**Specular Fading over Distance on Wrinkled Surfaces Applied for Game Engines**

Omitted for Blind Review

Omitted for Blind Review

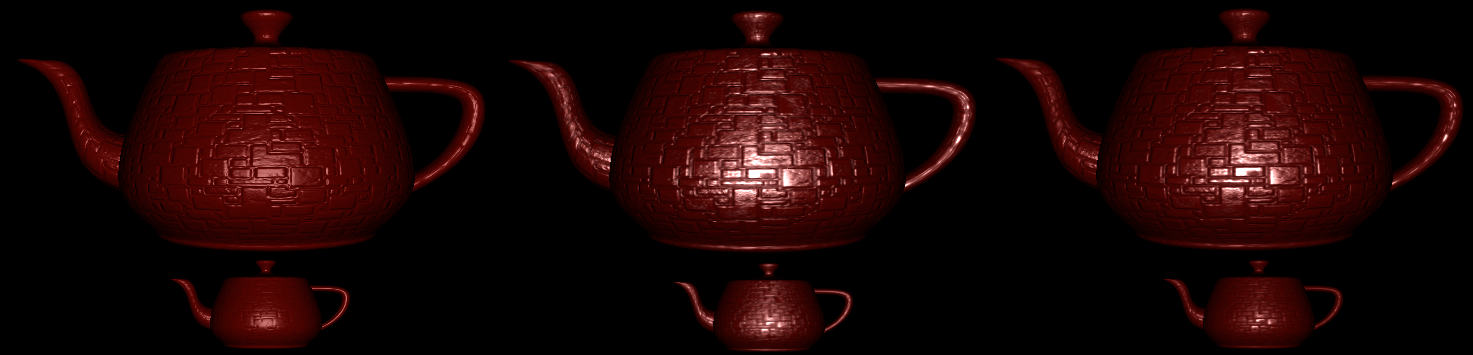


Figure : Bump mapping with Blinn-Phong, LEAN Mapping and Specular fading on a teapot..

**Abstract**

In 1978, James Blinn introduced a new way to simulate surface roughness. Many changes were made in the proposed approach in order to allow greater performance and are commonly used nowadays in video games, allowing a richer visual experience without the usage of highly detailed meshes. However, seeing from afar, the bumps become blurred and can disappear making the surface duller. To fix this issue, a new approach was developed recently and the result if satisfying. However, the specular hightligh intensity made by the small bumps is unaffected by the distance. One way to reduce the specular is by using fog techniques, but since most algorithms changes only the alpha factor, the specular shape remains the same. To correct this issue, this paper proposes an alternative approach, that aims to solve this specular highlight intensity by applying a new term during the calculation of the specular component.

**Keywords**: bump-mapping, normal-mapping, lean, mapping, specular ilumination.

**Authors’ contact**:

Omitted for Blind Review.

# Introduction

To create the illusion of certain surfaces or surface structures, artists usually create bump maps. In many cases these bump maps contain random bumps to better simulate rough surface materials at close range. The surface of these perceived roughness should be the same when displayed at different distances to the viewer. However, the shading of these bump, which are mapped to surfaces, are often inconsistent across different levels of detail and distance, because the shading model at the coarsest level of detail does not correspond to the mapped bump at finer level of details. As a result, the bump map is not consistent with the shading model.

Bump mapping was originally introduced by Blinn [Blinn 1978]. He demonstrated how to simulate wrinkled surfaces by only perturbing the normal vector without changing the underlying surface itself. The new perturbed normal is used for lighting calculations instead of the original normal. For about thirty years, the bump mapping introduced by Blinn has been an effective method for adding apparent detail to a surface. Programmable GPUs made this technique available for real time rendering, being a common feature in almost any 3D game.

On this paper, we use the term bump mapping to refer to the original height texture that defines surface normal perturbation for shading, introduced by Blinn, and the more common normal mapping, in which the texture holds the actual perturbed surface normal.

The hardware bump mapping is done by providing per-pixel operations at the pixel shader stage. Because such per-pixel operations are very flexible, they allow the usage of complex reflection models [Kautz and Seidel 2000] instead of only bump maps with diffuse and specular reflections using a *Lambertian* reflection model or *Blinn-Phong* model [Blinn 1977][Heidrich and Seidel 1999].

