Specular Fading over Distance on Wrinkled Surfaces

Sibgrapi paper ID: 114562

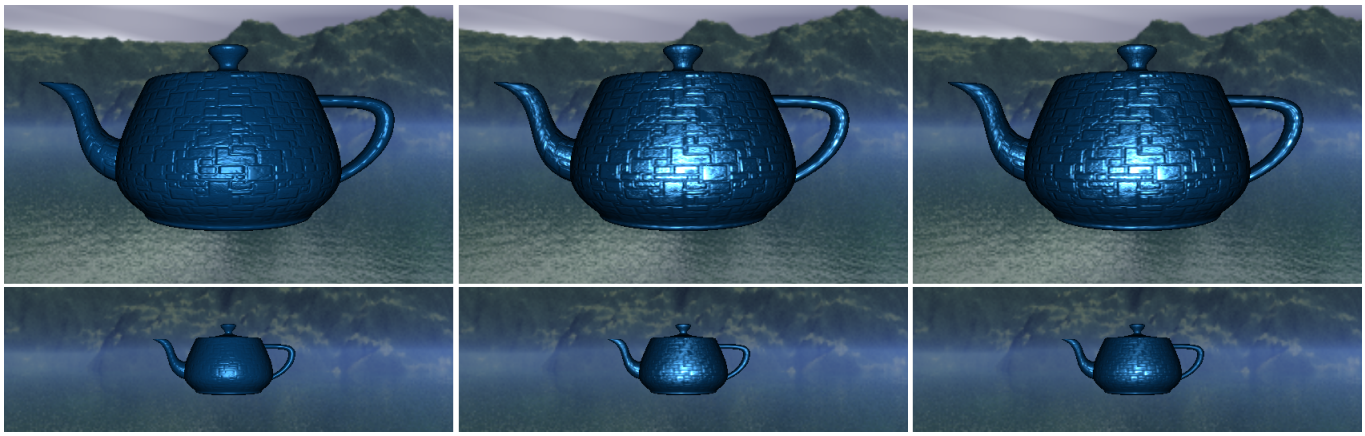

Fig. 1. Bump mapping with traditional Blinn-Phong shading (left), LEAN mapping (center), and our model (right) applied on teapots at different distances from the observer (top and bottom). In contrast to other approaches, our technique is capable to reproduce the natural fading of the specular component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

***Abstract*—Blinn-Phong shading model provides rich visual experience for bumped surfaces when used with traditional normal mapping techniques. 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 is a well-established tentative 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 an extension of LEAN mapping where the excess of energy in the specular highlight is corrected by introducing a fading term during the computation of the specular component of the illumination model. As a result, our approach produces more convincing specular effects for real-time graphics applications such as games and virtual reality.**

***Keywords*-bump mapping; normal mapping; LEAN mapping; specular illumination; fading**

I. Introduction

To create the illusion of details in surfaces, artists usually create and apply normal maps at the mesh description. 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 bump patterns combined to inappropriate shading models, the shading of wrinkled surfaces are often inconsistent across different levels of detail and distance. This happens because the shading model at the coarsest level of detail does not correspond to the mapped bumps at a finer level of detail. Traditional mipmapping-based filtering techniques may produce inconsistent results, with false attenuation of the specular effect.

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 virtually any 3D game.

Unfortunately, bump mapping has serious drawbacks with filtering and antialiasing. When the bumps are viewed at distance, the standard mipmapping technique applied on bump maps can work well for diffuse shading [2]. However, it fails to capture changes in specularity. When looking far away, the resulting look will be of a shiny and bumpless surface, appearing as if the bumps were duller. Many techniques that try 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.

This paper presents an extension of the LEAN mapping technique. By introducing a fading component in the computations, we show that it is possible to reduce the specular

highlight of shiny surfaces according to the distance of the object, hence reducing the extra energy produced by conventional LEAN mapping. For this purpose, we use the original LEAN mapping in order to avoid the blurring of the bumped surface and propose a modification to make the necessary change in the specular highlight, preventing the inclusion of additional energy on the shiny effect. We have implemented or approach as a vertex and a fragment shader. Since it does not introduce significant overhead on common transformation and lighting operations, it can be efficiently applied in real-time rendering of opaque specular materials.

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 the proposed approach and its implementation. Results are discussed in Section IV. Section V concludes the paper.

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 an empiric Bidirectional Reflectance Distribution Function (BRDF) for metallic and plastic surfaces. Cook and Torrance [8] have presented a microfacet-based BRDF model which 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 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. Cohen et al. [11] extended Blinn's 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 technique [6], in which this work is based on.

