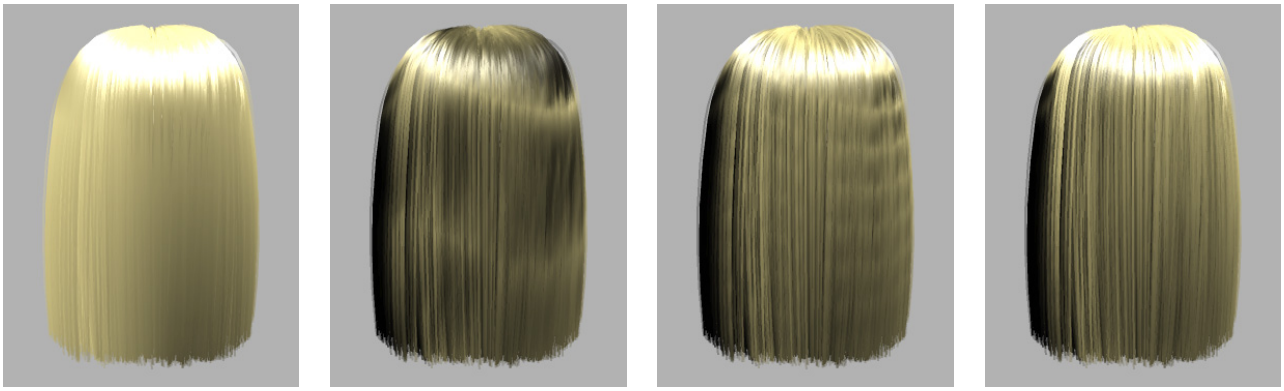
Deep Opacity Maps

Cem Yuksel
cem@cemyuksel.com
Department of Computer Science
Texas A&M University

John Keyser
keyser@cs.tamu.edu
Department of Computer Science
Texas A&M University



No shadows | Opacity Shadow Maps 4 layers | Opacity Shadow Maps 16 layers | **Deep Opacity Maps** 3 layers

Figure 1: Layering artifacts of Opacity Shadow Maps are apparent even with 16 layers, while Deep Opacity Maps can generate an artifact free image with only 3 layers.

Abstract

We present a new method for computing shadows from semi-transparent objects like hair. Extending the concept of opacity shadow maps, the deep opacity map method uses a depth map to obtain a per pixel distribution of opacity layers. This approach totally eliminates the layering artifacts of opacity shadow maps and requires much fewer layers to achieve high quality shadow computation. We provide qualitative comparisons to opacity shadow maps and give performance results. Our algorithm is easy to implement, fast, and memory efficient, enabling us to generate high quality hair self-shadows in real-time using consumer graphics hardware on a standard PC.

1 Introduction

Self-shadowing is an essential visual element for rendering semi-transparent objects like hair, fur, smoke, and clouds. However, for simple shadowing techniques, handling this transparency component is either inefficient or not possible. Various algorithms have been proposed to address this issue both for offline rendering [LV00] and interactive/real-time rendering [KN01; MKBvR04]. In this paper we present an algorithm that allows real-time rendering of human hair with dynamic lighting. Even though we focus on hair rendering, our method is applicable to rendering other semi-transparent objects.

Our deep opacity maps method combines shadow mapping [Wil78] and opacity shadow maps [KN01] to give a better distribution of opacity layers. We first render the hair geometry as opaque primitives from each light's view, recording the depth values on a shadow map. Next we render an opacity map from the light's view similar to opacity shadow maps. The novelty of our algorithm lies in the way that the opacity layers are distributed. Instead of using regular slices of the hair geometry in between two planes normal to the light direction, we use the depth information in the shadow map to create opacity layers that vary in depth from the light source on a per-pixel basis. This allows us to avoid the layering artifacts that are apparent in opacity shadow maps unless a very high number of layers are used. Moreover, far fewer layers are necessary to generate high quality shadows, since the shape of the layers are warped to coincide with the shape of the hair volume. Figure 1 shows a comparison of our deep opacity maps algorithm to opacity shadow maps. Layering artifacts in opacity shadow maps are significant when 4 layers are used (Figure 1c), and do not totally disappear even when 16 layers are used (Figure 1d). However, our method can produce an artifact-free image with only 3 layers.

In the next section we give an overview of the related work. Section 3 explains the details of our algorithm, Section 4 shows our results with comparisons to opacity shadow maps, and we conclude in Section 5.

2 Related Work

In this section we briefly overview related techniques for hair self-shadowing. For a more complete presentation of the previous methods please refer to [WBK*07] or [MKBvR04].

