Realistic Hair from a Sketch

Jamie Wither[1] Florence Bertails[2] Marie-Paule Cani[1]

[1]University of Grenoble, LJK, INRIA, France
[2]University of British Columbia, Vancouver, Canada

Abstract

This paper explores a sketch-based interface for quickly yet accurately creating visually realistic hair for virtual characters. Recently, physically-based models have proved successful for generating a wide variety of hair types, but they do not provide a straightforward method for designing target hairstyles. The contribution of this paper is to propose a user-friendly method for controlling such a physically-based model, requiring no specific knowledge of mechanics or hair styling: the user sketches example hair strands over a side view of the character's head, or alternatively annotates a picture of real hair viewed from the side serving as a reference. We show how the sketch can be used to infer the geometric and mechanical properties of the hair strands, to adjust the shape of the scalp, and to generate an adequate hair volume. Our method is validated on a wide variety of hair styles, from straight to curly and short to long hair.

1 Introduction

With the ability of graphical engines to display more and more complex scenes, the creation of visually plausible digital content has become a major problem: asking designers to manually shape each element of a scene is labour intensive, while automatic generation methods can restrict the variety of generated shapes and only offer indirect control. With these methods, the user has to set high level parameters from which the global shape emerges, often a trial and error approach if a specific design is to be reached. This is particularly true in the case of hair generation, on which this paper focuses.

Hair modeling has become an important problem due to the number of digital characters needed in many applications of Computer Graphics, from 3D feature films to video games. The increasing requirement for realism means that the use of simple surface models which was acceptable in the early years is no longer adequate: 3D hair made of several thousand individual strands has to be rendered, even when no hair animation will be computed. Instancing all these strands using standard modeling systems is tedious. Moreover, the results may lack realism: the observed shape of a head of hair results from a physical phenomenon, namely the interaction of thousands of elastic hair fibers, subject to gravity and tending to recover their natural curliness. Achieving the same effect manually for arbitrary hair types is extremely difficult.

This paper proposes a practical solution for quickly designing a variety of natural hair styles: we achieve realism by relying on an existing physically-based engine for hair generation, but set up a completely novel interface for it: the user controls the design by sketching the shape of the scalp and a few sample strands over a fixed view of the head, possibly using a picture of real hair as a guide. The ability to sketch what he wants provides him with direct control over the result. This paper shows how the various elements of the sketch are used to infer the individual geometric and mechanical parameters of the generated strands. Our results demonstrate that this method can be used successfully for quickly generating a large variety of hair from different ethnic origins, which resemble the sketches while being visually plausible.

1.1 Previous work

Due to the number of individual strands needed for creating a full head of hair, all the methods available so far model hair by positioning a few hundred guide strands, from which extra strands are generated at the rendering stage, using either interpolation, wisp-based models or a combination of both [3]. Both geometric and physically-based methods have been used in the past for shaping guide strands.

A number of dedicated, interactive modeling systems were proposed for hair design [5, 12, 27]. Most of them provide the user with precise control of the length, position and curliness of hair, but require a large number of successive manipulations, from the delimitation of the scalp to the positioning and shaping of guide strands. This can lead to several hours for the creation of a single head of hair. This pro-

cess can be made easier using multi-resolution editing [13]. Hair is shaped from coarse to fine using a hierarchy of hair wisps, and by copy-pasting some sub-wisp's style to others wisps. Skilled users can achieve visually realistic results but still requiring more than an hour for each head of hair.

An alternative to manually creating hair geometry is to reconstruct it from images [19, 26], an approach which achieves unsurpassed realism. However, using it to generate the desired hair for a CG character requires having a real life model with exactly the required style, in addition to the possible problems of adjusting the captured hair to a head of different morphology. In this work, we borrow the idea of re-using some data (such as hair color and examples of strand shapes) from a real photograph, but just use it as a guide and generate hair from scratch onto the desired 3D head.

Physically-based hair styling [2, 8, 28, 6] was introduced as a good alternative to purely geometric hair modeling, since it reduces the amount of work required by the user while enhancing realism. Recently, Bertails *et al.* [4] presented a validated static model for hair and used it to generate a number of natural hair styles through the tuning of a few mechanical parameters. Such parameters can be tedious to set up by hand when all the strands do not have the same length, curliness or stiffness. To ease the generation of various hair styles, the authors demonstrated the ability of their model to reproduce a classical hair styling process, from growing hair to wetting, cutting and drying it. Ward and Lin [24] took the same idea even further, they plugged a haptic interface into their physically-based engine for hair [25], enabling the user to feel hair while combing and styling it.

Here our goal is different: we do not aim to reproduce the process skilled hair-stylists have to go through in the real world, instead we want to provide CG artists with a straightforward method to control the generation of realistic hair. Therefore, we propose a sketch-based interface in which the user roughly sketches the desired result, and from this sketch we infer parameters which drive a physically-based hair generator (we are currently using the one in [4]).

Sketching systems have drawn increasing attention in the past few years. Seminal works have demonstrated their usefulness for quickly modeling and editing free-form shapes [9, 1, 21, 18, 11]. Sketching was also used in specific applications, where a priori knowledge ofthe object being sketched helps in making inferences about complex 3D shapes [10, 7, 23]. Such specific sketching approaches have been used previously for modeling geometric hairstyles, specifically Mao et al. [17, 16] and Malik [15].