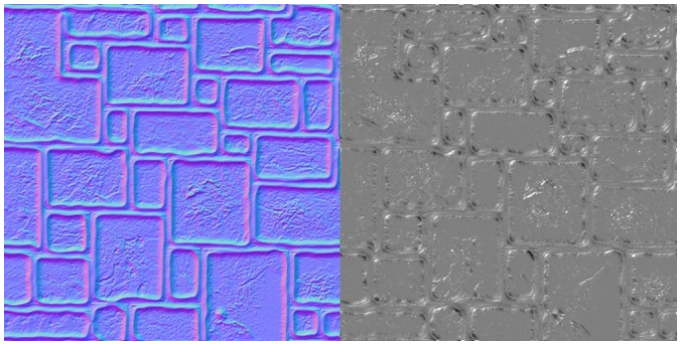

Fig. 2. First texture generated by LEAN. The first three color channels of the texture (R, G, and B) store the normal map, and the last one (A) stores M_z (defined by (3)). This texture was used in Figs. 1, 4, 5, 9 and 12.

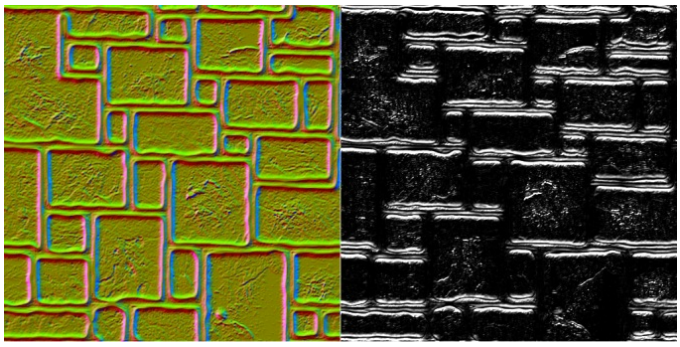

Fig. 3. Second texture generated by LEAN. The first two color channels (R and G) are used to store B (computed using (2)) while the last two channels (B and A) store, respectively, M_x and M_y (computed using (3)). This texture was used in Figs. 1, 4, 5, 9, and 12.

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 Ward's shading model [9]. In contrast to conventional filtering, the technique requires one additional mip-texture [12] 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, which is achieved with variance $1/s$ [6]. Given a normal vector retrieved from the normal map (N in (1)) and represented in tangent space:

$$N = (\vec{b}_x, \vec{b}_y, \vec{b}_z) \tag{1}$$

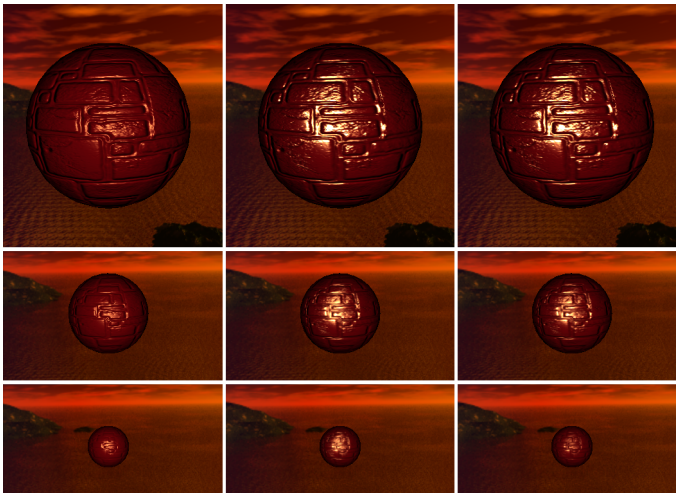

Fig. 4. Comparison between Bump Mapping (left column), LEAN mapping (center), and Specular Fading (right column) using a sphere model with 1984 triangles and 1226 vertices. Notice the difference in the specular highlights. At close range, it is very similar to LEAN's, but as the distance increases, the highlight diminishes while keeping the bumping aspect of the surface.

the LEAN map textures are computed as:

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

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

where $\vec{b}$ is the bump's normal. The textures storing 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. In order to store five values in textures, two texture units are required, which enables the possibility to store eight values altogether. The remaining three values can be used to store the conventional normal map. The advantages of storing all data instead of reconstructing it are twofold: some computation load is saved and the quality of the diffuse filtering is improved. As observed in [2], storing the normal map allows the usage of diffuse filtering with a non-normalized normal resulting from texture filtering. Figs. 2 and 3 illustrates the RGBA values produced for both LEAN textures applied in Figs. 1, 4, 5, 9, and 12

Given the LEAN textures values, B and M from (2) and (3), respectively, it is possible to compute the covariance Σ of the distribution with (4) by using mip-levels and any linear filtering as:

$$\Sigma = \begin{pmatrix} M_x - B_x\, B_x & M_z - B_x\, B_y \\ M_z - B_x\, B_y & M_y - B_y\, B_y \end{pmatrix} \tag{4}$$

The covariance controls the size and shape of the highlight center.

