Visual Comput (2008) 24: 577–585
DOI 10.1007/s00371-008-0238-8

ORIGINAL ARTICLE

**Rıfat Aras**
**Barkın Başarankut**
**Tolga Çapın**
**Bülent Özgüç**

# 3D Hair sketching for real-time dynamic & key frame animations

Published online: 5 June 2008


**Electronic supplementary material**
The online version of this article (doi:10.1007/s00371-008-0238-8) contains supplementary material, which is available to authorized users.

R. Aras (✉) · B. Başarankut · T. Çapın · B. Özgüç
Department of Computer Engineering, Bilkent University, Ankara, Turkey
{arifat, barkin, tcapin, ozguc}@cs.bilkent.edu.tr

**Abstract** Physically based simulation of human hair is a well-studied and well-known problem. But the "pure" physically based representation of hair (and other animation elements) is not the only concern of the animators, who want to "control" the creation and animation phases of the content. This paper describes a sketch-based tool, with which a user can both create hair models with different styling parameters and produce animations of these created hair models using physically and key frame-based techniques. The model creation and animation production tasks are all performed with direct manipulation techniques in real-time.

**Keywords** Sketching · Direct manipulation · Key frame · Hair animation

## 1 Introduction

As one of the hardest parts of the overall character animation, realistic hair is also one of the most important elements for producing convincing virtual human/animal agents. Physically based simulation of hair is a well-studied and well-known subject, but for animators and artists that create computer animation content, physical reality is not the only concern. They have to be provided with intuitive interfaces to create such content. Therefore, direct manipulation techniques for user interfaces, which are emerging as a major technique for user interaction, can be used as a means of 3D content creation.

In this paper, we propose such a sketch-based tool, with which an artist can create hair models including the stylistic properties of hair intuitively with direct manipulation. With the proposed tool, it is also possible to create physical and key frame animations in a short time with minimum user interference. The key frame and physically based animations are realized effectively by using GPU programming, thus enabling the created animations to be controlled interactively. Another property to be mentioned is that the created hair content is subject to no extra mapping process or database lookups (except the gesture recognition phase to solve ill-defined problems). With this property, it is ensured that the created hair model looks and behaves as closely as possible to the sketched one.

## 2 Previous work

Different hair modeling techniques have been proposed to serve for different purposes. Individual particle-based methods [1, 5, 11], real-time animation solutions [8, 10, 16], representing detailed interactions of the hair with each other [12] and interactive hairstyling systems [2, 7] have all addressed different parts of the hair modeling and animation problem. Our proposed tool deals with three different aspects of the problem: (1) modeling the hair along with its stylistic properties, (2) creating and controlling the animation of the hair model, and (3) performing these tasks with a direct manipulation interface. Therefore, it would be appropriate to examine the previous work relevant to our method, with respect to these different aspects.

*Hair modeling and animation.* Choe et al. [2] present a wisp-based technique that produces static hairstyles by employing wisp parameters such as length distribution, deviation radius function and strand-shape fuzziness value. On top of this statistical model, a constraint-based styler is used to model artificial features such as hairpins. Although the generated styles are realistic, real-time operation is unavailable due to excessive calculations. Oshita presents a physically based dynamic wisp model [10] that supports hair dynamics in a coarse model and then extends it to a fine model. In the dynamic wisp model, the shape of a wisp and the shapes of the individual strands are geometrically controlled based on the velocity of the particles in the coarse model. The model is designed to work on GPUs with operations performed in real-time.

*Controlling animations.* Physically based modeling of hair and other natural phenomena creates very realistic animations, but controlling these animations to match designers' unique needs has recently become an important topic. In Shi and Yu's work [14], liquids are controlled to match rapidly changing target shapes that represent regular non-fluid objects. Two different external force fields are applied for controlling the liquid: feedback force field and gradient field of a potential function that is defined by the shape and skeleton of the target object. Like water, controlled smoke animations have also been studied. Fattal and Lischinski [3] drive the smoke towards a given sequence of target smoke states. This control is achieved by two extra terms added to the standard flow equations that are (1) a driving force term used to carry smoke towards a target and (2) a smoke gathering term that prevents the smoke from diffusing too much. Treuille et al. [15] use a continuous quasi-Newton optimization to solve for wind-forces to be applied to the underlying velocity field throughout the simulation to match the user-defined key frames. Physically based hair animation has also been a subject of animation control. In Petrovic's work [11], hair is represented as a volume of particles. To control hair, this method employs a simulation force based on volumetric hair density difference between current and target hair shapes, which directs a group of connected hair particles towards a desired shape.

