Filtering Normal Maps and Creating Multiple Surfaces

Alain Fournier

Department of Computer Science
University of British Columbia
Vancouver, BC
V6T 1W5
Canada
fournier@cs.ubc.ca

Abstract

"Bump" mapping is a variant of texture mapping where the texture information is used to alter the surface normal. Current techniques to pre-filter textures are all relying on the fact that the texture information can be linearly "factored out" of the shading equation, and therefore can be pre-averaged in some way. This is not the case with bump maps, and those techniques fail to filter them correctly. We propose here a technique to pre-filter bump maps by building a pyramid where each level stores distributions of normal vectors reconstructed from the distribution given by the underlying bump map. The distributions are represented as sums of a small number of Phong-like spreads of normal vectors.

The technique, besides allowing an effective and smooth transition between a bump map and a single surface description, gives rise to the concept of a *multiple surface*, where each point of the surface is characterized by more than one normal vector. This allows the description of visually complex surfaces by a trivial modification of current local illumination models.

When a surface has an underlying microstructure, masking and self-shadowing are important factors in its appearance. Along with the filtering of normals we included the filtering of the masking and self-shadowing information. This is accomplished by computing the limiting angles of visibility and their variance along the two texture axes for the reconstructed distribution of normals.

These techniques allow the modelling of any surface whose microstructure we can model geometrically. This includes complex but familiar surfaces such as anisotropic surfaces, many woven cloth, and stochastic surfaces.

CR Categories: I.3.3 [**Computer Graphics**]: Picture/Image Generation I.3.5 [**Computer Graphics**]: Computational Geometry and Object Modeling I.3.7 [**Computer Graphics**]: Three-Dimensional Graphics and Realism

General Terms: algorithms, models.

Additional Keywords and Phrases: local illumination, bump map filtering, multiples surfaces.

1. Introduction

To achieve any degree of realism in computer generated images, shape modelling is only the beginning of the struggle. The other two fundamental steps are *modelling the interaction with light* and *animation*. Of course these steps cannot always be neatly separated. In many cases realistic animation cannot be achieved unless the motion is built into the geometric model, and there are complex interactions between shape models and illumination models. In fact the appearance of most objects results from such interactions between their *shape*, as modelled by geometry, and the way they reflect light. This dichotomy is only for descriptive convenience. This paper addresses more particularly the modelling of local interaction between material surfaces and light. It is easy to overlook the importance of these effects for our interpretation of the world, the subtlety of our visual system and the distance between our current illumination models and the

behaviour of real surfaces. As an example, consider how easily we recognize material such as velvet, silk or denim, by their colour and reflectivity structures, even at large distances where the details are not distinguishable (see Figure 1).

Figure 1.
Types of material (real) (see colour print)

1.1. Scales

The type of interactions between light and matter is largely dependent on the scale at which we consider the material. Even if we keep to a level of details well above the wavelength of light to avoid considering interference and diffraction, we can either observe the effect of individual surface details or merely the average of such, depending on the scale and viewing distance from the surface. Taking woven cloth as an example, we might have to consider a geometric model for each strand of material, each yarn, each thread, or the average of the whole weaving pattern, depending on viewing conditions.

For most levels of details modelling and rendering techniques have been developed in computer graphics. From top to bottom, the usual levels are the *geometric model* (such as polygons, parametric surfaces), *displacement maps*, *bump maps* [Blin78a, Max88a], and the *local reflection models*[1].

Traditionally, reflection models have been divided into two components: *diffuse* and *specular* [Torr66a, Blin77a, Cook82a, Beck87a]. The diffuse component takes into account the light that inter-reflects onto elements of a same surface and is reemitted equally in all directions. To model the specular reflection, Torrance and Sparrow[Torr66a] assume that the surface is made of highly reflective microscopic facets distributed in v-grooves. If the facets are randomly distributed over the surface, shadowing and masking functions can be statistically estimated, and, for a given distribution of slopes of the facets, the light reflected in a particular direction can be approximated. This is a clear example where the local illumination model is based on some model for the geometry of the surface at a smaller scale than the visible details. One of course can go further. In recent work to improve the generality of illumination models, Xiao *et al* [He91a] started from Kirchhoff equation for the light reflected at a surface (approximated by its tangent plane at the point considered) and added interreflection and self-shadowing/masking factors derived from statistical techniques (the surface height is assumed to follow a Gaussian distribution). While their model extends considerably the flexibility and accuracy of traditional local models, there are still many surfaces which do not meet these assumptions. The present work, addressing the surface description at a higher level of details, is in fact quite orthogonal to their approach.

