Rendering Hair using Pixel Blending and Shadow Buffers

André M. LeBlanc

Russell Turner

Daniel Thalmann

Computer Graphics Laboratory
Swiss Federal Institute of Technology
CH-1015, Lausanne, Switzerland

ABSTRACT

A technique is described for adding natural-looking hair to standard rendering algorithms. Using an explicit hair model in which each individual hair is represented by a three-dimensional curved cylinder the technique uses pixel blending combined with Z-buffer and shadow buffer information from the scene to yield a final anti-aliased image with soft shadows. Although developed for rendering human hair, this technique can also be used to render any model consisting of long filiments of sub-pixel width. The technique can be adapted to any rendering method that outputs Z-buffer and shadow buffer information and is amenable to hardware implementation.

KEYWORDS

Hair, Human Animation, Alpha-Blending, Shadow Buffers, Particle Systems.

1. INTRODUCTION

The rendering of natural scenes with a high degree of complex detail remains one of the most important and difficult goals of current image synthesis research. In the field of human animation, hair presents perhaps the most challenging rendering problem and therefore has been one of the least satisfactory aspects of human images rendered to date. The difficulties of rendering hair result from the large number and detailed geometry of the individual hairs, the complex interaction of light and shadow among the hairs, and the small scale of the hair width in comparison with the rendered image. The rendering of hair therefore constitutes a considerable anti-aliasing problem in which many individual hairs, reflecting light and casting shadows on each other, contribute to the shading each pixel.

In a more general sense, we have to ask the question: what is it that we see when we recognize human hair? We see not only the individual hairs, but also a continuous image consisting of regions of hair color, shadow, specular highlights and, under backlight conditions, haloing. The hair image itself can be totally opaque or it can obscure its background with varying degrees of transparency and cast shadows on it. Despite this complexity, the hair image usually has a definite pattern and texture to it which indicates to the viewer, in aggregate form, the underlying geometrical distribution of the hairs themselves.

Although the hair rendering problem has been addressed by a number of researchers, in some cases with considerable success, we know of no practical method in the literature for rendering hair that is efficient enough for animation purposes and generates acceptable results. In this paper, we will present such a method, which is based on techniques such as pixel blending and shadow buffering. It uses an explicit hair model, where each strand of hair is represented individually, and can be used to add hair to images rendered by any technique that generates Z-buffer and shadow buffer information.

2. PREVIOUS APPROACHES

Several researchers have published methods for rendering human hair, or the more limited problem of rendering fur. The problem itself falls into the category of rendering naturalistic phenomena, and has a lot in common with the problem of rendering grass and trees, which has been addressed with much success.

2.1. Hair Properties

A human hair is, geometrically, a long, thin, curved cylinder. Typically, there can be from 100,000 to 150,000 individual hairs on a human scalp. Depending on the individual, the hairs can vary in width from 0.05mm to 0.09mm. The strands of hair can have any degree of waviness from perfectly straight to kinky. The hair color itself (i.e. its diffuse reflectance) can vary from white through grey, yellow, red, and brown, to black, depending on the degree of pigmentation. Hair also has a strong specular component.

2.2. Volume Density Models

2.2.1. Hypertextures

Very good results have been obtained for the more limited problem of rendering fur, which can be considered very short hair. Perlin and Hoffert [PERL89] employed volume densities, controlled with pseudo-random functions, to generate soft furlike objects.

2.2.2. Texels

Perhaps the most impressive rendering of fur to date was achieved by Kajiya and Kay [KAJI 89] for a teddy bear using a generalization of 3D texturing known as texels. Texels are a type of model intermediate between a texture and a geometry. The image was rendered using raytracing and, using 16 processors, took a total wall-clock time of 2 hours. Although the results are spectacular, it is not clear how well the method can be applied to long hair and the rendering times are impractical for animation purposes.

2.3. Explicit Hair Models

The most obvious, brute-force approach to rendering hair is to model each individual hair as a curved cylinder. This immediately runs into several serious problems. For most practical scenes, the hair width is quite a bit less than the size of a pixel. This can easily be justified with the following argument: with a common perspective matrix set at an angle of 45 degrees, the projection of a 0.1 mm wide segment onto a screen of 1000 by 1000 resolution from a distance of 40 cm covers only slightly less than one third of a pixel. This results in a serious aliasing problem. The sheer number of cylinder primitives, together with the large amount of oversampling necessary to overcome aliasing can easily overwhelm most raytracing programs. The problem of hair self-illumination cannot normally be handled by raytracing. Using a hardware-based graphics engine can speed up the drawing of primitives, but aliasing, shadowing and lighting problems remain.

2.3.1. Rendering Fur

Csuri et al. [CSUR 79] were the first to render fur-like volumes. Each hair was modeled as a single triangle laid out on a surface and rendered using a Z-buffer algorithm for hidden surface removal.

