Specular Fading over Distance on Wrinkled Surfaces

Troy Kohwalter, Esteban Clua, Leandro Fernandes
Instituto de Computao
Universidade Federal Fluminense
Niteri RJ, Brazil
tkohwalter, esteban, laffernandes@ic.uff.br

Fig. 1. Bump mapping with traditional Blinn-Phong shading (left), LEAN mapping (center), and our model (right) applied on teapots at different distances from the observer (top and bottom). In contrast to other approaches, our technique is capable to reproduces the natural fading of the specular component over distance without loss of contrast (left) or without the introduction of additional energy (center).

***Abstract*—Blinn-Phong shading model is commonly used in real time graphics. When combined with Phong shading and normal mapping, this model allows a rich visual experience. However, when the observer is far from the surface, the bumped regions may become blurred and disappear, thus making the surface duller than it is. The Linear Efficient Antialiased Normal (LEAN) mapping was developed to correct this issue. However, in that approach the specular highlight intensity of small bumps is unaffected by distance, leading to brighter specular highlights than one would expect. We present and extension of LEAN mapping where the excess of energy in the specular highlight is corrected by the introducing a fading term during the computation of the specular component of the illumination model. We show that our simple but effective approach is capable to produce plausible visual effects to be used by real time graphics applications such as games and virtual reality.**

***Keywords*-bump mapping; normal mapping; LEAN mapping; specular ilumination;**

I. Introduction

To create the illusion of details in surfaces, artists usually create and apply bump maps in rendering. In many cases, these maps contain random bumps to better simulate rough surface materials at close range. The surface of these perceived roughnesses should be the same when displayed at different distances to the viewer. However, due to filtering issues regarding non-regular and near-stochastic patterns combined to inappropriate shading models, the shading of mapped bumps are often inconsistent across different levels of detail and distance. It is because the shading model at the coarsest level of detail does not correspond to the mapped bumps at finer level of detail.

Bump mapping was originally introduced by Blinn [1]. He demonstrated how to simulate wrinkled surfaces by only perturbing the normal vector without changing the underlying surface itself. The perturbed normal replaces the original normal vector of the surface while lighting is computed. For about thirty years, the bump mapping 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.

Unfortunately, bump mapping has serious drawbacks with filtering and antialiasing. When the bumps are viewed at a distance, the standard mipmapping technique for bump map can work well for diffuse shading [2]. 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. Many techniques that tries to solve the highlight aliasing problem were proposed, but most of them require pre-processing stages [3], [4], [5]. To correct this issue, a new mapping approach called Linear Efficient Antialiased Normal (LEAN) mapping was developed by Olano and Baker [6]. With this approach, the bumps are preserved at high distances. Nevertheless, the specular highlights produced by small bumps are also visible, making the surface artificially shiny.

Contributions: This paper describes an improvement of the LEAN mapping. By introducing a fading component in the computations we show that it is possible to reduce the

specular highlight according to the distance of the object. For this aim, we use the original 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, preventing the inclusion of additional energy on the shiny effect.

The remaining of the paper is organized as follows: Section II presents some ground basement of previews works in the area, as well as an overview of the proposed method. Section III presents our modified method. Section V and Section IV discusses results and implementation details. Section VI concludes the paper with some observations and directions for future exploration.

II. Related Work

There are many approaches that model how light reflects at an opaque surface. One of the first well established models was proposed by Blinn [7], which used a simple shading equation based on microfacets to represent micro details as a bidirectional reflectance distribution function (BRDF). Cook and Torrance [8] have presented a microfacet-based BRDF model which also added shadowing and a Fresnel term to make the representation of metallic and plastic materials more realistic. In contrast to previous work, Ward [9] has developed a BRDF modeling anisotropic Gaussian distribution of microfacet, and Ashikhmin et al. [10] have developed a way of generating reflection models with arbitrary normal distributions.

While micro details of opaque materials can be represented by BRDFs, small bumps on the underlying surface need a more descriptive representation. Blinn [1] was the first to to use a height field mapped to the surface to model small (not micro) details on objects represented by coarse triangular meshes. From the height information mapped to a given point on the surface and the normal vector on that point (computed from the original mesh), he demonstrated how to perturb and replace the original normal in order to simulate wrinkles on the surface. Cohen et al. [11] extended Blinns ideas and has used textures to store normal vectors and map them directly to the surfaces, avoiding the computation of perturbed normal from a height map. This technique is called Normal Mapping.

