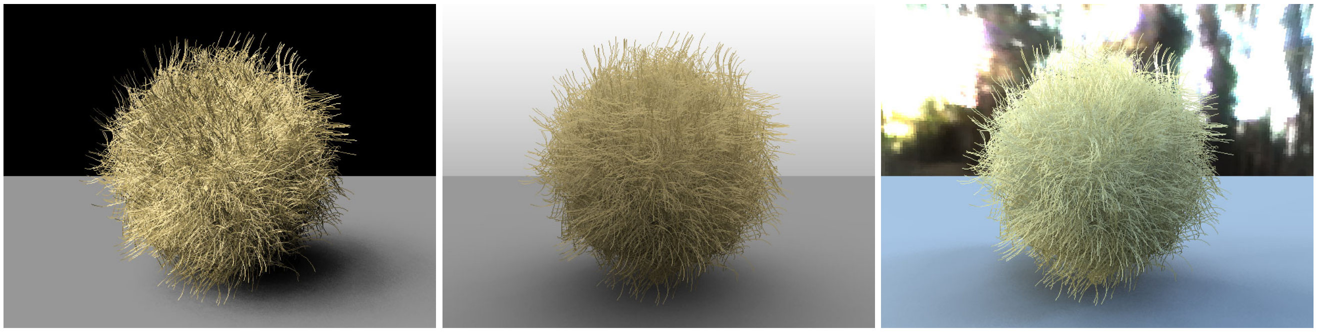
# Rendering Hair-Like Objects with Indirect Illumination

CEM YUKSEL and ERGUN AKLEMAN*
Visualization Sciences Program, Department of Architecture
Texas A&M University

TR0501 - January 30th 2005



Our method with an area light (left image), an ambient occlusion (center image) and with a HDR map (right image)

## Abstract

In this paper, we present a new approach that allows rendering hair like structures with global illumination. We use a projection-based method to calculate the visibility on the shading point. Our technique is particularly suitable for hair-like long and thin semi-transparent objects that can be approximated as a series of line segments. In this projection method, we subdivide the hemisphere centered on the shading point into segments that have the same area. The result of our subdivision resembles an igloo structure. We first project this structure on the plane tangent to the top of the hemisphere and then compute visibility on this plane. We also describe the implementation of a simplification technique that computes visibility using only a portion of the hair strand segments.

## 1 Introduction

Rendering fur and hair has long been a challenge for computer graphics. Early methods used texturing techniques [Kajiya and Kay 1989; Goldman 1997] to avoid rendering of high complexity hair models. These models often consist of more than ten thousand hair strands, formed by a number of line segments that varies according to the length of the hair. Recent advances in hair modeling, animation and rendering have made it possible to use hair in virtual environments [Lokovic and Veach 2000; Thalmann and Hadap 2000; Kim and Neumann 2002]. Advanced shadow computation methods [Lokovic and Veach 2000; Kim and Neumann 2002] that allow transparent hair self-shadows made more realistic hair rendering practical. However, all of the hair shading calculations that these methods use are applicable only for direct illumination of point light or single directional light sources. Skylight and area light sources, as well as indirect illumination cannot be computed by these methods.

Hair rendering using only direct illumination in scenes rendered by global illumination methods creates inconsistencies in the overall appearance. Same global illumination techniques that are based on radiosity [Cohen and Greenberg 1985] and ray-tracing [Cook et al. 1984] are not practical for high quality hair rendering. Radiosity suffers from the high complexity of hair models, which can even be more complex than the rest of the scene. In the case of ray-tracing, it is hard to detect ray-strand collisions, as the hair strands are extremely thin.

In this paper we present a method for efficiently computing the visibility at each shading point by using the projection of hair strand segments. This visibility information is used to calculate the indirect illumination on both hair strands and surrounding objects can be computed. The directional information is used to calculate specular reflections in addition to the diffuse illumination. This technique may also be used for sky light, area light sources, image based illumination [Debevec 1998] and final gathering stage of photon mapping [Jensen 1996].

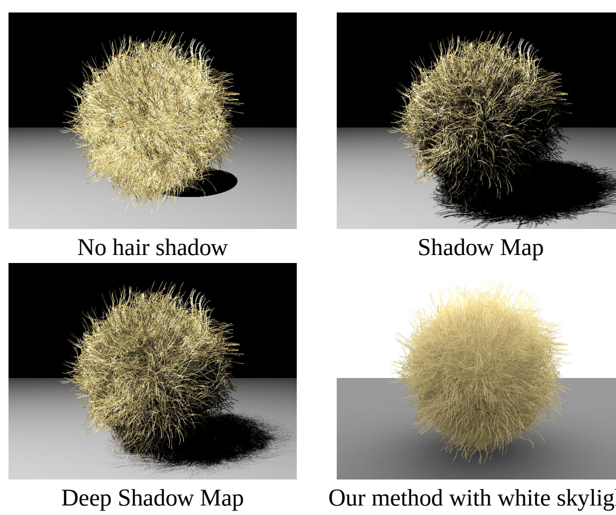



Figure 1: Comparison of our method.

Hair self-shadows are extremely important in visualizing hair structure. The Figure 1 shows a comparison of our technique with

*Address: C418 Langford Center, College Station, Texas 77843-3137. emails: cem@viz.tamu.edu, ergun@viz.tamu.edu. phone: +(979) 845-6599. fax: +(979) 845-4491.

shadow map based methods. The extreme simplicity of the hair model allows us to identify the contribution of our rendering method to the appearance. As seen in the Figure 1, the basic shadow mapping method represents a significant improvement, but it cannot handle transparency. The deep shadow map method improves this result by taking transparent shadows into account. However, both of these methods work only for direct illumination by point and single directional light sources. In our method, we calculate the visibility for each shading point, so that indirect illumination coming from every direction can be calculated.

### 1.1 Contributions

Our main contribution in this paper is is to develop a **framework** for Indirect illumination of Hair-Like Structures (IIH) which has never been done before. This framework also allows us to render images that have not been created before.

It is not feasible to develop a IIH framework based on any derivative of ray-tracing. That is why Global Illumination methods using ray tracing have never been implemented for hair-like structures. On the other hand, our framework is based on projection. We have created such a framework that it is possible to implement IIH using any projection method such as Hemicube or Igloo that is introduced in this paper. Based on our results any good graduate student can implement our framework even using Hemicube.

### 1.2 Justification for Igloo

Our framework can be implemented by any hemisphere based projection method. Our choice of Igloo over Hemicube is a design decision that guarantees our results is accurate and our code is error free. Since there is no prior IIH implementation with any other hemisphere approximation such as Hemicube, it is not easy to compare our implementation with an unimplemented hypothetical cases. We, initially, considered Hemicube, but since the IIH framework has never been investigated before, we needed a simple and robust structure that allows us easily debug the code. And also the number of projection computation (which is the bottle-neck of rendering algorithm) is less in the case of Igloo.

We do not claim that the Igloo structure can be useful anywhere beyond this specific case. It may be but we do not know yet. We already know how to handle triangles but we did not develop Igloo for other purposes such as radiosity or visibility calculation of triangles. We feel that it is not appropriate to attach more mission to Igloo and criticize the paper just because of that.

### 1.3 Line Projections to Igloo

Igloo is very easy to implement for lines: the problem reduces to a line-circle intersection in an infinite plane. It is also easy to create Igloo in different resolutions. Computation is also fast. These properties are crucial to keep the method sample and easy for debugging the code.

Moreover, once we made solid angles equal for Igloo (In practice, the error is negligible) the problem greatly simplified. We knew that Igloo does not introduce any additional error. However, in the case of Hemicube not all the pixels on the Hemicube corresponds to the same solid angle and it must be handled more carefully.

Because of the accuracy of Igloo, we are comfortably able to make simplifying assumptions and test them. In fact, the images we have created without the assumptions were exactly same with the assumptions. That is why we did not include any renderings without the simplifying assumptions. Note that here we are talking about several hours versus 10 minutes on a Pentium4 3GHz processor.

### 1.4 Code Debugging with Igloo

While working on the paper, we observed that it would not be easy to use Hemicube in the development of research code. If we have made any mistake, it would have been very hard to detect it. Again note that when we started we did not have any prior work we can compare with. Of course, Hemicube could potentially be faster if it is implemented in hardware. However, even if it is implemented, we will still need an igloo as a benchmark to identify any programming errors.

Why is debugging hard in Hemicube? For debugging purposes, we needed to develop an algorithm that would provide us Hemicubes with different resolutions. If we made any mistake in code in computing solid angles, it would be impossible to find out. Moreover, in Hemicube, there are 5 planar faces. One line can be projected to 3 faces of a Hemicube. That means Hemicube could potentially be 3 times slower from an Igloo with the same resolution. (Note that we had to implement Hemicube in software.) This is in a nutshell why we chose to develop Igloo.