Unfortunately, bump mapping has serious drawbacks with filtering and antialiasing. When the bumps are viewed at a distance, the standard MIP mapping technique for bump map can work well for diffuse shading [Kilgard 2000]. However, it fails to capture changes in specularity. When looking far away, the result will be a shiny and bump-less surface, appearing as if it were duller. To correct this issue, a new mapping approach was developed [Olano and Baker 2010]. With this approach, the bump is preserved at high distances. Nevertheless, the specular highlights produced by these small bumps are also visible, making the surface artificially shiny.

This paper proposes the introduction of a fog-like effect in order to reduce the specular highlight according to the distance of the object. For this aim, we use the LEAN mapping in order to avoid the blurring of the bump surface and propose a modification to make the necessary change in the specular highlight, reducing the shiny effect. Note that this only affect the specular term, so it can be used in conjunction with fog algorithms, making so that with the increase of distance, the object will suffer light scattering and gradually lose the specular highlight.

This paper is organized as follows: Section 2 presents some ground basement of previews works in the area, as well as an overview of the proposed method. Section 3 presents our modified method. Section 4 show the results of the method and Section 5 show some implementation details. Finally, Section 6 presents the conclusion of this work.

# Related Work

There are many approaches to represent bumps in an object's surface. One of the first well established models originated from Blinn [Blinn 1977], which uses a simple shading equation based on microfacets. That model is a modification of the Phong model [Phong 1975], enhancing its visually appealing for small wrinkled materials. After this proposal, many other ways to represent surface bump were developed with better results and more flexibility. One of such is the storage of the surface normal in texture maps [Heidrich and Seidel 1999] [Kautz and McCool 1999] to render diffuse and specular reflections from the small bump in the surface. Another model that is based on microfacets is Cook-Torrance [Cook and K. E. Torrance 1982], which also adds shadowing and a Fresnel term to make it more realistic. Ward [Ward 1992] introduced an anisotropic BRDF model that is based on an anisotropic Gaussian microfacet distribution and Ashikhmin [Ashikhmin et al. 2000] have developed a way of generating reflection models from arbitrary normal distributions.

In this section, after briefly present the basics of Bump Mapping, we discuss about the Normal Mapping and the Linear Efficient Antialiased Normal Mapping (LEAN) model, in which this work was based on. Since this paper proposes an improvement of the LEAN model, we will also present in more details the Distance Fog, a technique used to enhance the perception of distance by simulating fog and light scattering, causing distant objects to appear with lower contrast.

## Bump Mapping

In 1978, Blinn [Blinn 1978] first proposed the bump mapping as a way of adding details in the object's surface by using a texture in order to define the changes of the object's underlying surface. The bump maps have the advantage of being able to extend to true displacement maps, but doing so requires more computing operations, such as the partial derivatives at shading time. Figure 2 illustrates the process of generating a wrinkled surface using bump map and Figure 3 an example of bump mapping. Because of this extra computing, most recent approaches use normal maps [Cohen et al. 1998], which is defined in the surface tangent space [Peercy et al. 1997].

However, a problem with highlight aliasing was noticed by Williams [Williams 1983] during texture filtering, which was identified by Kajiya [Kajiya 1985] as part of a hierarchy of scales from surface changes from bumps to BRDF. Many techniques that try to solve this problem were proposed, but they require too many pre-computation, such as [Cabral et al. 1987][Becker and N. L. Max 1993][Westin et al. 1992].

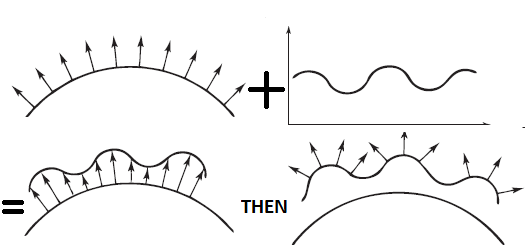


Figure 2: Bump Mapping. Smooth Surface + Wrinkle Function = Wrinkled Surface and then perturbed Normals. Figure adapted from [Blinn 1978].