Normal Mapping has been combined with BRDFs in games and other real-time graphics applications. However, the filtering process implemented by graphics hardware makes the normal vectors fetched from normal maps not suitable for rendering shiny surfaces at different distances from the observer with existing BRDF models. When viewed from distance, the sampled surface normals will be filtered and the result of changing the normal to an average value makes the bumps no longer visible, which causes the impression of a duller surface. This problem was addressed and partially solved by the LEAN mapping [6] technique, in which this work was based on.

A. LEAN Mapping

The LEAN mapping [6] is a simple model that is compatible with existing filtering for diffuse bumps. It has a low precomputation and run-time cost while also being compatible with existing BRDFs such as the Blinn-Phong model proposed by Blinn [7].

The LEAN mapping is a modification of Wards shading model [9]. In contrast to conventional filtering, the technique requires one additional MIP [15] texture lookup per shading evaluation and also manages to capture antialiasing of the highlight shape and the transition of anisotropic bumps into an anisotropic highlight. In render time, the technique uses an existing height or normal map to generate two auxiliary LEAN map textures on GPU [6].

The use of LEAN mapping with existing Blinn-Phong-based shaders requires the equivalence of the Blinn-Phong model with the symmetric Ward model using a Beckmann distribution. Given a normal vector, (1), retrieved from the normal map and represented in tangent space:

$$N = (b_x, b_y, b_z) \tag{1}$$

the LEAN map textures are computed as:

$$B = (\tilde{b}_x, \tilde{b}_y) = \left(\frac{\vec{b}_x}{\vec{b}_z}, \frac{\vec{b}_y}{\vec{b}_z} \right), \text{ and} \tag{2}$$

$$M = ({b'_x}^2, {b'_y}^2, b'_x b'_y) \tag{3}$$

where b is the Bump's Normal. The textures sampling the five values described in (2) and (3) gives an anti-aliased, filtered blend of the bumps and specular shading for any view. Refer to [6] for details. To store five values in textures, it is required the usage of two texture units, which enables the possibility to store eight values. The remaining three values can be used to store the conventional normal map in order to save computation and improve the quality of the diffuse filtering, instead of reconstructing it. Figs. 2 and 3 illustrates the RGBA from both generated LEAN textures. As observed in [2], storing the normal map allows the usage of diffuse filtering with a non-normalized normal resulting from texture filtering.

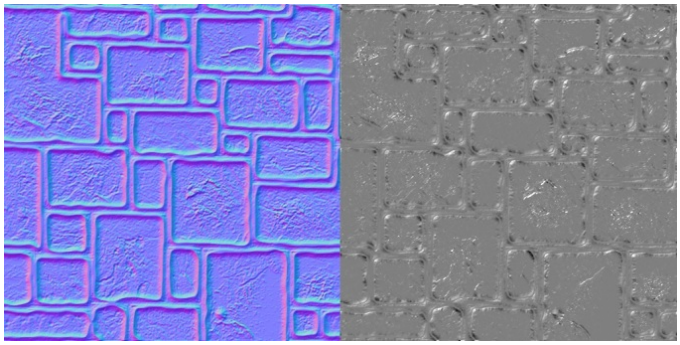

Fig. 2. First texture generated by LEAN. The first three color channels of the texture (R, G, and B) store the normal map, and the last one (A) stores M.z (defined by (3)).