In [Poul90a] we used a "hidden" level of geometry made of cylinders to allow the computation of anisotropic reflection. Our anisotropic reflection model corresponds to inserting two new geometric levels between the mapping (displacement or bump) and the isotropic reflection model. The isotropic reflection model, like the facets model of Torrance and Sparrow, characterizes the surface nature of each cylinder. Set of groups of adjacent cylinders can be used to represent yet another intermediary level.

While all these techniques are effective within their intented scale, a major unsolved problem is to allow their simultaneous use, and in particular ensure smooth transition between models when the

1. It is important to note that *local* is a relative term. In practice local means at a scale of the order of the area which is to be represented by a single illumination computation.

scale changes. For instance in our anisotropy model one cannot see the individual cylinders, no matter how close one gets to the surface. Another way to formulate the same problem is that one should be able to find a way to represent the average effect of one level in terms of the level above. As it will turn out, one very constraining, though unnecessary, assumption made in all the local models is that a single normal at each sampled point or area has to account for all the reflection effects.

1.2. Local Illumination Models

To begin to solve that problem, it is important to understand what "average" means in this context. The smallest scale we will consider is the one at which we can define and compute an effective *bidirectional reflectance distribution function* (BDRF). Now of course this function can be itself the result of considering a surface with a structure at a scale below the one considered. The important point is that this structure will never be explicit or visible in the rendering. The BDRF is then probably itself an average. The trouble right away is that a complete BDRF is a function of 4 variables $\rho(\theta_v \phi_v \theta_r \phi_r)$, 2 for the incident direction and two for the reflected direction. This can (and most often is) reduced to two variables for *isotropic* surfaces, but most natural surfaces are not isotropic (and even more important we are very sensitive to the case where they are not). To be able to develop models above this, we have to compute the light reflected by a given area of the surface given the local BDRF, the local geometric characteristics of the surface (tangent plane, curvatures), the masking/shadowing and the amount of interreflection. This has to be averaged over the area considered, preferably as a close form solution, but barring this in a form that can be quickly computed. In addition, since at this scale the structure can actually be seen under some viewing conditions, we should be able to vary smoothly from the values given by the original BDRF to the average given by the new model. The same requirements and exercise can be repeated at an higher level.

1.3. Bump Maps and Filtering

Texture and texture filtering has been investigated often in computer graphics, and in addition to the surveys by Greene and Heckbert [Gree86a], Greene [Gree86b], and Heckbert [Heck86a, Heck86b] techniques are described in [Dung78a, Crow84a, Will83a] and [Four88a].

Bump maps is the traditional (if inelegant) name given to texture maps when the values (discrete values of texture maps are often called *texels*) are used to define or modify the normal to the surface prior to shading computations. They were originally introduced by Blinn [Blin78a] and have been widely used since. Unfortunately the filtering techniques listed above do not apply and they have not been successfully filtered yet. The reason is straightforward. All the pre-filtering techniques developed so far rely on the fact that they filter some quantity that can be factored out of the shading computation when the result is averaged over an area or discrete samples. If we represent the local illumination equation as:

$$I_R = I_I f(\mathbf{N},\mathbf{E},\mathbf{L},\mathbf{C},\mathbf{S})$$

for a given surface element dA over which all these terms can be considered to be constant, where I_R is the intensity (this assumes a quantity of dimension Power/Area $\times$ Solid Angle) of the reflected light, I_I the intensity of the incident light, $f()$ the function implementing the reflection model, $\mathbf{N}$ the surface normal, $\mathbf{E}$ the viewing direction (or in general the direction for which the reflected light is computed), $\mathbf{L}$ the direction of the incident light, $\mathbf{C}$ the colour of the surface and $\mathbf{S}$ a vector of surface characteristics (such as albedos, characteristics of normal distributions, etc.). It is important to note that this formulation has already assumed that the effect of the intensity is *linear*, and can be factored out. For a finite area A, the equation becomes:

$$\int_{allA} w(\mathbf{P})\ I_R\ dA = \int_{allA} w(\mathbf{P})\ I_I f(\mathbf{N},\mathbf{E},\mathbf{L},\mathbf{C},\mathbf{S})\ dA$$

where $w(\mathbf{P})$ is a weighting function of the point $\mathbf{P}$ within the area A (its dimension is an inverse of area). Of course usually the integral is wanted because some terms can be considered constant over A, usually at least $\mathbf{E}$, an often $\mathbf{L}$ as well. Any quantity that enters only linearly in $f()$ can

be brought out and integrated separately. This is usually the case of colour [2], whether it is represented in some **RGB** space, spectral samples or linear combinations of basis functions. In this case the equation becomes, if everything else is constant:

$$\int_{allA} w(\mathbf{P})\ I_R\ dA = I_I f(\mathbf{N},\mathbf{E},\mathbf{L},\mathbf{S}) \int_{allA} w(\mathbf{P})\ \mathbf{C}\ dA$$

This is the basis of all the texture filtering techniques using *pre-filtering.*

Such an operation is not possible with normals, since they are used in non-linear formulae by all current illumination models. As a trivial counter example, consider the case where the normals over the area considered are of two kinds, symmetric with respect to the average normal and equally distributed (see Figure 2).