Mao et al. [17] present a system for creating geometric models of cartoon character hairstyles. The user first sketches the scalp boundary through a series of strokes directly onto a 3D model of the head from user chosen viewpoints, he then draws one stroke specifying the silhouette of the hairstyle viewed face on. This stroke is used to infer hair volume and global shape. In [16] this work is extended to allow greater local control. Global shape is now controlled by specifying four silhouette strokes and a hair partition line. Hair clusters are created which conform to this global shape. Individual clusters can then be reshaped by sketching strokes onto either a front or side view of the head if desired.

Malik [15] presents a comprehensive suite of sketching tools for creating and editing virtual hairstyles. The approach emulates the real-world hairstyling process by providing operations such as cutting, combing, curling, frizzing and twisting - controlled by gestures sketched onto arbitrary views of a 3D head model. A tablet is required as pen pressure is used to determine the scope of many operations.

In both of these previous works the ultimate goal is to create a geometric model for a head of hair which in some way corresponds to the set of sketched inputs. Our goal is different. Our ultimate goal is to create a set of parameters which control a physical model, which would in turn generate a geometric hair model corresponding to the set of sketched inputs.

Although our goal is different, we share the desire to make the whole process easy to use. We also use the idea of generating hair from a few strokes drawn from a fixed view of a head. However, we ask the user to draw some sample guide hairs rather than only a silhouette in order to allow more variety in the resulting hair styles, and we use the sketch to infer some mechanical parameters rather than to create geometry directly.

We don't currently offer as comprehensive a range of manipulation tools as Malik [15]. For example we don't currently support the concept of hairpins and so can't create pony-tails. However our approach doesn't preclude the development of such tools as future work. By constraining our manipulations to only the parameters required by the physical model we are effectively setting the initial state for a head of hair which could later be used for animation.

1.2 Overview

This paper presents the first sketching interface for physically-based hair styling. To our knowledge, combining sketching and physically-based modeling has never been done in the past. Relying on a physically-based engine for designing hairstyles has clear advantages over geometric methods, such as the ability to quickly change hair properties afterwards (dry/wet hair, smooth/curly hair ...) as well as animating the resulting hair style later on, in a straightforward way. However, this combination raises a major difficulty: the user would expect to get a hair style

very close to what is conveyed by his sketch, while the final shape given by the method will result from the convergence of a physically-based simulation. The main issue is thus to infer physically-based parameters from a sketch, while keeping this process completely hidden from the user.

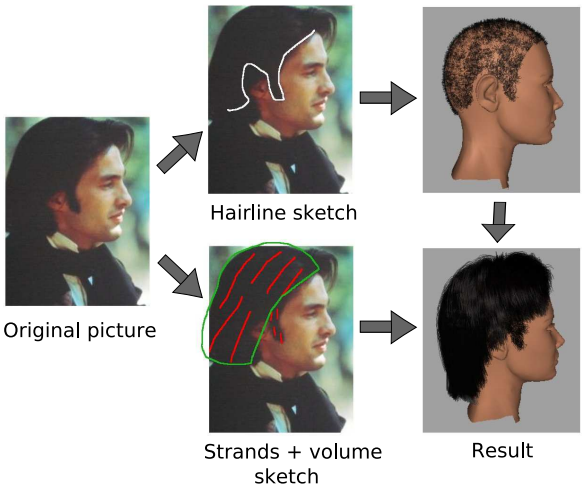

Figure 1. A quick process (10 min) for designing a hairstyle from a photograph, requiring only a few strokes from the user.

Our contributions for solving this issue are:

- A system for inferring the geometric and elastic parameters of individual hair fibers from just a 2D line sketching the final aspect of a sample hair strand.
- The generalization to the sketch-based generation of a full head of hair. Many guide strands are parameterised from a few drawn examples. The hair volume is inferred from the sketch of the hair silhouette, which can also be used to refine the cutting. The shape of the scalp is inferred from the sketch.
- A user friendly interface in which the user can either sketch hair from scratch over a 3D head model (as done in previous work [15, 16]) or over a photo of real hair. In the latter case, the photo is used as a guide: the user can trace the main wisps of hair, but may easily deviate from the photo if he wants to. The photo can also serve as a reference for hair color. The inferred hair is created on a 3D head, and an image of the result can be compared to the original photo for validation.

2 Hair strand from a sketch

In this section we outline a system for inferring 3D, physically-based strands from 2D sketches of hair strands drawn over a 2D side view of the head model.

In the case of hair, considering the strand the user draws as a target shape to be exactly fitted by a physically-based model - as in standard inverse engineering problems - would not be appropriate: due to the large number of hair strands, our system will need to generate a large number of plausible 3D strands from only a few sketched ones. Therefore, the strands the user sketches are rather *interpreted as examples*, used to infer the high level, local properties in terms of length, curliness and final shape under gravity the user wants. In practice, the system will interpolate these parameters to get the appropriate strand parameters everywhere on the scalp, and then generate physically-based strands, which will have the same overall visual aspect, but not fit exactly with any of the sketched strokes.