Better results were obtained by Gavin Miller, who rendered furry animals [MILL88a], [MILL88b] by rendering images made of explicit hairs. Each hair was modeled with triangles to form a hexagonal pyramid. Oversampling was used to avoid aliasing. Although the number of hairs was relatively small and their thickness was large, these techniques were nonetheless rather computationally intense. Presumably, it would become impractical when scaled up to a full human head of finer hair.

2.3.2. Human Hair using Z-Buffer

Watanabe and Suenaga [WATA 89] modeled human hairs as connected segments of triangular prisms and were able to render a full head of straight human hair in a reasonably short time using a hardware Z-buffer renderer with Gouraud shading. Although the hair model had a realistic number of hairs (more than a million primitives), the illumination model was quite simplistic and apparently no attempt was made to deal with aliasing. As a result, the images have a stiff, wire brush-like appearance.

2.3.3. Grass using Particle Systems

Perhaps the closest thing to a practical technique for rendering properly anti-aliased hair in the literature to date is the Reeves' rendering of grass [REEV 85] with particle systems. The term "particle systems" was first used by Reeves in [REEV 83] but now encompasses a range of techniques. For our purposes, a particle system is an animation technique based on dimensionless points which represent objects that can be smaller than a pixel. Since the original purpose of the technique was to animate particles in motion (for a sequence in the film "*Startrek II: The Wrath of Khan*"), the points were displaced along their paths of motion during one frame to simulate motion blurr, yielding three-dimensional line segments. Later, the same technique was used to represent static images with thin filaments, such as grass. As a result, the term "particle" often refers, in fact, to a filament which may have a considerable length, but are usually very thin.

Although the particle's structure is smaller than a pixel, it does manifest itself on a larger scale by the way it reflects light, casts shadows, and obscures objects behind it. Therefore, the subpixel structure of the particle needs to be represented only by a model that represents these properties. A particle system is rendered by painting each particle in succesion onto the frame buffer, calculating for each pixel what the particle's color contribution is to that pixel and combining it with the pixel's existing color.

Using particle systems and a frame buffer, Reeves was able to render an anti-aliased, very natural looking field of grass containing over half a million particles in a reasonable amount of CPU time. The technique has several limitations, however. In particular, shadowing is limited, using a stochastic model for local self-shadowing and a simple texture map projected onto the grassy field for shadows of external objects.

3. LIGHTING MODEL

We believe that a particle system type of rendering technique, where hair filaments of sub-pixel width are blended at each pixel to create an anti-aliased image, to be the most promising approach to achieving natural-looking hair, and forms the starting point for our approach. The technique has several problems, however, which have to be addressed when applying it to the rendering of hair.

Because hair is not confined to a specific surface volume, explicit control over the hair length, curve and orientation is necessary. An appropriate lighting model has to be developed. It must be capable of being integrated with other existing rendering algorithms for rendering the hairless objects and should be able to use existing frame-buffer hardware. Most importantly, shadowing has to be handled properly.

3.1. The Geometrical Model

3.1.1. Explicit model of each hair

Simple stochastic models are not sufficient to represent the type of order in a typical human head of hair. Therefore, we regard the problem of hairstyle modeling as a formidable topic of research in itself and choose to work here with an explicit hairstyle model in which each hair is represented individually, allowing us to concentrate on the rendering problem.

3.1.2. Hair is cylindrical 3Space curve

A strand of hair is a long, curved cylinder of constant thickness. It can therefore be represented sufficiently for rendering purposes as a three-dimensional curve segment together with a radius. We represent this curve as a polyline with segments varying in length depending on the waviness of the hair. The complete hairstyle data structure therefore consists of a very large number (over one million) of three-dimensional line segments, together with a scalar thickness value.

3.2. The Lighting Model

When viewed as an aggregate material, consisting of many strands, hair exhibits subtle optical properties which affect its appearance. The specular highlights in particular, owing to the geometry of the individual strands, is highly anisotropic. Furthermore, the interaction of light between the strands of hair is highly complex. Each strand reflects light on adjacent hairs and casts shadows. In backlighting situations, a significant amount of light is also diffusely transmitted through the hair itself, which is partially translucent. Finally, the hair width is small enough to cause, in some situations, noticable diffraction effects.

All of this serves to make the problem of finding an accurate hair illumination model an extremely difficult one. Attempting to explicitly render the individual cylindrical surfaces of each hair is clearly impractical. Acceptable results can be achieved in most situations by using an illuminiation model consisting of a modified anisotropic diffuse and specular reflectance together with proper shadowing.

