Optimizing fur rendering algorithms

Essay 2

Adam Alsegård*

TNCG13 - SFX: Tricks of the trade
Linköping University - Media Technology

November 28, 2017

Abstract

*Fur and hair rendering is very computationally expensive. This essay will explore two recent methods that strive to reduce the rendering cost for furry objects. The first method is proposed in **Cone Tracing for Furry Object Rendering** by Qin et al. [1]. Their method reduces the rendering time by improving a cone based ray tracing algorithm. The second method is presented in **An Efficient and Practical Near and Far Field Fur Reflectance Model** by Yan et al. [2]. They improve the local reflectance model for fur and introduces a multi-scale rendering solution where they combine near and far field computations. The essay ends with a discussion of the methods contribution to the field of fur rendering.*

I. Introduction

Among the most important features in avatar personalization we have fur and hair. Over the years researchers have developed multiple efficient techniques to render fur and hair photo-realistic, taking into account the complex visual effects including transparency, self-shadowing and multi-scattering. However these techniques are still very time consuming. This essay will compare two recent methods that reduces the computation time for rendering fur and hair.

First we have Qin et al. [1] that improved a cone-based ray tracing algorithm to significantly reduce the rendering time of furry objects. It is especially efficient for typical ray tracing effects as reflection and refraction as well as camera effects such as depth of field (DOF).

On the other side we have Yan et al. [2] who uses a global illumination model with path tracing. They have improved on a previous local reflection model for fur, also developed by Yan et al. [3], by reducing the numbers of lobes in the model. Furthermore they introduce a multi-scale rendering technique that switches resolution depending on how close the fur fibers are.

*adaal265@student.liu.se

II. Cone-based ray tracing

Ray tracing is a classic technique to render photo-realistic images, and there already exist ray tracing methods for fur. However, these methods have a problem with noise and aliasing, as well as a high rendering cost. The very thin fur fibers require extremely high supersampling rates to produce an aliasing-free image, especially when rendering with a short depth of field (DOF). The already vast amount of rays needed to produce the anti-aliasing sample gets even higher when fur transparency is calculated. The usual way to reduce the number of rays needed is to bundle them together, for examples as cones. This section will describe how Qin et al. [1] reduced the number of rays by using a cone based ray tracing algorithm.

By aggregating all sampling rays in a pixel as a single cone Qin et al. reduce the high supersampling rate drastically. Their approach is based on cone tracing for a specific type of scene primitive, i.e. fur fibers represented as a series of connected line segments with linearly interpolated per-vertex widths. Each fur fiber geometry is therefore a generalized cylinder with the connected line segment as its axis. Cones were chosen instead of other bundle representations as its shape more closely represents the image filter used

in downsamling anti-aliasing samples. Note that other parts of the scene were still rendered with regular ray tracing.

i. The challenges

The two main challenges Qin et al. focused on was the high cost of intersecting fur fibers with cones and how to handle transparency within each cone.

i.1 Intersecting fur fibers with cones

The proposed algorithm first constructs a bounding volume hierarchy (BVH) for all fiber geometry and then traverses the BVH to find all fibers that may intersect each cone. The projections of these fibers onto the image plane is then approximated as a set of ribbons (or quadrilaterals), each which corresponds to a line segment of a fiber. Finally they evaluate the intersection area of each ribbon with the cone, which suffices for further compositing and filtering computations. The following section describe this technique in more detail.

For each line segment they first construct an axis aligned bounding box (AABB) for the two spheres centered at the two ending vertices of the segment. Each sphere has a radius equal to the width value at the vertex. This AABB is the basic geometry when building the BVH. During ray tracing, for each cone, the BVH nodes are first checked against the ray cone using a fast separating axis theorem test to exclude many non-intersected nodes. For each AABB passing the test they then construct a ribbon for its corresponding line segment to approximate the projection of the fiber geometry bounded by the AABB to the image plane.