Figure 2.
Averaging the normals

The average of the normals will be the normal of the average plane, and this will cause highlights where the real surface has none, and not show highlights where the real surface has strong ones.

2. Filtering Bump Maps

2.1. Reconstructing a Distribution of Normals

The simple example above suggest a solution. If we somehow knew that the surface has only two values of normal vectors equally distributed, these two normals alone are enough to compute the reflected light at any scale. In general, we can look at the normals over the area as defining an underlying distribution of normals $\mathbf{N}(\theta,\phi)$ defined for each direction (θ,ϕ) on the hemisphere [3]. The problem is to approximate this distribution by a weighted linear sum of a small number of discrete functions, each function being used then to compute their contribution to the total reflected light. One could use a variety of techniques to accomplish this. The main criterion to take into account is the ease with which the functions can be used to compute their contribution to the total reflected light. The ease of fitting is not *per se* a decisive factor, since it only has to be done once for a given texture, and therefore is pre-processing. Classic techniques such as spline interpolation and spherical harmonics were possible candidates. Splines have the advantage of

2. Note that sometimes even the average colour of the surface is affected by self-shadowing/masking; for example consider materials woven with threads of different colours, such as denim or some taffeta.

3. In fact some normals could point below the horizon, but we will assume they cannot be seen.

familiarity and flexibility, but a piecewise approach would have added considerably to the cost of shading. Spherical harmonics are better candidates, but again the cost of shading would have been higher, and in the traditional ways to compute the terms the number of terms required for a given precision seems to be high (this is not based on an actual implementation of spherical harmonics, however).

2.2. Phong Peaks

The method chosen and implemented is to represent the reconstructed distribution of normals as a weighted sum of a small number of *Phong peaks*, that is of functions of the form:

$$\mathbf{N}(\theta,\phi) = \mathbf{U}(\theta,\phi) \sum_{i=0}^{n-1} a_i \cos^{n_i}(\alpha_i)$$

where $\mathbf{U}(\theta,\phi)$ is the unit vector in the direction (θ,ϕ), α_i is the angle between the direction (θ,ϕ) and the direction of peak i: (θ_i,ϕ_i). Each peak has therefore 4 variables to fit, and non-linear least squares are used to compute the fit. In practice n is kept small, ≤ 7, so the number of variables to fit is ≤ 28. It should be noted that Phong peaks are very close to Gaussian distributions especially for high values of n, and the latter could have been chosen as well. They would have had the advantage that they are defined for any angle values, when the Phong peaks are undefined for $\alpha > \frac{\pi}{2}$. The main argument in favour of Phong peaks was familiarity.

The original function $\mathbf{N}()$ is computed by *spreading* the normals given by the initial bump map, or the relevant section of it. Spreading consists in computing for every point on the hemisphere the sum of the density of normals given by the Phong $\cos^n(\alpha)$ formula. In this formulation each normal of the bump map can have an exponent n associated with it. This is consistent with the interpretation of the Phong model as the result of the distribution of the orientation of purely specular *microfacets*, similar to the Torrance-Sparrow model [Blin77a]. In effect the Phong function serves as a reconstruction filter for the definition of $\mathbf{N}()$.

2.3. Non Linear Least Squares

The goal of non-linear least squares is the same than with linear least squares, that is to minimize the residual variance, the sum of squares of the differences between the original data and the fitted data. There are different methods (some of them are described in[Foo188a]) and many routines are available from numerical libraries. They all rely on some form of iteration and therefore one or more stopping criteria. We have used in this work *NL2SOL*, a routine from the IMSL library. It uses automatically a variety of stopping criteria, and has proven to be extremely robust and reliable. In this case the input data points are the values of $\mathbf{N}()$ at the directions around the hemisphere. Discrete samples of the directions are taken in polar coordinates (in all the examples given, at every 5 degrees in θ and ϕ, which gives an array of samples 18x72). In addition, to deal with the singularity at the pole, a special sample point is used for all directions such that θ is less than a given limit (it is equal to 5/12 degree for the case given above, so that the area covered by this case is about half the area of the adjacent sectors).