### 1.5 Other Details

The data structures we used are very simple and straight forward. Therefore, we did not discuss it in the paper. Moreover, we decided not to include details about global rendering pipeline (such as HDR map sampling and shading) since they are not directly related to indirect illumination calculation.

One of our simplifying assumptions comes from the fact that the real hair strands usually form clusters. Although, our hair model is completely random, our results show that the assumptions still hold. Therefore, we believe that we will not have any problem with real hair models.

In addition, we did not include any tricks like using slightly different colors for each hair strand or different hair colors for root and tip that will make the images look more realistic. Still, even in very uniform lighting conditions, hair self shadows help to see the 3D structure of the hair, which would otherwise look flat.

Although we did not include any animated sequence, we do not expect to get any artifacts with motion since our random strand segment selection is deterministic.

Because the prior methods cannot handle indirect illumination, we cannot compare our method to the prior methods (such as deep shadow maps) in the same lighting conditions. That is why, the comparison is shown only in the introduction section to emphasize the difference in the lighting condition that cannot be computed by the prior methods and can be computed using our method.

## 2 Methodology

To compute the visibility of a shading point on a surface, we need to calculate the visibility of each direction over the hemisphere centered at the this point. In the case of hair strands, we use two hemispheres facing opposite directions to take all directions into

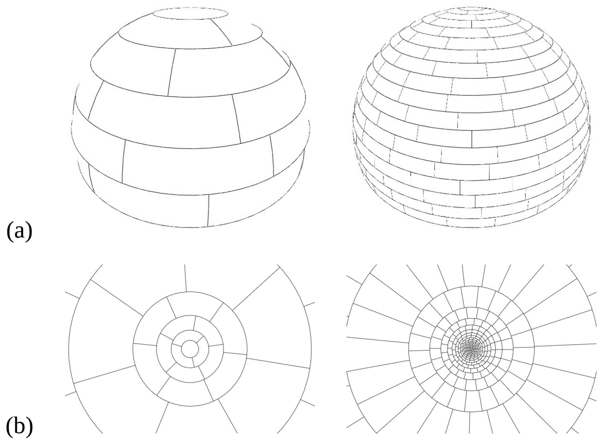



Figure 2: (a) Igloo model created by using our iterative procedure. (b) Projections of Igloos on the tangent plane.

account. The orientation of these hemispheres are chosen arbitrarily to eliminate aliasing artifacts. We subdivide each hemisphere into faces of equal area. The subdivision method that we use results in an igloo like structure. We project this structure onto the infinite plane tangent to the top of the hemisphere. At this point we have all the hemisphere subdivision faces on this plane and we can make the calculations on the plane, rather than on the actual hemisphere surface. Hair strand segments are then projected onto this plane as line segments and checked for intersections with hemisphere faces on the plane. The area of each face that is covered by the line segment decreases the visibility of the face.

We can safely ignore the inter-reflections between hair strands because of the high forward scattering property of hair[Steve Marschner and Hanrahan 2003]. As hair strands are thin and transparent, we assume that strands do not occlude each other.

## 2.1 Making an Igloo

Figure 2a shows igloos that are created by out iterated subdivision procedure, which is explained above.

We will explain our procedure in spherical coordinate system $(r,\theta,\phi)$. Starting from a top circle a new circular region is created in each iteration. The sizes of these circular regions are chosen in such a way that each circular region can be further divided into equal area rectangular regions.[1] Let $C$ denote the user-defined area. Using this area, we first compute the top circle. For small values of $C$, its area can approximately be given as $C=\pi\theta_0^2$. Therefore, we can choose $\theta_0=\sqrt{C/\pi}$.

Then user also defines the desired number of circles along the $\theta$ direction. Let $N$ denote the desired number of circles, then we compute approximate $\Delta\theta=(\pi/2-\theta_0)/N$.

The iteration starts from $n=1$. Let $\theta_n$ denote total angle change in $n_th$ iteration. Using $\theta_{n-1}$, we compute the total area of circular region that will be created in $n^{th}$ iteration as $A_n=\pi(Sin(\theta_{n-1}+\Delta\theta)+Sin(\theta_{n-1}))\Delta\theta$. Our goal is to separate this circular area into $k_n$ equal parts such that $C$ approaches $\frac{A_n}{k_n}$.

[1] These are not really rectangles in the spherical domain. Their edges are circular.

As a result, we can simply choose $k_n$ and $\theta_n$ as

$$k_n=\lfloor\frac{A_n}{C}+0.5\rfloor$$

and

$$\theta_n=\theta_{n-1}+\frac{k_nC}{\pi(Sin(\theta_{n-1}+\Delta\theta)+Sin(\theta_{n-1}))}.$$

To create each rectangle boundary, we start from a random angle $\phi_n$ and divide the circumference into $k_n$ equal parts.

In the last iteration, we cannot adjust $\theta_n$ freely since $\theta_n=\pi$. Therefore, we will not be able to get an exact integer value for $k_n$. Thus, we will have an error. Fortunately, this error is negligible in practice.

## 2.2 Projecting the Igloo to the Infinite Tangent Plane

The projection of the igloo to the infinite plane tangent to the top of the hemisphere, gives us concentric circles. It is easy to calculate the intersections of projected line segments with these concentric circles. If we keep the radii of these circles as 32 bit single precision floating point numbers, we can still represent horizontal angles down to $10^{-37}$ degrees and the accuracy of the radius values always stay the same. Figure 2b show projections of the Igloos.

In the infinite plane, each concentric circular region is divided into $k_n$ equal angle faces. Each face corresponds to a face of the igloo. To create these faces, we do not have to make any projection. We already know that the circular region $n$ consist of $k_n$ number of faces that are divided by the line segments that goes through the origin.

## 2.3 Computing Intersections with Line Segments

As we discussed earlier, we project hair strands to the infinite plane. We will first assume that a hair strand consists of line segments $L(v_1,v_2)$ where $v_1$ and $v_2$ denote two endpoints of the line segment, and we assume that $|v_1|\leq|v_2|$. Let

$$n_0=\frac{v_2-v_1}{|v_2-v_1|}$$

be a unit vector in the direction of the line. To simplify the intersection of the hair line segments with nested circles, we will introduce a new point. Let

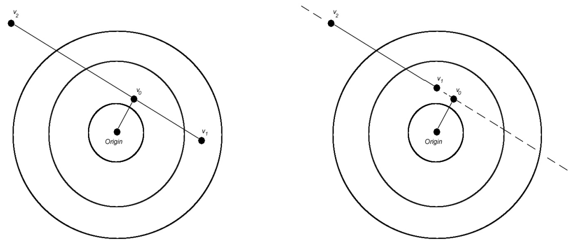
$$v_0=v_1-(n_0\bullet v_1)n_0$$

be the point closest to the origin on the infinite line that includes the line segment $L(v_1,v_2)$. As seen in Figure 3, there are two cases:

- **Case 1.** $v_0\in L(v_1,v_2)$: In this case, we separate the line into two pieces starting from $v_0$ as
  Line 1: $v=v_0+n_0t;\quad 0\leq t\leq|v_2-v_0|$
  Line 2: $v=v_0-n_0t;\quad 0\leq t\leq|v_1-v_0|$
- **Case 2.** $v_0\notin L(v_1,v_2)$: In this case, we use the
  Line 1: $v=v_0+n_0t;\quad |v_1-v_0|\leq t\leq|v_2-v_0|$

Using a line equation starting from the nearest point to the origin, we greatly simplify the intersection with nested circles. Let $r_i$ denote the radius of nested circle $i$, then the circle can be represented by the implicit equation

$$v\bullet v=r_i^2.$$



Case 1 Case 2

Figure 3: Intersection of a line segment with nested circles.

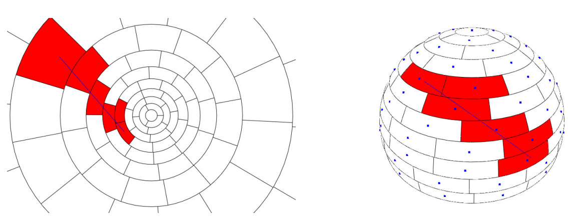
Combining the line equations and the circle equations we find

$$t = \mp\sqrt{r_i^2 - v_0 \bullet v_0}.$$

Based on this equation, the algorithm becomes extremely simple:

- **Case 1.** Start from the first circle in which $r_i^2 > v_0 \bullet v_0$ then
  - Find two intersection points $t_{1,2} = \mp\sqrt{r_i^2 - v_0 \bullet v_0}$ that correspond to line until $r_i^2 > v_1 \bullet v_1$, then
  - Find only one intersection point $t_1 = \sqrt{r_i^2 - v_0 \bullet v_0}$ until $r_i^2 > v_2 \bullet v_2$.
- **Case 2.** Start from the first circle in which $r_i^2 > v_1 \bullet v_1$ then
  - Find only one intersection point $t_1 = \sqrt{r_i^2 - v_0 \bullet v_0}$ until $r_i^2 > v_1 \bullet v_1$.