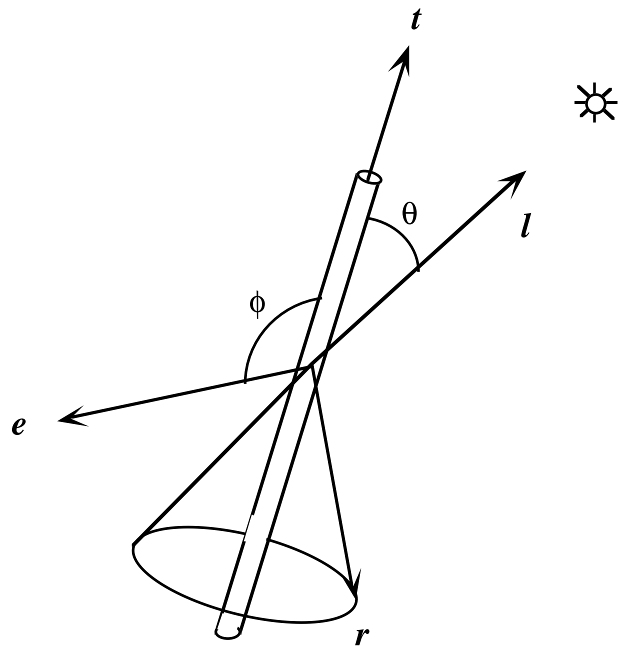

Fig. 1: Hair Lighting Geometry

3.3. Illumination Model Diagram

The hair illumination model we use, based on [KAJI89], follows directly from the underlying cylindrical nature of the hair strand. Figure 1 shows a segment of hair and the important vector quantities used in the model. The unit tangent vector, **t**, represents the direction of the hair's axis. The light vector, **l**, points in the direction of the light source. The reflection vector, **r**, points in the direction of reflected light and is the direction of maximum specular reflection. The eye vector, **e**, points in the direction of the viewer. The angles θ and ϕ are the angles from the tangent vector to the light vector and eye vectors respectively.

3.4. Diffuse Component

The diffuse component of the reflectance can be obtained by integrating the Lambert cosine law along the illuminated half of the cylinder [KAJI89] to yield a simple function of the angle θ:

$$\text{Diffuse Reflectance} = K_D \sin(\theta)$$

Where K_D is the coefficient of diffuse reflectance. This does not take into account self-shadowing, that is, the fact that half of the hair strand is in shadow. Therefore, for an opaque cylinder, the diffuse component will vary with what visible fraction is in shadow, which is dependant on the direction of the eye vector, **e**. However, human hairs are actually quite translucent so that the area in shadow transmits diffusely an amount of light comparable to the amount reflected diffusely on the non-shadow side. As a result, this simple formula makes an approximation that works quite well in practice.

3.5. Specular Component

The specular component is derived essentially by taking a Phong specular distribution and rotating it around the hair axis. This is motivated by making a symmetry-based argument [KAJI89]. At any section along the length of the hair, there are surface normals pointing in all directions perpendicular to the tangent vector, **t.** Therefore, the 180 degree semi-circular surface of the hair which is not in shadow will directly reflect light in a cone-shaped 360 degree radial pattern formed by rotating the **r** vector around the hair axis. This cone represents the angles of maximum specular reflection. The cylindrical Phong specular component is calculated by taking the angle between this cone and the eye vector, **e**, which equals $\phi + \theta - \pi$, and using this as the angle for the standard Phong equation:

$$\text{Specular Reflectance} = K_S \cos^n(\phi + \theta - \pi)$$

Theoretically, an integral should be made around the circumfrence of the hair to yield a much more complicated expression, so this is only an approximation. However, the Phong model itself is only an approximation and experience has shown that the cylindrical version of the Phong model generates quite acceptible results.

3.6. Intensity Equation without Shadows

The diffuse and specular components, together with an ambient component result in a hair intensity equation of:

$$H = L_A K_A + \sum L_i \left(K_D \sin(\theta) + K_S \cos^n(\phi + \theta - \pi)\right)$$

where H is the hair intensity, defined as the power emanating from the hair strand per unit projected area per unit solid viewing angle, L_i is the light intensity, defined as the power per unit area incident on the hair strand from the light source i, K_A a is the ambient reflectance coefficient and L_A is the ambient light power incident per unit area.

4. PIXEL BLENDING

Given the illumination model, the next task is to render the individual hairs in a manner such that aliasing artifacts are avoided. This is accomplished using a pixel blending technique which is similar to a method developed for particle system rendering. We will present the technique first without shadowing and then in the next section show how full shadowing can be incorporated into it.

4.1. The aliasing problem

Aliasing is an artifact that results whenever a source of data is sampled that contains higher resolution information than the sampling resolution. In the case of image synthesis, aliasing becomes a problem whenever the scene to be rendered contains detail smaller than a pixel. This is particularly true of hair where, in a normal viewing configuration, the thickness of a human hair is much less than a pixel width and tens or hundreds of hair strands contribute to the intensity of a single pixel.

4.1.1. Box Filtering

A straightforward solution to the aliasing problem is to treat each pixel as a square and then consider that portion of the hair strand passing through the projection of the pixel. The light intensity contributed by that segment of the hair strand within the boundaries of the pixel projection is considered to be that hair's contribution to the pixel intensity.

Such an antialiasing method is called box filtering because the antialiasing filter is likened to a box constructed above the pixel. The uniform height of the box represents equal weight averaging over the entire pixel area.

4.1.2. Pixel Coverage of Hairs

Therefore, to correctly render a single box-filtered pixel, the light intensity contributed by every hair passing through the projection of the pixel must be blended together properly, along with the underlying hairless image, to form the final pixel intensity. This process is called pixel blending.

