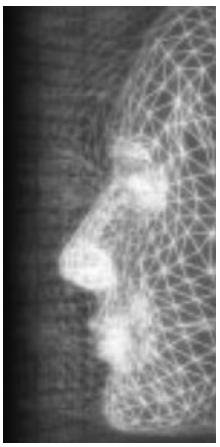
COMPUTER ANIMATION AND VIRTUAL WORLDS
*Comp. Anim. Virtual Worlds* 2010; **21**: 311–320
Published online 25 May 2010 in Wiley InterScience
(www.interscience.wiley.com) DOI: 10.1002/cav.361

# Real-time fur simulation and rendering



*By Jun Lee, DongKyum Kim, HyungSeok Kim*, Carloa Henzel, Jee-In Kim and MinGyu Lim*

***Real-time simulation of fur is a key element in many virtual reality applications. However, it is still difficult to simulate effects of fur animation in real-time. In this paper, we propose an interactive fur simulation method, to calculate effects of external forces, such as wind and gravity, and direct manipulation for real-time interactive animation. The proposed method consists of two layered textures for rendering. The first texture represents volumes of fur, and the second texture covers and laps the edge of the first texture. Our approach unifies these two structures using a shared vertex array to enhance rendering performance. We modify the shared vertex array with external force simulations and render it into the graphics pipeline. The proposed method is applied to virtual fashion design and 3D story telling as an example.***


KEY WORDS: fur simulation; fur rendering; real-time simulation

## Introduction

Various computer graphic technologies, such as texture mapping, lighting, and animation techniques are utilized to visualize tiny mammalian hair. Fur rendering algorithms can deal with bunches of long hair and millions of small hairs altogether.[1–4] These rendering methods can be adopted in various real-time applications. For example, in a virtual try-on system, users can easily have indirect experience of wearing fur clothes before their purchase.[5] This requires real-time simulation and animation of user interactions, such as touch and effects of wind.

Existing fur rendering algorithms mostly focus on rendering static states of fur.[4,6,7] However, it is still a challenging issue to simulate a fur with external forces caused by wind and touch in real-time. Dave *et al.*[2] utilized high quality polygons with mass spring structures for realistic animation of fur to animate the results of physical dynamic simulation. Although the simulation result is promising, it still requires high performance computing power.

In this paper, we propose a real-time simulation method that combines two layers of real-time representation for rendering: volume and lapping textures of edges. They are unified into a shared vertex array. This is simulated for the real-time animation. Each layer is based on the Shell and Fin method.[4] The main contribution of this paper is on the unified structure for simulation. We also propose a systematic modeling method to generate realistic animation based on the proposed structure.

The remainder of this paper is organized as follows. In the next section, we present related work on fur rendering. In Section 3, an overview of the proposed methods is presented. We describe the system architectures and the implementation details in Section 5. We apply the proposed methods to real world scenarios in Section 8. We discuss results and experiments in Section 9. We conclude the paper in Section 10.

## Related Work

### Fur Rendering

Several research streams visualize realistic fur characters in movies in the computer animation industry. They proposed complex equations, polygon-based simulations, and volume texture techniques.[1,2,8]

However, much time and memory is required to represent the fur of virtual characters realistically.

*Correspondence to: H. Kim, Department of Internet & Multimedia Engineering, Konkuk University, Seoul, Korea.
E-mail: hyuskim@konkuk.ac.kr

### Fur Animation

In addition to the visualization of realistic static fur, it is also important to animate fur, as it increases the level of believability in a virtual environment. We can classify previous research into long and short fur animation according to the fur length.

The animation of long fur is similar to hair simulation in humans. Since the method proposed by Ward *et al*.,[9] the rendering method for hair with different styles[3] and simulation method for animation[2] have been proposed. Those methods were applied to the natural fur animation of animals.[8]

Research on short fur animation focused on animating tiny hairs of mammals or plants. Bakay *et al*.[10] proposed a GPU-accelerated fur rendering algorithm to simulate lawn in real-time. Bruderlin[11] devised a scheme to generate the animation of wet and broken-up animal fur.

However, the animation of natural fur simulation usually takes long time. It is still hard to show interactive simulation of fur in real time.

### Shell and Fin Structures[4]

We need to provide an efficient structure to represent complex fur appearance to animate fur in real time. Many methods give optical illusion through lapping several textures to reduce the calculation time and memory space,[4,6–8,12] with respect to treating each strand as polygons. We adopt Shell and Fin structures of Lengyel's work to enable real-time animation of fur.[4]

Shell is a structure to visualize the volume of fur. It consists of several lapped textures with controlled opacity. A fin structure is utilized to fill gaps among lapped textures.