Given the LEAN textures values, B and M from (2) and (3), respectively, it is possible to compute the covariance Σ with (4) by using MIP levels and any linear filtering. The covariance

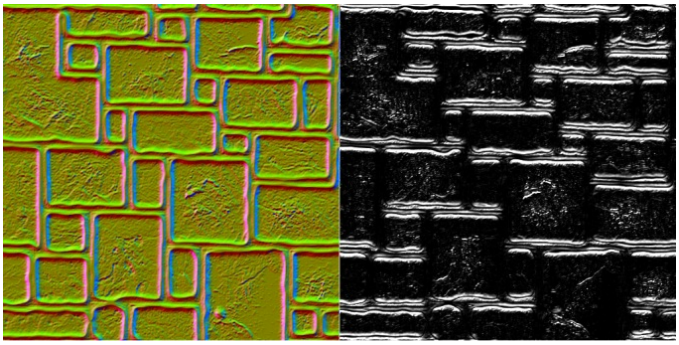

Fig. 3. Second texture generated by LEAN. The first two color channels (R and G) are used to store B (computed using (2)) while the last two channels (BA) store M.x and M.y (computed using (3)).

controls the size and shape of the highlight center. With the covariance, it is possible to compute Σ^{-1} to calculate the LEAN specular term in (5):

$$\Sigma = \begin{pmatrix} M_x - B_x * B_x & M_z - B_x * B_y \\ M_z - B_x * B_y & M_y - B_y * B_y \end{pmatrix} \quad (4)$$

$$\frac{1}{2\Pi\sqrt{|\Sigma|}} e^{-0.5(h_n - b'_n)^T \Sigma^{-1}(h_n - b'_n)} \quad (5)$$

where h is the half way vector between the vector from the surface toward the viewer and the vector from surface toward the light.

To use LEAN mapping with Blinn-Phong model, it is necessary to add the $1/s$ term in the final shading when constructing the covariance , adding it in the $M.x$ and $M.y$ terms [6]. Doing so will result in introducing the Blinn-Phong specularity effect in the texture.

III. Proposed Specular Fading Approach

The Ward model assumes perfectly reflective microfacets. This feature is inherited by the LEAN mapping technique, leading to specular shininess even on huge distances from the observer. The central contribution of our work is to deal with this problem by introducing a fading term on the computation of the specular component of LEAN mapping approach. Our 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 the result is identical to LEAN. This new configurable empirical specular fading term is added in a shader as a power of the dot product between Normal and Light vectors.

This new term allows the specular control over distance, which is made during the shading process without significant computational overhead. It acts as an attenuator component based on the distance, reducing the specular highlight. In addition, our distance-based term actually reduces the shiny appearance without interfering with the alpha component or other light model components.

The specular fading control sf is computed as:

$$sf = clamp(1 + (V_z * Fade)^2, 1, P_max) \quad (6)$$

where V is the normalized view vector, $Fade$ is a positive, and greater than zero, parameter used to configure the fading rate, which is also used to adjust to the world scale. The $Fade$ parameter determines the specular energy loss rate by distance (V_z). P_max is the maximum specular fade desired by the user and is used in conjunction with $clamp$ to control maximum specular loss. This is done so because even if $Fade$ is carefully chosen and controlled in order to avoid illumination glitches, the distance is not. And since specular energy is lost with distance, $clamp$ is used to force sf minimum an maximum values to avoid cases of gaining energy.

The power of two present in (6)is optional and was used for the sole purpose of increasing the energy loss rate in order to increase visibility in the illutrations presented throughout this paper. Doing so allowed to notice the changes made in the specular highlights while also maintaing the illustrations at a suitable size. This specular fading term, sf, is then applied in the intensity of the diffuse light ($NdotL$), as shown by (7), where Ks is the material specular component, sf the specular fading computed by (6), and $LEAN_spec$ is the LEAN specular term calculated from (5).

$$TotalSpecular = Ks * NdotL^{sf} * LEAN_spec \quad (7)$$

IV. Implementation

Tests were made in order to compare the new model with the traditional LEAN technique. The tests were performed on a GeForce GTX550Ti, installed in a Windows 7 64-bit machine. The recorded frames per second (fps) were ranging from 1000 to 1600 for both methods. The frame rate varied, with the camera range and the object in the scene. The difference was about only 10 fps between the two techniques. The geometric model used on the tests (Fig. 14) is comprised of 256 triangles and 200 vertices. With the teapot model containing 79600 triangles and 44366 vertices (Fig. 13), the frame rates varied from 570 to 790 fps, with a difference of approximately 5 fps between techniques.