*Sketch-based interaction.* Creating 3D content in an intuitive way has become an active research area recently. Sketch-based techniques have gained popularity to achieve this task. A number of researchers have proposed sketch-based techniques for creating hair animations. Wither et al. [17] present a sketching interface for physically based hair styling. This approach consists of extracting geometric and elastic parameters of individual hair strands. The 3D, physically based strands are inferred from 2D sketches by first cutting 2D strokes into half helical segments and then fitting these half helices to segments. After sketching a certain number of guide strands, a volume stroke is drawn to set the hair volume and adapt the hair cut. Finally, other strands are interpolated from the guide strands and the volume stroke. Because of the mentioned fitting process, it is not possible to obtain a resultant physically based strand that matches the user's input stroke. Another physically based hair creation technique is proposed by Hernandez et al. [6]. In this work, the painting interface can only create density and length maps, therefore hairstyle parameters such as curliness and fuzziness cannot be created easily with this technique.

In contrast to these physically based sketching tools, Malik [9] describes a tablet-based hair sketching user interface to create non-physically based hairstyles. In this approach, hair is represented as wisps, and parameters such as density, twist, and frizziness of a wisp are used to define the style of the hair. Additionally, with the help of gesture recognition algorithms and virtual tools such as virtual comb or hair-pin, the style of drawn hair can be changed interactively. Another non-physically based hairstyle design system is proposed by Fu et al. [4]. Their design system is equipped with a sketching interface and a fast vector field solver. The user-drawn strokes are used to depict the global shape of the hairstyle. The sketching system employs the following style primitive types: stream curve, dividing curve and ponytail. In this system, it is hard to provide local control over hairstyle without losing real-time property.

### 2.1 Our contribution

In this paper, we propose an easy-to-use sketch-based system for creation and animation of hair models, using both physically based and key frame-based animation techniques. In contrast to the previous work, our system is capable of completely preserving the drawn properties of the hair without any intermediate mapping process. As a result, all types of hair drawings (e.g., curly, wavy hair) can be represented. Dynamic and key frame animations of hair can also be created and edited in real-time with a sketching interface. Statistical wisp parameters that have been previously employed in a static context [2] (such as fuzziness and closeness) are employed in a dynamic wisp model. Physically based constraints are used in conjunction with key frame animations to create hybrid animations. Finally, a wide range of hair animation effects, such as growing hair, hairstyle changes, etc., are supported by the key framing interface via the proposed wisp matching and hair mass sampling techniques.

## 3 Hairstyle modeling

In our tool, hair is represented as a group of wisps, and styling of the hair is achieved by manipulating wisp parameters such as fuzziness and closeness [2]. Hand-drawn 2D strokes are used as a means of input. The process

of converting 2D input values into 3D styling parameters consists of locating, recording and forming steps. The details of the steps of the process are explained in the following subsections. The flow diagram of the process is given in Fig. 1.

### 3.1 Skeleton strand root positioning

First, we define *skeleton strands* as the master strands in a wisp, which are responsible for the style and movement of other strands located on that wisp. The goal of the first step of the hair sketching tool is then locating the root point of the skeleton strand. To achieve this goal, we fit a Catmull–Rom patch on the surface of the head model [7]. The Catmull–Rom patch structure is used to hold location information on the head. The patch representation allows us to decrease the dimension of the location problem from 3D to 2D space. The underlying patch structure also makes it easy to find the neighborhood information within a wisp, which is used to distribute the imitator strands around the skeleton strand. When the tool is in idle state (i.e., a wisp is not being sketched), the user's first input point is considered as a root candidate. If the point is on the patch, the point is registered as a skeleton strand root. The 3D coordinates of the root point are converted to the $u$–$v$ coordinate system of the patch, in order to be used at a later stage, during the wisp formation phase.

### 3.2 Skeleton strand control point position recording

After the root of the skeleton strand is determined, other control points of the strand are recorded. Because users can only interact with the 2D display, these 2D points have to be converted to 3D coordinates. This is accomplished by employing OpenGL's depth buffers and invisible planar elements [13].

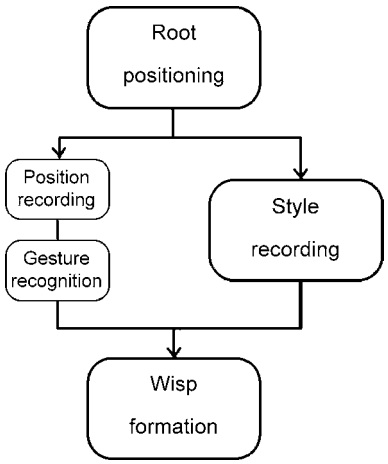


**Fig. 1.** Flow diagram of the hairstyle capture process

### 3.3 Style recording

The style of an individual wisp is represented by a number of style parameters, such as closeness $c_{i,j}$, fuzziness $f_{i,j}$, number of strands in a wisp $n_i$, and strand thickness distribution $t_{i,j}$ (where $i$ is the id of a particular wisp and $j$ is the number of recorded control points on that wisp).