A Fin represents the vertical information that cannot be represented by the Shell structures. Fin not only fills holes in the Shell structures, but also represents additional features to make plentiful fur. Furthermore, Fin provides more precise representation of fur, because its size is small and used for detailed representation.

Yang *et al*. proposed an enhanced algorithm based on Lengyel's research. His algorithm dynamically handles the number of Shell layers, using angles between the viewpoint and the plane of the Shell Structures.[7] However, the Shell layers require a large amount of memory, and it is difficult to edit complex fur in the hierarchical layers. In addition, with the separation of Fin and Shell structure, it is not easy to animate fur with respect to external forces, such as gravity or wind.

## Proposed Approach

### Unified Fin and Shell Structure

Most existing research on real-time fur rendering methods use Shell and Fin architectures.[4,7] However, it is hard to synchronize two different structures in interactive simulation and rendering. It is also difficult to represent natural curvatures of fur on the simulation in separate Fin structures. We propose a unified data structure of both Fin and Shell structures to solve these problems.

By the unification, it is possible to reduce the number of vertices using shared vertex information and to represent a more natural simulation of Fin, because Fins can utilize the Shell data in a shared architecture (as shown in Figure 1A).

Previous research used two Fin faces with static textures.[4] The result only shows static states of Fin, which looks like a line, as shown in Figure 1B. Our proposed system provides more natural representations of curvatures than the previous approach (as shown in Figure 1C).

As the complexity of 3D fur models increases, so does the number of polygons and size of textures for fur. Therefore, we reduce the quality of images to obtain reasonable performance for rendering fur of various 3D models in real-time. This section illustrates the modeling and rendering mechanism for this unified structure.

## Fur Modeling

### Generation of Unified Fin and Shell mesh

The proposed system creates the mesh data of Shell and Fin based on a base mesh. We use the following algorithm to generate the shared vertices of fur with Shell and Fin structure.

In the algorithm, NV is an array of shared vertices, in the unified structure. It stores positions of all vertices in Shell layers and Fins. A vertex $NV_{ij}$ is located in its defined distance $D_{ij}$ from the base mesh and direction vector $BN_j$ from $BV_j$, a vertex on the base mesh. We also can create several layers of shared vertices, as shown in Figure 2A.

*Step 1*: Set a target Shell layer $S_i$, a target layer to create the shared vertex.

*Step 2*: Get a current base mesh vertex $BV_j$ and normal $BN_j$.



DOI: 10.1002/cav

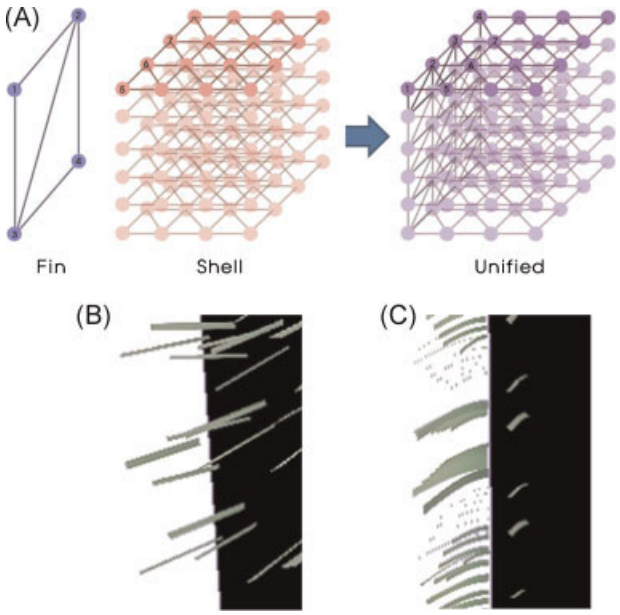



*Figure 1. (A) Fin and Shell unification. (B) Simulation of Fin with two faces. (C) Simulation of Fin with unified structure (unified with 32 shells).*

*Step 3*: Get a Shell layer distance $D_{ij}$ between the base mesh vertex $BV_j$ and the target Shell layer $S_i$.

*Step 4*: Create new shared vertices $NV_{ij}$, located in the new defined position, heading towards $BN_j$ with $D_{ij}$ distance.

*Step 5*: Iterate the above steps to create new shared vertices $NV_{ij}$ from all base meshes (1 to *j*).

*Step 6*: Iterate the above steps to create new shared vertices $NV_{ij}$ among all Shell layers (1 to *i*).