Depending on the type of ray the cones are formed differently. To simulate DOF Qin et al. uses an envelope of two cones, as illustrated in Figure 1. The first cone is formed by connecting the aperture center and the circumcircle of the focal plane projection of the pixel (red lines) and the other is formed by connecting the projected pixel center on the focal plane and the aperture circle (blue lines). When a fiber segment is projected onto the reference plane of the DOF cones, the average depth value of the segment is used to compute the sizes of the cross-sections of the DOF cones. The depth values are computed with respect to the view point.

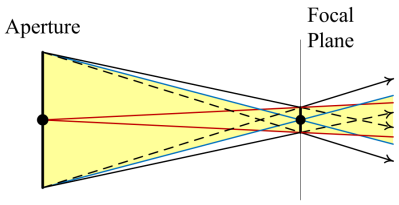

Figure 1: *Illustration of how the DOF cones are constructed.*

Reflection and refraction on the other hand cannot be directly applied to cones as rays within each individual cone may hit different reflective/refractive objects and diverge. Therefore a shading metric is used to cluster supersampled reflection/refraction rays that happen to be coherent and generate an aggregate cone for each group. The direction and origin of the new cone is determined by averaging the origins and directions of all rays in the group. A few more checks and a threshold makes sure that the generated cones does not grow too big to handle.

i.2 Handle transparency within cone

Qin et al. solved the expensive composition computations by approximating away in-cone variations of shading, opacity and occlusion. They assume that the depth order required for compositing transparent samples does not change within each cone and therefore perform the compositing on a per-cone basis. To facilitate such a compositing order they convert each cone-ribbon intersection into a single effective opacity according to the intersection area and an aggregated shading by further assuming shading and opacity are smooth within each cone.

ii. Results

With their improvements Qin et al. manage to reduce the rendering time by a factor of 3-6x depending on the scene. The image errors caused by the approximations can be reduced by increasing the supersampling rates or decreasing the cone size, which of course then

yields a longer rendering time. A comparison of images were showed to a group of 35 individuals with different experience of CG. About half of the subjects preferred the result of the new algorithm and the conclusion was that they were able to generate visually pleasing results without any supersampling.

The algorithm was implemented on a GPU with CUDA and can easily be integrated into a ray tracing framework. It handles view rays, reflection, refraction and shadow rays but not multiple scattering among fibers. As for the camera effects DOF can be accelerated but not motion blur, as it was found too difficult to formulate the 4D problem as 3D cones. But the same effect can be achieved by combining their method with shutter time supersampling.

The performance benefit is more significant for thin fibers (4-6x) than for wide fibers (1.5x). The approximation for the projected fiber geometry could generate self-intersecting ribbons if the fiber width is comparable to or larger than the line segment length, making the intersection areas incorrect.

III. Fur reflectance model

Physically based fur rendering is difficult. It is complicated due to the complex scattering paths through the medulla (central region of fur fibers). This leads to extensive pre-computation and limits to near field rendering only. Yan et al. [2] have developed several optimizations that improve the efficiency and generality of of these calculations without compromising the accuracy, leading to a practical fur reflectance model. They also propose a key method to support both near and far field rendering, that allows for a smooth transition between the two.

The specific contributions made by Yan et al. were a simplified reflectance model, improved accuracy and practicality, an analytic near/far field solution and a significant speed up. This section will go over these contributions in more detail.

i. Simple reflectance model

Most fur reflectance models represents individual fur fibers as cylinders. In the rendering equation, instead of BRDF, they use a BCSDF (Bidirectional Curve Scattering Distribution Function) to characterize how fur fibers scatter light,

$$L_r(\omega_r) = \int L_i(\omega_i) S(\omega_i, \omega_r) cos\theta_i d\omega_i \quad (1)$$

where ω_i and ω_r are the incident and outgoing directions and L_i and L_r are the incoming and outgoing radiance. S represents the BCSDF. This equation is then parameterized and separated into longitudinal and azimuthal events.