Although these parameters can be recorded by different means of input, in our tool, a tablet stylus pen is used for their recording with a direct manipulation interface. For example, the pressure information from the stylus pen is mapped to the closeness parameter $c_{i,j}$ (as the applied pressure increases, the closeness value increases also), and the tilt angle information is used to represent the fuzziness parameter. These parameters, except the number of strands and the thickness distribution, are recorded for each control point of the skeleton strand, so that it is possible to vary them within a single wisp as can be seen in Fig. 2.

### 3.4 Gesture recognition

The gesture recognition step operates on the drawn hair wisps and detects if there are any loops. These loops define curling hairstyles, and are used to create 3D curling hair. Gesture recognition represents hair strands as a list of segments: a hair strand is formed by $n$ mass nodes, forming $n-1$ segments. These segments may intersect with any other segment drawn on the same wisp. By calculating the respective positions of each segment on the 2D viewport, we detect these possible intersections as follows.

The intersections that have a potential to form curly hair are selected if they satisfy the following two constraints:

1. A hair segment might be intersected by more than one segment. If such a condition occurs, the segment that is nearest is chosen and the others are discarded.
2. After the intersecting segment is chosen, if the segments between the chosen pair do not produce a convex polygon, this means that the loop does not represent a curling hair and shall be discarded.

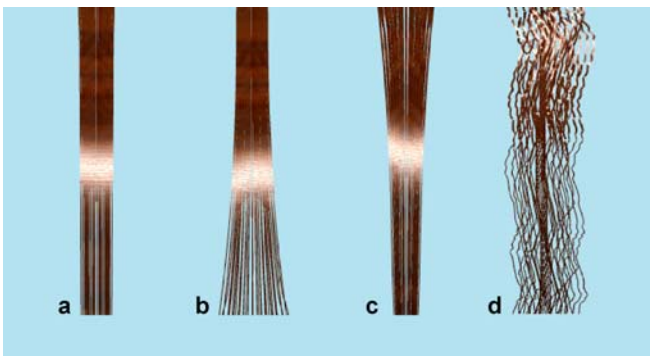


**Fig. 2a–d.** The effect of wisp parameters. **a** Constant closeness value. **b** Increasing closeness value. **c** Decreasing closeness value. **d** The effect of fuzziness value

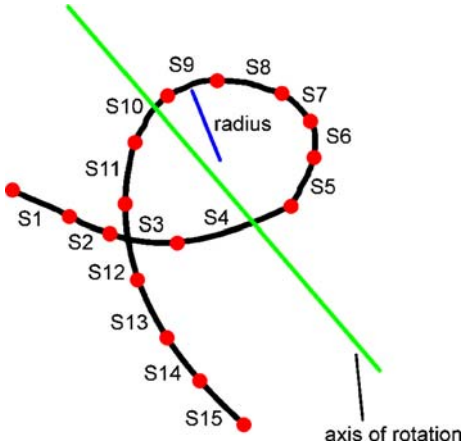


**Fig. 3.** The segments S3 and S12 are intersecting. Therefore, mass nodes forming the loop are mapped to a helix via the found axis of rotation

If all these constraints are satisfied, the drawing is detected as a loop. The focal point of the loop is found, and a principal 3D axis of rotation is established at the focal point. With this axis, the mass nodes in this loop are mapped to a helical structure, thus producing a 3D curly hair (Fig. 3).

### 3.5 Wisp formation

After the recording of the segments is completed, an individual wisp is formed according to the recorded control points and wisp style parameters. The captured control points and parameters are fed into a GPU vertex shader to create Catmull–Rom splines of the skeleton and imitator strands. The imitator strand root points are distributed around the skeleton strand root uniformly, using the employed patch structure and Archimedes' spiral (Fig. 4).

Archimedes' spiral is a spiral with polar equation:

$$r(\theta) = \alpha\theta, \tag{1}$$

where $\alpha$ controls the distance between successive turnings that matches our closeness style parameter. If we adapt this to our model, the equation becomes:

$$r(\theta) = c_{i,j}\theta. \tag{2}$$

Because a patch structure is employed for locating strands in a wisp, we can map patch coordinates to polar coordinates as follows:

$$\begin{aligned} u &= r\cos\theta \\ v &= r\sin\theta \end{aligned} \tag{3}$$

Replacing $r$ with Eq. 2, we get the patch parameters as follows:

$$\begin{aligned} u &= c_{i,j}\theta\cos\theta \\ v &= c_{i,j}\theta\sin\theta \end{aligned} \tag{4}$$