Since the lighting model we have used represents the total intensity of the projected hair strand, its intensity contribution to the pixel is equal to the intensity of the hair, H, times the fractional area of the pixel that the hair occupies. The total color value of the pixel, or pixel intensity P, therefore is:

$$P = F_B B + \sum F_j H_j$$

where B is the intensity of the hairless image's pixel, F_B is the fraction of pixel area covered by the hairless image, H_j is the hair intensity of hair j, and F_j is the fraction of pixel area covered by hair j. Note that the sum of all F_j plus F_B is equal to one.

4.1.3. Overlapping of Hairs

As the figure 2 illustrates, there are several factors which determine the hair strand's pixel area fraction: the width of the strand, the length of the segment within the pixel, and overlapping strands. Because of the problem of overlapping hairs, finding the exact value of F_j is a considerable geometrical problem and would normally be prohibitively expensive.

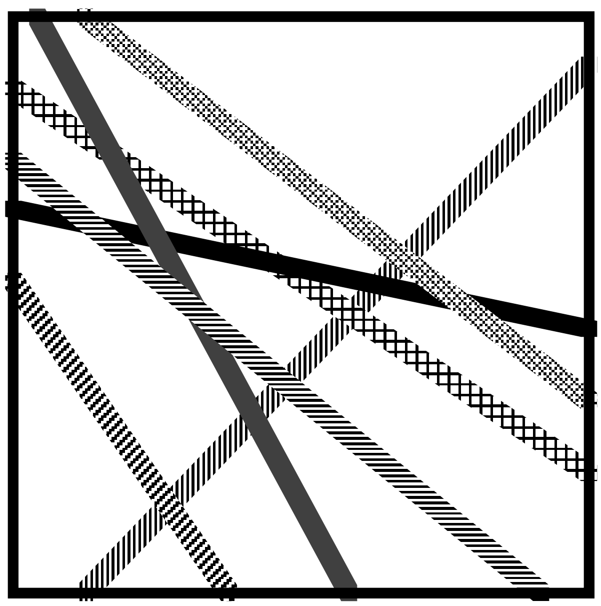

Fig. 2: Hair Strand Pixel Coverage

4.2. Pixel Blending Solution

Fortunately, by making a simplifying assumption, it is possible to reduce the complexity of this problem and make it amenable to standard screen-buffer rendering techniques. This assumption is that, on average, hair intensity information is distributed evenly about the area of the pixel projection. If this is true, then the effect on the overall pixel intensity of laying down one hair strand on top of all the other strands is the same, on average, as laying down one strand on top of an equivalent background of uniform intensity.

4.2.1. Blend pixel color in proportion to coverage

The assumption of uniform intensity distribution allows the complex geometry of all the overlapping hairs to be represented by a single intensity value. This leads to a simple iterative algorithm to blend each hair in its correct proportion. If the pixel intensity is initialized to the hairless image's value, then each hair strand's intensity can be blended into the pixel in back-to-front order in proportion to its projected surface area. The intensity of the pixel P_n after blending in the nth hair strand is therfore:

$$P_n = (1 - A_n)P_{n-1} + A_n H_n \quad , \quad P_0 = B$$

where B is the hairless image intensity, H_n is the intensity of the nth hair and A_n is the area of the hair strand's projection.

4.2.2. Alpha Blending

This iterative function corresponds to the "over" compositing operator [PORT 84] used in compositing raster images with alpha channels. This compositing technique, called "alpha-blending" allows raster operations to be performed on a frame buffer according to an alpha channel. The alpha channel is an additional component of each pixel, along with the red, green and blue components, which can be used to control the amount by which new source pixel values are blended into existing destination pixels. One of the advantages of using pixel-blending techniques is that they are easily implemented in hardware and many advanced frame-buffer architectures incorporate an alpha channel and pixel compositing operations.

Pixel blending is often used to simulate transparency effects, and it is not surprising that the same technique can be used for rendering hair because hair as an aggregate material has a transparent quality to it, if it is spread fairly thinly. Hair is different from a transparent material, however, in that it saturates after reaching a certain density. In other words, as an obscuring transparent material's thickness is increased, some percentage of the hairless image color will always get through, while adding more hair strands will eventually completely obscure its hairless color.

This points out a descrepancy, resulting directly from the assumption of equal intensity distribution, between the pixel blending based algorithm, which does not saturate, and the actual optical properties of hair, which does saturate.

4.3. Depth Information

The pixel blending based algorithm requires that, for each pixel, the hair strands are blended in back-to-front order. Also, only visible hair strands should be blended, that is, strands obscured by the hairless objects are not visible and therefore should not be blended. This requires depth information at each pixel for each strand and the depth values of the hairless object's pixels. The hair strands' pixel depths are then sorted. Pixels with depths greater than the hairless image are discarded and the remaining pixels are blended into the hairless image in reverse depth order.