After the generation of the shared vertices, the proposed system creates faces of Fins and Shells that are connected through the shared vertices (as shown in Figure 2A). In the process of creating Shell faces, the system recreates the same number of faces in the base mesh (as shown in Figure 2B-2). Faces of Fins are created by connecting neighbor vertices as shown in Figure 2B-3. All of faces are created after the iteration of these operations in all vertices (as shown in Figure 2B-4).

## Shell Creation

After the creation of Shell and Fin faces, a texture is generated to represent the shape of furs on Shells. The proposed method creates random Shell textures using a seed value.

The seed value is initialized by a threshold and random values. We utilize the seed value as a growing

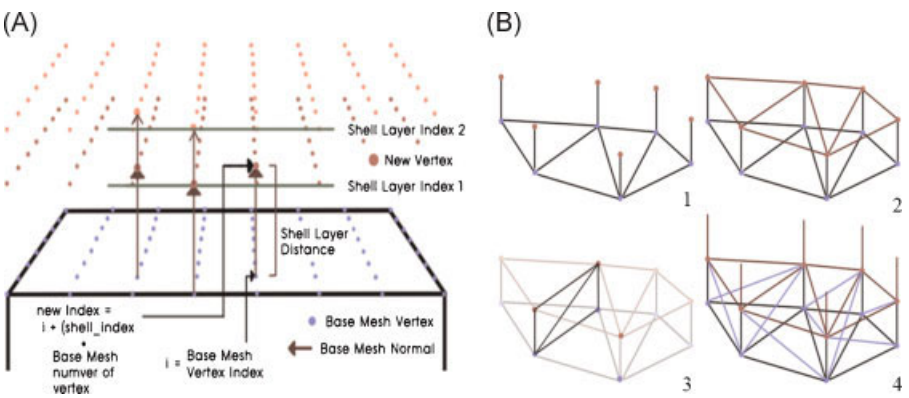



*Figure 2. (A) A shared vertex of a Fin and a Shell, (B) A list of Fins and Shell faces.*



DOI: 10.1002/cav

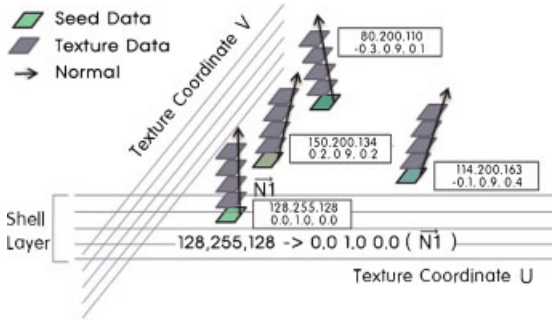



*Figure 3. Creation of random Shell textures using seed RGB values.*

vector to represent the random direction of each fur strand. Figure 3 shows that seed values determine the directions of growing vectors.

The seed value is converted to vector positions to create a texture of Shell layers. We arrange the Shell layers to represent three-dimensional positions of fur information. Using this process, it is possible to generate random fur shapes on Shell textures.

In addition, the proposed system can be customized, as it allows a user to manipulate different fur patterns during the creation process.

## Modelling Fur Patterns

The proposed system supports various configurations of fur patterns to realize natural representation of different fur styles.

We can create random directions of fur strands by changing positions of Shell and Fin vertices. That is, by modifying the normal vector of the base mesh ($N_j$ in the algorithm) we can have all normal vectors (as shown in Figure 4A–C, we can get different positions of textures and various directions of fur strands by changing directions).

If an artist requests the use of a special pattern, we can configure a desired texture pattern before the creation of a Shell texture (as shown in Figure 5). The given input texture is applied to generate the seed value for a random vector for texture generation.

If a fur strand has several irregularities of colors (e.g., different colors between the start and end of a strand), a color height map is used for natural fur representation (as shown in Figure 6).

Another feature of the proposed system is that it can provide various styles of fur strands, as shown in Figure D–F. We can also get different fur thicknesses and densities by resizing the input texture.

