RECONSTRUCTION FILTERS FOR BUMP MAPPING

Anders Hast

Creative Media Lab
University of Gävle, Kungsbäcksvägen 47, S-801 76 Gävle, Sweden. aht@hig.se

Tony Barrera

Cycore AB
Dragarbrunnsgatan 35, P.O. Box 1401, S-751 44 Uppsala, Sweden

Ewert Bengtsson

Centre for Image Analysis
University of Uppsala, Lägerhyddsvägen 17. S-752 37, Uppsala, Sweden. ewert@cb.uu.se

ABSTRACT

Textures that are magnified appears jagged, and therefore reconstruction filters using linear or cubic interpolation polynomials are often used to reduce aliasing. The same aliasing problem is noticed for bump mapping, where it is often solved by bilinear interpolation of the height map, in order to obtain intermediate values. Low-pass filtering of the height map is not suitable for the magnification problem since it will suppress the bumps. It will be shown that reconstruction filters can be used to interpolate gradients, rather than height values directly. The goal is to produce non jagged bumps, without removing high frequency details by low-pass filtering.

Keywords: Bump Mapping, Antialiasing, Reconstruction filters

1 INTRODUCTION

Bump mapping was introduced by Blinn [Blinn78] as a method to make surfaces appear rough or wrinkled without increasing the number of polygons. Instead, the normals used in the lighting computations are perturbed to achieve this effect. A bump map is used to store the information about the height. Blinn used a bilinear interpolation scheme of four sample points from this map to obtain intermediate values. By computing four such values from this map, the gradients in the parametric, u and v directions, are obtained. Hence, this method require a total of 16 sample points. The surface will look smooth, but as noticed by Foley et al. [Foley97] and Schilling [Schil97], this is because low-pass filtering in general will smooth out the bumps. That is, filtering will remove bumps, which is undesirable. In this paper it will be shown how reconstruction filters can be used for bump mapping in order to produce non jagged bumps, without flattening them.

1.1 Anti aliasing of Bump Maps

Blinn use an approximation for the perturbed normal, which depend on the surface normal and the height information in the bump map. Other approaches are discussed by Ernst et al. [Ernst98] In general, the bump map normal can be obtained from the height map as:

$$\mathbf{N}' = (-F_u, -F_v, 1), \tag{1}$$

where F_u and F_v are the gradients of the height map.

Some efforts, trying to remove aliasing from bump mapped objects have previously been described. Schilling [Schil97] use a roughness pyramid that contain roughness information for the texels. The roughness is computed from a covariance matrix containing squares of the standard deviation of the gradients F_u and F_v. The distribution of the normal vectors could be obtained from this matrix. This roughness information is used to change the exponent in the Phong-Blinn model [Blinn77]:

$$I_s = (\mathbf{N} \cdot \mathbf{H})^n \tag{2}$$

The effect will be a less specular surface for pixels that are more rough.

Becker and Max [Beck93] use the derivative of a cubic B-spline to reconstruct the gradients for bump mapping. Eight values are needed for each gradient. Two quadratic interpolations are done in each direction, which then are linearly weighted together. This approach will be taken a step further in this paper, by showing how a general reconstruction filter that will give intermediate gradients, can be formulated. It will be shown that bi-cubic interpolation will require sixteen values and that bilinear interpolation will require seven, eight or twelve values depending on how it is implemented. Fewer values will require a larger map, since it implies that values are precomputed.

1.2 Reconstruction filters

Reconstruction filters are often used for texture mapping. When the texture is sampled, several techniques could be used, as described in the standard textbook in image processing by Gonzales and Woods [Gonz97]. The *nearest neighbor* method is the fastest but gives the worst quality, since the u and v values are truncated and then these integer values are used to sample the texture map. *Bilinear interpolation* will use the fraction of the u and v coordinates in order to obtain an interpolation between four neighboring texels. The approach of taking more than one sample and then combining them is known as *super sampling.*

When an image is magnified the texels will appear as angular squares in the image, i.e. the image appears jagged. Bilinear interpolation could be used to diminish this effect. The individual texels are themselves regular samples of some texture, maybe a photograph of a real life object. Bicubic interpolation will do a better job to reconstruct the details in the image that are missing, which lies between the texels. Mitchell and Netravali [Mitch88] show how piecewise cubic polynomials could be used as reconstruction filters. An overview of different spline filters is given by Unser [Unser99].