Once intersections with two consecutive nested circles are computed, it is simple to identify intersections of the line segment with the faces within the given circular region. Note that intersections with two consecutive nested circles give us the first and last face the line has intersected while passing through from the circular region. To find the intersection of the lines we intersect this line with the line segments that separate the faces. Figure 4a shows an example of line and face intersection. Figure 4b shows the corresponding intersection with the faces of igloo.



(a) (b)

Figure 4: Line and face intersections.

### 2.4 Visibility Computation

To compute the visibility, each hair strand segment is projected onto the tangent plane and intersections with the igloo structure are computed on the plane. Let $s$ denote the visibility effect of a hair strand segment that is determined using the opacity and the volume of the hair strand segments. The total decrease in visibility caused by a segment is $\delta V = s/d$, where $d$ is the distance of the segment to the shading point. This decrease is distributed to the faces that the projected segment intersects. The decrease in the visibility of the face $i$ is computed as

$$\delta V_i = \omega_i \delta V,$$

where $\omega_i$ is the proportion of the segment area on the face to the total area of the projected segment. Notice that, by doing so we include the orientation of the segment in the visibility calculation.

When a certain number of hair strand segments fall on a face, the visibility of the face might become zero. After this, hair strand segments falling on this face will have no additional effect.

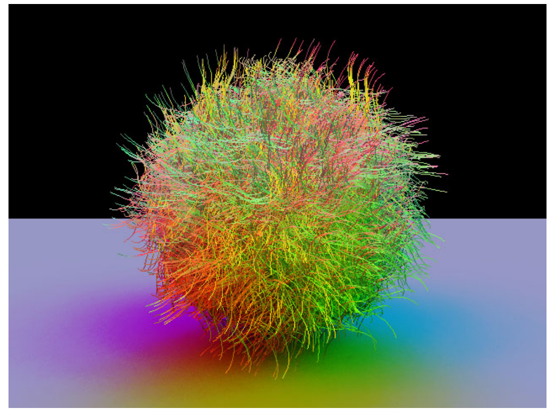


Figure 5: Hair model illuminated by three colored area lights.

## 3 Implementation and Results

By using an igloo it is possible to get extremely accurate results. However, note that the computation we have described above has to be done for every shading point on the image. Moreover, a simple hair consists of a huge number of hair strands and each strand represents a number of line segments. Therefore, we need to improve computation speed as much as possible.

We make five simplifying assumptions to speed up the computation. We have observed that these assumptions do not have visible effects on the final results. The five simplifying assumptions are the following:

- If a line segment intersects with a face, regardless of its length in the face, it makes the same contribution. This assumption allows us to not compute exact lengths of the line segments on each face. The segment shadow is distributed evenly to all the faces that the segment intersects with.
- Face visibility is calculated by adding the visibility effect of each line segment that intersects with the faces. By doing so, we assume that the line segments falling on the face are evenly distributed and line segments do not fall on top of each other.
- We do not need to use all hair strands for shadow computations. We can use a portion of the hair strands by adjusting the segment shadow accordingly. In our experiments we found that when we used only 10% of all hair strands, we achieve the same visual quality as when using all strands.

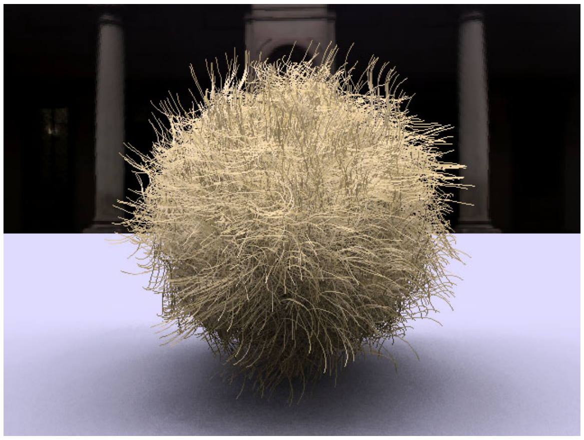


Figure 6: Hair model illuminated by a HDR map.

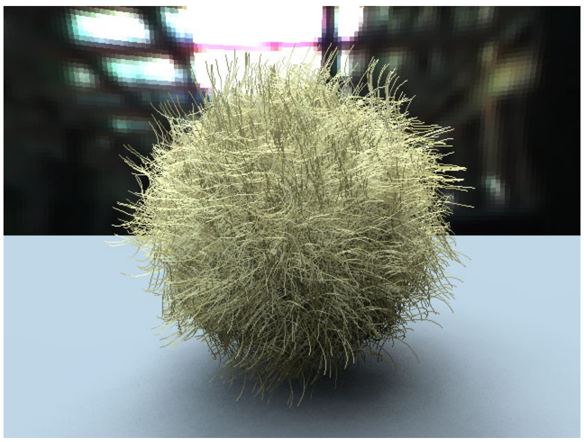


Figure 7: Hair model illuminated by a HDR map.

- We do not have to use high precision hair strands for shadow computations. We can represent a hair strand using a much lower number of line segments.
- Instead of dividing the segment shadow by the segment distance to find the visibility effect of the segment, we use a selection scheme that skips segments according to their distance from the shading point. As a result, distant segments affect the visibility more than they should. However, as we use only a portion of these hair segments, the total visibility change can be kept approximately constant.

Depending on the last three assumptions we can speed up the computation by 2 to 3 orders of magnitude. We can complete final rendering of a $720 \times 486$ image in less than 10 minutes for 10,000 hair strands, with each hair strand consisting of 32 line segments. Figure 5 shows the result of illumination using three colored area lights. As can be seen from Figures6-7, using our method we can illuminate hair with HDR maps publicly available at www.debevec.org. The images do not include anti-aliasing.

## 4 Conclusion

In this paper, we have presented a new projection-based method to render hair like structures with indirect illumination. Our results show that this technique can effectively be used for illuminating hair-like structures by sky light, area light sources, or image based illumination. This technique may also be used in final gathering stage of the photon mapping method. By the help of this method, it is possible to render hair with consistent illumination in the scenes rendered using global illumination or image based lighting methods.

## References


COHEN, M. F., AND GREENBERG, D. P. 1985. The hemi-cube: A radiosity solution for complex environments. In *Proceedings of SIGGRAPH 1985*, ACM Press / ACM SIGGRAPH, Computer Graphics Proceedings, Annual Conference Series, ACM, 31–40.

COOK, R. L., PORTER, T., AND CARPENTER, L. 1984. Distributed ray tracing. In *Proceedings of SIGGRAPH 1984*, ACM Press / ACM SIGGRAPH, Computer Graphics Proceedings, Annual Conference Series, ACM, 137–145.

DEBEVEC, P. 1998. Rendering synthetic objects into real scenes: Bridging traditional and image-based graphics with global illumination and high dynamic range photography. In *Proceedings of SIGGRAPH 1998*, ACM Press / ACM SIGGRAPH, Computer Graphics Proceedings, Annual Conference Series, ACM, 189–198.

GOLDMAN, D. B. 1997. Fake fur. In *Proceedings of SIGGRAPH 1997*, ACM Press / ACM SIGGRAPH, Computer Graphics Proceedings, Annual Conference Series, ACM, 127–134.

JENSEN, H. W. 1996. Global Illumination Using Photon Maps. In *Rendering Techniques '96 (Proceedings of the Seventh Eurographics Workshop on Rendering)*, Springer-Verlag, 21–30.

KAJIYA, J. T., AND KAY, T. L. 1989. Rendering fur with three dimensional textures. In *Proceedings of SIGGRAPH 1989*, ACM Press / ACM SIGGRAPH, Computer Graphics Proceedings, Annual Conference Series, ACM, 271–280.

KIM, T.-Y., AND NEUMANN, U. 2002. Interactive multiresolution hair modeling and editing. In *Proceedings of SIGGRAPH 2002*, ACM Press / ACM SIGGRAPH, Computer Graphics Proceedings, Annual Conference Series, ACM, 620–629.

LOKOVIC, T., AND VEACH, E. 2000. Deep shadow maps. In *Proceedings of SIGGRAPH 2000*, ACM Press / ACM SIGGRAPH, Computer Graphics Proceedings, Annual Conference Series, ACM, 385–392.