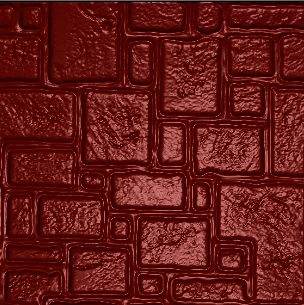


Figure : End result of Bump Mapping

## Normal Mapping

Because of the computational restrictions for the original Bump Mapping, a new technique was created. The normal mapping [Cohen 1998] replaces the object's normal entirely by doing a look-up into a normal map, while bump mapping changes the existing surface normal. For that, the normals of the normal map are stored in tangent space. By using normal maps instead of bump maps, the perturbation is no longer limited by object space. Because of this generality, it is possible to enhance visual quality by mapping the normals from a high-resolution surface and store as a texture map. Figure 4 illustrates a typical example of Normal Mapping.

To generate a texture map containing the new surface normal, each vertex normal from the high-resolution surface is mapped to a texel and will correspond to a point in the low resolution surface. Figure 5 illustrates the usage of such technique for a model using only 975 triangles. The left model is using a normal map, previously created with a higher resolution surface and applying in a less detailed mesh, and the right is using per-vertex normal..

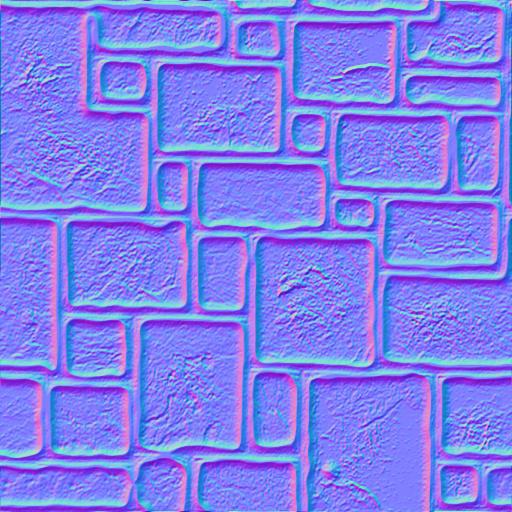


Figure : Typical example of Normal Map



Figure : Normal map comparison. Figure adapted from [Cohen et al. 1998].

It is also possible to achieve bump mapping using Normal mapping as an offline process, evaluating the perturbed normal for every texel in the normal map that corresponds to a surface position. A downside is that such object space normal maps depend on the used surface for bump. In contrast, when bump mapping, the height map is independent of bumped surface and can be used to perturb the normals of any other surface. Clearly, this very strong advantage should be preserved.

## Lean Mapping

As previously observed, Bump Mapping has a highlight problem, which is caused by filtering. When viewed from distance, the sampled surface will be filtered and result in changing the normal to the average value computed that as a result make the bumps no longer visible, which causes the impression of a duller surface. To deal with it, a new technique was developed called LEAN mapping [Olano and Baker 2010]. The LEAN mapping is a simple model that is compatible with existing diffuse bump filtering. It has a low pre-computation cost, low run-time cost and it is compatible with existing shading models based on Blinn-Phong [Blinn 1977] or Beckmann-distribution [Beckmann and Spizzichino 1963]. In addition, it allows several approaches to the combination of multiple bump layers. The Beckmann distribution uses a Gaussian distribution of normal slopes and is evaluated at the half vector to give the expected number of facets that the distribution predicts will be perfectly oriented to reflect view vector to light vector.

The LEAN mapping bump specularity is a modification of the Ward shading model [Ward 1992], which assumes perfectly reflective micro facets that are randomly distributed around the overall surface normal. This new model requires one additional MIP [Williams 1983] texture lookup per shading evaluating and manages to capture antialiasing of the highlight shape and the transition of anisotropic bumps into an anisotropic highlight. For the computation, the LEAN uses an existing height or normal map to generate the LEAN map on GPU.

Because of its simplicity, low computational overhead and compatibility with Blinn-Phong, it can be easily adopted for usage in games arts. To use LEAN with existing Blinn-Phong-based game assets, it is necessary to use an equivalence of the Blinn-Phong model with a symmetric Ward model using Beckmann distribution.

Given a bump normal, the top level of a LEAN map texture is seeded with:

Where is a division the coefficient by *b.z*. Using standard filtering of the resulted textures sampling of these five values will give as a result an anti-aliased, filtered blend of the bumps and specular shading. Since to store five values in textures will be necessary two textures, which enables the possibility to store eight values, the bump normal can also be stored on the remaining three values to save computation and improve quality of the diffuse filtering, instead of reconstructing it from . Figure 6 and Figure 7 illustrates both generated textures. This method also allows diffuse filtering using non-normalized normals after texture filtering, as observed by [Kilgard 2000].