The local illumination model used by Yan et al. builds upon a double cylinder model representing the cuticle-cortex structure of fur fibers. They have unified the cortex and the medulla's indices of refraction (IORs) removing most of the complicated types of light interactions between them. This means that instead of 11 light lobes they now only need 5. More specifically R, TT and TRT from traditional hair rendering is kept, while only two extensions TT^s and TRT^s are added for the scattering. In these notations R stands for Reflectance and T for Transmission. Uppercase letter correspond to the outer cylinder and lowercase to the inner cylinder. The authors concluded that many of the more complex types, such as TrT and TtrtT, were often too weak to be observed and could therefore be left out. An illustration of how the paths are computed can be seen in Figure 2 and an example of the contributions of the different light paths can be seen in Figure 3.

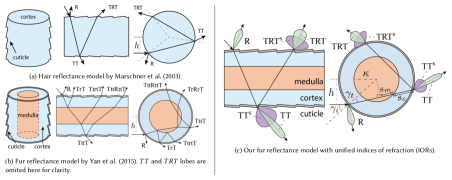

Figure 2: *Examples of different reflectance models and their light paths.*

One more benefit that the reduced number of lobes introduces was that the light paths now are the same as for hair and therefore it unifies fur and hair rendering techniques and thus the method can be implemented into an existing hair rendering systems with minimal effort.

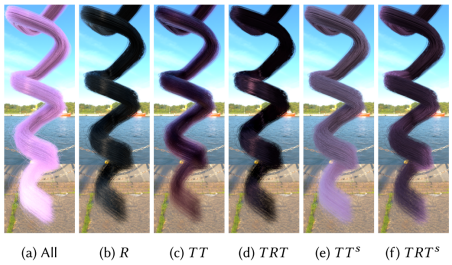

Figure 3: *The contributions of the different light paths.*

ii. Improved accuracy & practicality

The accuracy of the model was improved by considering different roughnesses of azimuthal and longitudinal sections (see Figure 4 for an illustration). Moreover they found that the medulla is not purely scattering but also absorbs light.

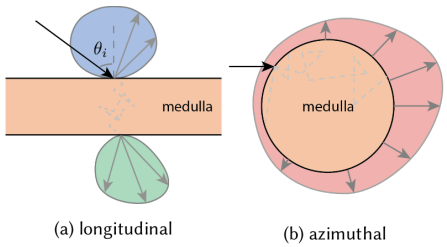

Figure 4: *Illustration of longitudinal and azimuthal scattering.*

As an addition they also introduced a compression scheme. They treated the precomputed longitudinal and azimuthal scattering profiles as 4D tensors and used a tensor decomposition technique to compress them. Tensor decomposition is the high dimensional analogue to a 2D singular value decomposition (SVD). Together with an alternating least squares algorithm (CP-ALS) they could compress 600MB of raw data for medulla scattering to 150KB of storage size. It took less than one minute to decompose with a maximum of 500 iterations. They found that a rank up to 16 was good enough to accurately capture the complex shapes of the data.

iii. Analytic near/far field solution

By exploiting piece-wise analytic integration, the proposed method enables a multi-scale rendering scheme that transitions between near and far field rendering smoothly and efficiently for the first time.

The model starts with analytic near field computations, as opposed to implicit ray tracing required previously. The near field model is thereafter integrated over the azimuthal section by partitioning the range of integration into a few segments. The partitioning is done quadratically to make up for the more rapid change caused when the incident position is further away from the center. For un-scattered lobes 5 segments are enough in practical renderings while 8 segments generate indistinguishable scattering profiles. For scattered lobes, since they are even smoother, 4 segments are good enough throughout the computations. Finally Yan et al. integrate for each segment using piece-wise linear approximation.