2 RECONSTRUCTION OF BUMP MAPS

For image reconstruction a suitable family of the cubic curves, or any other order for that matter, can be chosen and then applied in two dimensions as this tensor product form shows:

$$f_c(x, y) = \sum_{n=0}^{3} \sum_{m=0}^{3} f(n, m)\kappa(x - n)\kappa(y - m). \tag{3}$$

This equation is separable, which means that we could use an one dimensional equation for each row to obtain four values that are then used in one computation for the columns. This is principally the same equation used for spline surfaces by Hill [Hill01] and Hearn and Baker [Hearn97]. However, the matrix form will be used subsequently:

$$F = \mathbf{U} \cdot \mathbf{M} \cdot \ \mathbf{G} \cdot \mathbf{M}^T \cdot \mathbf{V}^T, \tag{4}$$

where $\mathbf{U} = (u^3, u^2, u, 1)$, $\mathbf{V} = (v^3, v^2, v, 1)$, $\mathbf{M}$ is the basis matrix that tells which family of splines that are used, and $\mathbf{G}$ is the geometry matrix which contains the control points, or as for the case of bump mapping, the height values. If u' and v' are the coordinates in the height map then $u = u' - \lfloor u' \rfloor$ and $v = v' - \lfloor v' \rfloor$. To obtain the partial derivatives we must differentiate in the u and v directions respectively:

$$F_u = \frac{\partial \mathbf{U}}{\partial u} \cdot \mathbf{M} \cdot \ \mathbf{G} \cdot \mathbf{M}^T \cdot \mathbf{V}^T, \tag{5}$$

$$F_v = \mathbf{U} \cdot \mathbf{M} \cdot \ \mathbf{G} \cdot \mathbf{M}^T \cdot \frac{\partial \mathbf{V}}{\partial v}^T. \tag{6}$$

Note, that these equations are valid for any degree of polynomials used for interpolation or reconstruction. Also note, that the sums of the blending polynomials are zero for the partial derivatives. Moreover, for a reconstruction filter for images the sum of the blending polynomials would be equal to 1, which assures that the output signal is constant if the input signal is constant.

3 COMPARISON OF RECONSTRUCTION FILTERS

If the reconstruction filter is sampled at integer points, the original samples are obtained. If the original samples are not obtained, then the reconstruction filter will not reconstruct the original signal properly. Such filter is said to perform low-pass filtering below the Nyquist limit, and could

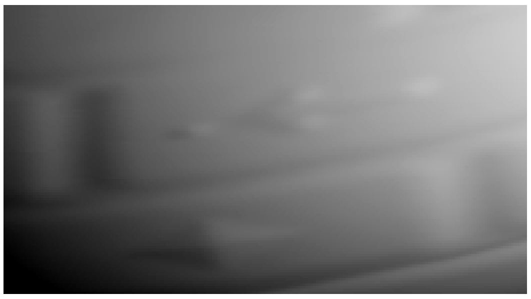
Figure 1: A bump-mapped torus sampled using B-spline reconstruction filtering

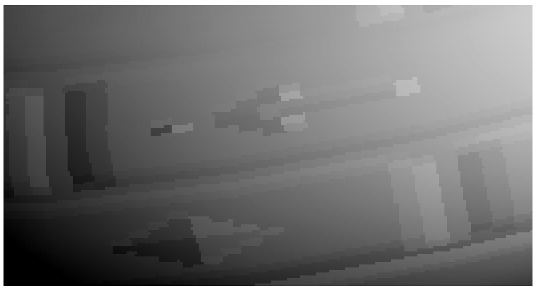
Figure 2: The torus sampled using the nearest neighbor method

be useful if we want to smoothen the original signal. Cubic B-splines and Catmull-Rom spline filters will be examined to ascertain this property. The effect of different filtering techniques will be shown in a number of images. A torus is bump mapped and a portion of the torus is magnified so that the effect of the filters could be compared. The bump that the arrow in the images is pointing at is a one texel bump, that is, one height value differs from the others surrounding it.

3.1 B-spline Filters

B-splines do not intersect the control points, but they have C^2 continuity. Therefore, the height map is filtered below the Nyquist limit. Figure 1 shows how the bumps are much smoother than the bumps in Figure 2 where the nearest neighbor technique is used.

3.2 Catmull-Rom Filters

Catmull-Rom [Catm74] splines have only C^1 continuity, but on the other hand, they do intersect the control points. Moreover, Catmull-Rom filtering will not low-pass filter the height values as B-spline filters will. The bumps will not be suppressed, although, the jaggedness apparent in Figure 2 for the nearest neighbor method is removed. The result for Catmull-Rom filtering is shown in Figure 3.

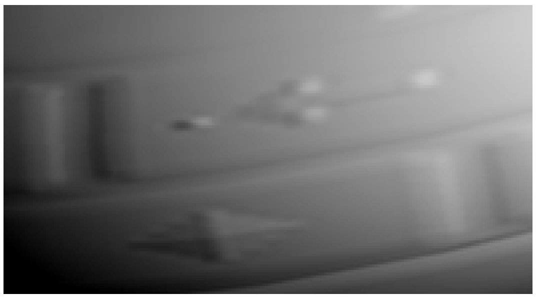
Figure 3: A bump-mapped torus sampled with Catmull-Rom reconstruction filtering

