**Specular Fading over Distance on Wrinkled Surfaces**

Troy C. Kohwalter Esteban W. G. Clua

Universidade Federal Fluminense, Instituto de Computação, Brazil

**Abstract**

In 1978, James Blinn introduced a new way to simulate surface roughness. Later, some changes were made in the proposed approach to allow greater performance and is now commonly used 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 to appear dull. 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 aims to solve this specular highlight intensity by applying a new term during the calculation of the specular component.

**Keywords**: bump, normal, lean, mapping, fade.

**Authors’ contact**:

{tkohwalter, esteban}@ic.uff.br

# Introduction

Artists to create the illusion of certain surfaces or surface structures 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. On this paper, it is used 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 normal map for bumping is extremely common in video games, where the additional surface detail allows for a richer visual experience without highly detailed meshes.

Nowadays, graphics hardware supports bump mapping by providing per-pixel operations such as dot products. Because such per-pixel operations are very flexible, they allow the usage of complex reflection models (KAUTZ; SEIDEL, 2000) instead of only bump maps with diffuse and specular reflections using a *Lambertian* reflection model or *Blinn-Phong* model (BLINN, 1977)(HEIDRICH; 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 saw 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; 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 but be bumped but shiny.

What this paper proposes is to apply a fog-like effect in order to reduce the specular highlight according to the distance of the object. For this aim, it was used the LEAN mapping in order to avoid the blurring of the bump surface and applied a modification to make the necessary change in the specular highlight. Note that this only affect the specular term, so it can be used in conjunction of 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 aiding the understand of the proposed method. Section 3 presents the method proposed on this paper. Section 4 show the results of the method and Section 5 show the implementation. Finally, Section 6 presents the conclusion of this work.

# Related Work

There are many ways to represent bumps in an object's surface. The first implementation originated from Blinn (BLINN, 1977), which implements a simple shading model based on microfacets. This model is a modification of the Phong model (PHONG, 1975), making it more visually appealing. But due to the way it was proposed, other ways to represent surface bump were adopted because they offer better results and are more flexible. One of such is the storage of the surface normal in texture maps (HEIDRICH; SEIDEL, 1999) (KAUTZ; 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; TORRANCE, K. E., 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.

This section will briefly explain the basics of Bump Mapping, then the more used model of Normal Mapping and the Linear Efficient Antialiased Normal Mapping (LEAN) model in which this work was based on. Since this work is to improve the LEAN model, it will also be presented 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) invented 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 1 illustrates the process of generating a wrinkled surface using bump map and Figure 2 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; MAX, N. L., 1993)(WESTIN *et al.*, 1992).

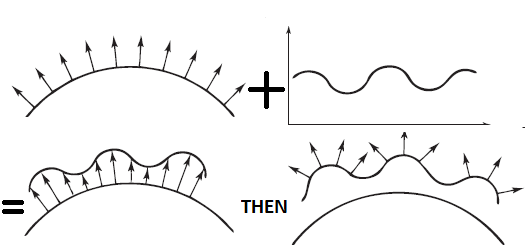


Figure 1: Bump Mapping. Smooth Surface + Wrinkle Function = Wrinkled Surface and then perturbed Normals. Figure adapted from "Simulation of Wrinkled Surfaces".

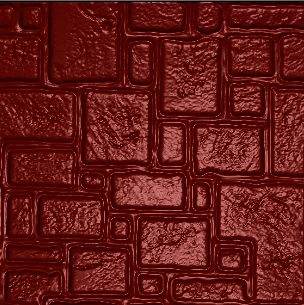


Figure : Example of Bump Mapping

## 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. So, 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 3 illustrates a typical example of Normal Mapping.