STEVE MARSCHNER, HENRIK WANN JENSEN, M. C. S. W., AND HANRAHAN, P. 2003. Light scattering from human hair fibers. In *Proceedings of SIGGRAPH 2003*, ACM Press / ACM SIGGRAPH, Computer Graphics Proceedings, Annual Conference Series, ACM, 780–791.

THALMANN, N. M., AND HADAP, S. 2000. State of the art in hair simulation. In *International Workshop on Human Modeling and Animation*, Korea Computer Graphics Society, International Workshop on Human Modeling and Animation, 3–9.

# A Appendix: Larger Images

In this section, we provide larger versions of images.

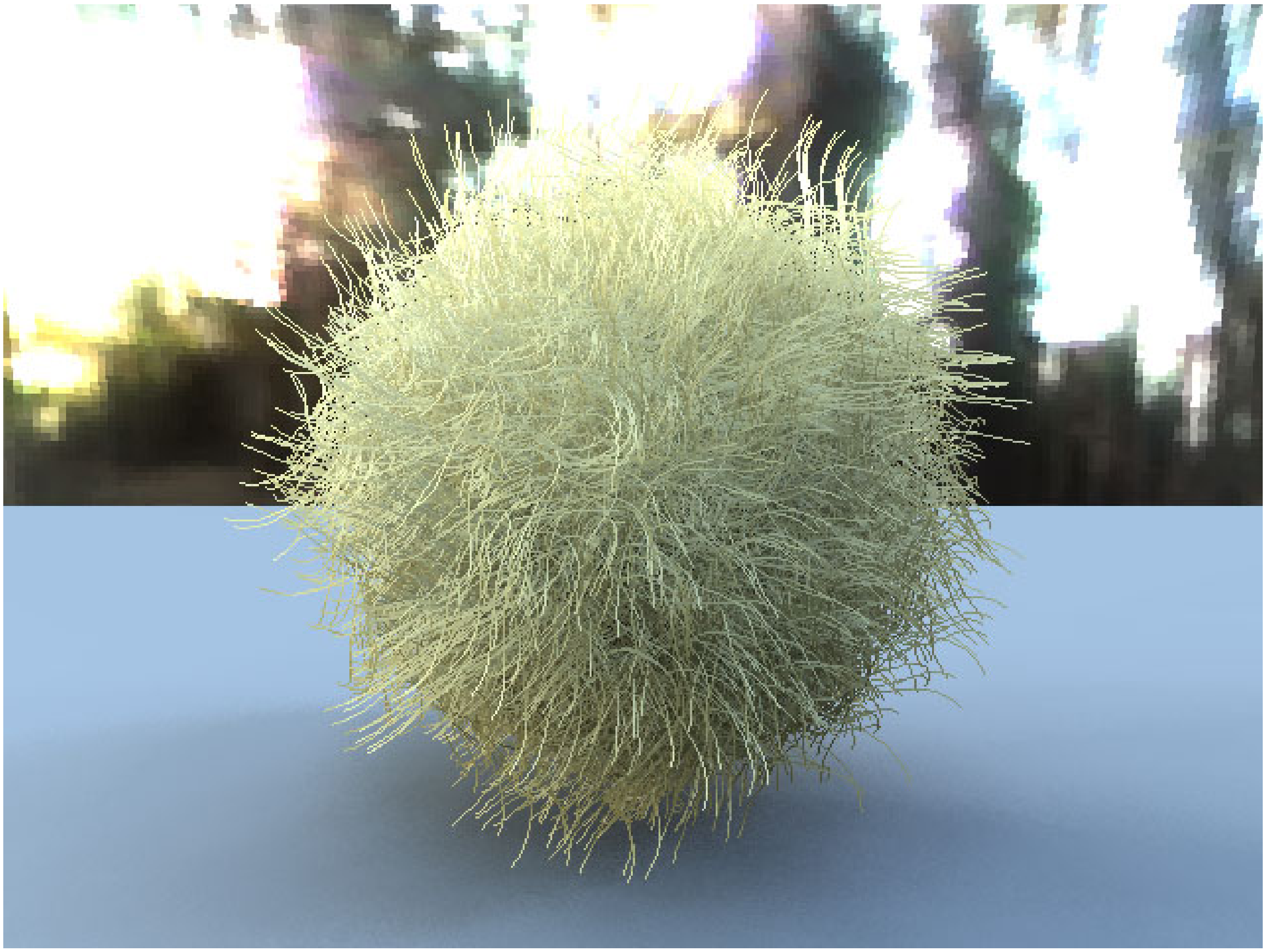


Figure 8: Our method by using a white area light.

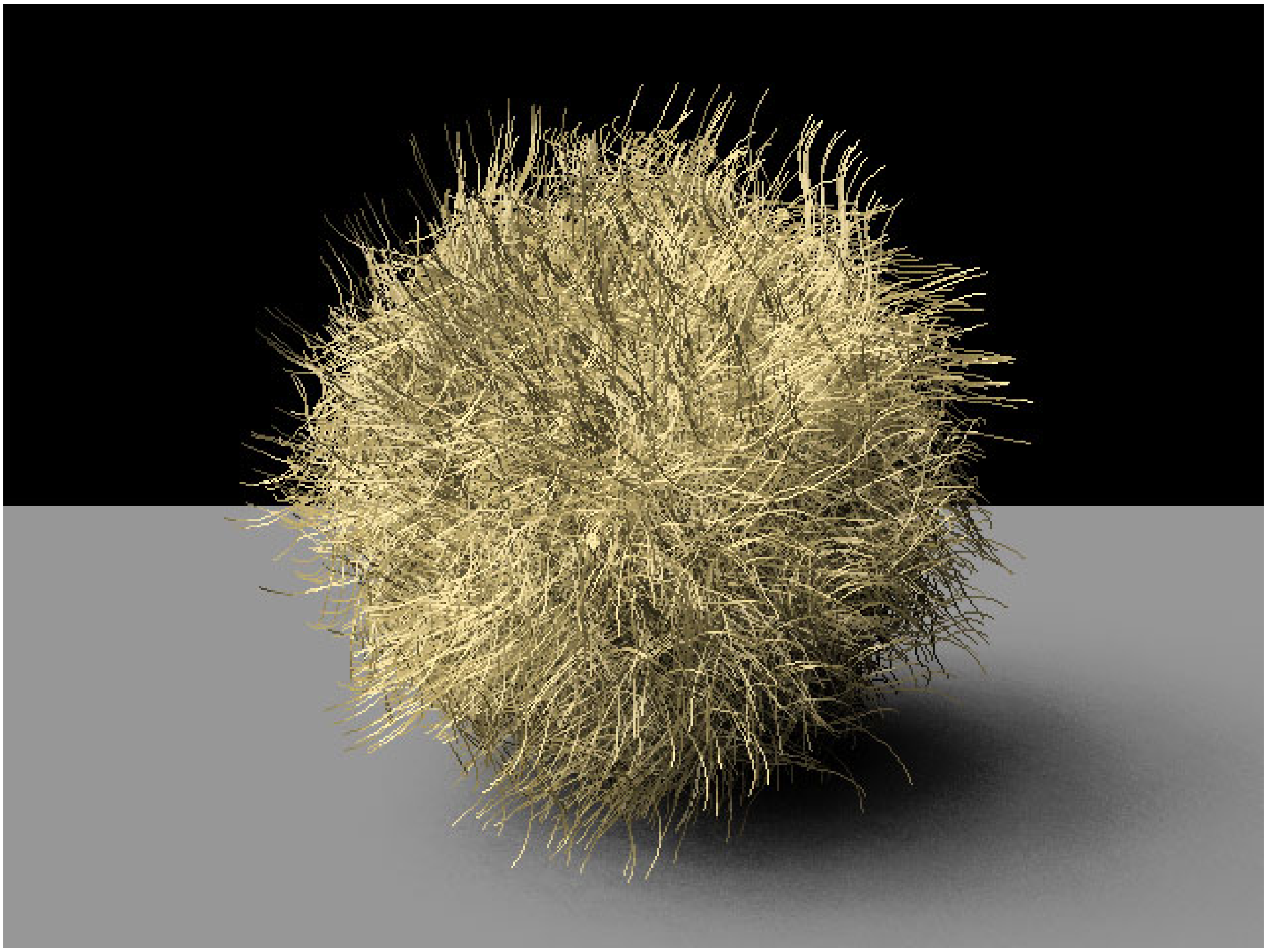


Figure 9: Our method by using a white area light.