The main difficulty with non-linear least squares in this application is the necessity to generate automatically reasonable first guesses of solutions. The algorithm could otherwise easily fail to converge, and it behaves in this respect like Newton iteration, and for the same reasons. To avoid this, the first guess is taken to be the position and amplitude of the maximum of $\mathbf{N}()$ (actually its discretized version), and the exponent is estimated from the rate at which the neighbours of that maximum deviate from it. The distribution is then fitted with one Phong peak, and the second guess is made with the peak resulting from the first fit plus the peak estimated from the maximum of the distribution of the residuals. This is then iterated until either both the root mean square of the residuals is less than a fraction of the original root mean square and the absolute maximum of the residual is less than a fraction of the original maximum (in practice we used 5% for both of these ratios), or the maximum number of peaks (7) has been used. If the original number of normals is less than or equal to 7 there is no neeed to do a fit, and these normals are

taken as the answer (with their amplitudes and exponents).

It is remarkable that the bump maps we examined yield a fit to within a few percent under these circumstances.

Figure 3.
Normal distributions and their fits. (see following pages)

Examples in Figure 3 show the original distributions and the fits obtained for a bump map of an hemisphere (equal distribution of normals in all direction), a cylinder (all normal vectors in the same plane, equally distributed in that plane), and a weaving pattern (a denim). The first two examples are actually the most challenging, because there are sensitive to good initial guesses. The sphere shows that 7 peaks are enough for even a wide spread of normals, the cylinder that even a severely restricted distribution can be well modelled. In this case the difficulty is to satisfy a wide distribution in one direction and a very narrow distribution in the perpendicular direction. The denim is more typical of the kind of bump maps that are useful but difficult to deal with so far. Stochastic distributions of normals can of course be deduced from the distribution of heights or other information, but it is not necessarily easy to go from there to an effective shading method, as opposed to this approach.

The building of the pyramid is a time-consuming operation. For the cylinder (64x64) it takes about 1 hour real time on a Personal Iris (with shadowing/blocking computations, see below). For the sphere (32x32) it takes about 1:20 hour. In the sphere pyramid 73 texels needed least square fitting, and 5 of those went to the maximum number of Phong peaks (7). The pyramid for the denim (64x64) took almost 5 hours! 85 texels required fitting, and 63 of those required the maximum number of peaks. Again the pyramid is computed only once for a given texture, and loaded by the display program at run time.

2.4. Building and Using the Pyramid

This reconstruction is performed in a pyramidal fashion, and the organisation of the data is similar to that used in [Will83a] and [Four88a]. One difference from MIP maps is that each level of the pyramid is not computed from the preceding one, but from the bottom level (the original distribution). The examples of fits given above are actually those obtained for the top of the pyramid, where the resolution is 1x1. At each level each cell stores a distribution of normals representing all the normals below it in the hierarchy.

To use the pyramid, the appropriate level is selected based on the size of the pixel (or ray "extent" in ray tracing) in texture space. Of course for this purpose the mapping of the texture on the object and the object scale have to be taken into account. Then the relevant texel is selected, and the distribution for this texel, in the form of the Phong peaks and their coefficients, is used for shading. This of course integrates easily into any current shading software/hardware. In practice for smooth transitions one has to interpolate between texels and between levels as is done for other pyramidal structures. The important difference is that the contributing peaks are not averaged (this would take us right back to the original problem), but their amplitude a_i is multiplied by the suitable interpolation weights, and they are all used to compute the shading. If a maximum of 7 peaks is used for each texel, each level can contribute 28 peaks if bilinear interpolation between 4 texels is used, and 56 peaks can be used between two levels. This might seem high, but we will see that the additional shading cost is actually not very high.

3. Illustrations

The initial data for the textures comes in all our examples from a sampling of normals generated by a ray-tracer in an orthographic view of the geometric model. The texture sizes are rather small, in our examples 32x32 or 64x64. The amplitude of the sampled normals is corrected by the cosine of the angle between the line of sight and the normal, so that the sampling is weighted correctly as a function of the object surface area.

Figure 4 shows a cylinder mapped on a plane, unfiltered, filtered and super sampled (with 16 samples). The filtered version still exhibits some aliasing because the method to estimate the ray extend is not very accurate when the ray is very slanted with respect to the surface. It is clear that a high number of super samples would be necessary to remove all the aliasing in the super-sampled version. It should also be noted that even in the unfiltered version there is a bilinear interpolation between four texels of the original texture.

Figure 4.
Cylinder texture on plane (see colour print)

Figure 5 shows the same variants with a sphere texture.