In our implementation we adapted the equation in the original LEAN shader. Our GLSL shader is presented in listing 1. For the LEAN stage, we generated lean maps textures (Figs.2 and 3, 4 and 5) as described in [6], and unpack N (line 5), B, M terms from the texture (lines 11 and 12). We then converted M to (line 14) and computed the LEAN specular term (lines 15 to 19). Then we compute the intensity of the diffuse light (line 21) and the diffuse component (line 22). After this process, we compute the specular fading sf (lines 24 to 27), the fresnel component (line 29) and the final specular component (line 30). Lastly, the final color using the diffuse and specular components with the light color (line 32).

Listing 1. GLSL pseudo code for specular fading applied in LEAN shader in tangent space.

```
1   vec3 fvLightDirection = normalize(
        LightDirection);
2   vec3 fvViewDirection = normalize(
        ViewDirection);
3   vec3 Half = normalize(fvViewDirection +
        fvLightDirection);
```

```
4           vec4 t1 = texture2D(Lean1, gl_TexCoord
                [0].st);
5           vec3 Normal = vec3(t1.xyz) * 2.0 - 1.0;
6            // Equivalence for the Blinn-Phong is 1
                 / s, where s is the specular
                expoent.
7            // This is done here to remove the
                dependancy on the lean map creation
                .
8           float equivalence = 1.0 /
                MaterialShininess;
9           // Unpack B and M
10          vec4 t2 = texture2D(Lean2, gl_TexCoord
                [0].st);
11          vec2 B = (t2.xy * 2.0 - 1.0);
12          vec3 M = vec3(t2.zw + equivalence, t1.w
                * 2.0 - 1.0);
13          // Convert M to E
14          vec3 E = M - vec3(B * B, B.x * B.y);
15          float Det = E.x * E.y - E.z * E.z;
16          // Compute LEAN Specular term
17          vec2 h = Half.xy / Half.z - B;
18          float e = (h.x * h.x * E.y + h.y * h.y *
                 E.x - 2.0 * h.x * h.y * E.z);
19          float LEAN_spec = (Det <= 0.0) ? 0.0 :
                exp(-0.5 * e / Det) / sqrt(Det);
20          // Normal dot LightDirection
21          float NdotL = clamp (dot (Normal,
                fvLightDirection , 0.0, 1.0);
22          fvTotalDiffuse = fvMaterialDiffuse *
                NdotL;
23          // Spec Fading Computation
24          float sf = 1.0 + (pow(ViewDirection.z *
                Fade, 2.0) );
25          sf = clamp(sf, 1.0, P_max);
26          // Add specFading to Specular component
27          fvTotalSpecular = fvMaterialSpecular *
                pow(NdotL, sf) * LEAN_spec;
28          // Compute Fresnel
29          float fFresnel = clamp(FresnelBias + (
                FresnelScale * pow(1.0 - dot(Normal,
                 fvViewDirection), 5.0)), 0.0, 1.0);
30          fvTotalSpecular *= fFresnel;
31          // Final Color
32          color = (fvTotalDiffuse +
                fvTotalSpecular).xyz * LightColor.
                xyz;
```

V. Result display

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. The generated LEAN maps from Figs.2 and 3, 4 and 5 were used to generate Figs. 6, 7, 8, 9, 10, 11 and 12 to illustrate the usage of specular fading, LEAN and bump mapping. Figs.13 and 14, which also used Figs.2 and 3, show the difference between normal LEAN technique and LEAN with specular fading. As shown by these figures when using specular fading, the farther the object is from the camera, the lower the specular highlight is. This happens because of the attenuator component based on the distance.

This also corrects the specular highlight from fog algorithms, which only changes the alpha component making it transparent over distance. Notice that this method only attenuate 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.