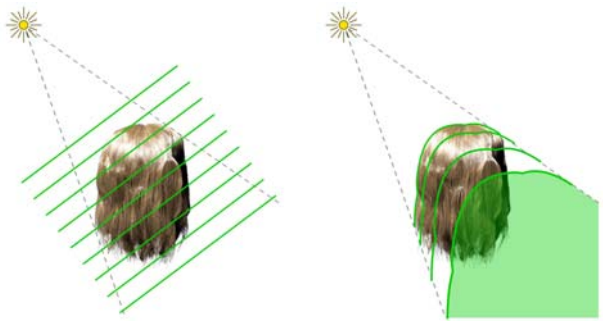
Most shadow computation techniques developed for hair are based on *Shadow Maps* [Wil78]. In the first pass of shadow mapping, shadow casting objects are rendered from the light's point of view and depth values are stored in a depth map. While rendering the scene from the camera view in the second pass, to check if a point is in shadow, one first finds the corresponding pixel of the shadow map that the point lies under, and compares the depth of the point to the value in the depth map. The result of this comparison is a binary decision, so shadow maps cannot be used for transparent shadows.

Deep Shadow Maps [LV00] is a high quality method for offline rendering. Each pixel of a deep shadow map stores a 1D approximate

transmittance function along the corresponding light direction. To compute the transmittance function, semi-transparent objects are rendered from the light's point of view and a list of fragments is stored for each pixel. The transmittance function defined by these fragments is then compressed into a piecewise linear function of approximate transmittance. The value of the transmittance function starts decreasing after the depth value of the first fragment in the corresponding light direction. The shadow value at any point is found similar to shadow maps, but this time the depth value is used to compute the transmittance function at the corresponding pixel of the deep shadow map, which is then converted to a shadow value.

Opacity Shadow Maps [KN01] is essentially a simpler version of deep shadow maps that is designed for interactive hair rendering. It first computes a number of separating planes that slice the hair volume into layers (Figure 3a). These planes are perpendicular to the light direction and are identified by their distances from the light source (i.e. depth value). The opacity map is then computed by rendering the hair structure from the light's view. A separate rendering pass is performed for each slice by clipping the hair geometry with the separating planes, and the hair density for each pixel of the opacity map is computed using additional blending on graphics hardware. The slices are rendered starting from the nearest slice to the light source in order, and the value of the previous slice is accumulated to the next one. Once all the layers are rendered, this opacity map can be used to find the transmittance from the occlusion value at any point using linear interpolation of the occlusion values at the neighboring slices. Depending on the number of layers used, the quality of opacity shadow maps can be lower than deep shadow maps, since the interpolation of the opacities between layers generates layering artifacts on the hair. This artifact can be reduced by using more and more layers, however artifacts remain visible unless a very high number of layers are used.

To bridge the gap between opacity shadow maps and deep shadow maps, [MKBvR04] proposed an approach that calculates different depths for the opacity layers per pixel. At runtime, for each pixel, separate rendering passes compute statistical information about the transmittance function. This statistical information is then used to determine a range in which to place opacity layers, and the individual hairs are clustered into a set number of opacity layers in that range during another render pass. An iterative process can be used to improve the positioning of layers and clustering of hairs into the layers.



(a) Opacity Shadow Map (b) Deep Opacity Map

Figure 3: Opacity shadow maps use regularly spaced planar layers. Our deep opacity maps use fewer layers, conforming to the shape of the model.

3 Deep Opacity Maps

Our deep opacity maps extend the idea of opacity shadow maps by warping opacity layers to the shape of the hair structure (Figure 3). We describe and evaluate the algorithm here.

3.1 Algorithm

Our algorithm has 3 steps: the first two prepare the deep opacity map, and the last renders the final image, using this map to compute shadows.

The *first step* prepares the separators between the opacity layers. We render a depth map of the hair as seen from the light source. This gives us, for each pixel of the depth map, the depth z_0 at which the hair geometry begins. Starting from this depth value, we divide the hair volume within this pixel into K layers such that each layer lies from $z_0 + d_{k-1}$ to $z_0 + d_k$ where $d_0 = 0$, $d_{k-1} < d_k$ and $1 \leq k \leq K$. Note that the spacing $d_k - d_{k-1}$ does not have to be uniform. The final shape of the separators between layers is not planar, but is similar to the shape of the hair structure (Figure 3).

The *second step* renders the opacity map using the depth map computed in the previous step. This requires rendering the hair only once. All computation occurs within the fragment shader. As each hair is rendered, we read the value of z_0 from the depth map and compute the depth values of the layers on the fly. We assign the opacity contribution of the fragment to the layer that the fragment falls in and to all the other layers behind it. The total opacity of a layer at a pixel is the sum of all contributing fragments. We represent the opacity map by associating each color channel with a different layer, and accumulate the opacities using additive blending on the graphics hardware. We reserve one color channel for the depth value, so that it is stored in the same texture with opacities. With a single color value, we can thus represent three opacity layers, and by enabling multiple draw buffers we can output multiple colors per pixel to represent more than three layers. Obviously, using more than three layers will also require multiple texture lookups.