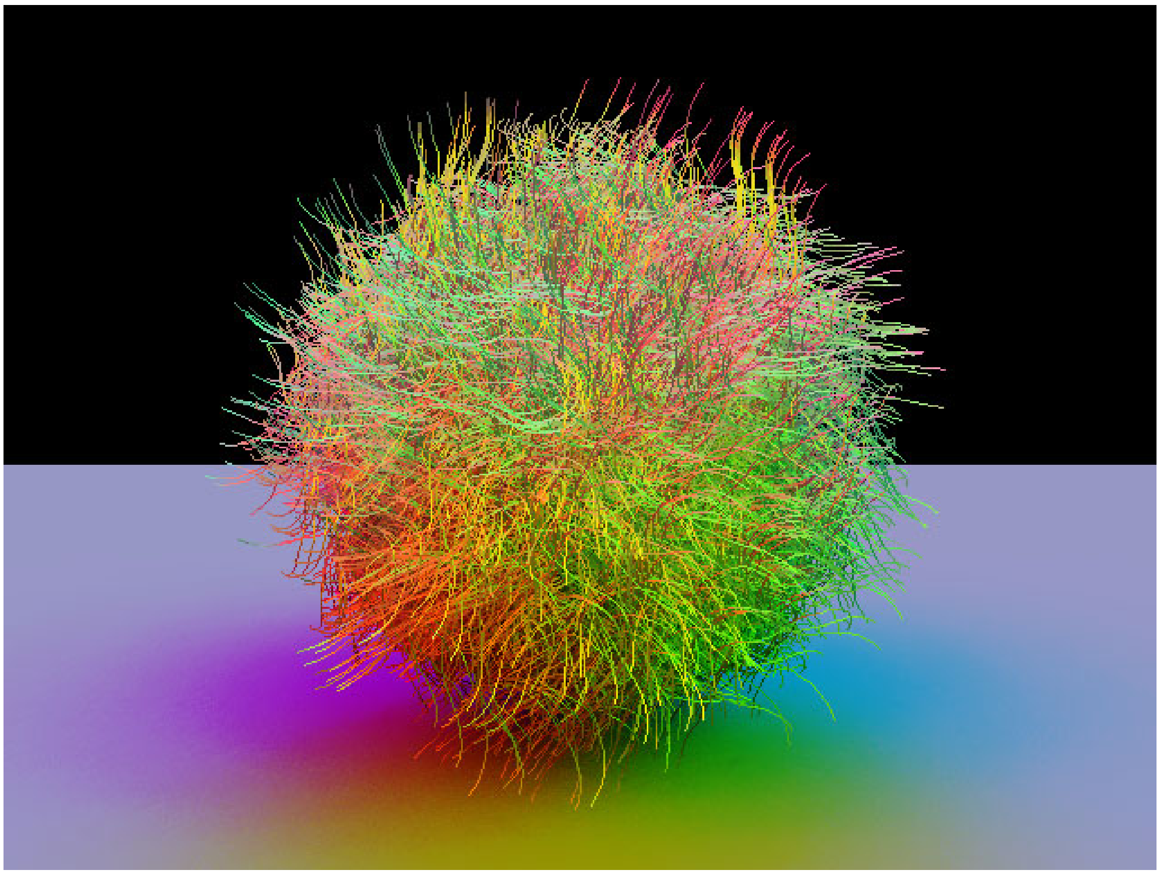


Figure 10: Our method by using 3 (red, green and blue colored) area lights.

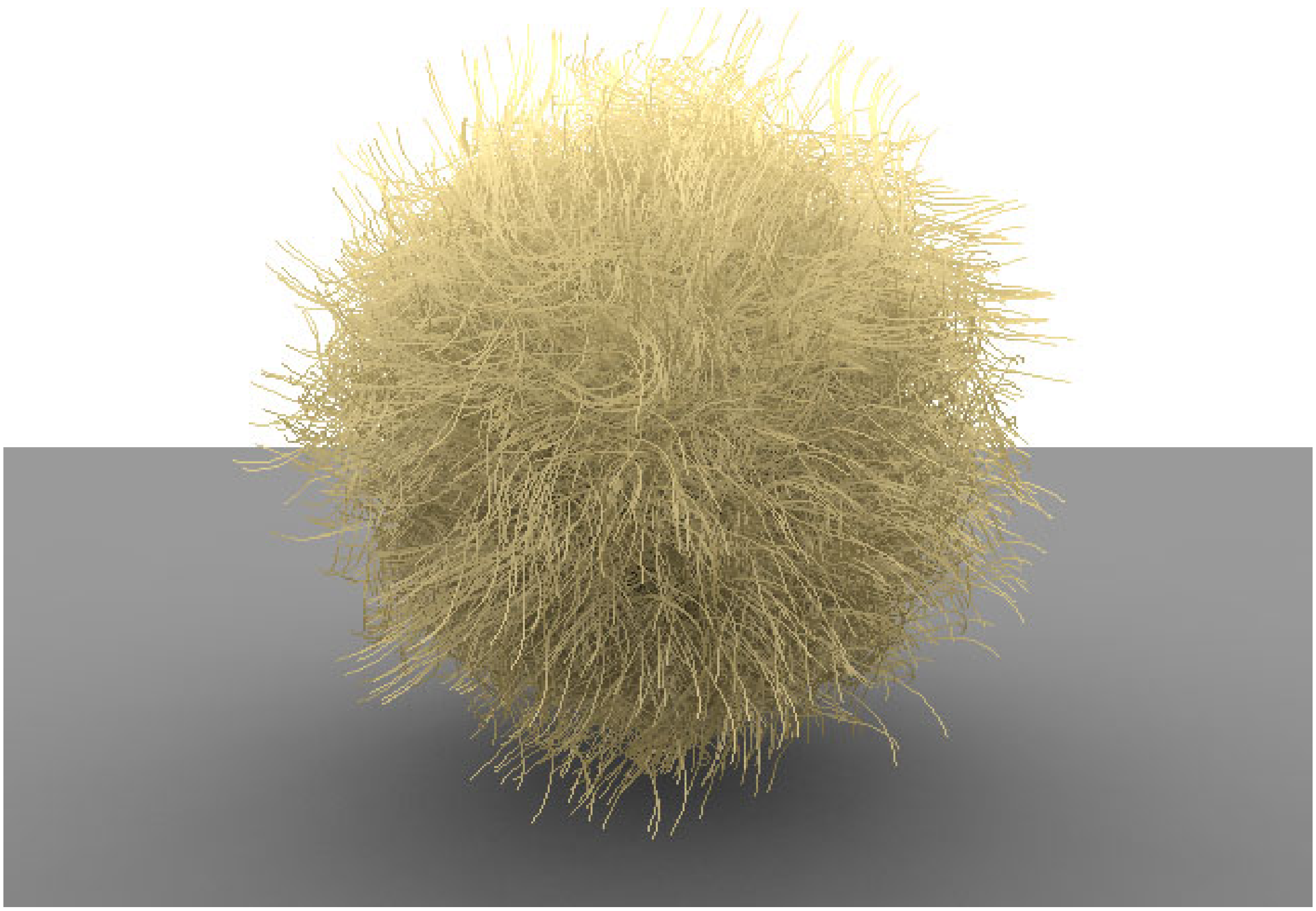


Figure 11: Ambient occlusion with our method. A white environment map is used.