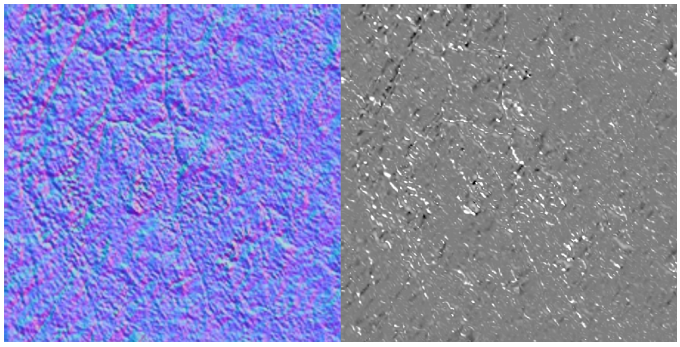

Fig. 4. First LEAN texture generated for Figs.6, 7, 8, 13 and 14

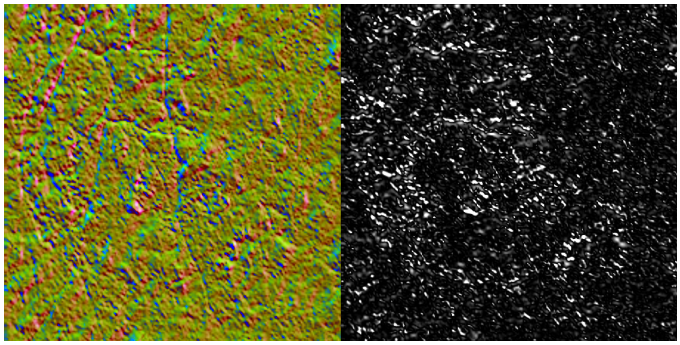

Fig. 5. Second LEAN texture generated for Figs.9, 10, 11 and 12

VI. Conclusion

This paper extended the LEAN mapping technique in order to deal with its problems with excessive specular highlight over distances. The proposed solution consist of a fading term that modules the shininess of the material. Since our fading only affects the specular term, it can also 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, increasing the visual appeal.

A performance comparison was made between conventional LEAN mapping and the proposed approach. It was concluded that the computational cost of our technique does not have noticeable impact on frame rate. Therefore, our approach can be used in real time applications such as games and virtual reality.

Acknowledgment

The authors would like to thank NVIDIA, CNPq, FAPERJ, and CAPES for the financial support

References

[1] J. F. Blinn, "Simulation of wrinkled surfaces," in *Proceedings of the 5th annual conference on Computer graphics and interactive techniques*, ser. SIGGRAPH '78. New York, NY, USA: ACM, 1978, pp. 286–292. [Online]. Available: http://doi.acm.org/10.1145/800248.507101

[2] M. J. Kilgard, "A practical and robust bump-mapping technique for today's gpus," 2000.

[3] B. Cabral, N. Max, and R. Springmeyer, "Bidirectional reflection functions from surface bump maps," in *Proceedings of the 14th annual conference on Computer graphics and interactive techniques*, ser. SIGGRAPH '87. New York, NY, USA: ACM, 1987, pp. 273–281. [Online]. Available: http://doi.acm.org/10.1145/37401.37434

[4] B. G. Becker and N. L. Max, "Smooth transitions between bump rendering algorithms," in *Proceedings of the 20th annual conference on Computer graphics and interactive techniques*, ser. SIGGRAPH '93. New York, NY, USA: ACM, 1993, pp. 183–190. [Online]. Available: http://doi.acm.org/10.1145/166117.166141

[5] S. H. Westin, J. R. Arvo, and K. E. Torrance, "Predicting reflectance functions from complex surfaces," in *Proceedings of the 19th annual conference on Computer graphics and interactive techniques*, ser. SIGGRAPH '92. New York, NY, USA: ACM, 1992, pp. 255–264. [Online]. Available: http://doi.acm.org/10.1145/133994.134075

[6] M. Olano and D. Baker, "Lean mapping," in *Proceedings of the 2010 ACM SIGGRAPH symposium on Interactive 3D Graphics and Games*, ser. I3D '10. New York, NY, USA: ACM, 2010, pp. 181–188. [Online]. Available: http://doi.acm.org/10.1145/1730804.1730834

[7] J. F. Blinn, "Models of light reflection for computer synthesized pictures," in *Proceedings of the 4th annual conference on Computer graphics and interactive techniques*, ser. SIGGRAPH '77. New York, NY, USA: ACM, 1977, pp. 192–198. [Online]. Available: http://doi.acm.org/10.1145/563858.563893

[8] R. L. Cook and K. E. Torrance, "A reflectance model for computer graphics," *ACM Trans. Graph.*, vol. 1, no. 1, pp. 7–24, Jan. 1982. [Online]. Available: http://doi.acm.org/10.1145/357290.357293