Our work relies on the physically-based system presented Bertails *et al.* [4]. It provides a set of strand parameters: length l, natural curvature κ_0 (a constant value along the strand, representing the way it would curl without gravity), ellipticity e and stiffness[1] k. The ellipticity stands for the shape of the strand's cross-section, which, as demonstrated in [4] (denoted "eccentricity" here), affects the distribution of curls along a strand hanging under gravity: with elongated cross-sections (none zero ellipticity), curls tend to be evenly distributed as in African hair, while the strand tends to be straight at the top and curly at the bottom when the cross-section is circular (zero ellipticity). The last parameter, stiffness, controls how strongly the hair fiber tends to recover its natural curliness, and thus how much curliness balances the effect of gravity. Although this parameter value can be measured on natural hair, curls can be made stiffer using styling products, so we need to infer stiffness from the sketch for allowing the design of fancy hair styles.

As shown in Bertails *et al.* and depicted in Figure 2, hair strands under gravity tend to take helical shapes at rest, where the helical parameters vary along the strand. This gives us the main clue we are using for inferring the strand's parameters from a sketched 2D strand: we first detect the inflection points along the 2D stroke, and use them to cut the stroke into sections representing a half turn of a helix; we assume each section is modelled by a half helix of constant parameters, infer the arc length of such a helix, and use the sum of the inferred lengths to infer the total length of the hair strand in 3D; we use the inverse of the maximum radius of the helical segments to estimate the natural curvature κ_0. We infer the ellipticity parameter from the distribution of inflection points along the length of the strand. We get the stiffness parameter by matching the span of the strand the user has drawn to the span of a physically modeled strand given the length and curvature values already determined. If the span cannot be matched by adjusting stiffness, as a

[1] To ease understanding, in contrast with [4], we are considering that the natural twist is always zero; we use the term stiffness here for the sake of simplicity, but it stands for the Young modulus in Bertails *et al.*'s model.

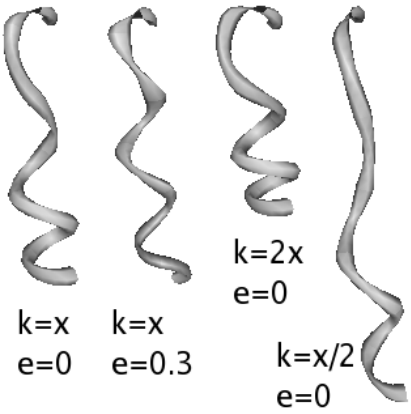

Figure 2. A physically-based hair strand is controlled by 4 main parameters: length, natural curliness, ellipticity and stiffness. Several strands hanging under the effect of gravity are depicted. All have the same length and natural curliness, but different values for the ellipticity e and the stiffness k. x=1.6GPa

final step we can adjust the mass per unit volume of the hair to find a better span match. The remainder of this section details each of these steps.

2.1 Cutting a 2D stroke into half helical segments

We wish to divide the stroke into segments representing one half turn of a helix.

We make the assumption that the user will draw a stroke along a roughly straight centre axis. Given this assumption we can apply principal component analysis to the positions of the stroke samples to determine the direction of this principal axis and the orthogonal secondary axis. By translating our points to this new set of axes we determine the points of inflection along the stroke by looking for a change in sign in the first derivative along the secondary axis.

We now identify separate half helical segments as the bounding boxes aligned to the principal axis and covering the regions between successive zero crossings of the principal axis. The width of each bounding box is defined by the position of the inflection point. The ends of the stroke (outside these bounding boxes) could be assumed to be straight and their length measured in 2D. In practice we include the first and last samples of the stroke as inflection points when performing the calculations.

We could remove the requirement that the user's stroke be along a roughly straight line by instead analyzing the 2D geometric curvature of the drawn stroke. The points of maximum curvature would correspond to the inflection points of the analysis above. The principal axis for each helical section could be defined as the direction between successive points where the curvature value changes sign. However in practice we found the first approach gave reasonable results.

2.2 Fitting a half helix to each segment

This step is, again, purely geometric and does not require any optimization process. We wish to infer the physical length and curvature of the helix modeled by the half helical segment identified in the previous section.

The equation for a circular helix can be expressed parametrically in Cartesian coordinates as:

$$x = r\cos(t) \qquad y = r\sin(t) \qquad z = ct$$

for $t \in [0, 2\pi)$, where r is the radius of the helix and $2\pi c$ is the vertical distance between consecutive loops (known as the "step" Δ [3]).

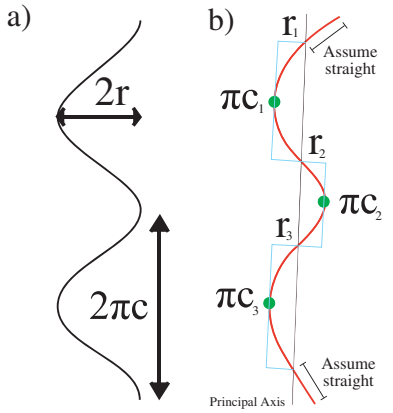

Figure 3. a. The orthogonal projection of a circular helix, and the measurements which can be made on it. b. Inferring the length of a stroke using half helical segments.