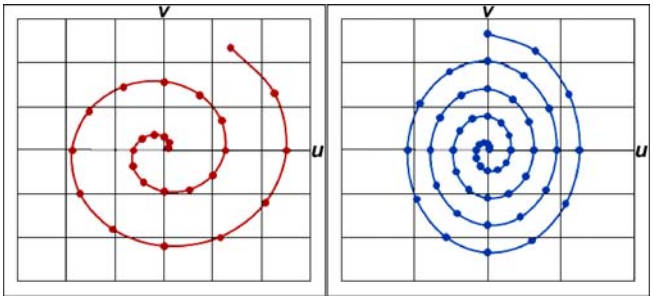


**Fig. 4.** The Archimedes spiral on the Catmull–Rom patch $u$–$v$ space. The points are obtained with 30 degree increments. The *left hand* spiral closeness parameter $c_{i,j}$ is greater than the *right hand* spiral closeness parameter

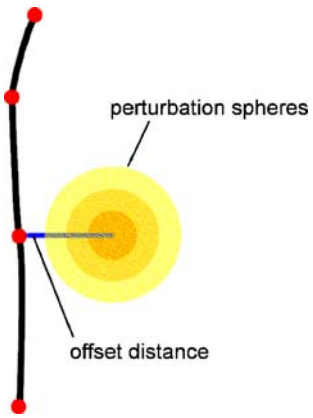


**Fig. 5.** The *blue vector* represents the offset distance for that control point. According to the fuzziness parameter, the corresponding control point of the imitator strand is perturbed by randomly replacing it in the volumes defined by perturbation spheres

The distances between the skeleton strand root point and the distributed imitator strand roots define the offset distances of the remaining control points of the imitator strands from the control points of the skeleton strand. In a wisp, in other words, if closeness is kept constant and no fuzziness is applied to the remaining control points, imitator strands keep these offset distances.

The role of the fuzziness style parameter is to produce uneven looking wisps. The increased value of fuzziness parameter results in a more perturbed imitator strand control point location (Fig. 5).

## 4 Animation creation

### 4.1 Dynamic model

Our model separates the dynamical properties of our skeleton strand structure from the stylistic properties,

using Choe et al.'s approach [2]. We decompose the master strand into two components – outline and detail components – in order to separate the intrinsic geometry of the strand from the deformations applied to it.

#### 4.1.1 Dynamic representation of skeleton strand

When a skeleton strand is drawn, it is processed by the dynamic model, in order to extract its *physical* and *detail representative components*. The component extraction process consists of a linear regression model as described below in which *physical representative components* are aligned with the axis of regression, and *detail representative components* become the vertical distance vector between the axis of regression and the skeleton strand points (Fig. 6).

1. After the skeleton strand is drawn, the axis of regression – vector starting from root ending at last control point – is found.
2. Each control point of the skeleton strand is projected onto this axis, thus forming the physical masses of the strand.
3. Vectors starting from physical masses ending at corresponding control points make detail components, and vectors connecting neighbor physical masses make physical components.
4. Physical components are used to preserve the distance between their connected physical masses.
5. Once the above steps are complete, when simulating the created hair model using physically based techniques, input forces act on the created physical masses.

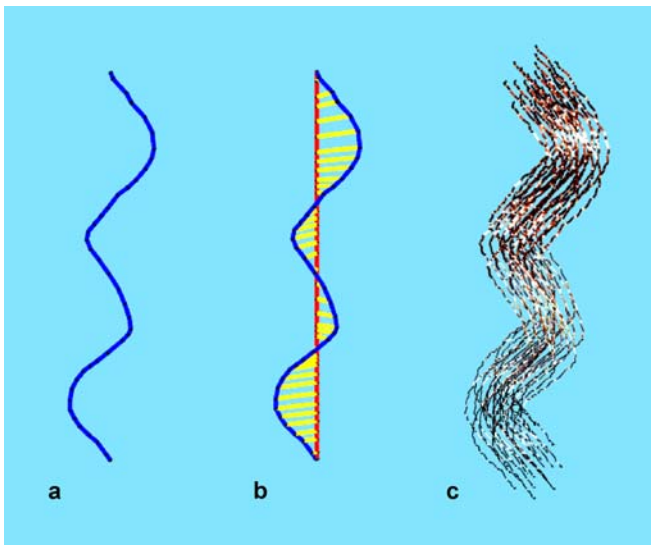


**Fig. 6a–c.** Extraction of physical and detail representative components from a skeleton strand. **a** The sketched skeleton strand. **b** The extracted components *red rods* are physical components and *yellow rods* are detail components. **c** The wisp generated

#### 4.1.2 Global simulation force stroke

Our tool, besides providing full control for creating hairstyles, also aims at providing control while animating the created hairstyle. We propose two approaches in this paper. The first method is *global simulation force stroke* (GSFS). The second method is the key frame model, which will be discussed in the next section. GSFS enables the user to intuitively manipulate the physical environment via the drawing interface. When the tool is in the physical animation mode, a drawn stroke on the screen is recognized as a GSFS, thus creating a virtual force field following the GFSF's pattern. Creating a force field requires a grid structure underneath. Field elements are calculated and stored in grid nodes, which will be later accessed by physical masses that are located inside them (Fig. 7).