iii.1 Multi-scale integration

The far field approximation is accurate when hair or fur fibers are thinner than a pixel. When viewed up close this will however produce a ribbon-like appearance and therefore near field rendering is required up close. This leads to a multi-scale BCSDF model. Yan et al. [2] uses each pixel's coverage on fur fibers to decide the range of azimuthal offset and integrates per pixel instead of per fiber. The azimuthal scattering profile for the integration is given by

$$N(\phi) = \frac{1}{h_2 - h_1} \int_{h1}^{h2} A(h) * D(h, \phi) dh \quad (2)$$

In the equation $1/(h2 - h1)$ guarantees energy conservation. When a pixel fully covers the entire azimuthal section the equation degenerates to far field case and when h_1 and h_2 are infinitesimally close it becomes a near field scattering representation. The pixel coverage is calculated by assuming that the pixels are round instead of square and using the pixel's diameter to tell its hit point in world coordinates. This technique is very similar to ray differentials (which is used in the cone tracing algorithm). The projected pixel at the hit point then becomes a disk, facing along the camera's look-at direction. Last step is to

project the disk towards the hit fiber and compare the fiber's radius to the pixels coverage.

$$C_{fiber} = (\omega_{fiber} * \omega_{lookat})C_{hit}/r_{fiber} \quad (3)$$

This leads to a ray-cylinder intersection. However, unless viewed extremely close, we can replace ω_{fiber} by ω_{camera} and simplify the equation somewhat.

iv. Results

The proposed model gives a 6-8x speed up over Yan et al. previous work [3], when using the multi-scale algorithm. It also converges twice as fast. However, the results are slightly more blurred longitudinally, indicating that a larger longitudinal roughness is fitted. Both this and all previous methods still have a problem with some of the benchmark objects (such as human hair). A test on a human hair model however showed the importance of medulla even for human hair which means that the multi-scale integration benefits both hair and fur rendering.

By unifying the IORs of cortex and medulla, the simplified model is capable of representing complex scattering within hair and fur fibers with only 5 lobes. By introducing medulla's absorption and different longitudinal and azimuthal roughness, and using tensor approximation to minimize the storage overhead, the proposed model achieves both accuracy and practicality. Moreover the analytic integration scheme for efficient far field approximation and the multi-scale extension unifies hair and fur reflectance models, as well as near and far field rendering schemes.

IV. Summary

Fur and hair rendering is an advanced area. Many of the presented equations in the papers require a lot of previous knowledge to fully understand and I feel that I have only scratched the surface of this field. Both of the presented algorithms builds upon dozens of previous papers and to explain them thorough you would need hundreds of pages, which this essay does not have. As such I have tried to focus on the main contributions and how these can be implemented in an already existing system.

This is also one of the pros for both papers. They both have a section explaining all non-trivial implementation specifics so their methods can be easily integrated into an existing system. For example a SFX company like MPC could easily integrate these methods into their existing fur and hair renderer *Furtility* and it would make a direct improvement.

The two papers are trying to achieve the same goal, to render fur photo-realistically for a lesser cost, but they do it in two completely different ways. While the cone based ray tracing cannot handle inner material scattering that is all the second paper is about. However, the biggest contribution in my mind is the near/far field transition. It lowers the cost tremendously and it will probably have a more lasting effect in the field than the rest.

References

[1] Hao Qin, Menglei Chai, Qiming Hou, Zhong Ren, and Kun Zhou. Cone tracing for furry object rendering. *IEEE Transactions on Visualization and Computer Graphics*, 20(8):1178–1188, August 2014.

[2] Ling-Qi Yan, Henrik Wann Jensen, and Ravi Ramamoorthi. An efficient and practical near and far field fur reflectance model. *ACM Transactions on Graphics (Proceedings of SIGGRAPH 2017)*, 36(4), 2017.

[3] Ling-Qi Yan, Henrik Wann Jensen, Chi-Wei Tseng, and Ravi Ramamoorthi. Physically-accurate fur reflectance: modeling, measurement and rendering. *ACM Transactions on Graphics (TOG)*, 34(6):185, 2015.