Figure 5.
Sphere texture on plane (see colour print)

In Figure 6 the sphere texture is mapped on a sphere. In this case the mapping is non-affine, and care has to be taken to take the Jacobian into account. Because of the relatively large scale of the texture, the differences are less dramatic. It can be seen that the texture is slightly more aliased near the north pole with filtering than with super sampling. The aliasing is actually from insufficient sampling of *directions* in the reconstructed distribution of normals.

Figure 6.
Sphere texture on Sphere (see colour print)

Figure 7 shows the denim texture, again unfiltered, filtered and super sampled. It should be noted than the threads of the weave were made quite specular, which is not the case for real denim. Figure 8 shows the same texture on a model of drapery (the geometric model is from Frederic Taillefer). Note that the geometry is rather coarse (a 11x40 grid).

The following table gives some timings for these images, all in minutes, 512x512 resolution, on a 20 MIPS processor with 16 MB. There was no attempt to make the code efficient, but it should be noted that filtering about doubled the cost of using the same texture without filtering; on the

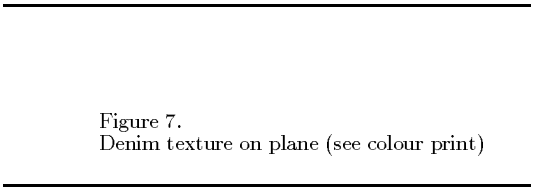

Figure 7.
Denim texture on plane (see colour print)

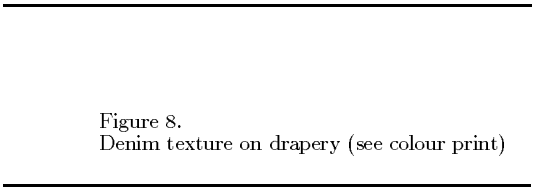

Figure 8.
Denim texture on drapery (see colour print)

other hand, supersampling (at 16 samples per pixel) predictably multiplied the cost by 16 for a picture usually exhibiting more aliasing.

Image	No texture	Unfiltered	Filtered	Supersampled
Cylinder on plane	1:03	3:30	7:38	52:39
Sphere on plane	1:03	3:02	5:39	46:14
Sphere on sphere	0:45	1:37	2:30	26:43
Denim on plane	1:03	3:34	8:34	53:06

4. Multiple Surfaces

The building of the pyramid has a not totally unexpected side benefit: at each texel of each level of the pyramid one gets a local illumination model, and at the top level one gets a model valid when all of the original bump map is comprised within the local "sample". The model is expressed simply as a surface with more than one normal per sample, which we can call a *multiple surface*. The description of the multiple surface implies a new BDRF extracted from the bump map. The BDRF can be obtained by computing the reflected intensity for every direction o f incoming and reflected light. It is not as general as an arbitrary BDRF but it has the advantages of being very compact and made of simple elements, namely Phong peaks. This makes the shad - ing computation with this model a trivial modification of the computation normally done with a single Phong peak.

Of course values for multiple surfaces do not have to be extracted from bump maps. One can deduce them in simple cases, such as a sphere or a cylinder, or invent them for "special effects" . For example Figure 9

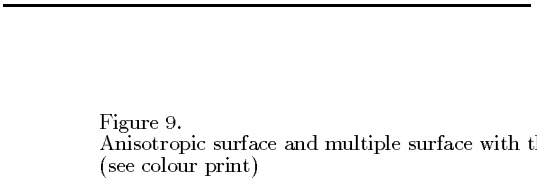

Figure 9.
Anisotropic surface and multiple surface with three normals
(see colour print)

shows a sphere with an anisotropic surface and with a multiple surface with three normals: one with amplitude 0.4, direction (0.6, 0, 0.8) and exponent 100, an other with amplitude 0.4, direction (-0.6, 0, 0.8) and exponent 100 and the third with amplitude 0.4 and direction (0,0,1), that is the geometric normal. The normal vectors are defined in the surface local frame of reference (this means that a consistent tangent has to be defined for each point on the surface). One can se e than an anisotropic surface is obtained in a very straightforward manner. Figure 10 and Figure 11 show the same geometry

Figure 10.
Velvet (see colour print)

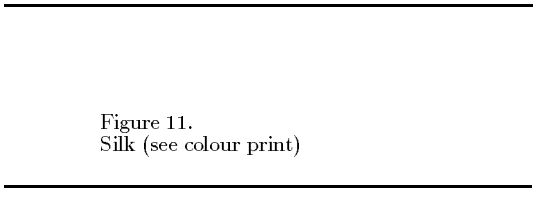

Figure 11.
Silk (see colour print)

with two different multiple surfaces, one defined to simulate the reflective behaviour of velvet, the other a rough approximation of silk. The parameters are given in the following table.