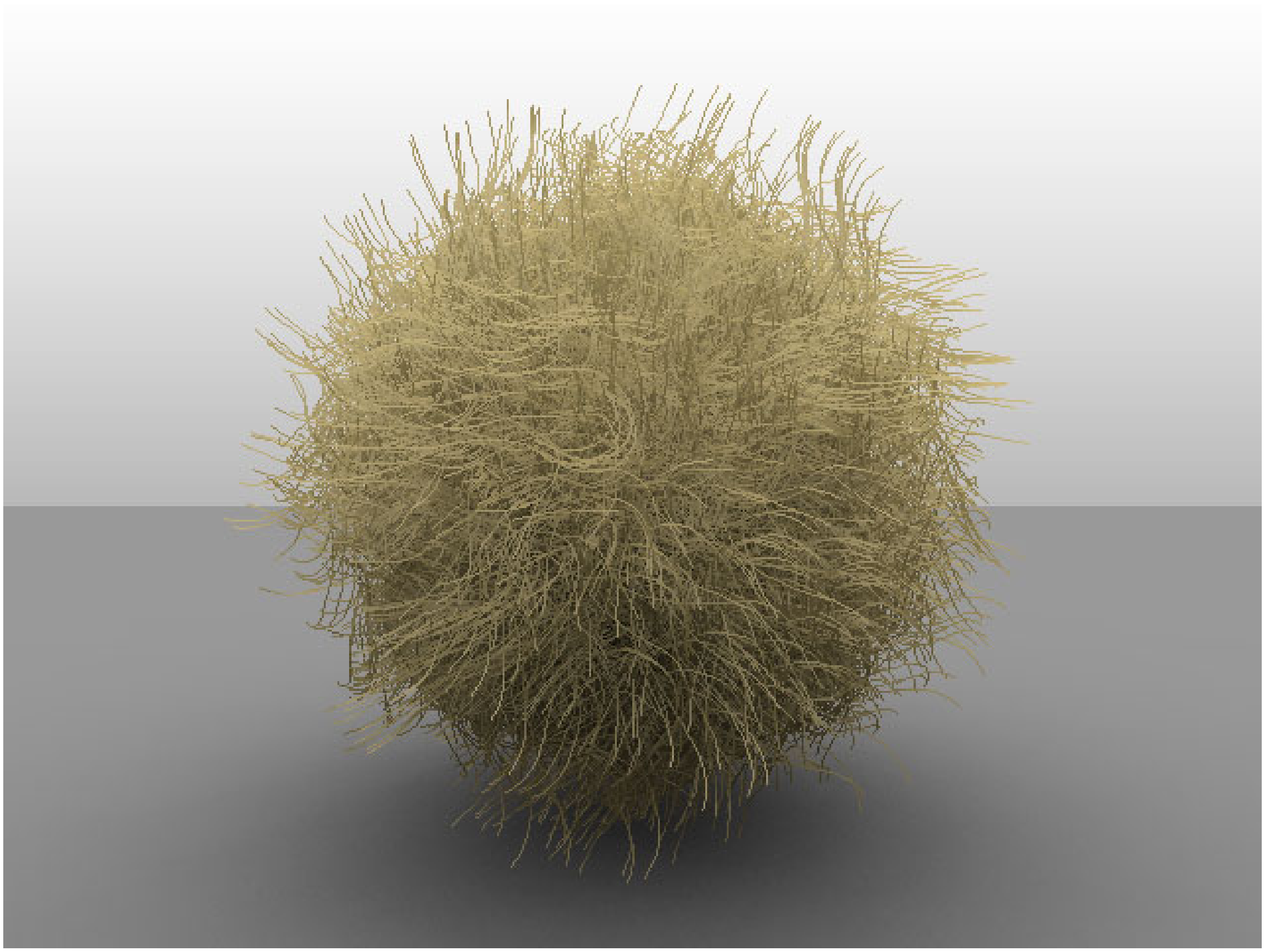


Figure 12: Another ambient occlusion with our method. Environment map is brightest at the top.

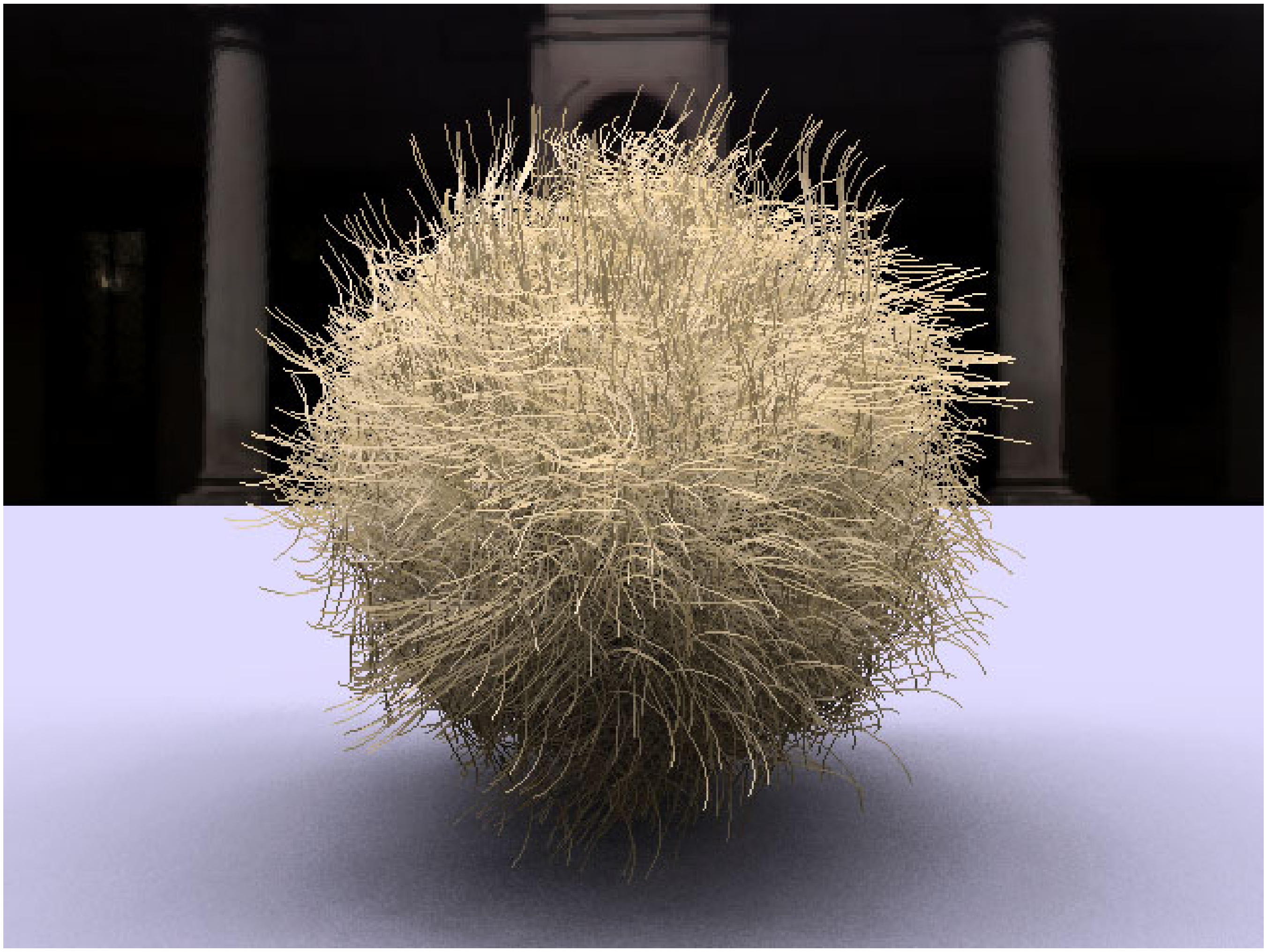


Figure 13: Our method by using Debevec's HDR map.