### 4.2 Key frame model

We also propose a key frame animation interface for creating hair simulations. Hair wisps are drawn on the 3D head model and their positions are recorded as key frames. After key frame creation is finished, the in-betweening stage operates.

The in-betweening stage is responsible for calculating the transition functions and mapping of wisps between the key frames. It is the most crucial stage of key frame-based hair animation, since it fills the gaps between the key frames provided to the interface, with correctly calculated in-between frames.

The stage consists of three steps: wisp matching between key frames, wisp mass sampling, and path function creation.

#### 4.2.1 Wisp matching between key frames

There can be any number of key frames provided to the tool. Each key frame can also consist of up to hundreds of individual hair wisps. Hair wisps of each key frame should be correctly mapped to the hair wisps on the next key

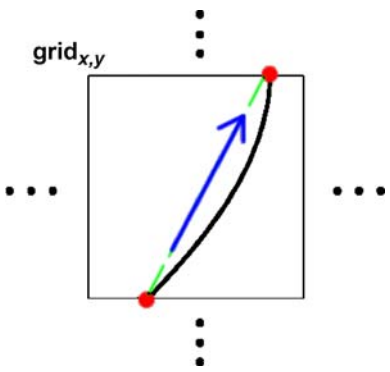


**Fig. 7.** The intersection points of the GSFS and the walls of the grid are found and a force vector between these points is formed

frame to achieve a realistic simulation. For instance, a hair wisp located above the right ear in a key frame should be mapped to a hair wisp again located above the right ear on the next key frame. Without this constraint, the effect would be similar to mapping the right leg to left leg in a walking animation.

To prevent this unwanted effect from occurring, we assume two constraints. The in-betweening stage controls each two consecutive key frames and all of the wisps created during these key frames. The skull is parameterized as a $6 \times 6$ 3D Catmull–Rom patch. All hair wisps have their roots on this patch, and their root coordinates are located as a 2D point in $u$, $v$ coordinates on this patch. By operating on this 2D coordinate system, the most suitable wisp root is found (the closest one in 2D Euclidian distance in parametric space). At the end, the wisps having the closest roots are mapped to each other.

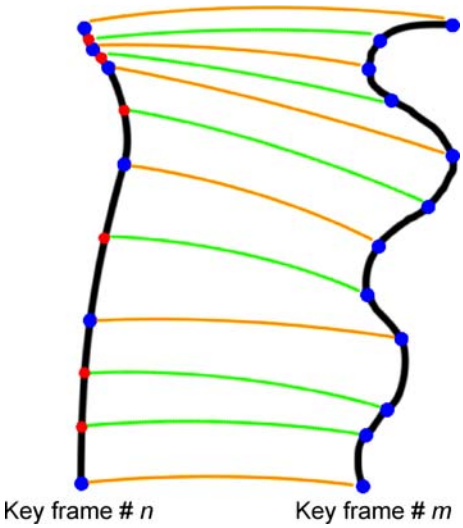


**Fig. 8.** Strand having a smaller number of mass nodes is sampled to higher dimensions via addition of extra mass nodes. *Blue nodes* are original mass nodes and *red nodes* are newly created. The original and extra mass nodes of the *left hand side* strand are mapped to *right hand side* strand nodes

#### 4.2.2 Wisp mass sampling

After the wisp mapping stage is completed, the mass sampling stage begins. Hair wisp lengths can be different than the mapped hair wisp. For example, a hair strand having 17 nodes might be mapped to a hair strand having 25 nodes. To correctly animate each hair strand, the number of nodes in each key frame should match. Therefore, between two key frames, we select the hair strand having fewer numbers of mass nodes, and resample its mass nodes to match the second key frame. We achieve this by creating new mass points between the old mass points (Fig. 8).

For example, let there be 12 mass nodes in the longer hair strand and 6 mass nodes in the shorter one (Fig. 8). We detect that we need 6 extra mass nodes to match the target number. In the short hair, there are 5 segments connecting these 6 mass nodes. We traverse each of these segments from bottom to top and while traversing, we put an extra node to each segment. When we reach the top, if the strand does not have the same number of nodes (in this example it will not, it will have 11), the operation is repeated until the number of nodes are equal. After the traversal, according to the number of extra nodes introduced into each segment, the coordinates of each extra node are determined (they are positioned in the line connecting the original nodes of the strand to avoid shape changes). So 2, 1, 1, 1, 1 extra nodes are introduced into segments 1, 2, 3, 4, 5, respectively.

Now the animation between these two hair wisps can be processed. Extra mass nodes are added to the line connecting the older mass nodes uniformly so that their addition does not cause any change in the hair wisp's shape.

#### 4.2.3 Path function creation

In each key frame, there may be hundreds of hair wisps and each of these wisps will probably contain tens of mass nodes. Each of these mass nodes should go to the correct target positions in the amount of time given (i.e., the number of in-between frames).