The depth of each hair strand at its projection onto to pixel can be determined by transforming it into the viewing coordinate system. The depth of the hairless objects must be determined by the rendering software used to generate the hairless image. Since pixel blending is an image composition technique, it works by post-processing an existing image and therefore can be used on any image rendered by a method that generates depth information for each pixel. This depth value, which is called a Z-buffer value, is then used, together with the renderer's camera viewing parameters, to determine the hair strand's visibility with respect to the hairless scene.

4.4. Line Color Interpolation

As described so far, the hair pixel blending algorithm has major complexity problems brought about by the need to break each hair strand up into pixel-sized segments. Most straight to moderately wavey hair, however, can be represented fairly accurately by straight line segments with a projected length of many pixels. Not only are these segments straight, but their color tends to vary fairly continuously over the length of the segment.

A single hair can therefore be rendered quite well by simply breaking it into appropriate-lengthed segments, calculating the hair intensity at each of the the segment end-points, and then drawing each of the line segments into a frame buffer with linear color interpolation between the colors at the end-points. The result is a single hair strand rendered with a smoothly varying intensity along its length.

Rendering a hair strand in this manner takes advantage of the coherence in intensity along the length of the strand. It also allows the use of standard anti-aliased algorithms which have been implemented in hardware in many commercially available graphics hardware systems. The actual hair intensity, based on the lighting model, only needs to be calculated at the end-points of the segments and then the line-drawing and color interpolation algorithms are used to calculate the intermediate pixel values. The same technique can be extended to render many strands by using pixel blending and interpolating the alpha channel as well.

5. ADDING SHADOWS

Shadowing is an essential part of rendering realistic looking hair. This was pointed out by Kajiya [KAJI 89], who gave examples of fur rendered with and without shadowing. Long hair presents a wider variety of shadow effects due to the more complex geometry and the interaction between the hairless image and the hair. A solution to this problem is incorporated into our algorithm for rendering hair.

5.1. The Shadowing Problem

Given the large number of hair strands involved, rendering shadows is obviously a very difficult problem. Raytracing is usually the most accurate method of calculating shadows, by casting secondary rays, but it is impractically expensive for rendering hair. Also, a raytracing shadow method requires that the hair rendering be incorporated into the hairless scene rendering algorithm, while the pixel blending algorithm allows the hair to be rendered in a completely separate post-processing step.

To render a naturalistic human head with hair, shadows must be cast not only from the head onto the hair, but from the hair onto the head. The hair strands can also cast shadows on other strands and this has a quite noticable effect on the appearance of hair.

5.2. Shadow Buffer Solution

Fortunately, there exist shadow rendering techniques which operate on raster images with depth maps. Although shadow buffer algorithms can not usually produce shadows with sharp edges, they can create very acceptible soft shadows with penumbra. This is very appropriate for hair because hair almost always casts somewhat diffuse shadows. Most importantly, shadow composition is fast enough to be practical for animation purposes [REEV 87].

For rendering hair, we combine shadow buffer algorithms and pixel blending to produce very naturalistic results with reasonable efficiency.

5.2.1. Shadow Projection Masks

The simplest shadowing technique is to simply project a two-dimensional shadow pattern onto an object. [REEV 85] from the point of view of each light. This method is obviously limited to certain geometrical configurations where the three-dimensional nature of the shadow casting objects are not significant and it is not adequate for rendering hair.

5.2.2. Shadow Buffers

By adding depth information to a shadow pattern, however, the full three-dimensional information of the shadows is retained, resulting in a general-purpose shadowing method [WILL 78], [REEV 87] that is quite efficient. We call a raster image containing only depth information of a scene from a light source a shadow buffer. It is created by projecting the scene onto a pixel array from the point of view of the light source. Only one piece of information, the depth or Z value, is stored for each pixel.

This results in a shadow buffer which records the region of space which is visible to the light source and therefore not in shadow. A shadow buffer is necessary for each light source together with the projection matrix used to create it.

Using this method, it can be determined whether any point in world-cordinate space is in shadow by projecting the point onto each of the light source's shadow buffers. If the projected depth is greater than the depth in the shadow buffer, the pixel is in shadow for that light source.

5.2.3. Percentage Closer Filtering

Compositing shadows using shadow buffers has an obvious aliasing problem due to the sampled nature of the image and the shadow buffers themselves. In fact, Reeves et al [REEV 87] point out that there are two aliasing problems. The first problem of aliasing is caused by depth sampling to create the shadow buffers. This can be solved by using stochastic sampling, or jittering, to create the depth maps.

The second aliasing problem arises when the shadow buffers themselves are sampled to determine if a point is in shadow. The solution to this, proposed by Reeves et al, is to use percentage closer filtering. In this technique, a region representing the pixel to be rendered is projected back onto the shadow buffer. Generally, this will cover a region of several pixels in the shadow buffer. Each one of these depth values is then compared to the depth of the rendered pixel to determine if it is in shadow, resulting in a binary image in the sample region representing the shadow's edge. This image is then filtered, using some kind of weighted average, to obtain a gray-scale value that indicates the degree of shadowing for that light.

Percentage closer filtering results in shadows with soft edges which resemble real penumbra. The degree of softness can be varied to simulate a diffuse light source by enlarging or reducing the area of the sample region.