The arc length of the helix is given by $s = (\sqrt{r^2 + c^2})t$. Thus the arc length of one complete turn of the helix is given by s when $t = 2\pi$, and a half turn is given by $t = \pi$. Therefore the physical length of the half helical segment can be estimated if we measure r and c.

As shown in Figure 3a the parameters r and c can be easily measured on an orthographic 2D projection of a helix when viewed from a direction perpendicular to its main axis. For each half helical segment we measure the extent of the bounding box along the principal axis to give πc and the extent of the box along the secondary axis to give r.

By taking the sum of the arc lengths for each half helical segment we have an estimate for the actual length of

the hair strand. We estimate the natural curvature κ_0 of the hair strand as being $1/max(r)$ measured from the set of all segments.

2.3 Identifying straight strands

As the radius of a helix decreases, the projected image of the helix approaches the appearance of a straight line. Also - if the user draws a very straight stroke then many inflection points will be detected due to freehand sketching as the stroke will closely follow the major axis. These inflection points will have very little variation along the secondary axis and thus create narrow segments which imply high curvature. To discriminate between these cases and thus correctly detect straight strokes we apply the following tests.

1. Sum of squared errors (SSE). If the SSE of the stroke samples with respect to the secondary axis is below a small threshold - we assume the line is close to parallel to the axis and so this is a straight line. Set the curvature to zero.

2. Ratio of mean segment length to segment width. We measure the mean segment length and widths. If the ratio of length to width is above a threshold (we use 20) then we assume we have detected few narrow segments which implies a straight hair strand and we set the curvature to zero.

2.4 Inferring ellipticity, stiffness and mass per unit volume

We estimate ellipticity from the distribution of all the inflection points measured along the strand. We take the median point from the set of inflection points and measure it's displacement along the major axis. If it is positioned towards the end of the strand then we assume the curls form at the end of the strand which implies a low ellipticity. If it is positioned before or around the middle of the segment we assume evenly distributed curls which implies a high ellipticity. The range of eccentricies observed for real hair is 0.0 (European) to around 0.2 (African), so we chose to use a linear scale for ellipticity clamped in a similar range [0,0.3] as the displacement of the median inflection points varies from 50% to 70% of the strand extent.

Stiffness and mass per unit volume are the only parameters for which we need to run an optimization process. The span of a strand of hair is the distance between the root and the tip. Assuming we have made close to correct estimates for the length and curliness of the strand then matching the span of the example to the span of the model by varying the stiffness will give a good visual correspondence for realistic example strokes. This matching allows the user to set much higher values of stiffness than would be found in natural hair. We use a routine that models the hair using a range of stiffness values and chooses the value that minimises the difference in span between model and example. Values of stiffness far from natural are penalised using a simple linear weighting scheme that ensures that when a range of stiffness values produce similar spans the closest to a natural value is chosen.

Should the stiffness determination routine not find a good span match - then we assume the user has drawn an unrealistically large radius of curvature. Natural hair would be too heavy to hold the curls drawn for any reasonable value of stiffness. In this case we use a similar fitting routine but this time alter the mass per unit volume to make the hair lighter and more able to hold the curl (Figure 4g).

Figure 4 shows various examples of sketched hair strands. Note that our method focuses on realistic hairstyles, and thus only allows for the creation of natural curls, which mostly follow helical shapes in the real world. The handling of unnatural hair shapes such as waves lying in a plane for example, obtained by using shaping tools like hair irons is beyond the scope of this paper.

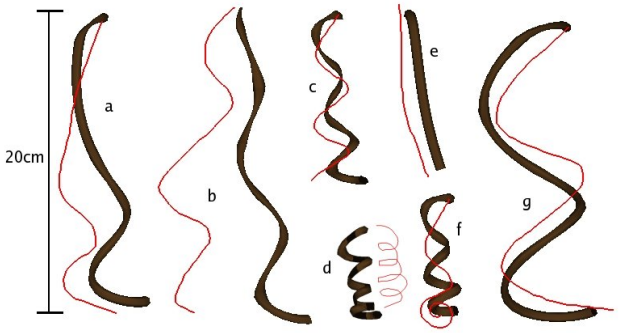

Figure 4. Single strands inferred from sketches: the sketch is interpreted as an example of the desired shape, our system generates similar, physically-based 3D strands. The unnaturally large strand (g) required automatic fitting of the mass per unit volume.

3 Full head of hair

This section generalizes the sketch-based generation technique we have just described for hair strands to the case of a full head of hair.

3.1 Overview of the user interaction

To keep the whole process as simple as possible and in contrast to previous work, we restrict all stroke input to a

side view of the head. Such a restriction removes the need for the user to position the viewpoint, while still allowing the user to draw strokes in all important regions of the head (fringe, side, back).

Throughout the process two viewports are provided. One is a fixed side view input area where strokes are entered. The other is a result view where the camera position can be freely altered by the user. The whole modeling process consists of three steps.

1. Define the scalp area
2. Provide example strands and optional volume stroke
3. Generate the full head of hair and view the result

First, the scalp is defined by drawing one stroke delimiting the scalp extent on one side of the head (step one above). The other side is deduced by symmetry. The user can redraw this stroke until happy with the scalp shape. At this point the scalp shape is fixed for the rest of the process and the user progresses to the second step.