[9] G. J. Ward, "Measuring and modeling anisotropic reflection," in *Proceedings of the 19th annual conference on Computer graphics and interactive techniques*, ser. SIGGRAPH '92. New York, NY, USA: ACM, 1992, pp. 265–272. [Online]. Available: http://doi.acm.org/10.1145/133994.134078

[10] M. Ashikmin, S. Premože, and P. Shirley, "A microfacet-based brdf generator," in *Proceedings of the 27th annual conference on Computer graphics and interactive techniques*, ser. SIGGRAPH '00. New York, NY, USA: ACM Press/Addison-Wesley Publishing Co., 2000, pp. 65–74. [Online]. Available: http://dx.doi.org/10.1145/344779.344814

[11] J. Cohen, M. Olano, and D. Manocha, "Appearance-preserving simplification," in *Proceedings of the 25th annual conference on Computer graphics and interactive techniques*, ser. SIGGRAPH '98. New York, NY, USA: ACM, 1998, pp. 115–122. [Online]. Available: http://doi.acm.org/10.1145/280814.280832

Fig. 6. Comparison among Bump Mapping (left column), LEAN mapping (center), and Specular Fading (right column) using a ninja head model. Notice the difference of the specular highlights as the distance increases.

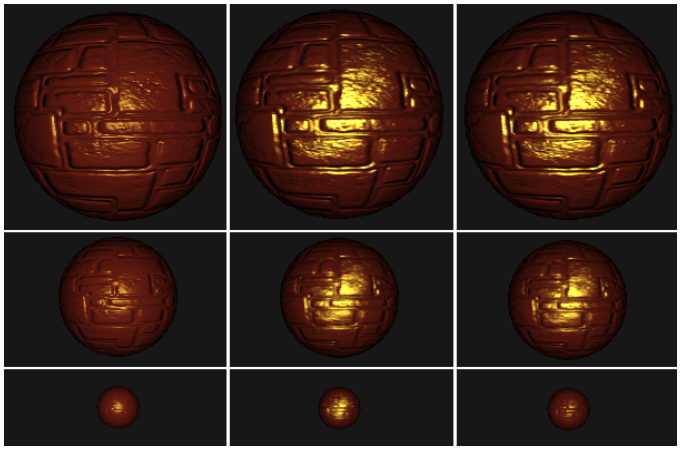

Fig. 7. Another comparison between Bump Mapping (left column), LEAN mapping (center), and Specular Fading (right column) using a sphere model. Notice the difference in the specular highlights. At close range, it is very similar to LEAN's, but as the distance increases, the highlight diminishes while keeping the aspect of bump surface.

Fig. 8. Another comparison between Bump Mapping (left column), LEAN mapping (center), and Specular Fading (right column). This time using a torus model.

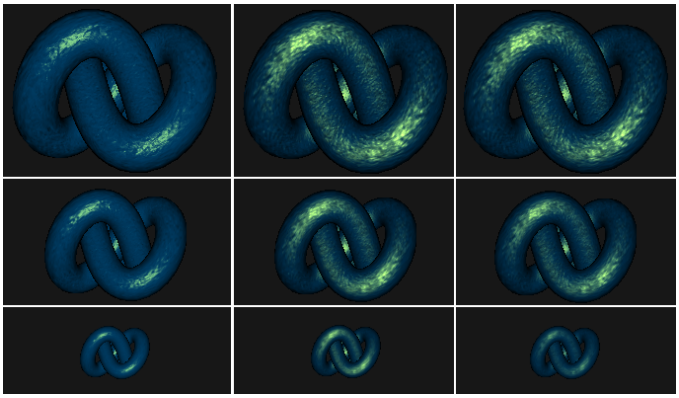

Fig. 9. Another comparison among Bump Mapping (left column), LEAN mapping (center), and Specular Fading (right column) using a torus model. This time the model and specular lights are in different colors. Notice the difference of specular highlights as the distance increases.