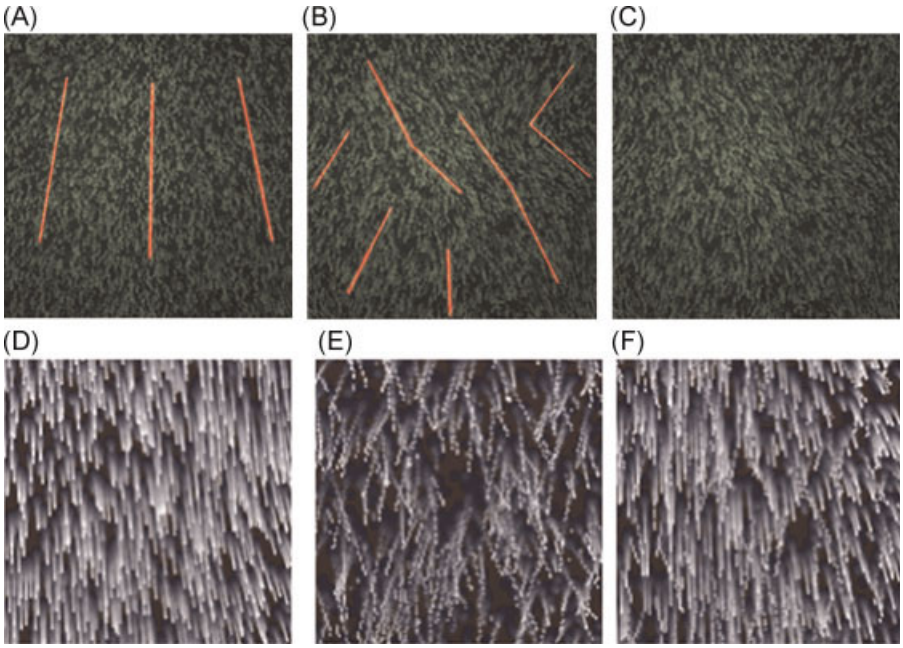



*Figure 4. (A) Before randomization of normal vectors, (B) after randomization of normal vectors, (C) regenerated result, (D) all straight, (E) all random, (F) result of mixed values.*



DOI: 10.1002/cav

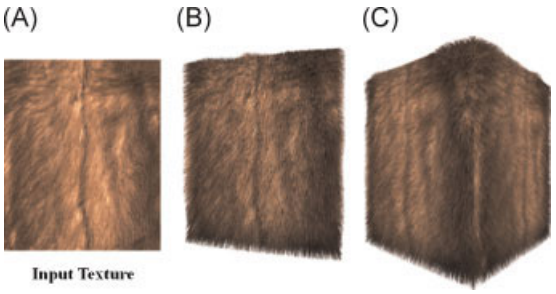



*Figure 5. (A) Input texture image, (B) result with a plane, (C) generated result with a 3D model.*

## Simulation

Figure 7 illustrates the overall procedure of the proposed fur rendering system. In the pre-processing stage, the system creates a unified architecture of both Shell and Fin textures from 3D models and texture resources. After the pre-processing step, the unified data is simulated with external forces, such as gravity and natural wind, and the simulation result is passed to the animation stage. The system animates the simulated data using linear interpolation.

The animation requires two steps. First, the system updates the positions of vertices based on the animation data. Second, the system renders the updated vertices. The rendering performance could be accelerated by algorithms using GPU.

We perform physical simulations with the interpolation of relevant data for fur animation. We consider three possible force sources for the physical simulations: gravity, wind, and touch.

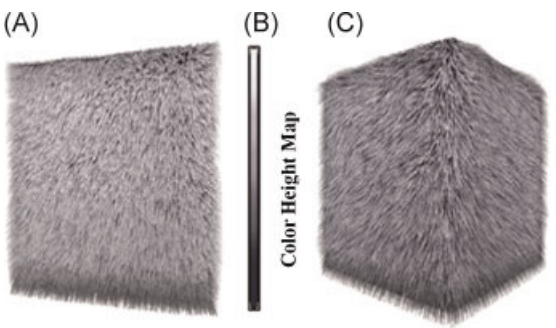



*Figure 6. (A) Original fur, (B) color height map with 32 colors, (C) regenerated fur with color height map.*

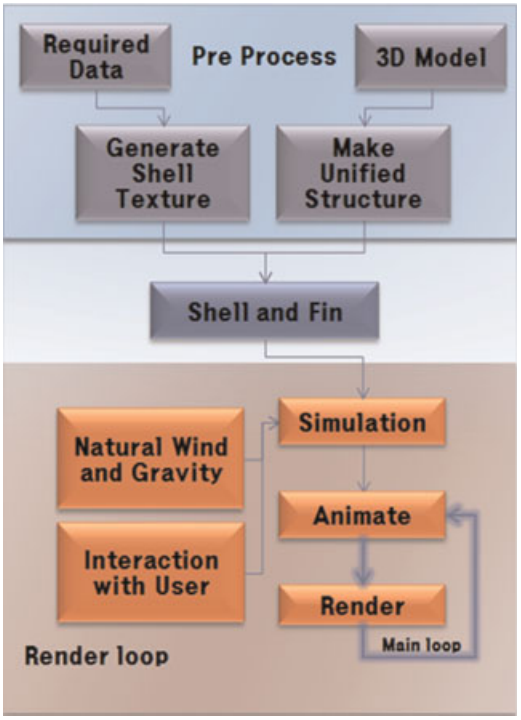



*Figure 7. System components.*

## Internal Force Simulation

The internal force simulation, such as the effect of gravity, on fur enhances the level of believability in a virtual environment. While fur strands initially head towards their original direction without gravity, gravity should pull them down to the ground, as if they are in the real world.