The user then draws example strands anywhere within the defined scalp area (step two above). These examples can be replaced or deleted. Each time an example stroke is drawn a corresponding physically modelled strand is updated in the corresponding area of the result viewport. The update is immediate (roughly a tenth of a second).

By selecting the volume stroke mode the user can also draw an optional volume stroke. This stroke determines the global volume of the style and can be redrawn at any point.

The user can request an update of the full head of hair using a keypress at any time (step three above). At this point all the guide hairstrands in the result view are updated using information from all the input strokes and the volume stroke (this takes a few seconds depending on the number of guide strands, roughly ten seconds[2] for the 40 guide strands in Fig 6). The user can continue to add, adjust or remove guide strands or the volume stroke once this has been done. The whole process is thus iterative, with the user moving back and forth between steps two and three until he is happy with the result.

The rest of this section describes each step of the whole process in more detail.

3.2 Defining the scalp

In contrast with [4], where the scalp region was predefined as a set of full triangles of the head mesh, our sketching system provides the user with an easy way to shape the scalp, which is delimited by an *arbitrary curve* lying on the mesh. This approach is similar to [17] except we usually restrict the curve to be drawn from the side view.

The 3D scalp is inferred using the following steps:

[2]Using a 2.8GHz Intel® Xeon®processor

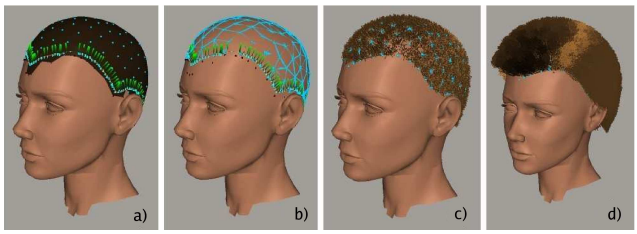

Figure 5. Generation of the hair scalp from a sketched hairline. (a) Generation of the scalp triangles T_{scalp} from the hairline. (b) Triangular tessellation of the scalp area. (c) Clamping hair on the scalp. (d) Growing hair.

1. The stroke is first projected on the 3D head mesh, following the view direction. We duplicate it using symmetry to get a closed 3D curve which delimitates the scalp.

2. We use the orientation of this curve, given by the direction of the user's gesture, to characterize the scalp, i.e. the closed region of the 3D head model delimited by the contour.

3. Finally, we generate two separate lists of triangles from the head mesh: the ones which are entirely within the scalp and the ones intersected by the contour line.

More precisely, let $\mathbf{\Gamma}^{3D}(s)$ be the closed and oriented scalp contour projected on the head mesh, defined as the ordered set of sample points $\{\mathbf{P}_k\}_{k\in[1..N]}$ (the projections on the mesh of the 2D stroke points and their mirror image on the other side of the head).

The mesh triangle T_k to which a projected point $\{\mathbf{P}_k\}$ belongs is computed on the fly. We define T_Γ as the list of triangles that intersect the scalp contour line, built by adding to $\{T_k\}_{k\in[1..N]}$ the set of triangles that intersect the hairline without containing any of the $\{\mathbf{P}_k\}$. This is performed through a search of neighbouring triangles along the closed hairline: finding intersecting triangles between two consecutive points $\mathbf{P}_k$ and $\mathbf{P}_{k+1}$ is achieved by projecting this segment onto T_k and finding the neighbouring mesh triangle in the direction given by the projected, oriented segment.

Then, the set T^{int}_{scalp} of triangles lying entirely within the scalp region is computed using the contour curve's orientation and a propagation mechanism towards the center of the scalp: we compute an oriented vector $\{\mathbf{b}_k\}_{k\in[1..N]}$ lying on each T_k on the contour as the cross product between the normal to the triangle and of an oriented tangent direction $\mathbf{t}_k = \frac{\mathbf{P}_{k+1}-\mathbf{P}_k}{\|\mathbf{P}_{k+1}-\mathbf{P}_k\|}$. This vector points towards the center of the scalp. We use it to propagate the property of belonging to

the scalp to the inward triangles, and stopping each time the border is reached again:

$$\text{if } T \in T_{\Gamma} \quad \begin{cases} \text{search for } T' = \text{ neighbour}(T) \text{ along } \mathbf{b}_k \\ \texttt{propagate} \text{ to } T' \end{cases}$$

$$\text{else} \quad \begin{cases} \text{if}(T)\text{belongs to } T^{int}_{scalp} \texttt{stop} \\ \text{else, add}(T)\text{to } T^{int}_{scalp} \\ \texttt{propagate} \text{ to all the neighbours of } T. \end{cases}$$

Finally, the set of scalp triangles T_{scalp} is defined as the union of T^{int}_{scalp} and T_{Γ}.

3.3 Clamping hair over the scalp

A regular triangulation of the scalp region will be needed to evenly clamp hair over the scalp. This new triangulation is computed using an algorithm inspired from Turk [22]: a given number N of particles are spread over the scalp triangles and uniformly distributed using a repulsion mechanism. This 3D set of vertices we get after convergence is tessellated using a Delaunay tetrahedralisation, from which we keep only the surface triangles.