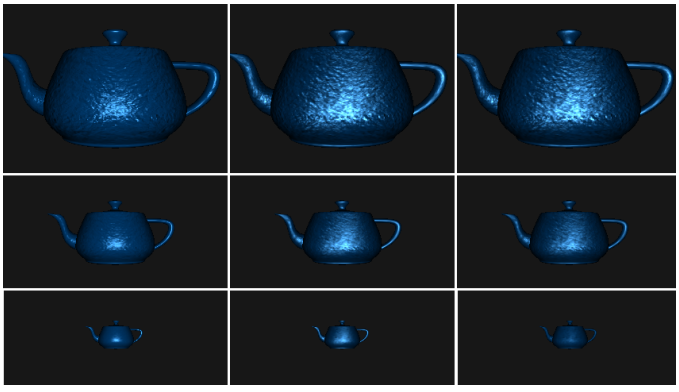

Fig. 10. Another comparison among Bump Mapping (left column), LEAN mapping (center), and Specular Fading (right column). The model used is a teapot with blue shading and light. While Bump mapping makes the teapot surface look more smoth as distance increases, LEAN keeps the bump aspect while also preserving the specular highlights. The specular fading preserves the bump surface while also diminishes the highlights.

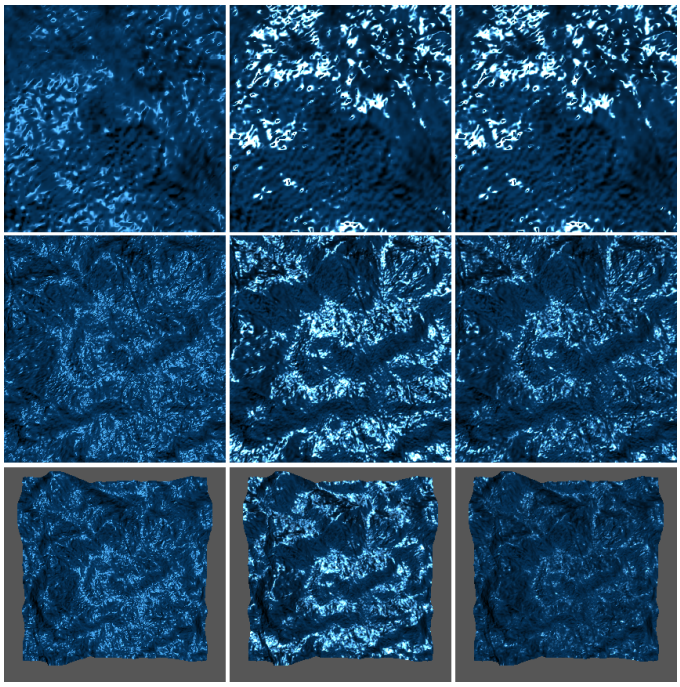

Fig. 11. Another comparison among Bump Mapping (left column), LEAN mapping (center), and Specular Fading (right column). This time with a terrain model at different distances. Notice the difference of specular highlights as the distance increases.

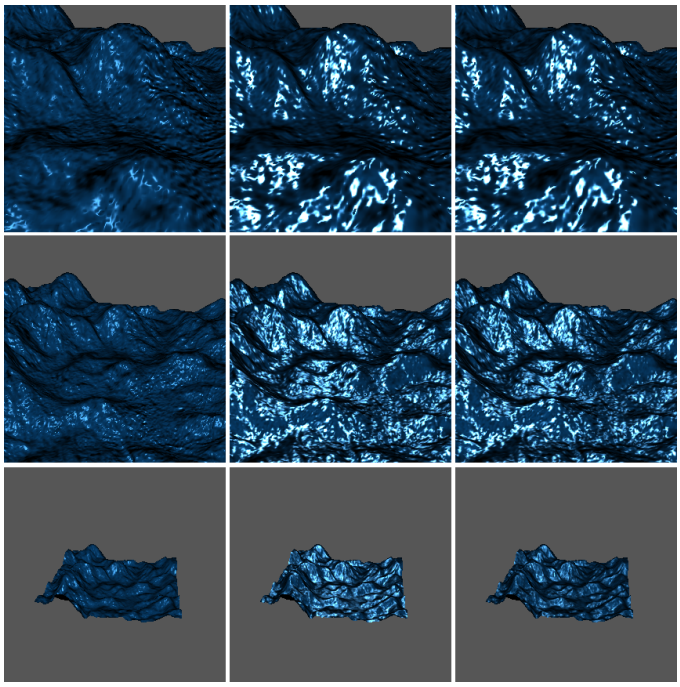

Fig. 12. The same terrain model but in a different angle.