After the wisp mapping stage successfully completes, the two wisps that are matched are known by the tool. Since the mass sampling stage is also completed, the matching wisps now have the same number of mass nodes. The path functions for each of these mass nodes are created using linear interpolation. According to the number of in-between frames and the distance to their locations on the next frame; their speeds are calculated and reflected to the animation.

During the animation stage, a real-time air viscous noise is reflected to the animating hair strip. Its aim is to give the hair a wavy look while it is being animated. To further enhance the quality of animation and provide smooth and realistic transitions, slow-in and slow-out systems are introduced, defined below.

#### 4.2.4 Slow-in slow-out and viscous air drag noise

The speed of key frame animation can be defined by changing the number of in-between frames, but the velocities of hair strands are constant between each key frame, which sometimes leads to non-natural movements of the hair strands. We introduce a slow-in/slow-out scheme for individual nodes for more natural interpolation. Slow-in/slow-out alters the speeds of the hair nodes according to the frame index of the in-between frame. The mid-in-between frame has the highest speed (normal speed $\times$ 2), while the beginning and end in-between frames have the

lowest (near 0):

$$s_{\mathrm{i}} = \lfloor n_{\mathrm{f}}/(n_{\mathrm{ib}}+1) \rfloor - n_{\mathrm{f}}/(n_{\mathrm{ib}}+1) \quad (5)$$

$$\mathrm{speed}_{\mathrm{new}} = \mathrm{speed}_{\mathrm{old}} \times (2 - \mathrm{abs}(s_{\mathrm{i}})), \quad (6)$$

where $s_{\mathrm{i}}$ is the speed index of the current in-between frame, $n_{\mathrm{f}}$ is the number of frames so far, and $n_{\mathrm{ib}}$ is the number of in-between frames chosen between two key frames.

To achieve more realistic animations using slow-in/slow-out, we introduced a direction check constraint in our tool. The slow-in/slow-out method checks the general direction of the hair strand's mass nodes in 3D space (for $x$, $y$ and $z$ directions). After finding the direction of each node, a mean direction value is calculated, which is used as the general direction of the hair wisp in 3D space. Having these values in hand, the interpolator switches to the next key frame (the target key frame) and performs the same travel direction check step and compares the general directions for each wisp. If a direction change is detected in any of $x$, $y$ or $z$ directions, the slow-in/slow-out system is operated for $x$, $y$ or $z$ values separately. Thus, if a hair is swinging to the left and then to the right, its $x$ direction component is changing its direction leading to a slow-in/slow-out effect performed on its $x$ direction speed.

Besides slow-in slow out, to improve the quality of the animation, a touch of randomness should be introduced into the system as well. In real life situations, since all of the elements around encounter a resistive force (friction), we should also include this into our animation system. The force is the viscous air resistance force, which in our situation is the force that gives our hairs a wavy randomly moving simulation look. To be able to reflect this into our key frame interface, we introduced an extra randomized noise value to our path function, which edits the positions of each mass node during run time. From root to top, this noise is introduced to the hair wisp but the values are different for different mass nodes. If the node is near to the surface, the viscous air drag force is not too effective but once we reach the top parts of the hair, it becomes more apparent, preventing the stick-like effects during the hair's motion by giving the hair a wavy effect.

## 5 Results and conclusion

In this paper, we proposed a sketching tool to create and animate hair models intuitively. To our knowledge, although there have been sketching-painting based solutions proposed for creating hair models, our tool's contribution is in the use of sketching as a means to create hair models and animate the created models by using physically or key frame-based techniques.

With the hairstyle capture subsystem, the drawn strokes are converted directly to wisps without a mapping process, except for the gesture recognition phase, in which 2D drawn loops are converted to 3D helices. This direct conversion of strokes into wisps makes it possible to create straight, curly, wavy and other arbitrary hairstyles. As the drawn strokes are decomposed into physical and detail representative components, these different types of hairstyles can all be handled by key frame and physically based animation subsystems correctly.

Although we currently cannot provide usability test results, it would be appropriate to perform a usability analysis to discuss the advantages and the limitations of our tool.

*Rapid hair model creation.* Our sketching tool makes it possible to create hair models with different stylistic properties in a short amount of time. For example, the results obtained in Figs. 9 and 10 are created by first-time users in nearly 3 min. By using direct manipulation techniques to create different hair styles, users do not need to use other interfaces and this property decreases the hair model creation time.

*Rapid hair animation prototyping.* Our tool provides creating two types of hair animations: physically based and key frame-based hair animations. Both of the animation techniques can be prototyped rapidly. The implemented GSFS is used for controlling physically based animation whereas sketching techniques are used for key frame animations.

*Limitations.* Although our tool enables rapid production of hair models and animation, it has some limitations. One limitation with our tool is that it is not possible to use artistic techniques like hachure and shading. Also, another problem with the tool is that for defining key

**Fig. 9.** A hair model created in under 3 min

**Fig. 10.** The model is animated with physical techniques

frames, a user has to re-sketch all of the hair wisps, which may be cumbersome when defining a large number of key frames.