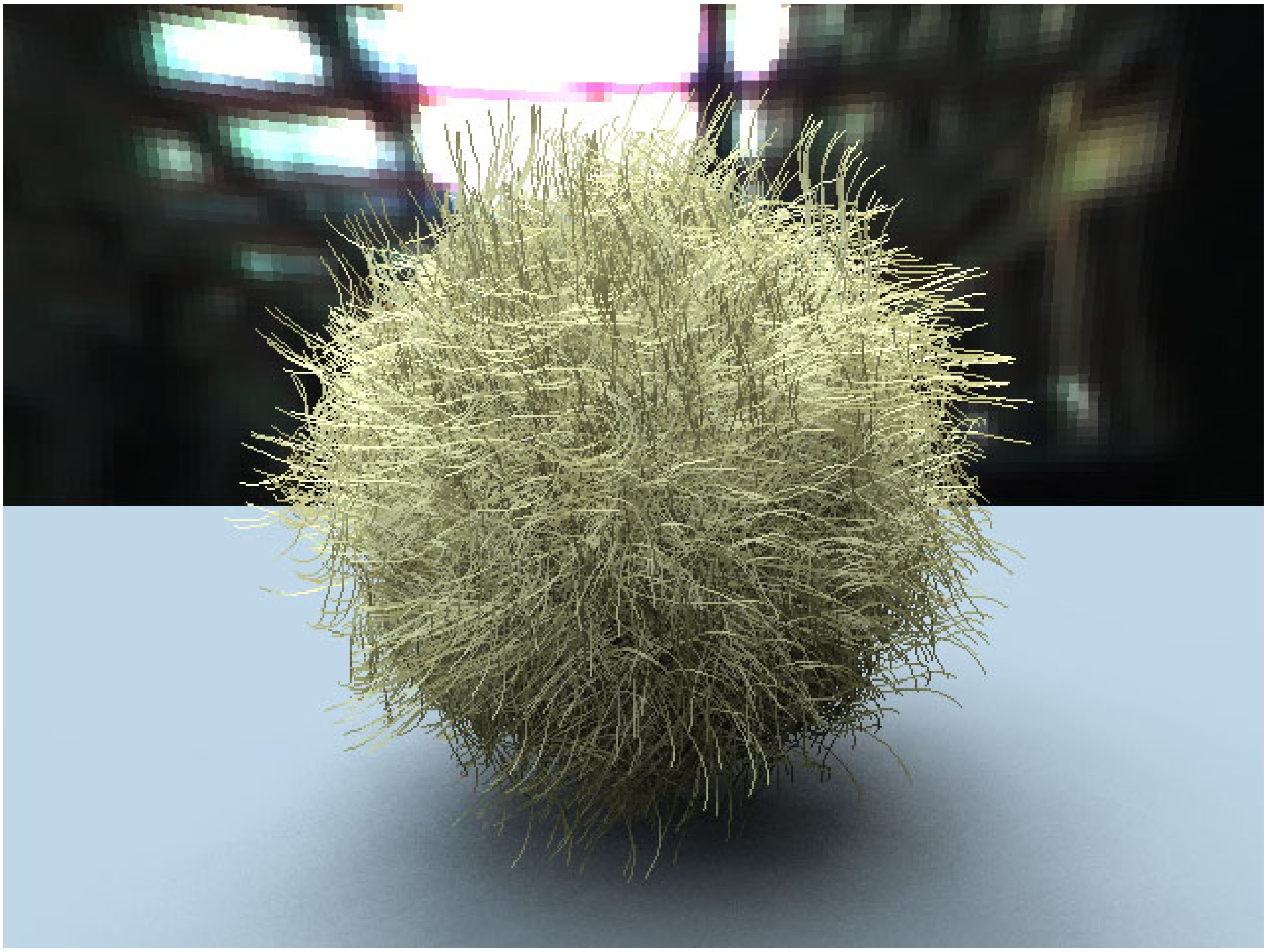


Figure 14: Our method by using Debevec's HDR map.

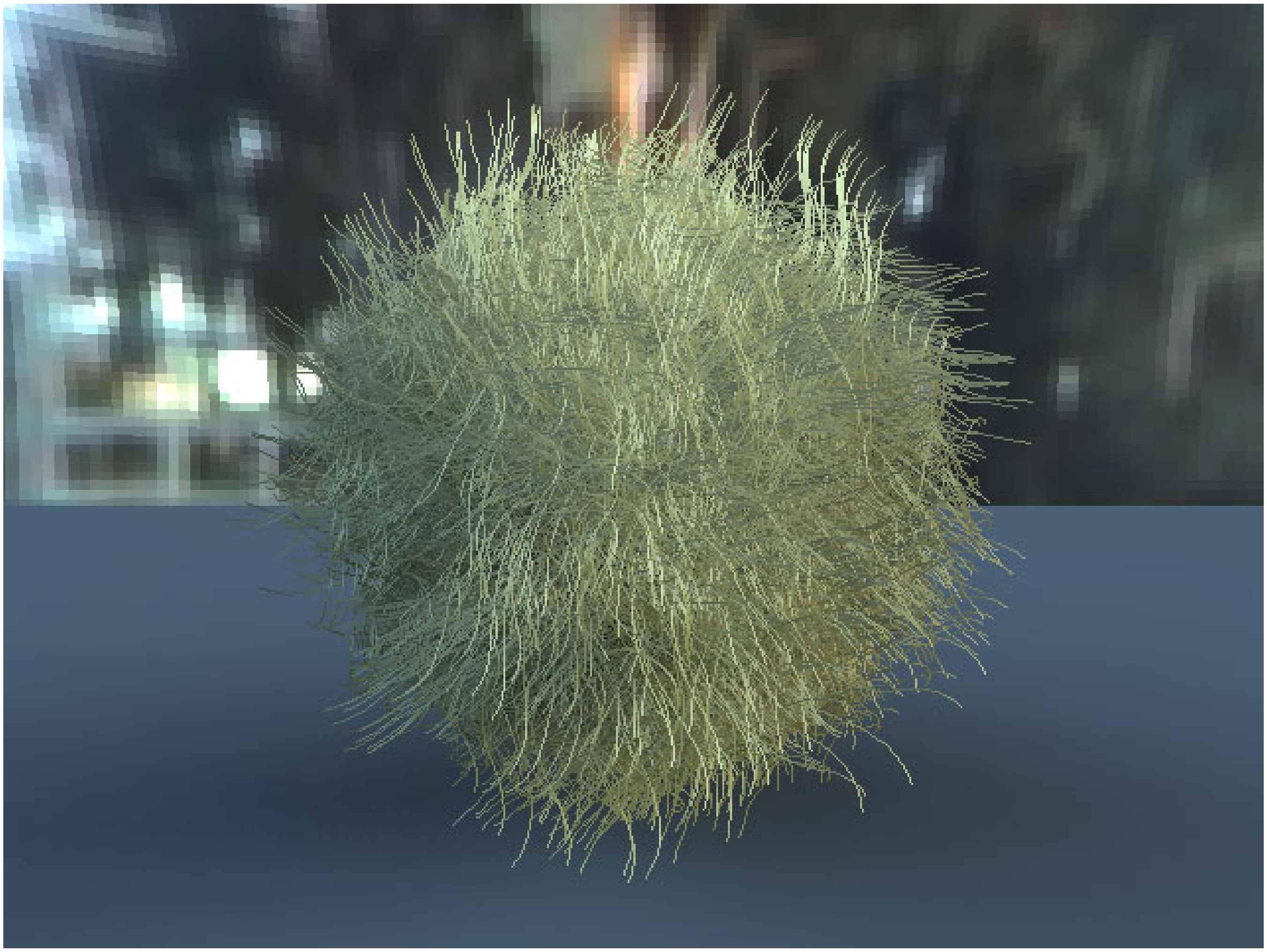


Figure 15: Our method by using Debevec's HDR map.

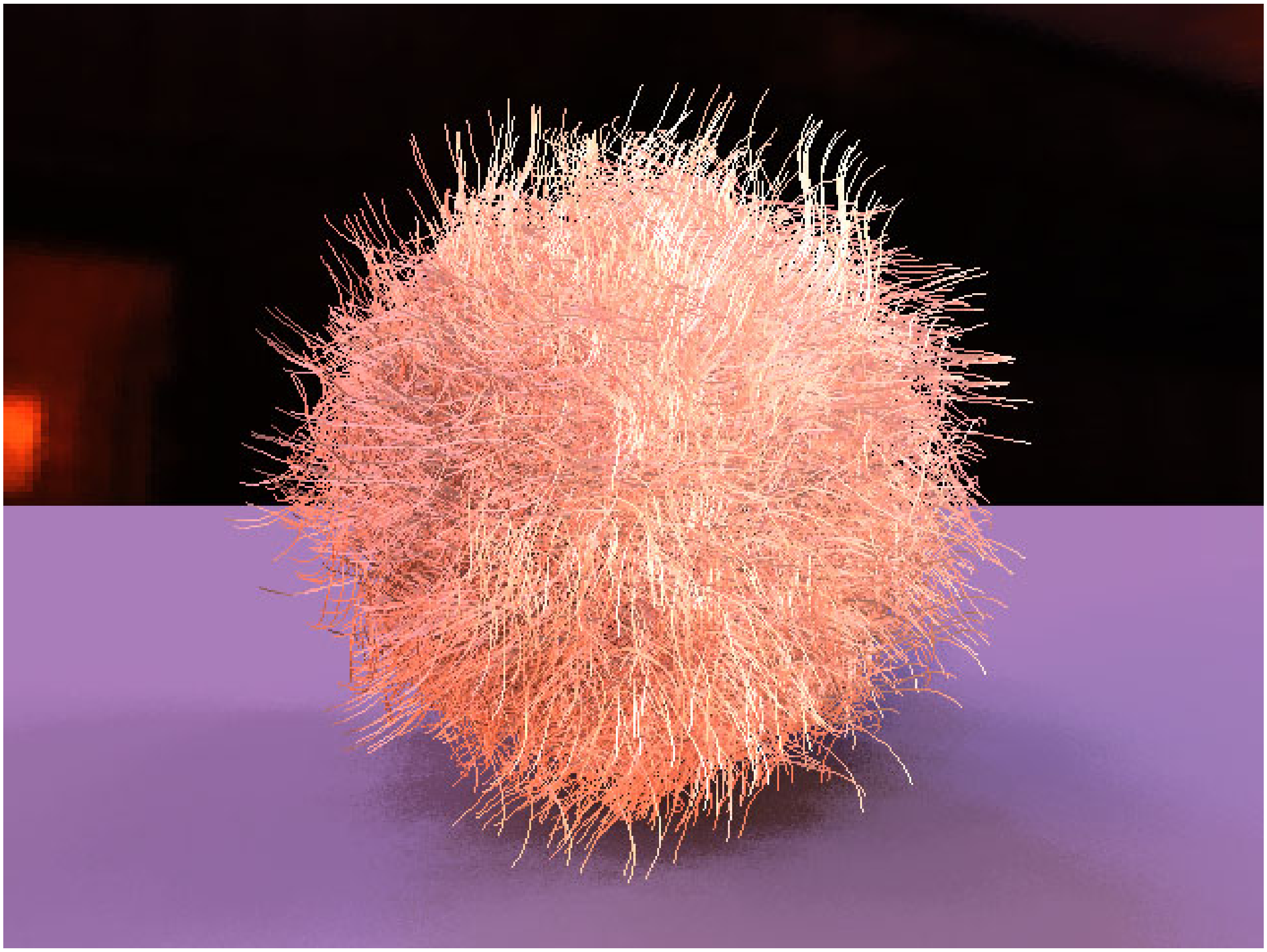


Figure 16: Our method by using Debevec's HDR map.