The inverse of the covariance (Σ^{-1}) is used to calculate the LEAN specular term as

$$LEAN_{spec} = \frac{1}{2\Pi\sqrt{|\Sigma|}} e^{-0.5(h_n - \tilde{b}_n)^T \Sigma^{-1} (h_n - \tilde{b}_n)} \tag{5}$$

Fig. 5. Comparison between Bump Mapping (left column), LEAN mapping (center), and Specular Fading (right column) using a ninja head model with 17832 triangles and 9136 vertices. Notice the difference of the specular highlights as the distance increases.

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 Σ, 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. When viewed on close range, the result is identical to LEAN. This new configurable specular-fading term is added in a shader as a power of the dot product between normal and light vectors.

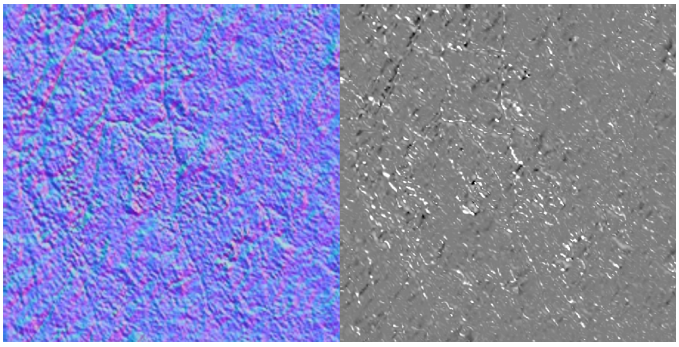

Fig. 6. First LEAN texture generated for Figs. 8, 10, 11, and 13

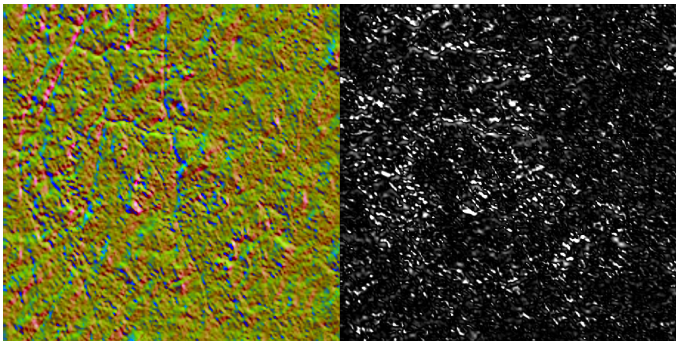

Fig. 7. Second LEAN texture generated for Figs. 8, 10, 11, and 13

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, $Fade \in (0, \infty)$ is a real parameter used to configure the fading rate, which is also related 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 it is used in conjunction with the $clamp$ function 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 of LEAN must be lost with distance, $clamp$ is used to force sf minimum and maximum values to avoid cases of gaining energy. The sf term must always be greater than one to avoid introducing 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 illustrations presented throughout this paper. Doing so allowed noticing the changes made in the specular highlights while also maintain 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):

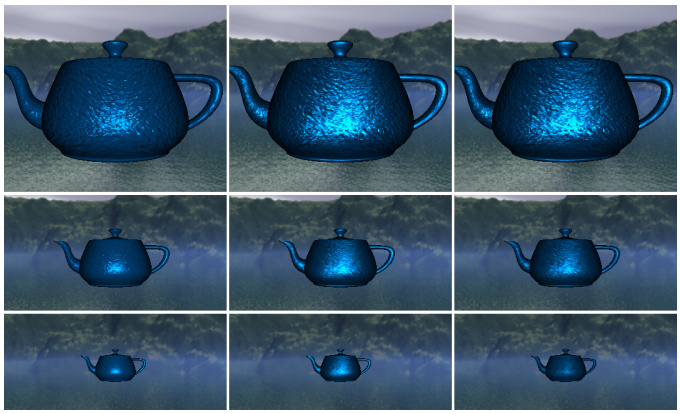

Fig. 8. The model used in this example is a teapot with blue shading and light with 79664 triangles and 44416 vertices. While Bump mapping makes the teapot surface lose contrast, and LEAN preserves the highlight, the specular fading preserves the bumped surface while diminishing the highlight.

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

where Ks is the material specular component, and $LEAN_{spec}$ is the LEAN specular term calculated from (5). Since sf is always positive and greater than one, the intensity of the diffuse light is between zero and one, and $Fade$ is constant, as sf increases the intensity of LEAN's highlight decreases.