Although key frame animation of the hair gives the capability of creating any type of animation (physically correct, incorrect, chaotic, etc.) it might become a bit difficult to imitate a completely real life situation. Slow-in slow-out and viscous air drag noise helps to overcome unwanted transition effects.

**Table 1.** Flow diagram of the hairstyle capture process

| No. of wisps | No. of strands | fps |
|---|---|---|
| 50 | 1250 | 36 |
| 50 | 5000 | 24 |
| 100 | 2000 | 25 |

In Figs. 9 and 10, examples of created hair models and animations by first-time users can be seen. These hair models were created and prepared for animation in under 3 min. Key frame and/or physically based animation sequences were also created by first-time users. The computation intensive physical animation sequences were produced with a standard laptop PC (Intel Core 2 Duo T7500 2.2 GHz CPU, 2 GB RAM) with a video card NVIDIA GeForce 8600M GT. Although the physical calculations are not ported to the GPU, we were able to obtain the results shown in Table 1, inclusive of the full hair-head collision detection procedures.

To achieve this objective, however, we did not use a hair–hair collision detection mechanism, which is considered as a future work by employing data structures and GPU computing techniques.

Rapid hair formation gives the capability to form any type of key frame hair animation rapidly. The results in

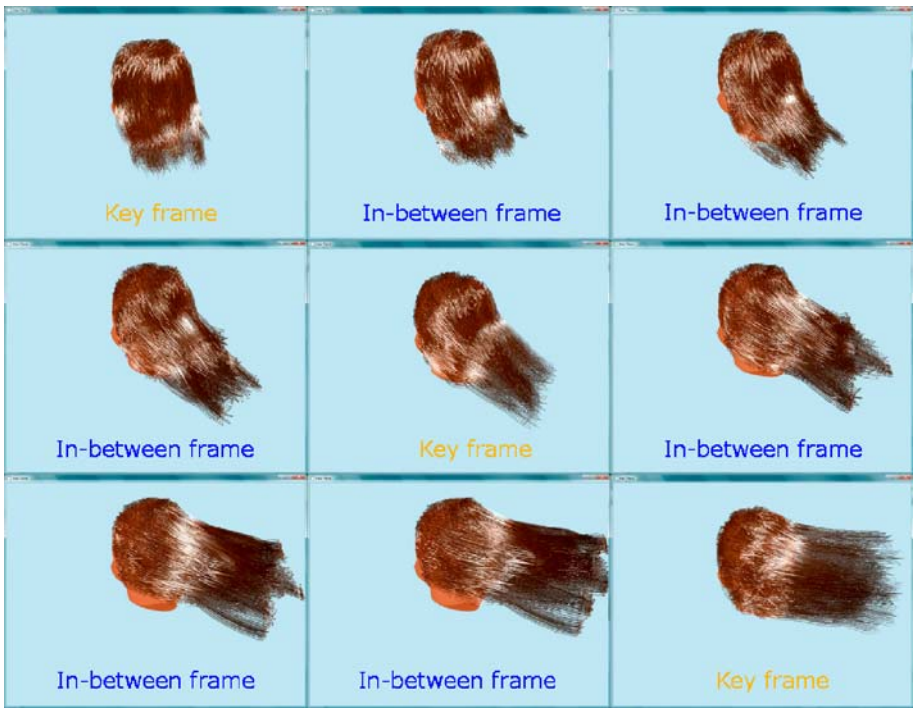


**Fig. 11.** A combination of consecutive key frame animation screenshots, showing how the in-between frames gradually transform the hair strands into the target key frames

Table 1 also apply to animations created using the key frame method. In 1 min, a 30 s key frame-based animation, similar to Fig. 11, can be created.

Also as an extension to our tool, we will implement a physical-key frame hybrid technique, in which physically based forces are used to drive hair strands from one key frame state to another one. Another feature that needs to be investigated is employing physical constraints to hair wisps to imitate hair styling products, braids and other cosmetics.