Material	Amplitude	Vector	Exponent
Velvet	0.1	0, 0, 1	1.0
	0.6	0.01, 1, 0	20
	0.6	0.01, 0.5, 0.866	20
	0.6	0.01, -0.5, 0.866	20
	0.6	0.01, -1, 0	20
	0.6	0.01, -0.5, -0.866	20
	0.6	0.01, 0.5, -0.866	20
Silk	0.6	0, 0, 1	1.0
	0.1	0.1, 1, 0	32
	0.6	0.8, 1, 0	32
	0.4	0.5, 1, 0	32
	0.1	0.1, 0, 1	32
	0.6	0.8, 0, 1	32
	0.4	0.5, 0, 1	32
	0.1	0.1, -1, 0	32
	0.6	0.8, -1, 0	32
	0.4	0.5, -1, 0	32
	0.1	0.1, 0, -1	32
	0.4	0.8, 0, -1	32
	0.6	0.5, 0, -1	32

The geometric model is identical (this is not of course the case for real materials, the geometry i s also a strong cue about the nature of the material) for both examples. This should be compared to the images of Figure 1.

5. Shadowing and Masking

When the surface structure is real some of the details block other from the light (*self-shadowing*) or from the view (*masking*, or *self-blocking*). It is important to model these effects for convincing rendering. Textures simulating hair or fur can be geometrically adequate, for example, but not very realistic if masking and shadowing are missing. Such problems, and solutions, has been addressed notably by Max and his collaborators [Max88a, Cabr87a]. Cabral *et al* specifically computed horizon angles in a large sample of directions (24), and used a triangulation of the bump map to compute a BDRF for the surface. They also addressed the problem of reflection by the bump mapped surface, a question we will not broach here. Depending on the context, we could claim that traditional bump-maps do not take these effects into account, and therefore since we just filter them we do not have to be concerned. The challenge, however, is hard to resist.

Within our model the complete answer would be to be able to express for each Phong peak at any level the fraction of area (therefore the fraction of amplitude) visible from a given direction. This requires a bivariate function description for each peak of each texel, quite a heavy burden. Noticing that the shadowing and masking does not have to be too accurate (just consistent and smooth), and that for a wide range of useful textures the bivariate function reduces to one or two separable unidirectional functions, this is what we implemented.

First a height map is obtained for the bump map to be used. This can be extracted from ray-tracing as the normal map is computed from the geometric model. Then for the X and Y axes in texture space and for each texel the slope of the horizon in both directions (that is left (negative X), right (positive X), top (negative Y) and bottom (positive Y)) is computed and stored as a map (see Figure 12).

Figure 12.
Horizon slopes at texels

This amounts to 4 values per texels, which can be encoded as 4 bytes, and therefore be similar to an image map in size. While rendering with the unfiltered texture, this information is compared at the relevant texel to the slope of the light vector and the eye vector in the XZ plane and YZ plane of the surface local frame. If the slope of the light vector is less than any of the stored values, then the texel is in shadow. If the slope of the eye vector is less than any of the stored values then the texel is masked by some other texel. In the latter case unfortunately we do not know which texel is visible instead. This could be deduced from the height map (we would have

to keep it around) by tracing from the hidden texel to the last visible one in the direction of the ray as is done in [Cabr87a] but that would not carry well to filtering.

To filter this information, we proceed as follows. When building the samples of $\mathbf{N}(\theta,\phi)$, for each sample direction the average slopes and their standard deviations are computed (the standar d deviation for the slopes of the bottom texture is 0, and each contribution is weighted by the vector amplitude). After the surface is fitted with the Phong peaks, each Phong peak "collects" the slope information, by essentially filtering the contribution of all the samples weighted by a $\cos^n(\alpha)$ term. Each Phong peak then ends up with four average slopes each with a standard deviation . When shading these are treated as described before, except that the shadow/masking variable is not a Boolean any more, but has real values in the range [0,1]. To simplify the computations, we actually assumed that the texel is entirely visible at the limiting slope plus the standard deviation, and entirely hidden at the limiting slope minus the standard deviation. In between the values are linearly interpolated as a function of the slope of the light or eye vector. The model therefore computes in the directions of the stored slopes the distribution of shadowing/blocking slopes for each Phong peak. The assumption is that the distribution can be modelled by a normal distribution around the average (it is important to note that it does not mean that we assume the normals themselves or the slopes in general are normally distributed, only that the average of blocking slopes for a given direction is, which is a very mild assumption).

The problem we had with masking with the original map (not knowing the actual visible texel) is solved in filtering if we assume that the original visible texel is within the texel at this level of the pyramid. In this case its contribution to the shading will be computed (note that in this case the contributions of the visible components of the texel should be correctly ratioed to correct for th e smaller visible area). Note also that this could help compute and filter the change of colour of the surface as well, if colour information is carried along with the distribution of normals (in ou r examples it is a completely separate texture map).