As the scalp region is bounded by an arbitrary curve, care must be taken when spreading the particles on the triangles belonging to T_{Γ}: each time a particle crosses the contour curve of the scalp, it is projected back onto it.

3.4 Distributing hair parameters over the scalp

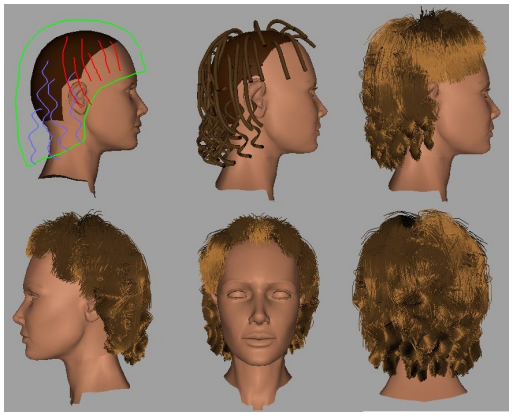

Figure 6. The input and resulting hairstyle from different viewpoints.

In order to generate a realistic looking head of hair the model requires at least twenty to thirty guide strands. However we do not want the user to have to supply an example strand for every guide strand. Therefore - having obtained the parameters for one or a few example strands sketched by the user - we need to extrapolate from this information in order to assign sensible values for all guide strands on the head.

We don't want to constrain the user more than necessary and so the minimum input requirement is just one example strand. We set no maximum constraint. Each example strand is required to start in the valid scalp zone and consist of a continuous stroke. An example strand can be replaced by redrawing the strand starting close to the root, or deleted by double clicking on its root. The user chooses when to update the result view using a keypress. We use the following scheme to extrapolate from the input:

1. One example strand - Assign parameters inferred from this example to all guide strands in model

2. Two example strands - Project guide strand root positions onto the vertical plane bisecting the head. Then project these points onto the line joining the two example root points. Linearly interpolate between the two sets of example parameters at this point.

3. Three or more example strands - Generate a Delaunay triangulation of the example root locations on the vertical plane bisecting the head. Interpolate between triangle vertices by projecting model strand root locations onto the Delaunay triangulation and determining the barycentric coordinates within the intersected triangle. For points outside the triangulation - project to the nearest edge of the triangulation.

A more complex interpolation scheme such as projecting onto a sphere and using spherical barycentric coordinates could be envisioned. However this simpler scheme provides visually convincing results, as illustrated in Figure 8.

3.5 Setting the hair volume and adapting the hair cut

So far the user provides local parameters for the hair model by providing examples of individual strands. However the volume of the overall hairstyle is a useful global parameter we would also like to be able to provide in a simple manner. To allow this we introduce a second type of stroke - the volume stroke.

With this stroke the user roughly indicates the outer boundary for the silhouette of the hair style. The part of the stroke above the scalp is used to determine a parameter controlling the volume of the style. Parts of the stroke below the scalp are used to trim back hairs which are longer than desired without having to redraw examples.

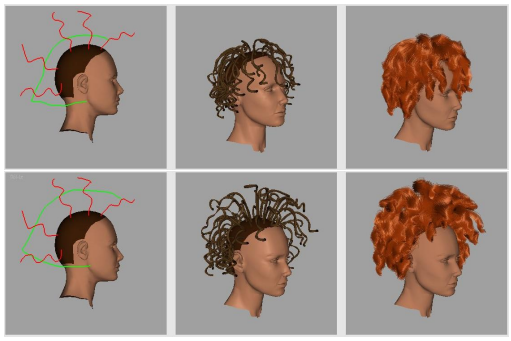

Figure 7. The effect of increasing the size of the volume stroke. Note the hairs at the back of the head have been cut where they extend below the stroke.

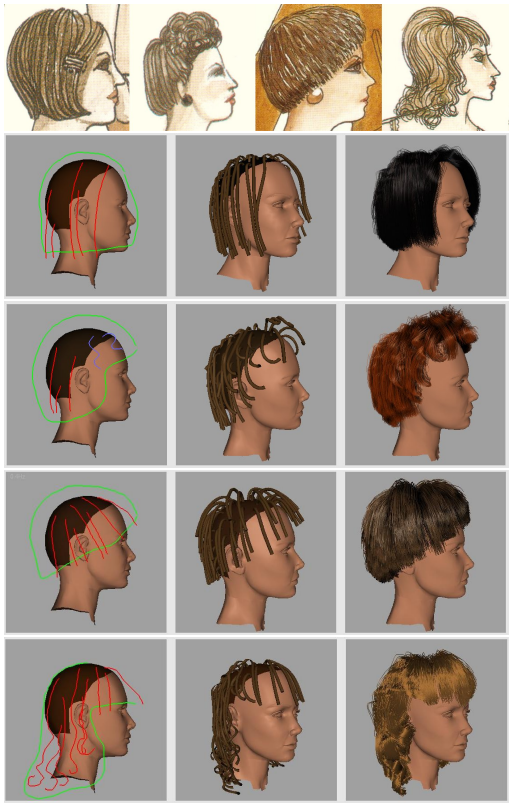

Figure 8. Styles from 1927, 1945, 1965 and 1971. Top: Drawings from [20].