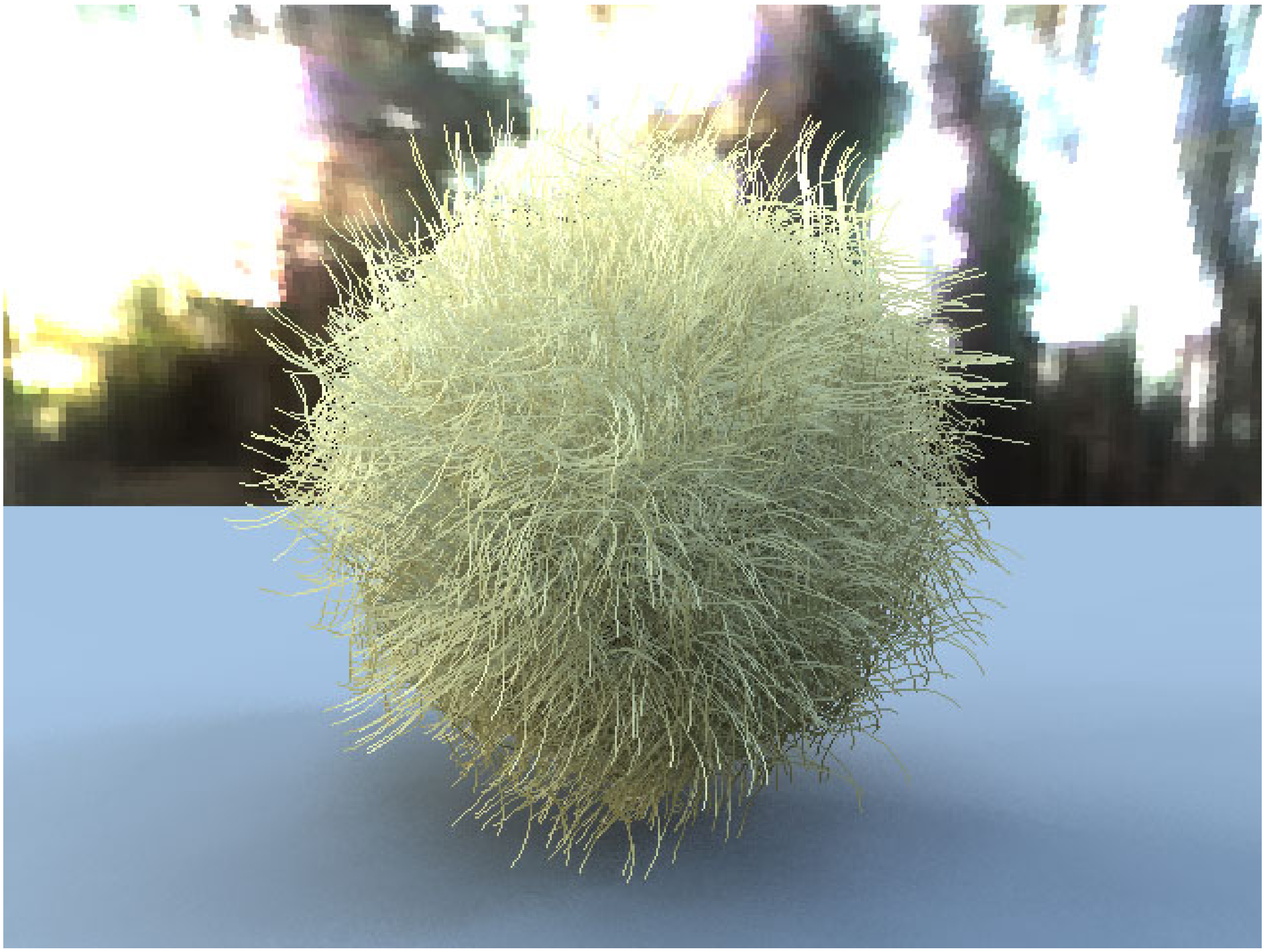


Figure 17: Our method by using Debevec's HDR map.

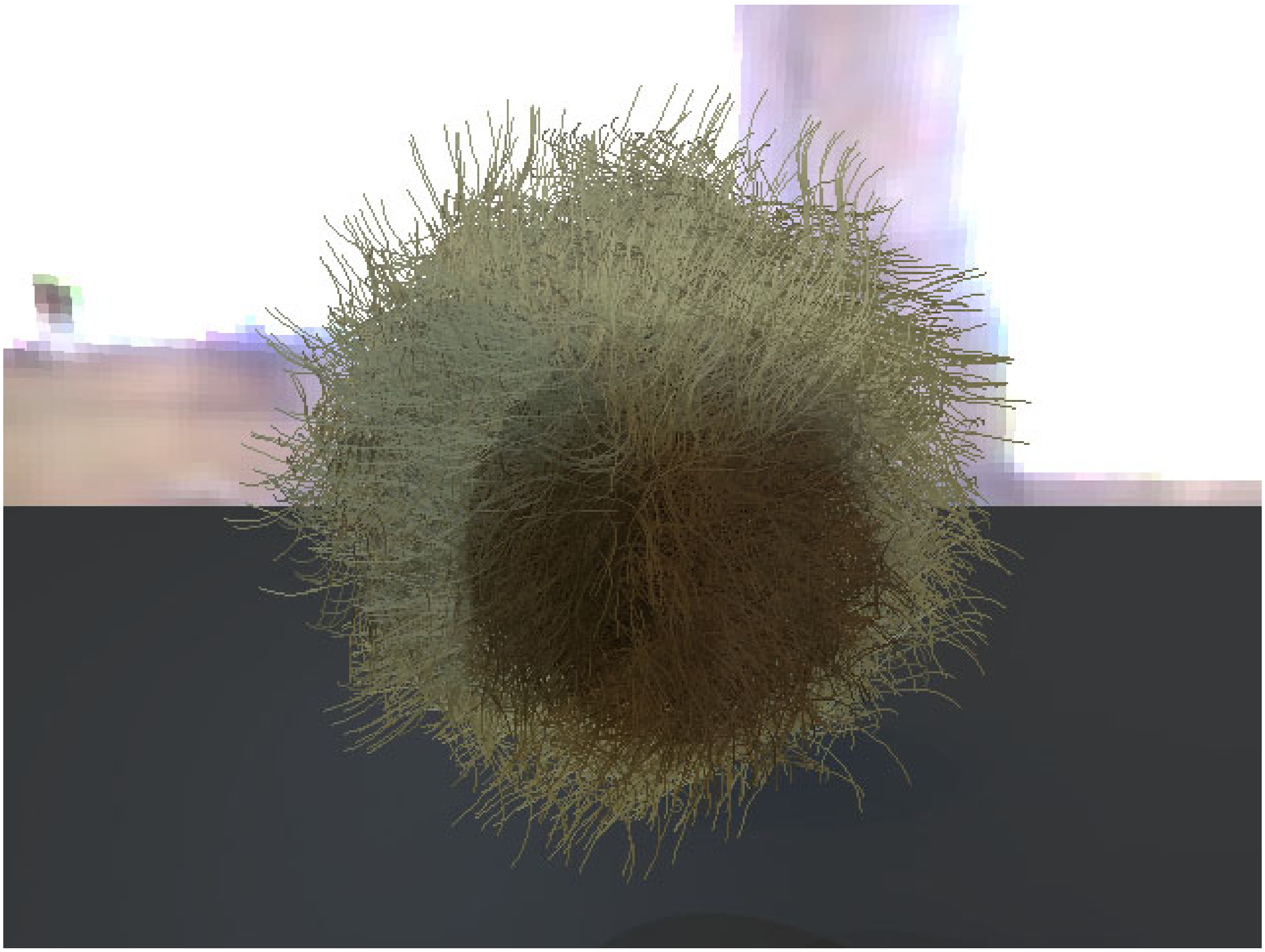


Figure 18: Our method by using Debevec's HDR map. This image is particularly interesting. Notice the shadow of sphere on hair.