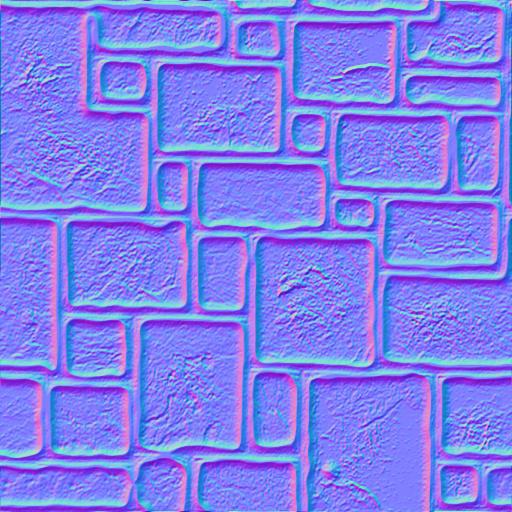


Figure : Typical example of Normal Map

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 4 illustrates the usage of such technique for a model using only 975 triangles. The left model is using a normal map created with a higher resolution surface and applying in the low resolution, and the right is using per-vertex normals.



Figure : Normal map comparison. Figure adapted from "Appearance-Preserving Simplification"

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 which 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 was observed earlier, Bump Mapping has a highlight problem. To deal with it, a new technique was developed called LEAN mapping (OLANO; 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 Blinn-Phong (BLINN, 1977) or Beckmann-distribution (BECKMANN; SPIZZICHINO, 1963) based shading models. 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 adopted for use for 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 by *b.z*. Using standard filtering of the resulted textures sampling of these five values will give as a result an antialiased and filtered blend of the bumps and specular shading. Since to store five values in textures will need 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 the quality of the diffuse filtering, instead of reconstructing it from . This method also allows for diffuse filtering using non-normalized normals after texture filtering, as observed at (KILGARD, 2000).

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 is the equation 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

Lastly, compute the specular term using the following equation:

Equation : LEAN specular term

Figure 5 shows the difference of the techniques LEAN mapping and Bump mapping.

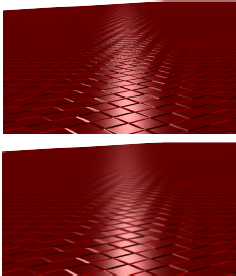


Figure : Bump Mapping versus LEAN Mapping: First picture is generating an ordinary bump map. Second picture was using LEAN mapping with trilinear MIP filtering. Figure adapted from LEAN Mapping article.

## Distance Fog

Distance Fog, which sometimes is called 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 can have as a result 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 6 illustrates an example of the fading effect from distance fog.



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.

Distance fog, or Range-based fog, is a more accurate way to determine the fog effects. In range-based fog, it is used the actual distance from the viewpoint to a vertex for its fog calculations and increases 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. There are three equations in Direct3D used for fog: One linear and two exponential fog equations.

Linear fog equation, know 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 used performs its own transformation and lighting, it is possible to insert the fog factor values manually to be applied by the system during rendering. When calculated the fog effects, the fog factor from one of the preceding equation is used in a blending equation. The blending equation below effectively scales the color of the current polygon C by the fog factor f, and adss 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:

# Method

The proposed shading model in this paper is a variation of the LEAN mapping technique for Bump Mapping. 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, specular are also maintained, even on huge distances, as shown in Figure 7. 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.



Figure : LEAN mapping specular highligh is present on huge distances. Notice the highlight at the upper right corner of the picture. Figures extracted from Lean Mapping paper.

To deal with this problem, a new term was added in 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 term varies according to the camera distance from the object. Both the minimum and maximum highlight specular intensity is configurable, and on close range is identical to LEAN. This new configurable term is added as a power of the dot product between *Normal* and *Light* vectors.

Inserting this new term will allow for a specular control over distance, which is made during the shading process without much overhead. Also, unlike fog that changes the alpha according to the distance and thus preserving the specular highlight, this term actually reduces it without affecting the alpha component. Below is the equation for it:

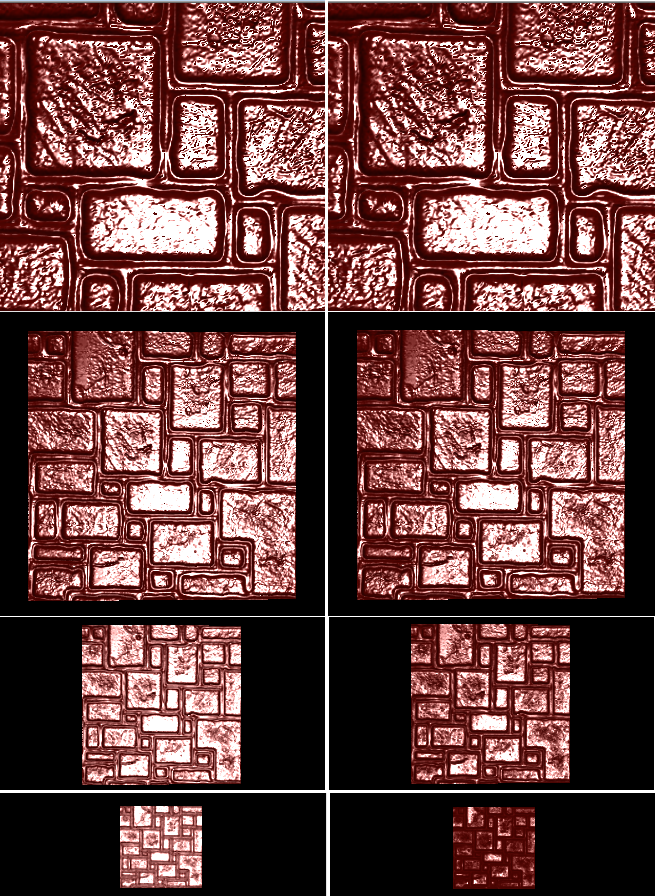
Where *V* is the view vector, *Fade* is a parameter to configure the fading rate, or adjust scale, and *maxP* is the maximum specular fade desired by the user. This term is applied as shown below:

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

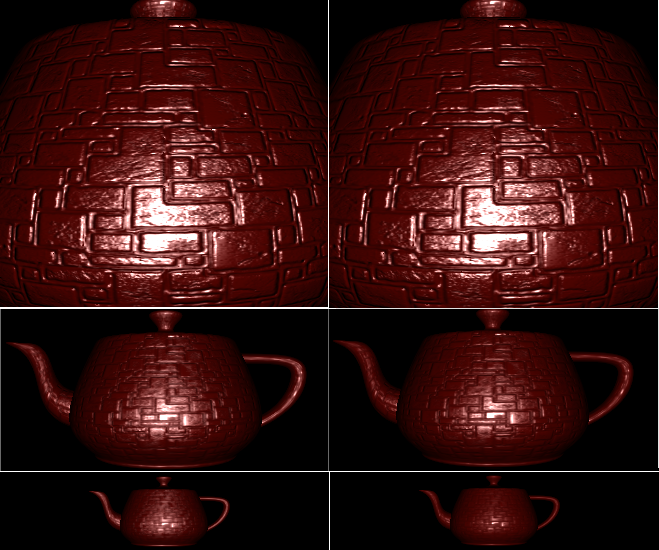
# Results

Comparing LEAN map with the proposed model, it is possible to notice the difference on specular highlight when the distance between the camera and the object changes. Figure 8 illustrates this difference, and 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 6, which maintains the highlight while fading.

Tests were run in order to compare the new model with the traditional LEAN technique. The computer which ran the tests use a geforce GTX550Ti, Windows 7 64-bit. The recorded frames per second were ranging over 1000 to 1600 FPS for both methods, which depends with the range of the camera to the object, with only a difference of 10 FPS. The model used on 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.



(a)



(b)

Figure : Comparison of LEAN with the 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 (a) represents a quad and (b) teapot.

# 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 9.

*[…]*

*//Calculate LEAN variables and the result spec*

*…*

*[…]*

*//Specular fading*

*אּ = 1.0 + (pow(V.z \* Fade, 2.0) );*

*אּ = clamp(אּ, 1.0, maxPower);*

*//Calculate Diffuse and Specular components*

*Td = Kd \* NdotL;*

*Ts = Ks \* pow(NdotL, אּ) \* spec;*

*[…]*

*//Fresnel term*

*Fresnel = clamp(FresnelBias + (FresnelScale \* pow(1.0 - dot(N, V), 5.0)), 0.0, 1.0);*

*Ts \*= Fresnel;*

*//Calculate final color*

*color = (Td + Ts).xyz \* LightColor.xyz;*

*[…]*

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

# Conclusion

This paper presented techniques for bump mapping and their corresponding problems and the specular highlight problem 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 be used in real time applications.