3.5.1 Setting hair volume

To control the hair volume we use the multiple layer hulls model of [14], also used in [4], which models the hair volume due to self interactions while avoiding the computations of collision detection and response between wisps of hair. In this model hair strands with roots higher up the head are tested for collisions against larger offsets of the head model. The controlling factor is the hair volume scaling factor. Larger values generate larger volumes of hair.

When the user enters a volume stroke we project the stroke onto the vertical plane bisecting the model of the head. This projected stroke is then interpreted in two ways. The first way is to determine the hair volume scaling factor. The second way is to determine which hairs to cut.

To determine the hair volume from the projected volume stroke we consider only points above the lowest point of the scalp model. For each of these points we calculate the distance from the nearest point of the head model (using a pre-calculated distance field of the head). We take the largest of these distances as the maximum offset of the hulls model and calculate the hair volume scaling factor which will produce this maximum offset.

3.5.2 Cutting the hair

To determine which hairs to cut we form line segments from the projected volume stroke samples according to the order in which they were drawn and flip the line segment normals so that they are all on the 'interior' side of the line. This is done via a simple voting system where for each segment we take the dot product of the segment normal with the vector from the centre of the volume stroke bounding box to the mid-point of the segment. If the dot product is positive then the normal is pointing away from the interior and the segment votes to flip all of the segment normals. If the majority of votes are to flip then all of the segment normals are flipped.

We then test the orientation of these normals to determine the action of this section of the stroke. If the normal is pointing roughly upwards (in a direction within 90° of the positive y-axis) then we wish to use this section to cut the hair. We form a plane by projecting the points along the x-axis and testing the modeled wisps for intersection with this plane. Any intersecting hairs are cut back to their length at the intersection point.

We also provide separate cut stroke functionality similar to that of Malik [15]. The user selects a cut stroke mode and draws strokes which are projected onto the hair model. Guide strands which intersect a cut stroke are cut back to that length.

3.5.3 Pushing back hair from the face

For long hair styles without a short fringe we need a method to ensure hair does not fall in front of the face. To address this we add an optional spherical force field tool in front of the face that the user can see, move and resize. If enabled this spherical force field is accounted for in the body colli-

sions routine. Ideally this spherical tool should be inferred from the sketch. This will be addressed in future work.

Such force fields could be used to create a 'comb' tool similar to Malik [15], allowing the user to influence the position of individual guide strands in the resulting model.

4 Annotation of photographs

As we emphasized in the introduction, the sketching system we just described is much easier to use for a designer than tuning indirect physically-based parameters, using hair-dresser tools, or shaping the hair by hand. However users with little sketching skill may like to use a real image as a guide. This can be useful if we want to quickly create a hairstyle inspired from a photograph. The user can reuse some aspects of the photo and modify others. The hair colour can be taken from the photo. It also enables visual validation of the resulting model.

When using a photo we must address the problem of the correspondance between the photo and the digital head. For average human heads, simply positioning and scaling the photo with the mouse is often adequate. This is the approach we have taken here. However more complex image registration techniques would be required for non-human head models.

Figure 9. Results of annotating photographs.

Figures 1 and 9 show various hairstyles generated from the annotation of photographs. As demonstrated in our results, the method allows for the quick creation of many different hairstyles involving various hair types (smooth, curly), only from the sketch of a few strokes. Note that Figure 1 depicts the importance of the hairline shape on the final hairstyle. The punk hairstyle in Figure 9 is an example which is hard to model using our constraints - obviously some form of stiffening product has been used for the top of the hair. However by setting a large volume offset using the volume stroke we can approximate it fairly well. A future improvement would be to allow the hull volume to vary according to the stroke drawn so that such volume discontinuities could be modeled, instead of picking just one offset value for the whole head. Again, the last hairstyle above shows an unnatural style, as the woman has had strong curls put in. However, the automatic fitting does a good job of matching the stiffness value and produces similar looking curls.

5 Conclusion

This paper presented the first sketching system for hair that targets realistic hair models. Various hairstyles from arbitrary ethnic origins can be created in a few minutes each without specialized user knowledge, either by over sketching on a view of the targeted 3D model or by annotating a photo, used as a guide. To reach this goal, we infer from the user's sketch the parameters of a physically-based model for hair. The latter can be seen as an easy way to incorporate the a priori knowledge of hair we have.

Although designed for realism, our system can also achieve fancy hair styles, since any mesh could be used as an input for the head and since quite artificial stiffness and volume values can be set up.

As future work, handling hair styles with some partings and constraints (ponytails, braids, etc.) would be very useful to achieve the generation of more fancy hairstyles. We also plan to allow for locally varying clamping directions as currently, hair root orientation is taken as being the surface normal vector at the root position, whereas in reality these directions don't vary so uniformly: they can be directly visible and affect the resulting hair shape. They could be specified, together with the possible parting, using an extra sketch over a front view of the head. Currently some of the physically-based strands we produce may end up crossing vertical sections of the desired silhouette line. This could be avoided by artificially reducing their stiffness if they do so. Regarding the interface, having to sketch several lines of different nature (i.e. the scalp line and the volume line in addition to sample strands) provides good control of the result, but may be distracting for skilled designers. An alternative would be to adjust generic version of these lines according the location and length of the sample strands, a process that could be made completely transparent to the user. We have however to evaluate whether too many samples would be needed in this case. Finally we would like to perform a user evaluation of our system. The feedback supplied by artists who regularly have to parameterise hair

models would be useful in improving our system.