## References


1. Bando, Y., Chen, B.-Y., Nishita, T.: Animating hair with loosely connected particles. Comput. Graph. Forum (Proceedings of Eurographics'03) **22**(3), 411–418 (2003)
2. Choe, B., Ko, H.: A statistical wisp model and pseudophysical approaches for interactive hairstyle generation. IEEE Trans. Vis. Comput. Graph. **11**(2), 160–170 (2005)
3. Fattal, R., Lischinski, D.: Target-driven smoke animation. In: Marks, J. (ed.) ACM SIGGRAPH 2004 Papers, Los Angeles, 8–12 August 2004, pp. 441–448. ACM, New York (2004)
4. Fu, H., Wei, Y., Tai, C., Quan, L.: Sketching hairstyles. In: Eurographics Workshop on Sketch-Based Interfaces and Modeling (SBIM 2007), 2–3 August 2007, University of California, Riverside (2007)
5. Hadap, S., Magnenat-Thalmann, N.: Modeling dynamic hair as a continuum. Comput. Graph. Forum (Proceedings of Eurographics'01) **20**(3), 329–338 (2001)
6. Hernandez, B., Rudomin, I.: Styling by painting and real time of hair using basis-dependent hair strands. In: WSCG 2004 (Poster), pp. 57–60 (2004)
7. Kim, T., Neumann, U.: Interactive multiresolution hair modeling and editing. In: Proceedings of the 29th Annual Conference on Computer Graphics and interactive Techniques, San Antonio, Texas, 23–26 July 2002, pp. 620–629. ACM, New York (2002)
8. Koh, C.K., Huang, Z.: A simple physics model to animate human hair modeled in 2D strips in real time. In: Hansmann, W., Purgathofer, W., Sillion, F. (eds.) Proceedings of the Eurographic Workshop on Computer Animation and Simulation, Manchester, UK, 2–3 September 2001, pp. 127–138. Springer, Berlin Heidelberg New York (2001)
9. Malik, S.: A sketching interface for modeling and editing hairstyles. In: Proceedings of Eurographics Workshop on Sketch Based Interfaces and Modeling (EGSBM), pp. 185–194. Dublin, Ireland (2005)
10. Oshita, M.: Real-time hair simulation on GPU with a dynamic wisp model. Comput. Animat. Virtual Worlds **18**(4–5), 583–593 (2007)
11. Petrovic, L., Henne, M., Anderson, J.: Volumetric methods for simulation and rendering of hair. Pixar Technical Memo #06-08 (2005)
12. Plante, E., Cani, M., Poulin, P.: A layered wisp model for simulating interactions inside long hair. In: Hansmann, W., Purgathofer, W., Sillion, F. (eds.) Proceedings of the Eurographic Workshop on Computer Animation and Simulation, Manchester, UK, 2–3 September 2001, pp. 139–148. Springer, Berlin Heidelberg New York (2001)
13. Shreiner, D., Woo, M., Neider, J., Davis, T.: Opengl(R) Programming Guide: the Official Guide to Learning Opengl(R), version 2, 5th edn. (Opengl). Addison-Wesley Professional, Boston (2005)
14. Shi, L., Yu, Y.: Taming liquids for rapidly changing targets. In: Proceedings of the 2005 ACM SIGGRAPH/Eurographics Symposium on Computer Animation, Los Angeles, 29–31 July 2005, pp. 229–236. ACM, New York (2005)
15. Treuille, A., McNamara, A., Popović, Z., Stam, J.: Keyframe control of smoke simulations. In: ACM SIGGRAPH 2003 Papers, San Diego, 27–31 July 2003, pp. 716–723. ACM, New York (2003)
16. Volino, P., Magnenat-Thalmann, N.: Real-time animation of complex hairstyles. IEEE Trans. Vis. Comput. Graph. **12**(2), 131–142 (2006)
17. Wither, J., Bertails, F., Cani, M.: Realistic hair from a sketch. In: Proceedings of the IEEE international Conference on Shape Modeling and Applications 2007, 13–15 June 2007, pp. 33–42. IEEE Computer Society, Washington, DC (2007)






RIFAT ARAS is a M.Sc. student at the Department of Computer Engineering, Bilkent University. He obtained his B.S. degree in computer engineering from Bilkent University in 2005. His research interests include augmented virtual reality and human-computer interaction. (arifat@cs.bilkent.edu.tr)

BARKIN BAŞARANKUT is a M.Sc. student at the Department of Computer Engineering, Bilkent University. He obtained his B.S. degree in computer engineering from Bilkent University in 2005. He has participated in various computer graphics related projects including virtual reality and physical simulations during his studies at the university. His research interests include augmented reality and computer animation.

TOLGA ÇAPIN is an assistant professor at the Department of Computer Engineering, Bilkent University. Before joining Bilkent, he worked at the Nokia Research Center as a Principal Scientist, where he led various graphics research activities. He has published more than 20 journal papers and book chapters, 30 conference papers, and a book. He has 2 patents and 10 pending patent applications. His current research interests include mobile graphics platforms, human-computer interaction, and computer animation.

BÜLENT ÖZGÜÇ received his Ph.D. in computer graphics from the University of Pennsylvania in 1978. He taught at the University of Pennsylvania, Philadelphia College of Arts, and the Middle East Technical University, Turkey, and worked as a member of the research staff at the Schlumberger–Fairchild Instrument and Optical Co research center. He joined Bilkent University, Department of Computer Engineering in 1986. His research topics are computer graphics and animation. His articles appear in journals such as *Computers and Graphics*, *Computer Aided Design*, and *The Visual Computer.*