We can simulate the gravity effect by applying a simple Equation (1) to every shell layer vertex. RN is the number of unvisited vertices in a strand and changes to 1 from Figure 8B to C. It can be seen as a weighting factor of curving with the gravity vector V2. *V* is the growing vector, and *stability* is a variable that relates to the inertia of the original shape (as shown in Figure 8). When the rotation angle of the current node is calculated, as shown in Figure 8B, the proposed system calculates each rotation angle of the remaining nodes using these variables. After the calculation, the positions of the remaining nodes are changed with the rotation angle, as shown in Figure 8C, where the rotation axis is given as the cross product between *V* and V2. The Equation is applied first to the Shell layer near to the base mesh (as shown in Figure 8B) and then to the vertices in the



DOI: 10.1002/cav

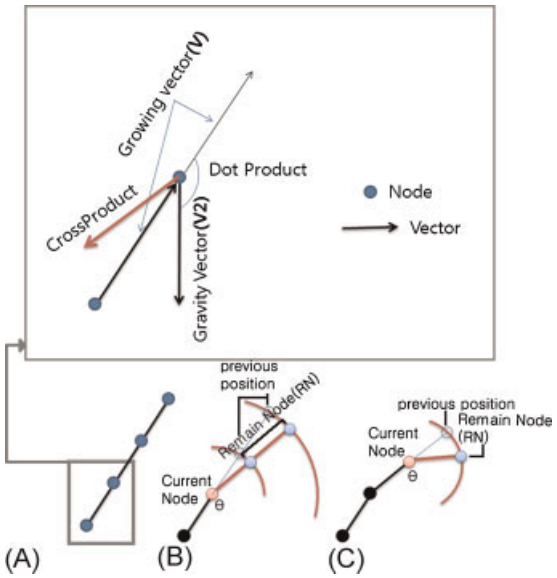



*Figure 8. Rotation angles to simulate the gravity effect.*

distant Shell layers.

$$\theta = a\cos(\mathrm{dot}(V, \mathrm{V2})) \times \mathrm{RN}/\mathrm{stability}$$

This procedure continues to other vertices in the Shell layer following the previous vertex (as shown in Figure 8C).

## External Force Simulation

We create a force field, which stores a force per vertex in a base mesh, to enable real-time simulation of various external forces. A force vector of each vertex in the force field affects and changes the positions of neighboring vertices of the Shell and Fin.

Each vertex simulates a rotation angle and a rotation axis using the inner and cross products between a force vector and a growing vector.

We allocate the initial wind force vector at each point of the force field to simulate the wind. We can also add a small random value to the original vector to visualize shaking fur.

The proposed system allows a user to pat the fur by controlling vector values at some parts of the force field. When a user touches part of the fur, the system calculates the 3D positions of the input. The method calculates an interaction vector using these two positions, and changes the vectors of the selected area in the force field. When a user takes his/her hand away from fur, the original values of the force field are restored.

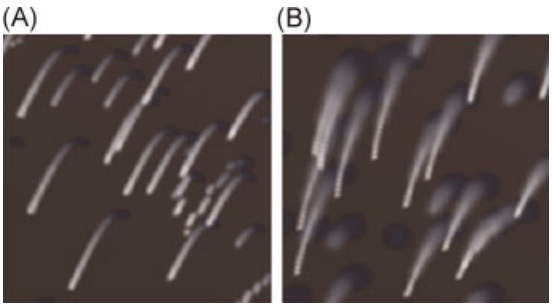



*Figure 9. (A) Before randomization of normal vectors, (B) after randomization of normal vectors.*

The positions of vertices in successive frames are computed by linear interpolation of simulated results instead of simulating it for each iteration to reduce the calculation cost.

## Applications

In this section, we explain the implementation of the proposed approach from the fur modeling to the real-time animation procedure in exemplar applications.

