This paper has been published in : *Graphics Gems, vol 4, p363-366, Eds Paul Heckbert, Academic Press, 1994*

A Fast Alternative to Phong's Specular Model

Christophe Schlick

LaBRI,[1] *351 cours de la libération, 33405 Talence (France)*
schlick@labri.u-bordeaux.fr

Despite its known faults (no physical validity, no energy conservation...) Phong's illumination model [1] is still the most popular reflectance model in computer graphics. The most time consuming point in that model is the computation of the specular term which involves an exponentiation. A fast, simple formula is proposed here to replace that exponentiation. The original expression of the specular term S_n in the Phong reflectance model is:

$$S_n(t) = t^n$$

where $n \in [1, \infty)$ is a parameter that controls the highlight size and t is the cosine of the angle either between the viewing direction V and the reflected light direction R [1] or between the normal vector N and the bisector vector H between the light and the viewing direction [2]. The left part of Figure 1 shows the shape of $S_n(t)$ for different values of n.

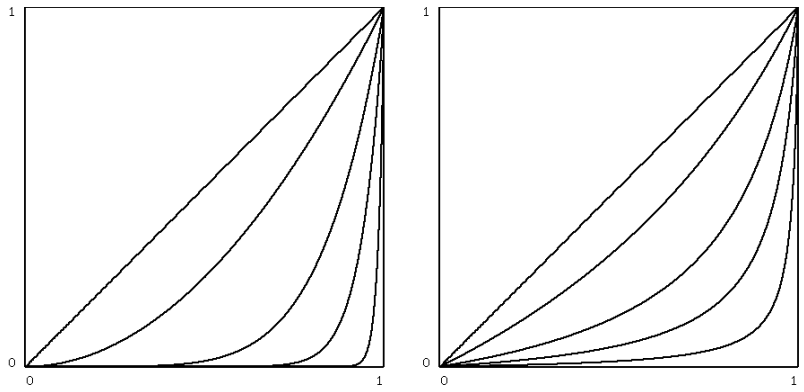

Figure 1: *$S_n(t)$ for $n = 1, 2, 6, 16, 64$.* LEFT : *Phong model,* RIGHT : *Alternative model.*

Previous Work

Several solutions have been proposed to reduce the cost of the exponentiation required by the Phong model. The most straightforward techniques for exponentiation, computing $\exp(n \log t)$, or multiplying t by itself $n-1$ times, are often too slow. A first alternative is that when the exponent n is a power of two, t^n can be evaluated by $\log_2 n$ successive squarings. This trick can be generalized to arbitrary integer exponents [6, p. 441]. Another solution is to tabulate $S_n(t)$ with a sufficient set of values and interpolate the missing ones [3]. When using a linear interpolation, the computation cost of such a technique is 1 addition, 1 subtraction, 2 multiplications, 1 integer part extraction and 2 table accesses. But linear interpolation often creates visible Mach banding that can only be eliminated by taking larger tables or higher order interpolations, leading to a memory/speed tradeoff. Moreover, a different table is needed for each value of parameter n, therefore the memory cost cannot be neglected for scenes with many different objects.

Another solution is to replace the function by its Taylor or Chebyshev approximation, in order to replace the exponentiation by a polynomial [5]. Such a technique works well for small values of n, but when n increases, keeping good accuracy rapidly requires high degree polynomials (and thus involves many multiplications).

[1] Laboratoire Bordelais de Recherche en Informatique (*Université Bordeaux I* and *Centre National de la Recherche Scientifique*). The present work is also granted by the *Conseil Régional d'Aquitaine.*

A last solution has been presented in [4] in which $S_n(t)$ is approximated by a piece-wise quadratic function. Unfortunately, the quadratic function acts directly on the angle and not on its cosine, therefore a call to `acos` (almost as expensive as an exponentiation) is needed.

Description

As previous papers have observed, there is no need for great accuracy when approximating $S_n(t)$, since the Phong model is totally empirical and is not intended for physical rendering: its only purpose is to give a visual impression of specularity by adding highlights on objects. Such highlights usually facilitate the understanding of a three dimensional scene by giving additional information about the curvature of an object. Therefore, every function that evokes a similar visual impression can be used instead of the exponentiation. We propose here such an alternative to the original formula:

$$S_n(t) = \frac{t}{n - nt + t}$$

In fact, several expressions with equivalent costs (1 addition, 1 subtraction, 1 multiplication and 1 division) can be found. This one has been chosen because the approximation of the Phong model is particularly close in the neighbourhood of $t = 1$ — where the values of $S_n(t)$ are high — instead of the neighbourhood of $t = 0$ as with classical approximation techniques.

The new formulation compares favorably with previous work. Compared to an implementation using Taylor or Chebyshev approximation, both the memory and the computation costs are lower. According to the architecture, the speed is more or less equivalent to look-up tables with linear interpolations (it depends on the cost of the division compared to the cost of the multiplication for the given architecture). But compared to look-up tables, the new implementation is not prone to Mach banding (all derivatives of the function are continuous) and does not require memory at all (only n has to be stored). Therefore, the formulation is particularly well adapted for hardware implementations.

The right part of Figure 1 shows the shape of $S_n(t)$ for different values of n. Compared to the left part, one can see that the decrease of the function starting from $S_n(1) = 1$ is slower as t decreases. Visually, it means that the main difference between the two models appears is that the highlight frontiers are broader with the new model than with the original one.

Bibliography

[1] B.T. Phong, *Illumination for computer generated pictures*, Communications of the ACM, v18, n8, p311-317, 1975.

[2] J.F. Blinn, *Models of light reflection for computer synthesized pictures*, Computer Graphics, v11, n4, p192-198, 1977.

[3] G.F. Bishop and D.M. Weimer, *Fast Phong Shading*, Computer Graphics, v20, n4, p103-106, 1986.

[4] A.A. Kuijk and E.H. Blake, *Faster Phong Shading via Angular Interpolation*, Computer Graphics Forum, v8, n4, p315-324, 1989.

[5] P. Poulin and A. Fournier, *A Model for Anisotropic Reflection*, Computer Graphics, v19, n3, p15-21, 1990.

[6] Donald E. Knuth, *The Art of Computer Programming, vol. 2, Seminumerical Algorithms, 2nd ed.*, Addison-Wesley, 1981.