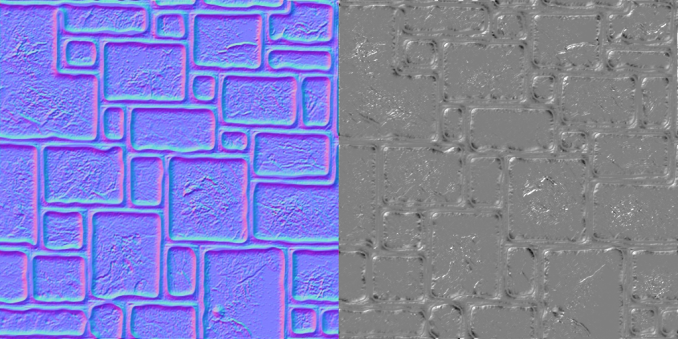


Figure : First texture generated by LEAN. The first three values from the texture (RGB) represents the bump normal, and the last one (Alpha) M.z.

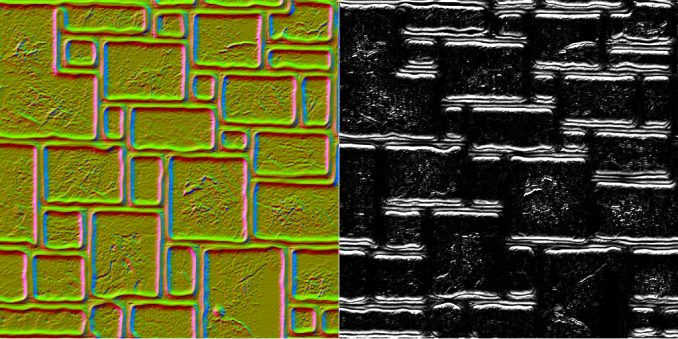


Figure : Second texture generated by LEAN. The first two values (RG) represent B while the last two (BA) M.x and M.y.

Given the filtered textures values, *B* and *M*, it is possible to reconstruct the covariance ∑, which is used to control the size and shape of the highlight, and then use ∑-1. Below are the equations used to construct ∑ from *B* and *M*. To use LEAN with Blinn-Phong specularity, it is only necessary to add the *1 / s* term during the final shading when constructing the covariance ∑, adding it in the *M.x* and *M.y* terms. Doing so will result in the effect of baking the Blinn-Phong specularity in the texture.

Equation : Covariance construction from B and M combining linearly

Equation : Covariance construction from B and M using MIP levels with any linear filtering

Lastly, the specular term is computed using the following equation:

Equation : LEAN specular term

Figure 8 shows the difference of the techniques LEAN mapping and Bump mapping.

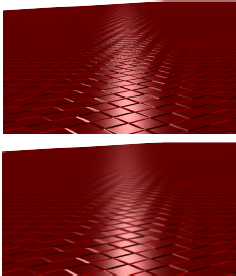


Figure : Bump Mapping versus LEAN Mapping: First picture is generating an ordinary bump map. Second picture was using LEAN mapping with trilinear MIP filtering. Figure adapted from [Olano and Baker 2010].

## Distance Fog

Distance Fog, also referred as true fog or range fog, tints the screen according to the distance that a given pixel is from the camera. Objects further away will appear to fade into the fog. To do this, distance fog need to check distances every frame, and as a result can have an impact on performance. Some limitations can be set to reduce this impact, like rejecting objects beyond a certain distance. The resulting effect of distance fog simulates light scattering, which causes objects that are more distant appear in lower contrast. This technique can also be used to obscure objects with a fog gradient by haze and aerial perspective. Figure 9 illustrates an example of the fading effect from distance fog.

The range-based fog approach, is a more accurate way to determine the fog effects. In method, it is used the actual distance from the viewpoint to a vertex for its fog calculations, increasing the effect of fog as the distance between the two points increases, rather than the depth of the vertex within in the scene, thereby avoiding rotational artifacts. Graphic APIs, such as Direct3D or OpenGL, have three interpolations for fog: One linear and two exponential.



Figure : Distance Fog representation. Note that the teapot begins to vanish when camera distance increases. Even so, the specular highlight is identical to the original model, minus the change in alpha to make the model transparent.