Note that our algorithm uses the depth map only for computing the starting points of layers, not for a binary decision of in or out of shadow. Thus, unlike standard shadow mapping, deep opacity maps do not require high precision depth maps. For the scenes in our experiments, we found that using an 8-bit depth map visually performs the same as a 16-bit floating point depth map.

3.2 Advantages and Difficulties

The main advantage of our method is that by shaping the opacity layers, we eliminate visual layering artifacts and move interpolation between layers to within the hair volume, thus hiding possible inaccuracies. This also allows high quality results with far fewer layers. When hair density is mostly uniform (as in most natural hair models), a very small number of layers is sufficient (3 layers were enough in our test cases).

Since we require far fewer layers, all information can be stored in a small number of textures (for 3 layers, in a single texture). This makes our algorithm memory efficient and also reduces load on the fragment shader.

We require as few as 2 render passes to prepare the opacity map, and only one of these passes uses blending. Opacity shadow maps could be generated in a single pass given enough draw buffers. However, to achieve comparable results, opacity shadow maps would require more draw buffers than current hardware supplies.

Figure 2: Dark hair model for qualitative comparison. Layering artifacts of Opacity Shadow Maps generate dark diagonal stripes, while Deep Opacity Maps generate similar images with any number of layers.

Like opacity shadow maps, our algorithm does not have any restrictions on the hair model or hair data structure. Since it does not need any pre-computation, it can be used when rendering animating dynamic hair or any other semi-transparent object that can be represented by simple primitives.

One disadvantage of using a small number of layers is that it can be more difficult to ensure all points in the hair volume are assigned to a layer. In particular, points beyond the end of the last layer $z_0 + d_k$ do not correspond to any layer (shaded region in Figure 3b). We have a few options: ignore these points (thus, they will not cast shadows), include these points in the last layer (thus, they cast shadows on themselves), or ensure that the last layer lies beyond the hair volume. While the last option might seem "ideal," it can lead to an unnecessary extra layer that adds little visual benefit, at more computational cost. We found that the second option usually gave reasonable results, since the transmittance of light is often near zero at that point, anyway.

4 Results

To demonstrate the effectiveness of our approach we compare the results of our deep opacity maps algorithm to opacity shadow maps. We extended the implementation of the opacity shadow maps using multiple draw buffers such that the opacity map can be prepared in a single-pass, as opposed to the multi-pass implementation proposed in the original method [KN01]. We obtained our results using a standard PC with a 2.13GHz Core2 Duo processor and GeForce 7900 graphics card.

We used line drawing for rendering these models, and a Kajiya-Kay shading model [KK89] because of its simplicity. Antialiasing is handled on the graphics hardware using multi-sampling. The transparency effect is achieved by dividing the hair strands into 3 randomized sets, rendering each set separately, and compositing the resulting images to produce the final hair image.

For our test scenes we used three different hair models: *straight* (Figure 1), *dark* (Figure 2), and *curly* (Figure 4); represented by 160 thousand, one million, and 1.5 million line segments respectively. Figures 1, 2 and 4 show qualitative comparisons of our deep opacity maps to opacity shadow maps. As can be seen from these images, while our algorithm can produce an artifact-free image with only 3 layers, layering artifacts in opacity shadow maps are prominent even with 16 layers; they are even more prominent in animated sequences. This shows that the opacity shadow maps method needs more than 16 layers to produce high quality shadows (this is also suggested by other authors [MKBvR04]). On the other hand, adding more than 3 layers to our deep opacity maps does not have a significant improvement on the image quality in our test scenes. '

We provide performance results in Table 1. Generating the deep opacity maps takes more time due to the overhead of depth map generation and per pixel on-the-fly layer placement used in both opacity map generation and sampling. Notice that this overhead is quite small even when the number of layers for both methods are close. However, since opacity shadow maps require far more layers, we can conclude that for comparable quality, deep opacity maps are faster.