5.3. Pixel Blending Hair with Shadows

Shadows can be added into the hair pixel blending algorithm by incorporating it into the lighting model used to calculate the hair intensity. Taking the hair intensity equation of section 3, we add a shadowing cooefficient, S_i , which represents the amount that the ith light should be attenuated by shadowing.

$$H = L_A K_A + \sum S_i L_i \left(K_D \sin(\theta) + K_S \cos^n(\phi + \theta - \pi)\right)$$

The value of S_i is obtained by percentage closer filtering the pixel at the hair strand's depth value against the shadow buffer for the ith light source. The resulting hair intensity value is then blended into the pixel using the hair blending algorithm described in section 4.

One possible drawback of this method is that shadow calculations are only made at the endpoints of the hair segments. As a result, shadows are interpolated along the length of each segment, resulting in a softening of the shadow boundaries. However, as long as the shadows are fairly diffuse, so that the penumbras are larger than the hair segments, this does not create a noticable problem.

5.3.1. Scene Buffer

Pixel blending hair with shadow buffers requires that the original hairless scene be rendered in such a way that shadow buffers are generated for each light source. This allows shadows to be cast from the scene onto the hair. To cast shadows from the hair onto the hair requires an augmented shadow buffer that includes the hair in the depth map.

5.3.2. Hair Shadow Buffer

This can be done by creating a separate shadow buffer for the hair and then combining the two depth maps. At each pixel, the depth values are compared and the lower value is retained, resulting in a hair and scene composite shadow buffer (see figure 3c). The hair shadow buffer can be created using any rendering technique that generates depth values, preferably a hardware-based one because of large number of hair segments involved. The composite shadow buffer can then be used to determine the shadowing coefficient, resulting in shadows cast on the hair by the hair as well as by the scene.

Creating the shadow buffers seperately allows the hair buffer to be rendered using a much more efficient technique than that used in the hairless image renderer. However, the use of a hair shadow buffer has an obvious aliasing problem in the depth domain, analogous to the aliasing problem for rendering. A single hair strand will not cast much of a shadow, but it will leave a depth value in the shadow buffer. In most cases this is not a problem if fairly diffuse shadowing is used.

5.4. Hair Shadows on Hairless Image

The use of composite shadow buffers in the pixel blending algorithm allows shadows to be cast from the scene onto the hair and from the hair onto the hair, but does not account for shadows cast from the hair onto the scene. To do this properly requires that the scene rendering software be able to import shadow buffers to use in its rendering algorithm. The composite scene-and-hair shadow buffer can then be fed into the renderer, generating a scene with hair shadows but no hair (see figure 3d). Hair with shadows can then be blended into the scene to form a final image with full shadowing (see figure 3f).

This type of shadowing is the only aspect of the hair rendering algorithm that requires internal access to the scene rendering software. The remaining parts can all be implemented as an image composition process on an existing rendered image.

6. HAIR RENDERING PIPELINE

The previous sections have described the various component techniques used in our hair rendering method. The full rendering pipeline is now summarized in the following algorithm and in figure 3.

1) Take the scene model description and project it onto each light source, creating one scene shadow buffer for each light source. Usually the scene rendering software can be adapted to perform this function.

2) Take the hair model and project it onto each light source, creating one hair shadow buffer for each light source. This can be done very efficiently by drawing each hair segment into a Z-buffer based frame buffer and extracting the resulting depth map.

3) Composite the depth maps for the scene shadow buffer and the hair shadow buffer, resulting in one single composite shadow buffer for each light source.

4) Generate the scene image and its Z-buffer, using the scene model description and the composite shadow buffers as input to the scene renderer, resulting in a fully rendered scene with hair shadows, but no hair.

5) Blend the hair segments into the scene image, using the scene's Z-buffer to determine visibility and the composite shadow buffers to determine shadowing, yielding the final image with hair and full shadows.

The hair blending algorithm mentioned in step 5 is as follows:

1) Load the scene image and its Z-buffer into the frame buffer.

2) Break each hair strand of the hair model into straight 3D line segments and determine the tangent vector at each segment's endpoints.

4) Project all the segments for all the hairs into the viewing coordinate system and sort by average depth.

5) Scan the hair segments in reverse depth order. For each segment:

a) Determine the intensity, H, of each of the segment's endpoints, using the hair intensity equation with shadows.

b) Transform the hair segment into viewing coordinates.

c) Determine fraction of pixel coverage, F, for the line segment based on width of hair and viewing projection. Set this as the line segments's alpha value.

d) Draw line segment as an alpha-blended line into the frame buffer, using Z buffering for hidden surface removal and linear color interpolation between the intensities of the endpoints.