The assumption of separability is the most restrictive, and easy to violate. A wide range of tex - tures, however, meet the assumption:

- isotropic textures (in 2D)
- unidirectional textures, such as parallel grooves, cylinders, etc. Of course either the X or the Y axis should be in that direction
- bidirectional textures, such as most weaving patterns.

The visibility information from both direction is blended by taking the product. Any number o f more sophisticated "blending functions" can be thought of, but we have not experimented with any yet. The slope information is of course to be transformed according to the texture and th e object transformations. We made no attempts here to take the object curvature into account, as is done in [Max88a]. Ignoring the curvature is reasonable as long as the bumps introduce much more local variation than the curvature. The mapping from texture to object can create areas such has almost everything is in shadow or not seen, such as the north pole of the sphere in ou r examples. If no scaling is applied to the slopes in these cases it is equivalent to assume that the microstructure modelled by the bump map is of constant physical size over the surface, which i s the case for most real microstructures. As an aside, it is interesting to note that most fractal surfaces have no limiting slope between adjacent "bumps", and therefore every point on the surface is masked and in shadow, and therefore should appear totally black.

Figure 13 shows the effect of self shadowing and masking on the cylinder texture applied an d filtered on a plane. This can be compared to Figure 5, the same texture without these effects. Figure 14 shows the surface obtained from the top level of the cylinder texture applied to a sphere. On the right, there is no shadowing or masking, and the surface looks like an ordinary anisotropic surface. On the left, with shadowing and masking, only the cylinders close to th e plane defined by the eye and the light vectors reflect strongly.

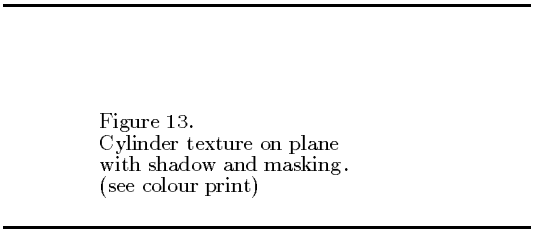

Figure 13.
Cylinder texture on plane
with shadow and masking.
(see colour print)

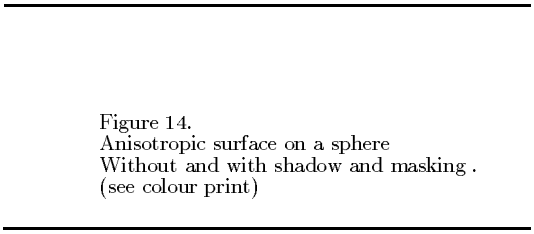

Figure 14.
Anisotropic surface on a sphere
Without and with shadow and masking.
(see colour print)

6. Conclusions

We have obtained an effective technique to filter bump maps, bridging the gap between bump maps and local illumination models. It lead us to a simple local illumination model where eac h point on a surface can have multiple normal vectors. We can with this model easily the reflective characteristics of cloth, stochastic surfaces and in general any microstructure whose geometry w e can model. We have also developed a technique to compute and filter the effect of self-shadowing and masking, again from basic and compact information extracted from the geometric model o f the microstructure.

One of the shortcomings of these techniques, besides the limiting assumptions made for the filtering of shadowing/masking is that the specular interreflections are ignored. Work is in progress to first assess how much does interreflection contribute to the illumination, an then try to account for it in the event that it contributes significantly. Another interesting issue is the use of the directions of the Phong peaks to model the reflection and refraction of the environment through the surface. This is, as mentioned before, the type of problems addressed in [Cabr87a] in particu - lar.

The relationship between BDRF and the distributions of normals we compute are also to explore. One issue is to characterize the BDRF which cannot be effectively represented that way, another is to try to extract automatically multiple surface characteristics from experimentally measured BDRF.

In addition to the shortcomings and remaining problems noted in the above, there are still many related larger questions to be addressed. For example most surfaces are not static. The surface o f water, as an important example, moves constantly. Statistical approaches to the computation of the light reflected by water exist [Krue88a] but are not well suited to animation. As another example, my initial interest in the problem of filtering surfaces arose in the context of stochastic modelling [Four82a] where a partially evolved surface "stands in" for the fully subdivided one. In this case the shading of each polygon should be the average of the shade of all the surface details not yet produced. Here again a single static texture is not enough. One has to create variou s "tiles" with statistics deduced from the process modelled. These two problems have similar elements and are shimmering targets above the horizon.