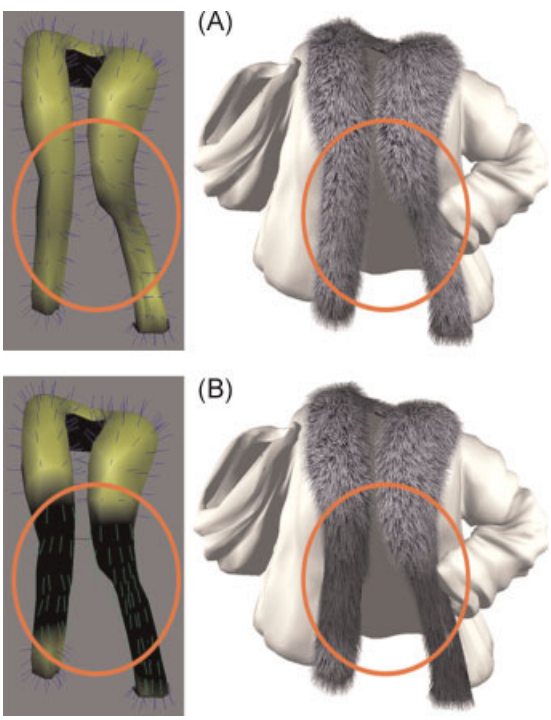



*Figure 10. (A) Before modifying alpha blur, (B) after modifying alpha blur.*



DOI: 10.1002/cav

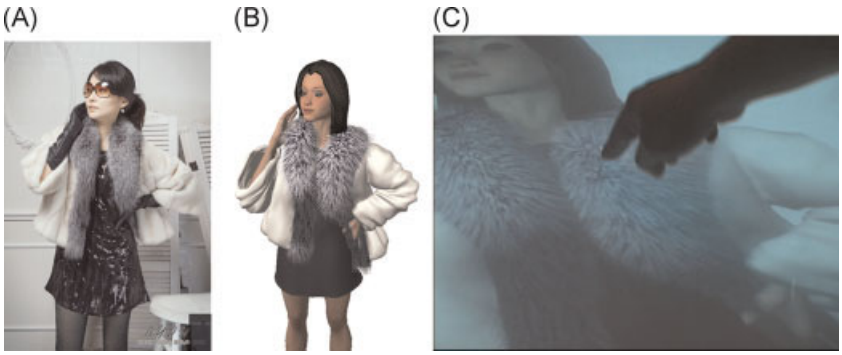



*Figure 11. (A) A real and (B) a virtual woman model with fur. (C) An interactive animation on a tabletop interface.*

| | Vertices | Faces | Eclqes | Texture size of Shell |
|---|---|---|---|---|
| Box | 296 | 588 | 882 | 1024 × 1024 |
| Bunny | 2520 | 5036 | 7554 | 1024 × 1024 |
| Boar | 2976 | 5688 | 5271 | 1024 × 1024 |
| Shawl | 261 | 420 | 680 | 512 × 512 |
| Women (with Shawl) | 18 324 | 35 639 | 680 | 512 × 512 |

**Table I. 3D models used in experiments.**

## Fur Modeling

First, we separate target meshes to add fur from an entire model. During the modeling process, we need to modify the texture map or the normal vector. The next step is to determine the direction of hair.

We can use two methods to decide the hair direction. First, we can give random vectors for the Shell texture data, as described in the previous section. Second, we can use a normal vector of the base mesh to change the direction of original fur strands. The directions of fur strands can be enhanced further by artists or a randomized calculation. Figure 9 shows the randomized case followed by customizing some normal vectors.

Then, we control the thickness of a fur strand by changing the size of the Shell texture or modifying the texture image to a thinner size. After this operation, the fur strands become thinner (as shown in Figure 10A). We applied $\alpha$ values to the root parts of the Shell layer texture using a blur method, which can present different thicknesses between the root and the fur end (as shown in Figure 10B).

There are three different options to color fur strands; a texture for the entire surface, a single color for every strand, and a color height map.