The Linear fog equation, known in Direct3D as D3DFOG\_LINEAR:

Exponential Fog equations:

D3DFOG\_EXP:

D3DFOG\_EXP2:

* *start* is the distance at which fog effects begin.
* *end* is the distance at which fog effects no longer increase.
* *e* is the base of natural logarithms (approximately 2.71828).
* *density* is an arbitrary fog density that can range from 0.0 to 1.0.
* *d* represents depth, or the distance from the viewpoint. For range based fog, the value for d is the distance between the camera position and a vertex. For non-range based fog, the value for d is the absolute value of the Z-coordinate in camera space.

The system stores the fog factor in the alpha component of the specular color for a vertex. If the application performs its own transformation and lighting, it is possible to insert the fog factor values manually during the rendering process. When the fog effects are calculated, the fog factor from one of the previews equation is used in a blending process. The blending equation below effectively scales the color of the current polygon C by the fog factor f, and adds the product to the fog color C, scaled by the bitwise inverse of the fog factor. The resulting color value is a blend of the fog color and the original color, as a factor of distance.

Blending Equation:

# Specular Shyness Correction

The proposed shading model in this paper is a variation of the LEAN mapping technique proposed by [Olano and Baker 2010]. As was presented on section 2, LEAN mapping is used to deal with the highlight aliasing and the blurred drawback generated from bump mapping when the object is viewed from certain distances. LEAN solves this issue very well, maintaining the clarity of the bumps in the surface. However, the specular shyness is also maintained, even on huge distances, as shown in Figure 10. As the figure illustrates, the ocean highlights caused from the bump deformations to simulate ocean waves are still present even on huge distance near the island's coastal. This happens because LEAN preserves the bump surface as well as the bump specularity, which is based on the Ward model. The Ward model assumes perfectly reflective microfacets that are randomly distributed around the overall surface normal.

To deal with this problem, we propose a modification of the shading specular component, similar to the effect of distant fog, but affecting only the specular highlight intensity, reducing it gradually up to a minimum accepted value defined by the user. This specular fading term varies according to the camera distance from the object. Both the minimum and maximum highlight specular intensity is configurable, and viewed on close range is identical to LEAN. This new configurable empirical specular fading term is added as a power of the dot product between *Normal* and *Light* vectors.



Figure : LEAN mapping specular highligh is present on huge distances. Notice the highlight at the upper right corner of the picture. Figures extracted from [Olano and Baker 2010].

This new term allows the specular control over distance, which is made during the shading process without too much computational overhead. In addition, unlike fog that changes the alpha according to the distance and thus preserving the specular highlight, this term actually reduces it without interfering with the alpha component. Below is the equation for the specular fading control:

Where *V* is the view vector, *Fade* is a parameter to configure the fading rate, which can also be used to adjust to the world scale, and *maxP* is the maximum specular fade desired by the user. This term is applied as shown below:

With *Ks* being the material specular component, אּ the specular fading term, and *spec* the LEAN specular term.

# Results

While comparing LEAN map with our proposed model, it is possible to notice the difference on specular highlight when the distance between the camera and the object changes. Figure 1 illustrates the specular fading, LEAN and Bump mapping, while Figure 12 and Figure 13 show the difference between normal LEAN technique and LEAN with specular fading. As shown in it, the farther the object is from the camera, lower the specular highlight is. This also corrects the specular highlight from fog algorithms, which changes the alpha component making it transparent over distance. Notice that this method only reduces the specular highlight, so it can be used in conjunction with distance fog without problems, allowing the fog algorithm to fade the object while changing the specular highlight, unlike shown in Figure 9, which maintains the highlight while fading.

Tests were run in order to compare the new model with the traditional LEAN technique. The computer that ran the tests uses a geforce GTX550Ti, Windows 7 64-bit. The recorded frames per second were ranging over 1000 to 1600 FPS for both methods, which varies with the camera range and the object, The difference with only of 10 FPS between the two techniques. The model used on these tests had 256 triangles and 200 vertices. With a teapot model containing 79600 triangles and 44366 vertices, the frame rates were around 570 to 790 FPS depending on the range with minimal difference of 5 FPS.