Acknowledgements

I gratefully acknowledge the support of NSERC through operating grants and an equipment grant which considerably facilitated the research described here. The support of the University of British Columbia in establishing a computer graphics laboratory in our department is greatly appreciated, as is the support of IBM Canada, which established GraFiC, an invaluable facility. Collaboration with Eugene Fiume at the University of Toronto has been useful, and is made easier by the support of the Province of Ontario through an Information Technology Research Centre Grant. Frederic Taillefer and Guang Huo collaborated on the development of models for woven materials. Peter Cahoon pointed me towards the right numerical libraries. Chris Romanzin helped in the production of depth maps and slope maps. Pierre Poulin and John Buchanan often helped with suggestions, criticisms and advice (not always taken).

References

Beck87a.
P. Beckmann and A. Spizzichino, *The scattering of electromagnetic waves from rough surfaces,* Artech House Inc, 1987.

Blin77a.
J.F. Blinn, "Models of Light Reflection For Computer Synthesized Pictures," *Computer Graphics*, vol. 11, no. 2, pp. 192-198, July 1977.

Blin78a.
J.F. Blinn, "Simulation of Wrinkled Surfaces," *Computer Graphics*, vol. 12, no. 3, pp. 286-292, August 1978.

Cabr87a.
B. Cabral, N. Max, and R. Springmeyer, "Bidirectional Reflection Functions from Surface Bump Maps," *Computer Graphics*, vol. 21, no. 4, pp. 273-281, July 1987.

Cook82a.
R.L. Cook and K.L. Torrance, "A Reflectance Model for Computer Graphics," *ACM Trans. on Graphics*, vol. 1, no. 1, pp. 7-24, January 1982.

Crow84a.
F.C. Crow, "Summed-Area Tables for Texture Mapping," *Computer Graphics*, vol. 18, no. 3, pp. 207-212, July 1984.

Dung78a.
W. Dungan, A. Stenger, and G. Sutty, "Texture Tiles Consideration for Raster Graphics," *Computer Graphics*, vol. 12, no. 3, pp. 130-134, 1978.

Foo188a.
A. V. Foo1 and J. E. Foo2, *Numerical Recipes in C,* Addison-Wesley, 1988.

Four82a.
A. Fournier, D. Fussell, and L. Carpenter, "Computer Rendering of Stochastic Models," *Comm. of the ACM*, vol. 25, no. 6, pp. 371-384, June 1982.

Four88a.
A. Fournier and E.L. Fiume, "Constant-Time Filtering with Space-Variant Kernels," *Computer Graphics*, vol. 22, no. 4, pp. 229-238, August 1988.

Gree86b.
N. Greene, "Environment Mapping and Other Applications of World Projections," *IEEE Computer Graphics and Applications*, vol. 6, no. 11, pp. 21-29, November 1986.

Gree86a.
N. Greene and P. S. Heckbert, "Creating Raster Omnimax Images from Multiple Perspective Views Using the Elliptical Weighted Average Filter," *IEEE Computer Graphics and Applications*, vol. 6, no. 6, pp. 21-27, June 1986.

He91a.Xiao D. He, Kenneth E. Torrance, Francois X. Sillion, and Donald P. Greenberg, "A comprehensive physical model for light reflection," *Computer Graphics (SIGGRAPH '91 Proceedings)*, vol. 25, no. 4, pp. 175-186, 1991.

Heck86a.
P.S. Heckbert, "Filtering by Repeated Integration," *Computer Graphics*, vol. 20, no. 4, pp. 315-321, August 1986.

Heck86b.
P.S. Heckbert, "Survey of Texture Mapping," *IEEE Computer Graphics and Applications*, vol. 6, no. 11, pp. 56-67, November 1986.

Krue88a.
W. Krueger, "Intensity fluctuations and natural texturing," *Computer Graphics*, vol. 22, no. 4, August 1988.

Max88a.
N.L. Max, "Horizon mapping: shadows for bump-mapped surfaces," *The Visual Computer*, vol. 4, 1988.

Poul90a.
P. Poulin and A. Fournier, "A Model for Anisotropic Reflection," *Computer Graphics (SIGGRAPH '90 Proceedings)*, vol. 24, no. 4, pp. 273-282, held in Dallas, Texas; 6-10 August 1990, July 1990.

Torr66a.
K.E. Torrance and E.M. Sparrow, "Polarization, Directional Distribution, and Off-Specular Peak Phenomena in Light Reflected from Roughened Surfaces," *J.Opt.Soc.Am.*, vol. 56, no . 7, 1966.

Will83a.
L. Williams, "Pyramidal Parametrics," *Computer Graphics*, vol. 17, no. 3, pp. 1-11, July 1983.