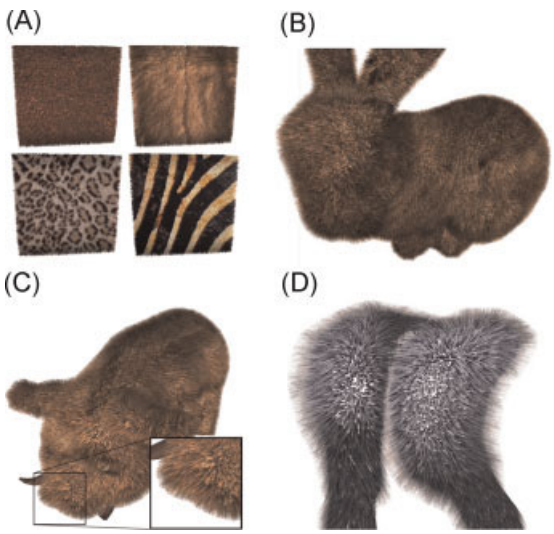



*Figure 12. (A) Rendering result of 3D box models, (B) a bunny model, (C) a boar model, (D) a shawl model.*



DOI: 10.1002/cav

| Model | Box | Bunny | Boar | Shawl | Women |
|---|---|---|---|---|---|
| Shared vertices (proposed, 30 faces for a fin) | 5032 | 42 840 | 31 632 | 4176 | 22 500 |
| Shell and Fin vertices (separated, two faces for a fin)[4] | 6796 | 57 948 | 42 174 | 5536 | 23 860 |
| Shell and Fin vertices (separated, 30 faces for a fin)[4] | 31 492 | 275 460 | 189 762 | 24 576 | 42 900 |
| Number of faces (proposed, 30 faces for a fin) | 9996 | 85 612 | 61 304 | 6720 | 42 359 |
| Number of faces (separated, 30 faces for a fin)[4] | 36 456 | 312 232 | 219 434 | 27 120 | 62 759 |
| FPS | 109.93 | 14.4 | 20.42 | 149.72 | 64.87 |

**Table 2. Performance evaluation in the numbers of vertices and faces and average FPS values of various 3D models using the proposed approach and the separated Fin and Shell structure[4]**

## Applications of Real-time Fur Simulations

Digital fashion is a promising application of the proposed system. For example, in a design process, it is possible to check previews of fur cloths prior to production by simulating 3D modeling and real-time animation with different external forces (as shown in Figure 11A and B).

We can utilize these digital contents in a marketing area using a tabletop interface[13,14] or a wall display.[15] As shown in Figure 11C, a user can have various experiences with natural interactions, such as touching, a wind simulation, and stroking down the fur. It may give an opportunity for customers to virtually try on a cloth before buying a real fur cloth.

## Results and Experiments

We conducted fur rendering tests with several 3D models to test the performance of the proposed system, as described in Table 1. Figure 12 shows various applications using the proposed method. In these examples, the textures are sampled in 16 layers.

The experiment was conducted on a desktop PC with Core2 Quad CPU and nVIDIA GeForce 8800GTX. We compared the number of vertices of the proposed approach with Lengyel's research.[4] Table 2 shows that our proposed system reduces about 25% of the extra costs of the redundancy of memory space using the shared vertices architecture. If we compare this to straightforward dynamic fur simulation using 30 faces of Fin, our proposed system can reduce around 85% of the extra memory space.

## Conclusion

In this paper, we proposed an algorithm to enable real-time fur rendering using a unified architecture that consists of Shells and Fins. Users can manipulate and interact with fur in various simulations, such as through applying wind and through touching it. The results of these user interactions are applied to real-time animation. The proposed system can be used in various application areas, such as representation of 3D characters, designing fashion content, and marketing.

In future work, we will try to overcome and enhance current limitations of the proposed system. We plan to investigate a level of detailed techniques to further reduce the amount of data to be processed. The number of Fin faces determines the rendering performance for fur rendering; we can reduce the number of Fin faces, if a virtual character is located distant from the viewpoint. We can also reduce a memory space of faces using tessellation techniques on GPU. In the initial result, we could achieve around five times the speed. Using this speedy simulation, we will also develop more external simulations, such as wet simulation and grasping fur.

We also plan to develop an authoring tool for the proposed system. This will enable fashion designers to use the CAD system to automatically create fur textures.

#### ACKNOWLEDGEMENTS

This work was supported by the Seoul R&BD Program (10581) and supported by the Industrial Source Technology Development Programs funded by the Ministry of Knowledge Economy (B0008583).