# Implementation

The implementation shown in this section shows only the application of the equation in the LEAN shader. For LEAN, generate lean maps textures normaly and unpack N, B, M terms from the texture. Convert M to ∑ and compute the specular term. After this process, compute the specular fading and the final color, as shown by Figure 11.

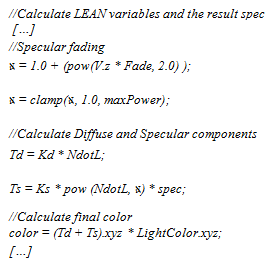


Figure : GLSL pseudo code for specular fading applied in LEAN shader.

# Conclusion

This paper presented improvement techniques for bump mapping and their corresponding problems with the specular highlight over distances, as well as the way the distance fog algorithm try to solve it. As a result, a new approach was developed and presented to affect the highlight according to the object distance. Since it only affects the specular term, it can also be used in conjunction with fog to increase visual appeal. Because this new approach is based on LEAN, a performance comparison was made between both approaches, concluding that the performance cost of the modification does not have noticeable impact on calculation speed. Since the base of this approach is LEAN mapping, it can also be used in real time applications.

# Acknowledgements

Omitted for blind review.

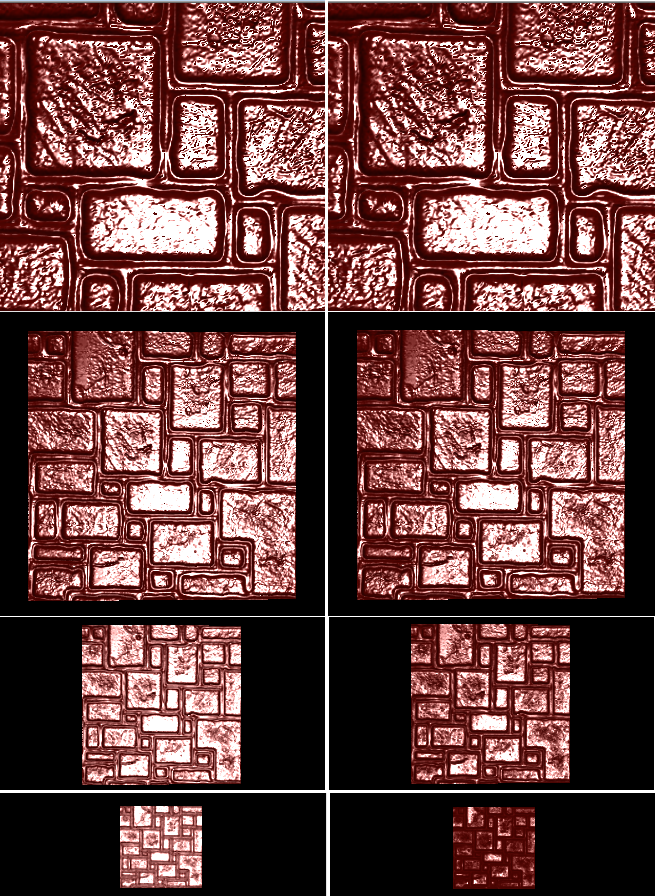


Figure : Comparison of LEAN with our proposed model. In the left is the traditional LEAN technique. To the right the proposed change. As shown in the figure, at close range they are identical, however the farther the camera is from the object, less specular highlight.

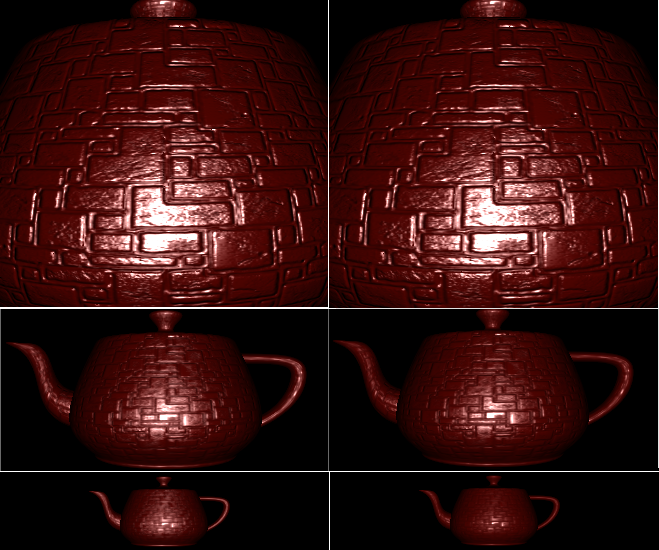


Figure : Comparison with a Teapot model

# References

Ashikhmin, M., Premoze, S. and Shirley, P., 2000. A Microfacet-based BRDF Generator. *In: Proceedings SIGGRAPH ’00*, pp.65–74.