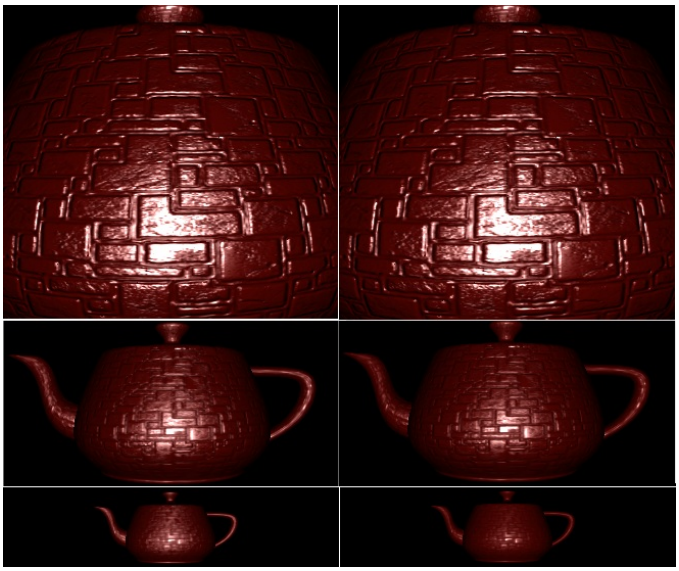

Fig. 13. Comparison between LEAN and our proposed improvement using a teapot model. In the left is the traditional LEAN technique and in the right the proposed improvement. As shown in the figure, at close range they are identical, however the farther the camera is from the object, less specular highlight.

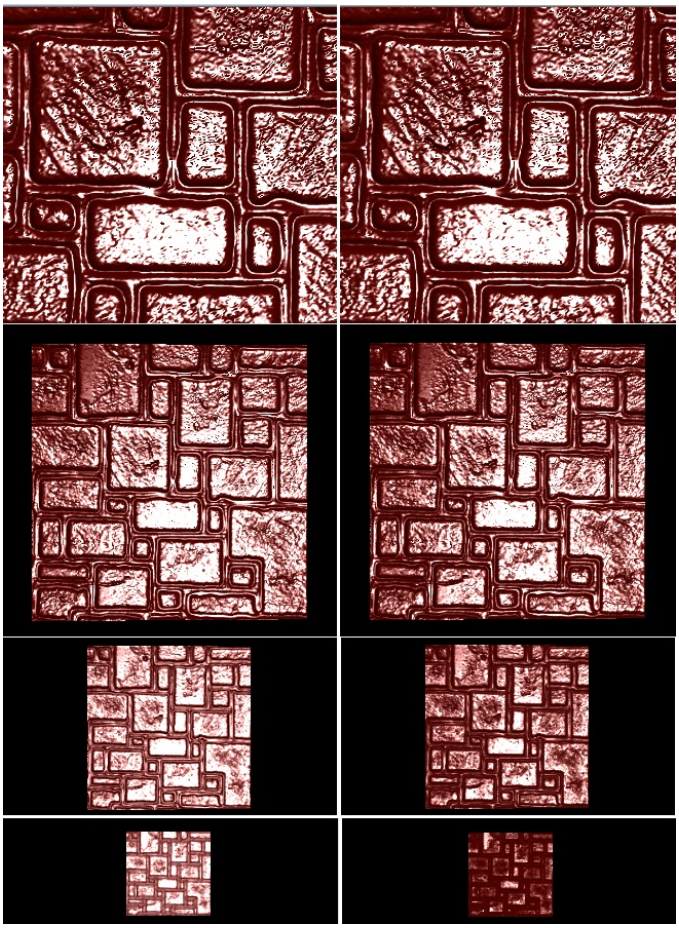

Fig. 14. Another comparison between LEAN and our proposed improvement using a square model. In the left is the traditional LEAN technique and in the right is LEAN with specular fading.