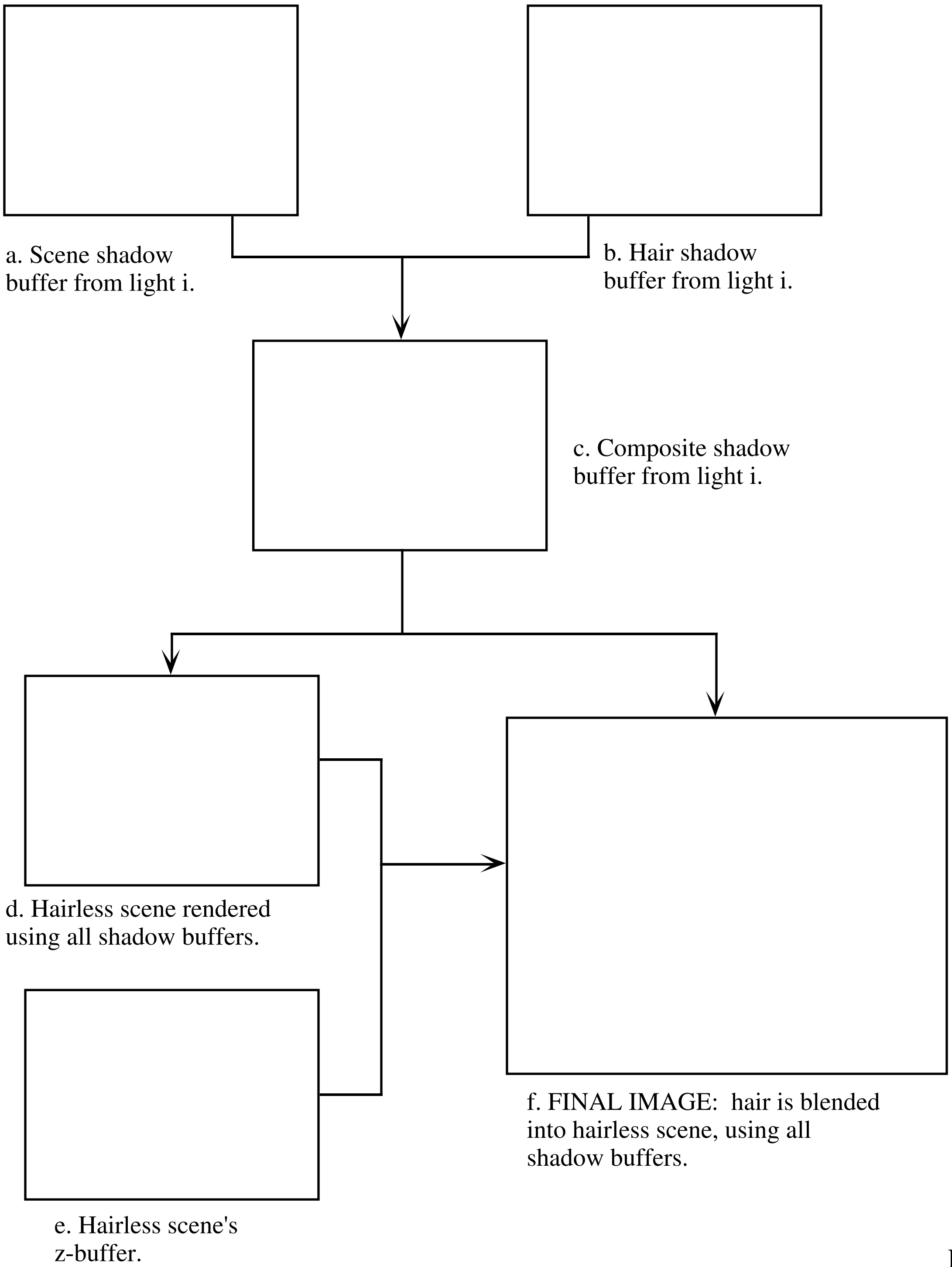

Figure 3. Hair Rendering Pipeline.

7. IMPLEMENTATION

Our hair rendering system is implemented on a Silicon Graphics Iris Power Series workstation with a VGX graphics board. This hardware configuration provides hardware support for coordinate transformations, Z-buffer hidden surface removal, alpha blending and linear color interpolation.

7.1 Memory size issues

Because the algorithm uses so many shadow buffers and because the hair model consists of such a large number of explicit hair segments, memory size can become an issue.

The hair segments themselves, which can sometimes number in the millions, are stored using three-dimensional 16-bit integer local coordinates to conserve memory. The local hair coordinates are multiplied by the local transformation matrix to yield their world coordinate values. The coordinate system is normalized to the bounding volume of the hair model to maximize the resolution.

7.2 Rendering Statistics

In Figure 3, the number of hairs rendered is 101608 with a total of 507836 segments. Three shadow buffers of 1000 by 1000 pixels resolution were used to render Figures 3d and 3f. The shadow buffer in Figure 3b was generated using a hardware based Z-buffer taking 24 seconds. Rendering the hairs into Figure 3f took 4 min 20 seconds. This includes 2 min 20 seconds for shading calculations distributed on four processors, 1 min 15 seconds for sorting and 11 seconds for pixel blending using a Silicon Graphics VGX board with anti-aliased line drawing and alpha-blending.

Figure 4 shows four balls enclosed a multiple shadowing environment.