A. 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 rates for the terrain model in Figs. 11 and 13 were ranging from 600 to 710 fps, depending on the distance of the object to the camera. This model is comprised of 122564 triangles and 62002 vertices. The difference in performance was barely noticeable between the two techniques. Another geometric model used on the tests was a torus (Fig. 9), which is comprised of 5824 triangles and 3718 vertices. Again, the difference in performance was also barely noticeable in this case. Even with the teapot model containing 79664 triangles and 44416 vertices (Fig. 12), the frame rates varied from 550 to 980 fps, depending on the distance, with a maximum difference of 5 fps between techniques.

In our implementation we adapted the equations in the original LEAN shader. Our GLSL fragment program is presented in Listing 1. For the LEAN stage, we generated LEAN map textures (Figs. 2 and 3, and Figs. 6 and 7) as described in [6], and unpacked N (line 5), B, and M terms from the textures (lines 11 and 12). In render time we convert M to Σ (line 14) and compute the LEAN specular term (lines 15 to 19). In turn, 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). The final color is computed by modulating

the color of the light (line 32) with the diffuse and specular components of the material.

Listing 1. GLSL pseudocode for specular fading applied as an extension of the 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 exponent.
7  // This is done here to remove the dependency 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;
```

IV. Results

While comparing LEAN mapping with the proposed model, it is possible to notice the difference on specular highlight when the distance between the camera and the object changes. The LEAN maps generated for our tests are presented in Figs. 2 and 3, and in Figs. 6, and 7. They were used in render time to produce the images in Figs. 4, 5, 8, 9, 10, 11, and 13. Those results illustrate the usage of specular fading, LEAN and bump mapping.

In all figures we placed the object at two or three different distances from the viewer. By doing so it is possible to visually inspect the difference between conventional LEAN technique and the LEAN mapping enhanced with the proposed specular fading. Fig. 12, which also used the maps in Figs. 2 and 3, presents a closer look on the teapot model.

