears darker
    - More colorless light reflected

The UNIVERSITY of NORTH CAROLINA at CHAPEL HILL



## Parameterization of Lighting

- Three main values influenced by water
  - Opacity  $\alpha$
  - Shininess  $s$
  - Anisotropic lighting  $f_a$
- Vary control parameter vector  $\mathbf{V}_p = \{\alpha, s, f_a\}$ 
  - Empirically obtain max and min
  - Linearly interpolate based on wetness in hair,  $f_{\text{wetness}}$

$$\mathbf{V}_p = \mathbf{V}_p^{\text{min}} + f_{\text{wetness}}(\mathbf{V}_p^{\text{max}} - \mathbf{V}_p^{\text{min}})$$

The UNIVERSITY of NORTH CAROLINA at CHAPEL HILL



## Organization

- Previous Work
- Influence of Water and Styling Products
- Dual-Skeleton System
- Modifications of Physical Properties
- Modifications of Rendering Properties
- **Implementation & Results**

The UNIVERSITY of NORTH CAROLINA at CHAPEL HILL



## Modeling Wet Hair

- Adjust mass and spring stiffness
  - Mass increases up to 45%
  - Stiffness increases by factor of 3
- Alter geometric representation
  - Radius contractions
- Parameterized lighting and shadow computations
  - Darker, shinier, more specular

The UNIVERSITY of NORTH CAROLINA at CHAPEL HILL



## Wet Straight Hair



The UNIVERSITY of NORTH CAROLINA at CHAPEL HILL



## Wet Curly Hair



The UNIVERSITY of NORTH CAROLINA at CHAPEL HILL





## Hair with Styling Products

- Adjust mass and spring stiffness
  - Larger increase in length stiffness retains curl
- Alter geometric representation
  - Changes volume, depending on product
- Dynamic bonds between strands
  - Increased adhesiveness and friction between strands
- Parameterize lighting and shadow

The UNIVERSITY of NORTH CAROLINA at CHAPEL HILL



## Hair with Styling Products



The UNIVERSITY of NORTH CAROLINA at CHAPEL HILL



## Performance

- Requires no pre-computations
- Runs at approximately the same rate with or without substances present
  - Simulation: 4.16 seconds/frame
  - Rendering: 0.34 seconds/frame

**Timings:** PC with 1.8 GHz processor, GeForce® 4 graphics card

**Hairstyle:** 16 nodes per skeleton and 9,680 rendered strands

The UNIVERSITY of NORTH CAROLINA at CHAPEL HILL



## Limitations

- Currently, wetness and styling products applied evenly to all hairs
  - Interface to place substance to specific hairs
  - Interactive hairstyling tool
- Approximate lighting equations used
  - Faster rendering algorithm
  - Easily integrated with multi-resolution hair simulation for interactive use

The UNIVERSITY of NORTH CAROLINA at CHAPEL HILL



## Summary

- Presented techniques for modeling hair with water and styling products
  - Dual-skeleton model for hair
    - Captures various hair motions
    - Efficient and accurate collision model
  - Modifications of physical and lighting properties
- Techniques can be used together or separately integrated with various hair modeling schemes

The UNIVERSITY of NORTH CAROLINA at CHAPEL HILL



## Current and Future Work

- Integrate techniques with multi-resolution hair modeling [Ward et al. 2003]
  - Interactive hair simulation and rendering
- Create 3D user interface
  - Direct application of substances
  - Manipulation of hair to create various hairstyles
- Account for plasticity of hair with water

The UNIVERSITY of NORTH CAROLINA at CHAPEL HILL



## Acknowledgements

- This research is supported in part by:
  - Army Research Office
  - Intel Corporation
  - National Science Foundation
  - Office of Naval Research
- For more information:  
<http://gamma.cs.unc.edu/HairWS>

The UNIVERSITY of NORTH CAROLINA at CHAPEL HILL