Figure 4: The torus sampled with bilinear interpolation of gradients

3.3 Bilinear Interpolation

It can be shown that bilinear interpolation of the height map will yield gradients that will only depend on one variable. The result will therefore be poor. However, a bilinear interpolation that will depend on both u and v for the derivatives can be considered as a bilinear interpolation between four normals. Equation (1) shows that this interpolation could be reduced to an interpolation of gradients as shown in fig 5:

$$\begin{aligned} F_u =& b-a+u(a-2b+c)+ \\ & v(a-b-d+e)+ \\ & uv(-a+2b-c-2e+d+f), \end{aligned} \tag{7}$$

$$\begin{aligned} F_v =& d-a+v(a-2d+g)+ \\ & u(a-b-d+e)+ \\ & uv(-a+b+2d-2e-g+h). \end{aligned} \tag{8}$$

It is possible to reduce the computation, at the cost of having a larger map, by storing precomputed values. Note, that the v-term of F_u and the u-term of F_v are equal. Therefore, only seven precomputed values have to be stored in a map using this scheme. Often normalized normals are stored in a normal map, but then one bilinear interpolation per element must be performed and the map will be three times larger than the height map. It is also possible to store the two gradients instead of a normal. The map will only be twice as large as the height map and eight values in total have to be retrieved from the map in order to

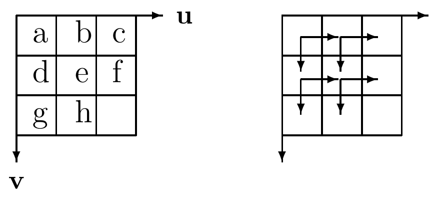

Figure 5: Eight sample points in a local neighborhood of the height map, and the corresponding gradients

perform bilinear interpolation. However, the gain will be very small since only eight values need to be fetched in order to compute these eight gradients anyway. Which scheme to use will heavily depend on the current hardware used. The result of using bilinear interpolation of two gradients is shown in Figure 4.

4 CONCLUSIONS

Several reconstruction filters and interpolation techniques for height maps, have been investigated. B-splines will perform low-pass filtering, and should not be used if this is not the purpose. However, they will produce smooth bumps due to the C^2 continuity of the spline. Cubic Catmull-Rom filters will give good results, since they will not low-pass filter the height map. Moreover, they will produce smooth bumps because of the C^1 continuity. The disadvantage with both of these reconstruction filters is that they are computationally expensive. Bilinear interpolation is computationally faster, but must be done for each gradient to give good results. Here, the original map could be used, or a map containing precomputed values. Which of the methods is the fastest will depend on the hardware, since some maps are quite large and may not fit into the cache or texture memory.

4.1 Future Work

Other filter types should also be investigated, as well as the effect of using these filters in combination with Mip-maps and Roughness maps. Special processor architectures like SIMD technology could make cubic filters more attractive, and therefore the impact of SIMD should be ascertained.

REFERENCES

[Beck93] B. G. Becker, N. L. Max, *Smooth Transitions between Bump Rendering Algorithms*, In Proceedings SIGGRAPH, pp. 183-190. 1993.

[Blinn77] J. F. Blinn, *Models of Light Reflection for Computer Synthesized Pictures*, In Proceedings SIGGRAPH, pp. 192-198. 1977.

[Blinn78] J. F. Blinn, *Simulation of Wrinkled Surfaces*, In Proceedings SIGGRAPH, pp. 286-292. 1978.

[Catm74] E. Catmull, R. Rom, *A Class of Local Interpolating Splines*, Computer Aided Geometric design, pp. 317-326. 1974.

[Ernst98] I. Enrst, H. Rüssler, H. Schultz, O. Wittig *Gouraud Bump mapping*, Workshop on Graphics Hardware, pp. 47-53. 1998.

[Foley97] J. D. Foley, A. van Dam, S. K. Feiner, J. F. Hughes, *Computer Graphics - Principles and Practice*, Addison-Wesley, pp. 617-646, 744. 1997.

[Gonz97] R. C. Gonzales, R. E Woods, *Digital Image Processing*, Addison-Wesley, pp. 300-302. 1993.

[Hearn97] D. Hearn M. P. Baker, *Computer Graphics - C version*, Prentice Hall, pp. 345. 1997.

[Hill01] F.S. Hill JR, *Computer Graphics - using OpenGL*, Prentice Hall, pp. 659. 1997.

[Mitch88] D. P. Mitchell, A. N. Netravali, *Reconstruction filters in Computer Graphics*, Computer Graphics, Vol. 22, No. 4, pp. 221-228. 1988.

[Schil97] Andreas Schilling, *Towards Real-Time Photo Realistic Rendering: Challenges and Solutions*, In Proceedings SIGGRAPH/EUROGRAPHICS workshop on Graphics Hardware, pp. 7-15, 1997.

[Unser99] M. Unser, *Splines - a Perfect Fit for Signal and Image Processing* , IEEE Signal Processing Magazine , Vol. 16 No. 6, pp. 22-38, Nov. 1999