In conclusion, we believe that the combination of sketching and annotation interfaces with procedural methods expressing a priori knowledge (such as physically-based models) is a promising approach for allowing artists to quickly create realistic digital content, while providing adequate levels of user control.

Acknowledgments We thank Basile Audoly for his original code for computing the statics of a single hair fiber. We are also grateful to Steve Marschner who provided us with his code for calculating hair local illumination and to DH Kong for the use of his Flaming Hair image (Fig 9). Jamie Wither is supported by a grant from the European Community under the Marie-Curie project VISITOR.

References

[1] A. Alexe, L. Barthe, M.-P. Cani, and V. Gaildrat. Shape modeling by sketching using convolution surfaces. In *Pacific Graphics*, Short paper, Macau, China, 2005.

[2] 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.

[3] F. Bertails, B. Audoly, M.-P. Cani, B. Querleux, F. Leroy, and J.-L. Lévêque. Super-helices for predicting the dynamics of natural hair. In *ACM Transactions on Graphics (Proceedings of the SIGGRAPH conference)*, August 2006.

[4] 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)*, August 2005.

[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] B. Choe and H.-S. Ko. A statistical wisp model and pseudophysical approaches for interactive hairstyle generation. *IEEE Transactions on Visualization and Computer Graphics*, 11(2), March 2005.

[7] F. S. Fabricio Anastacio, Mario Costa Sousa and J. Jorge. Modeling plant structures using concept sketches. In *Non Photorealistic Animation and Rendering (NPAR)*, Annecy, France, 2006.

[8] S. Hadap and N. Magnenat-Thalmann. Interactive hair styler based on fluid flow. In *Computer Animation and Simulation '00*, pages 87–100, Aug. 2000.

[9] T. Igarashi, S. Matsuoka, and H. Tanaka. Teddy: a sketching interface for 3d freeform design. In *Proceedings of the 26th annual conference on Computer graphics and interactive techniques*, pages 409–416. ACM Press/Addison-Wesley Publishing Co., 1999.

[10] T. Ijiri, S. Owada, M. Okabe, and T. Igarashi. Floral diagrams and inflorescences: interactive flower modeling using botanical structural constraints. *ACM Transactions on Graphics*, 24(3):720–726, July 2005.

[11] O. A. Karpenko and J. F. Hughes. SmoothSketch: 3D free-form shapes from complex sketches. *ACM Transactions on Graphics*, 25(3):589–598, July 2006.

[12] 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.

[13] 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.

[14] D.-W. Lee and H.-S. Ko. Natural hairstyle modeling and animation. *Graphical Models*, 63(2):67–85, March 2001.

[15] S. Malik. A sketching interface for modeling and editing hairstyles. In *Sketch Based Interfaces and Modeling*, pages 185–194, Dublin, Ireland, 2005. Eurographics Association.

[16] X. Mao, S. Isobe, K. Anjyo, and A. Imamiya. Sketchy hairstyles. In *Proceedings of Computer Graphics International*, 2005.

[17] X. Mao, K. Kashio, H. Kato, and A. Imamiya. Interactive hairstyle modeling using a sketching interface. In *ICCS '02: Proceedings of the International Conference on Computational Science-Part II*, pages 131–140, London, UK, April 2002. Springer-Verlag.

[18] A. Nealen, O. Sorkine, M. Alexa, and D. Cohen-Or. A sketch-based interface for detail-preserving mesh editing. *ACM Transactions on Graphics*, 24(3):1142–1147, July 2005.

[19] S. Paris, H. Briceño, and F. Sillion. Capture of hair geometry from multiple images. *ACM Transactions on Graphics (Proceedings of the SIGGRAPH conference)*, 2004.

[20] J. Peacock. *La Mode du XX siecle*. Thames & Hudson, 2003.

[21] R. Schmidt, B. Wyvill, M. C. Sousa, and J. A. Jorge. Shapeshop: Sketch-based solid modeling with blobtrees. In *Sketch Based Interfaces and Modeling*, pages 53–62, Dublin, Ireland, 2005. Eurographics Association.

[22] G. Turk. Re-tiling polygonal surfaces. In *Computer Graphics Proceedings (Proceedings of the ACM SIGGRAPH'92 conference)*, pages 55–64, New York, NY, USA, 1992. ACM Press.

[23] E. Turquin, J. Wither, L. Boissieux, M.-P. Cani, and J. Hughes. A sketch-based interface for clothing virtual characters. *IEEE Computer Graphics & Applications*, Jan. 2007. Special issue on sketch based modeling.

[24] K. Ward, N. Galoppo, and M. Lin. Interactive virtual hair salon. In *PRESENCE: Teleoperators & Virtual Environments (to appear)*, 2007.

[25] 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.

[26] Y. Wei, E. Ofek, L. Quan, and H.-Y. Shum. Modeling hair from multiple views. In *Proceedings of ACM SIGGRAPH'05*, 2005.

[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] Y. Yu. Modeling realistic virtual hairstyles. In *Proceedings of Pacific Graphics'01*, pages 295–304, Oct. 2001.