8. CONCLUSION

We have shown that, by combining a number of existing rendering and image composition techniques in new ways, it is possible to render naturalistic human hair free of aliasing artifacts and with full shadowing, in a reasonable amount of CPU time. The algorithm makes use of standard scan-conversion techniques which are available in many graphics hardware architectures and many other aspects of the algorthm could be easily implemented in hardware.

The technique does, however, make certain simplifiying assumptions which, in some cases, give unsatisfactory results. In particular, the depth aliasing problem with the shadow buffers tends to cause overly dark shadows cast by relatively transparent hair volumes. We are continuing to investigate several possible solutions to this problem.

Ultimately, the greatest challenge in creating realistic images of human hair is in modeling not just the optical properties but the complex mechanical behavior of hair as well. Any such models must take into account the interaction of the strands with each other to form an aggregate material with quite complicated properties. This remains a relatively unexplored area of research.

ACKNOWLEDGMENTS

We wish to thank Gavin Miller for reviewing the text. The hairless images were rendered using a modified version of Craig Kolb's rayshade 3.0 program.

REFERENCES

[BLIN 82] Blinn J.F. "*Light Reflection Functions for Simulation of Clouds and Dusty Surfaces*," Computer Graphics 16(3), 1982, pp. 21-29.

[BLOO 85] Bloomenthal J. "*Modeling the Mighty Maple*," Computer Graphics 19(3), 1985, pp. 305-311.

[CARE 85] Carey R.J. and Greenberg D.P "*Textures for Realistic Image Sythesis*," Computers & Graphics 9(2), 1985, pp. 125-138.

*[CARP 84] Carpenter L. "*The A-buffer, an Antialiased Hidden Surface Method*," Computer Graphics 18(3), 1984, pp. 103-108.

[CLAR 82] Clark J.H. "*The Geometry Engine: A VLSI Geometry System for Graphics*," Computer Graphics 16(3), 1982, pp. 127-137.

*[CSUR 79] Csuri C., Hakathorn R., Parent R., Carlson W. and Howard M. "*Towards an interactive high visual complexity animation system*," Computer Graphics 13(2), 1979, pp. 289-299.

[GARD 84] Gardner G.Y. "*Simulation of Natural Scenes Using Textured Quadric Surfaces*," Computer Graphics 18(3), 1984, pp. 11-20.

[GARD 85] Gardner G.Y. "*Visual Simulation of Clouds*," Computer Graphics 19(3), 1985, pp. 297-303.

[GOUR 71] Gouraud H. "*Computer Display of Curved Surfaces*," IEEE Transactions on Computers 20(6), 1971, pp. 623-629.

*[KAJI 85] Kajiya J.T. "*Anisotropic Reflection Models*," Computer Graphics 19(3), 1985, pp. 15-21.

*[KAJI 89] Kajiya J.T. and Kay T.L. "*Rendering Fur with Three Dimentional Textures*," Computer Graphics 23(3), 1989, pp. 271-280.

*[MILL 88a] Miller Gavin S.P. "*From Wire-Frame to Furry Animals*," Graphics Interface 1988 (Proc.), pp. 138-146.

*[MILL 88a] Miller Gavin S.P. "*The Motion Dynamics of Snakes and Worms,*" Computer Graphics 1988 22(4), pp. 169-178.

[PEAC 85] Peachey D.R. "*Solid Texturing of Complex Surfaces*," Computer Graphics 19(3), 1985, pp. 279-286.

*[PEAR 86] Pearce A., Wyvill B., Wyvill G. and Hill D. "*Speech and Expression: A Computer Solution to Face Animation*," Graphics Interface 1988 (Proc.), pp. 136-146.

*[PERL 89] Perlin K., Hoffert "*Hypertexture*," Computer Graphics 23(3), 1989, pp. 253-262.

[PORT 84] Porter T. and Duff T. "*Compositing Digital Images*," Computer Graphcis 18(3), 1984, pp. 253-259.

*[REEV 83] Reeves W.T. "*Particle Systems - A Technique for Modeling a Class of Fuzzy Objects*," Computer Graphics 17(3), 1983, pp. 359-376.

[REEV 85] Reeves W.T. and Blau R. "*Approximate and Probabilistic Algorithm for Shading and Rendering Structured Particle Systems*," Computer Graphics 19(3), 1985, pp. 313-322.

*[REEV 87] Reeves W.T., Salesin D.H. and Cook R.L. "*Rendering Antialiased Shadows with Depth Maps*," Computer Graphics 21(4), 1987, pp. 283-291.

[SMIT 84] Smith A.R. "*Plants, Fractals and Formal Languages*," Computer Graphics 18(3), 1984, pp. 1-10.

*[WATA 89] Watanabe Y. and Suenaga Y. "*Drawing Human Hair Using Wisp Model*," Computer Graphics International (Proc.), 1989, pp. 691-700.

*[WILL 78] Williams, L. "*Casting Curved Shadows on Curved Surfaces*" Computer Graphics 12, 3 (August 1978), pp. 270-274.