As illustrated by those examples, when using specular fading, the farther the object is from the camera, the lower the specular highlight is. On the other hand, with conventional LEAN, the highlight of distant objects does not reproduce the expected attenuation in function of the dispersion of the light in the medium.

Fig. 9. Comparison between Bump Mapping (left column), LEAN mapping (center), and Specular Fading (right column) using a torus model composed of 5824 triangles and 3718 vertices.

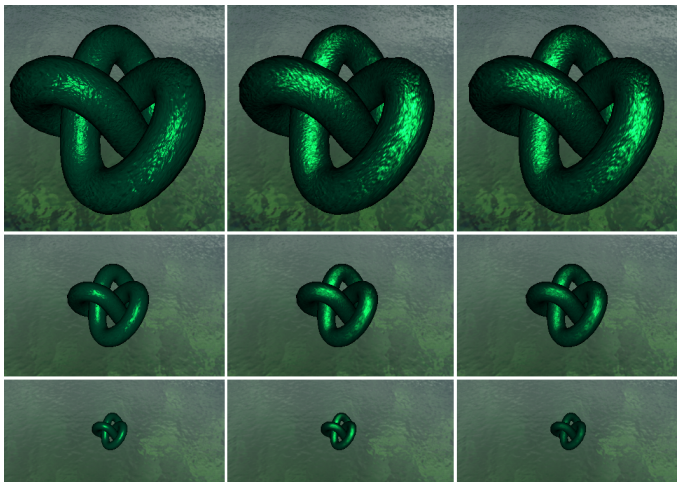

Fig. 10. Same comparison from Fig. 9 but with different colors and bumps.

Our specular fading also corrects the specular highlight from fog algorithms, which only changes the alpha component making it transparent over distance. Notice that our approach only attenuate the specular highlight, so it can also be used in conjunction with distance fog, allowing the fog algorithm to fade the object while changing its specular response.

V. Conclusion

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

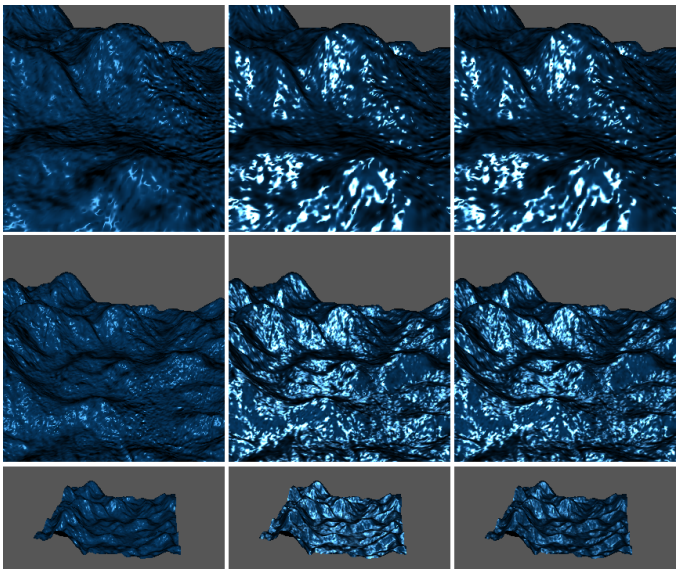

Fig. 11. Comparison between Bump Mapping (left column), LEAN mapping (center), and Specular Fading (right column) using a terrain model composed of 122564 triangles and 62002 vertices.

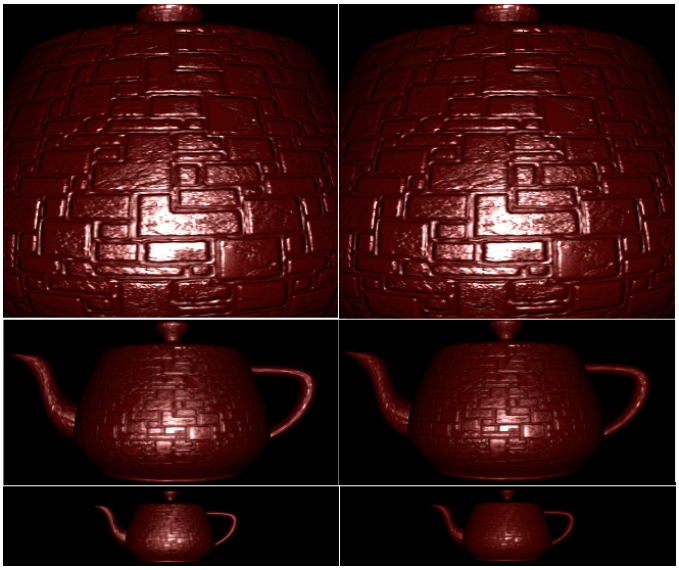

Fig. 12. Comparison between LEAN and our proposed improvement using a teapot model. In the left is the traditional LEAN technique and in the right the proposed improvement. As shown in the figure, at close range they are identical, however the farther the camera is from the object, less specular highlight is noticed.

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.

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. 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. 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. 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. 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. 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. 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. 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

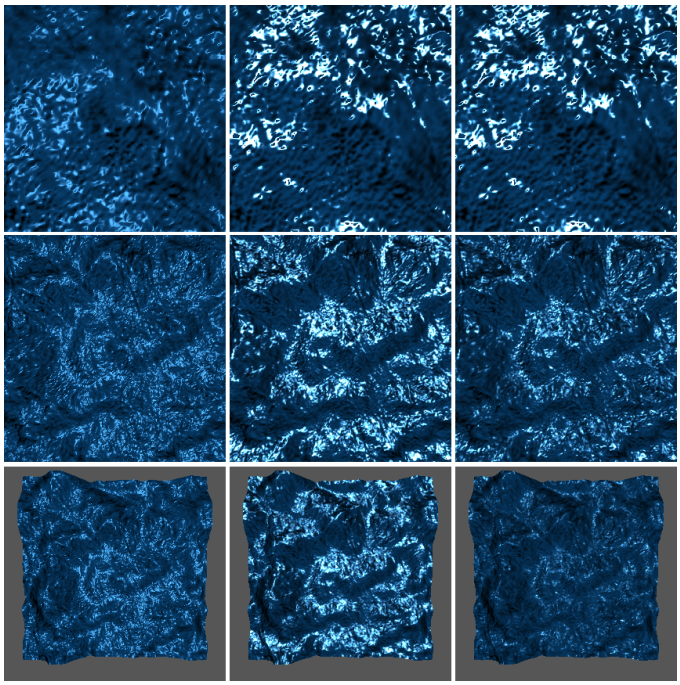

Fig. 13. Same comparison from Fig. 11 but at a different angle.

simplification," in *Proceedings of the 25th annual conference on Computer graphics and interactive techniques*, ser. SIGGRAPH '98. ACM, 1998, pp. 115–122. [Online]. Available: http://doi.acm.org/10.1145/280814.280832

[12] L. Williams, "Pyramidal parametrics," in *Proceedings of the 10th annual conference on Computer graphics and interactive techniques*, ser. SIGGRAPH '83. ACM, 1983, pp. 1–11. [Online]. Available: http://doi.acm.org/10.1145/800059.801126