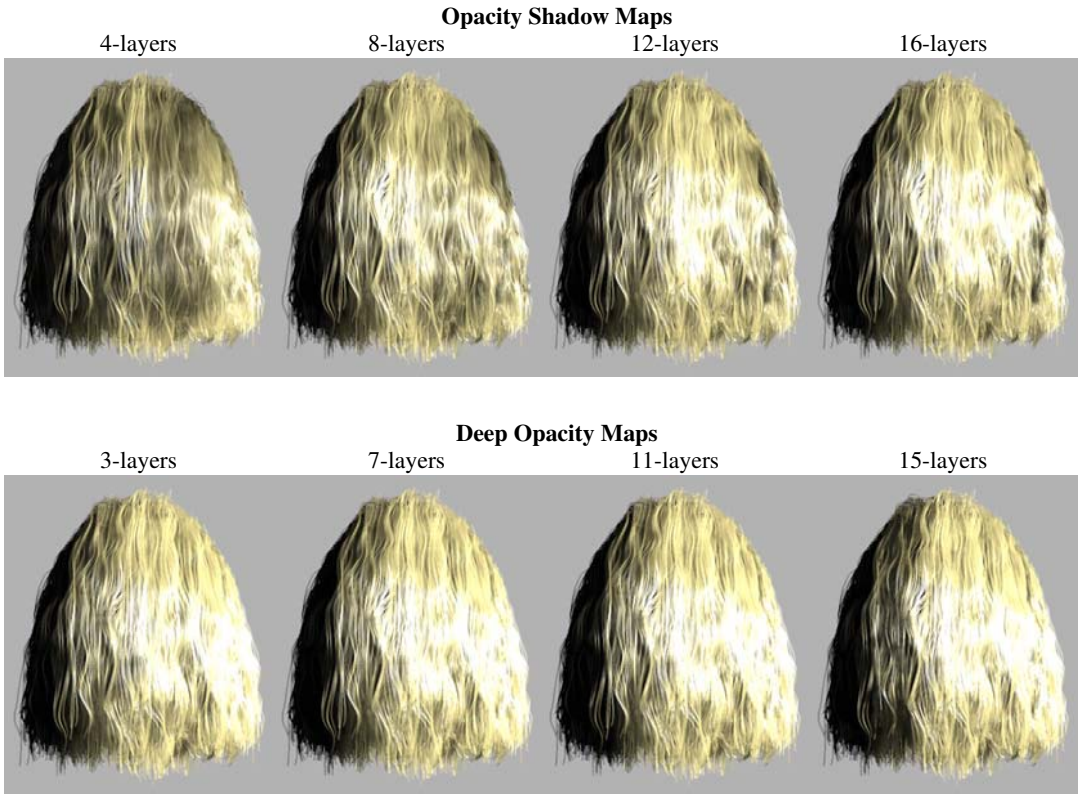


Figure 4: Curly hair model for qualitative comparison. Layering artifacts of Opacity Shadow Maps generate dark vertical stripes, while Deep Opacity Maps generate similar images with any number of layers.

Table 1: Time results in milliseconds

		straight	dark	curly
Opacity Shadow Maps	4-layers	11.9	23.6	29.2
	8-layers	22.1	42.9	41.2
	12-layers	30.3	57.7	53.2
	16-layers	39.6	72.9	65.1
Deep Opacity Maps	3-layers	14.8	31.6	39.2
	7-layers	22.7	44.4	47.2
	11-layers	31.8	60.9	61.2
	15-layers	42.3	80.9	80.5

5 Conclusion and Future Work

We have introduced the deep opacity maps method, which uses a depth map to achieve per-pixel layering of the opacity map for real-time computation of transparent shadows. Even though we restrict our implementation to real-time hair rendering, deep opacity maps may also be used to render other semi-transparent objects. Our results show that deep opacity maps are fast and can generate high quality transparent shadows with minimal memory consumption. We have compared our results to opacity shadow maps, and demonstrated how our approach eliminates layering artifacts that are prominent in opacity shadow maps unless a very high number of layers are used.

A possible future extension of our method is to handle opaque shadows from polygons within the deep opacity maps using the depth map component. However, this would not be a trivial extension, since in our approach depth map is only used for per-pixel layer placement, and the actual shadow computation is handled through the opacity map.

References

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

KIM T.-Y., NEUMANN U.: Opacity shadow maps. In *Proceedings of the 12th Eurographics Workshop on Rendering Techniques* (London, UK, 2001), Springer-Verlag, pp. 177–182.

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

MERTENS T., KAUTZ J., BEKAERT P., VAN REETH F.: A self-shadow algorithm for dynamic hair using clustered densities. In *Proceedings of Eurographics Symposium on Rendering 2004* (June 2004), pp. 173–178.

WARD K., BERTAILS F., KIM T.-Y., MARSCHNER S. R., CANI M.-P., LIN M.: A survey on hair modeling: Styling, simulation, and rendering. *IEEE Transactions on Visualization and Computer Graphics (TVCG) 13*, 2 (Mar-Apr 2007), 213–34. To appear.

WILLIAMS L.: Casting curved shadows on curved surfaces. In *SIGGRAPH '78: Proceedings of the 5th annual conference on Computer graphics and interactive techniques* (New York, NY, USA, 1978), ACM Press, pp. 270–274.