Becker, B.G. and Max, N.L., 1993. Smooth transitions between bump rendering algorithms. In SIGGRAPH ’93. Proceedings of the 20th annual conference on Computer graphics and interactive techniques. New York, NY, USA: ACM, pp. 183–190.

Beckmann, P. and Spizzichino, A., 1963. *The Scattering of Electromagnetic Waves from Rough Surfaces*, Pergamon Press.

Blinn, J.F., 1977. Models of light reflection for computer synthesized pictures. *SIGGRAPH Comput. Graph.*, 11(2), pp.192–198.

Blinn, J.F., 1978. Simulation of wrinkled surfaces. *SIGGRAPH Comput. Graph.*, 12(3), pp.286–292.

Cabral, B., Max, N. and Springmeyer, R., 1987. Bidirectional reflection functions from surface bump maps. *SIGGRAPH Comput. Graph.*, 21(4), pp.273–281.

Cohen, J., 1998. *Appearance-preserving simpli\_cation of polygonal models*. The University of North Carolina at Chapel Hill.

Cohen, J., Olano, M. and Manocha, D., 1998. Appearance-preserving simplification. In SIGGRAPH ’98. Proceedings of the 25th annual conference on Computer graphics and interactive techniques. New York, NY, USA: ACM, pp. 115–122.

Cook, R.L. and Torrance, K. E., 1982. A Reflectance Model for Computer Graphics. *ACM Trans. Graph.*, 1(1), pp.7–24.

Heidrich, W. and Seidel, H.-P., 1999. Realistic, hardware-accelerated shading and lighting. In SIGGRAPH ’99. Proceedings of the 26th annual conference on Computer graphics and interactive techniques. New York, NY, USA: ACM Press/Addison-Wesley Publishing Co., pp. 171–178.

Kajiya, J.T., 1985. Anisotropic reflection models. In SIGGRAPH ’85. Proceedings of the 12th annual conference on Computer graphics and interactive techniques. New York, NY, USA: ACM, pp. 15–21.

Kautz, J. and McCool, M.D., 1999. Interactive rendering with arbitrary BRDFs using separable approximations. In SIGGRAPH ’99. ACM SIGGRAPH 99 Conference abstracts and applications. New York, NY, USA: ACM, p. 253–.

Kautz, J. and Seidel, H.-P., 2000. Towards interactive bump mapping with anisotropic shift-variant BRDFs. In HWWS ’00. Proceedings of the ACM SIGGRAPH/EUROGRAPHICS workshop on Graphics hardware. New York, NY, USA: ACM, pp. 51–58.

Kilgard, M., 2000. A practical and robust bump-mapping technique for today’s GPU’s. *In: GDC 2000*.

Olano, M. and Baker, D., 2010. LEAN mapping. In I3D ’10. Proceedings of the 2010 ACM SIGGRAPH symposium on Interactive 3D Graphics and Games. New York, NY, USA: ACM, pp. 181–188.

Peercy, M., Airey, J. and Cabral, B., 1997. Efficient bump mapping hardware. In SIGGRAPH ’97. Proceedings of the 24th annual conference on Computer graphics and interactive techniques. New York, NY, USA: ACM Press/Addison-Wesley Publishing Co., pp. 303–306.

Phong, B.T., 1975. Illumination for computer generated pictures. *Commun. ACM*, 18(6), pp.311–317.

Ward, G.J., 1992. Measuring and modeling anisotropic reflection. *SIGGRAPH Comput. Graph.*, 26(2), pp.265–272.

Westin, S.H., Arvo, J.R. and Torrance, Kenneth E., 1992. Predicting reflectance functions from complex surfaces. In SIGGRAPH ’92. Proceedings of the 19th annual conference on Computer graphics and interactive techniques. New York, NY, USA: ACM, pp. 255–264.

Williams, L., 1983. Pyramidal parametrics. *SIGGRAPH Comput. Graph.*, 17(3), pp.1–11.