DOI: 10.1002/cav

## References

1. Siegel L, Kenkins S. Composite based refraction for fur and other complex objects on Bolt. *ACM SIGGRAPH 09 Talks*, 2009; 1–1.
2. Dave J, Hiebert B, Kim TY, Neulander H, Rijpkema H, Telford W. The Chronivles of Narnia: the lion, the crowds and rhythm and hues. *Proceedings of ACM SIGGRAPH 06, Course 11*, 2006.
3. Choe B, Choi MG, Ko HS. Simulating complex hair with robust collision handling. *Proceedings of the 2005 ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, 2005; 153–160.
4. Lengyel J, Praun E, Finkelstein A, Hoppe H. Real-time fur over arbitrary surfaces. *ACM Symposium on Interactive 3D Graphics 2001*, 2001; 227–232.
5. Volino P, Cordier F, Magnenat-Thalmann N. From early virtual garment simulation to interactive fashion design. *Computer-Aided Design 2005*; **37**(6): 593–608.
6. Kajiya J, Kay T. Rendering fur with three dimensional textures. *Proceedings of ACMSIGGRAPH 89*, 1989; 271–280.
7. Yang G, Sun H, Wang W, Wu E. Interactive fur modeling based on hierarchical texture layers. *Proceedings of ACM VRCIA 06*, 2006; 343–346.
8. Goldman D. Fake fur rendering. *Proceedings of ACM SIGGRAPH 97*, 1997; 127–134.
9. Ward K, Lin MC, Lee J, Fisher S, Macri D. Modeling hair using level-of-detail representations. *Proceedings of Computer Animation and Social Agents 03*, 2003; 41.
10. Bakay B, Lalonde P, Hedirich W. Real time animated grass. *Proceedings of Eurographics 02 (short paper)*, 2002.
11. Bruderlin A. A method to generate wet and broken-up animal fur. *Proceedings of Pacific Conference on Computer Graphics and Applications*, 1999; 242.
12. Praun E, Finkelstein A, Hoppe H. Lapped textures. *Proceedings of ACM SIGGRAPH 00*, 2000; 465–470. Microsoft Direct X 9.0 SDK, sample. Fur.
13. Microsoft Surface. http://www.microsof t.com/surface/
14. Lee JH, Lee J, Kim H, Kim JI. Believable interaction with a quasi-tangible tabletop interface. *Computer Animation and Virtual Worlds 2007*; **18**: 121–132.
15. Matsushita N, Rekimoto J. HoloWall: designing a Finger, hand, body, and object sensitive wall. *Proceedings of the ACM Symposium on User interface Software and Technology'97*, 1997; 209–210.

*Authors' biographies:*

**Jun Lee** received BS and MS degree in Computer Engineering from Konkuk University, Korea, 2004 and 2006. He is a PhD student in advanced technology fusion of Konkuk University. He currently researches a collaborative virtual reality, real-time rendering using a GPU architecture.

**DongKyum Kim** received his BS degree in Internet & Multimedia Engineering from Konkuk University, Korea, 2009. After graduation, He is currently working in a game company called AeonSoft of GALA Group.

**HyungSeok Kim** is an Associate Professor at the Department of Internet & Multimedia Engineering, Konkuk University, Korea from the year 2006. Before joining Konkuk University, he was a senior researcher at MIRALab, University of Geneva, being involved in research activities on virtual reality. He received his PhD in Computer Science in February 2003 at VRLab, KAIST. His major research field is real-time interaction in virtual environments including multi-resolution modeling of shape and texture and multi-resolution interaction mechanisms. His current research activities are focused on topics of shaped modeling for real-time rendering and evoking believable experiences in the virtual environment.

**Carola Henzel** received her BS degree in Media Computer Science at Fachhochschule Wiesbaden, University of Applied Sciences in 2008 after writing her Thesis at the Fraunhofer Institute for Computer Graphics Research (IGD). She currently is a MS student of Computer Science at Hochschule RheinMain, University of Applied Sciences in Germany and spent one exchange semester in VRlab at Konkuk University, Korea in 2009/2010.



DOI: 10.1002/cav

**Jee-In Kim** is a Professor of Internet and Multimedia at Konkuk University. He is Director of Digital Contents Research Center supported by Seoul Metropolitan Government. He received a BS degree in Computer Engineering from Seoul National University in 1982 and an MS degree in Computer Science from Korea Advanced Institute of Science and Technology in 1984. He received his PhD in Computer Science & Information in May 1993 from University of Pennsylvania. His major research fields include ubiquitous computing, virtual reality and HCI.

**Mingyu Lim** is an Assistant Professor at the Department of Internet & Multimedia Engineering, Konkuk University, Korea from the year 2009. Before joining Konkuk University, he was a senior researcher at MIRALab, University of Geneva, being involved in various research activities on networked virtual environments and ubiquitous computing systems. He received his PhD in Computer Science in February 2006 at ICU, Korea. His major research field is supporting scalability in networked virtual environments. His current research activities are focused on efficient communication middleware, event transmissions, and content distribution in networked ubiquitous computing systems.



DOI: 10.1002/cav