# Acknowledgements

The authors would like to thank (omitted) and (omitted) for the financial support of this work. Also for (rest is omitted because I did not write it yet).

# References

ASHIKHMIN, M.; PREMOZE, S.; SHIRLEY, P. A Microfacet-based BRDF Generator. **In: Proceedings SIGGRAPH ’00**, p. 65-74, 2000.

BECKER, B. G.; MAX, N. L. **Smooth transitions between bump rendering algorithms**. , SIGGRAPH ’93. New York, NY, USA: ACM. , 1993

BECKMANN, P.; SPIZZICHINO, A. **The Scattering of Electromagnetic Waves from Rough Surfaces**. [S.l.]: Pergamon Press, 1963.

BLINN, J. F. Models of light reflection for computer synthesized pictures. **SIGGRAPH Comput. Graph.**, v. 11, n. 2, p. 192–198, jul 1977.

BLINN, J. F. Simulation of wrinkled surfaces. **SIGGRAPH Comput. Graph.**, v. 12, n. 3, p. 286–292, ago 1978.

CABRAL, B.; MAX, N.; SPRINGMEYER, R. Bidirectional reflection functions from surface bump maps. **SIGGRAPH Comput. Graph.**, v. 21, n. 4, p. 273–281, ago 1987.

COHEN, J. **Appearance-preserving simpli\_cation of polygonal models**. [S.l.]: The University of North Carolina at Chapel Hill, 1998.

COHEN, J.; OLANO, M.; MANOCHA, D. **Appearance-preserving simplification**. , SIGGRAPH ’98. New York, NY, USA: ACM. , 1998

COOK, R. L.; TORRANCE, K. E. A Reflectance Model for Computer Graphics. **ACM Trans. Graph.**, v. 1, n. 1, p. 7–24, jan 1982.

HEIDRICH, W.; SEIDEL, H.-P. **Realistic, hardware-accelerated shading and lighting**. , SIGGRAPH ’99. New York, NY, USA: ACM Press/Addison-Wesley Publishing Co. , 1999

KAJIYA, J. T. **Anisotropic reflection models**. , SIGGRAPH ’85. New York, NY, USA: ACM. , 1985

KAUTZ, J.; MCCOOL, M. D. **Interactive rendering with arbitrary BRDFs using separable approximations**. , SIGGRAPH ’99. New York, NY, USA: ACM. , 1999

KAUTZ, J.; SEIDEL, H.-P. **Towards interactive bump mapping with anisotropic shift-variant BRDFs**. , HWWS ’00. New York, NY, USA: ACM. , 2000

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

OLANO, M.; BAKER, D. **LEAN mapping**. , I3D ’10. New York, NY, USA: ACM. , 2010

PEERCY, M.; AIREY, J.; CABRAL, B. **Efficient bump mapping hardware**. , SIGGRAPH ’97. New York, NY, USA: ACM Press/Addison-Wesley Publishing Co. , 1997

PHONG, B. T. Illumination for computer generated pictures. **Commun. ACM**, v. 18, n. 6, p. 311–317, jun 1975.

WARD, G. J. Measuring and modeling anisotropic reflection. **SIGGRAPH Comput. Graph.**, v. 26, n. 2, p. 265–272, jul 1992.

WESTIN, S. H.; ARVO, J. R.; TORRANCE, KENNETH E. **Predicting reflectance functions from complex surfaces**. , SIGGRAPH ’92. New York, NY, USA: ACM. , 1992

WILLIAMS, L. Pyramidal parametrics. **SIGGRAPH Comput. Graph.**, v. 17, n. 3, p. 1–11, jul 1983.