Interactive Rendering of Wavelet Projected Light Fields

Paul Lalonde[1]
Radical Entertainment
Vancouver, B.C.

Alain Fournier[2]
Department of Computer Science
University of British Columbia

Abstract

Light field techniques allow the rendering of objects in time complexity unrelated to their geometric complexity. The technique discretely samples the space of light rays exiting the boundary around an object and then reconstructs a requested view from these data. In order to generate high quality images a dense sampling of the space is required which leads to large data sets. These data sets exhibit a high degree of coherence and should be compressed in order to make their size manageable.

We present a wavelet-based method for storing light fields over planar domains. The parameterization is based on the Nusselt embedding, which leads to simplifications in shading computations when the light fields are used illumination sources. The wavelet transform exploits the coherence in the data to reduce the size of the data sets by factors of 20 times or more without objectionable deterioration in the rendered images. The wavelet representation also allows a hierarchical representation in which detail can be added incrementally, and in which each coarser view is an appropriately filtered version of the finer detail.

The wavelet coefficients are compressed by thresholding the coefficients and storing them in a sparse hexadecary tree. The tree encoding allows random access over the compressed wavelet coefficients which is essential for extracting slices and point samples from the light field.

Key words: image based rendering, wavelets, compression, light fields, Nusselt embedding.

1 Introduction

The traditional input to 3-D graphics systems is a description of a scene comprised of geometric primitives and their associated surface properties. For display some form of lighting simulation is performed and the results rendered. This process can be inexpensive and very approximate, or very complex with a high degree of physical accuracy. The main difficulty is that the time complexities of the rendering algorithms are all strongly dependant on the geometric complexity of the scene. This leads to very expensive computation when complex objects are included in the scene.

Recently a number of solutions to thses problems have been proposed that are based on interpolation of pixel data. Early results in this field include QuickTime VR [2], in which environment maps at a fixed point are used to generate continuous views of a scene as the user changes his viewing direction. A number of other systems then allow movement through the scene by performing view interpolation [3, 13]. These methods rely on having a depth map of the image that can be used to compute position of a pixel in a new view or else reconstruct the depth map from images captured at multiple viewpoints. The remaining challenge is in filling in the gaps where previously occluded regions become visible. These methods can be viewed as ways of estimating the radiance arriving at the eye from incomplete samples.

The plenoptic function completely describes the flow of light in an environment as a function of 5 variables: three to locate a point in space and two to specify the direction. If we assume that the light is travelling in a transparent medium then the radiance along a ray through empty space is constant. This observation can reduce the plenoptic function from a function of 5 variable to a function of 4, by considering only the boundary around an object, reducing the spacial component to two parameters on a surface. This observation leads directly to the *Light Field Rendering* and *Lumigraph* approaches of Levoy and Hanrahan [11] and Gortler *et al.* [6], respectively. In these systems they parameterize the space of positions and directions by the intersection of a line with two parallel planes, one of which represents a viewing window while the other specifies a direction. A set of views of the volume are stored, and a new view is generated by interpolating from these views. Levoy and Hanrahan use a lossy compression system based on vector quantization. Gortler *et al.* do not compress their data, keeping a large number of views in memory and using hardware texture mapping to perform the interpolations. Neither use their results for rendering applications, outside of generating views of their objects.

A related set of results can be found in the field of illumination engineering, where near field photometry is concerned with representing the light field emitted from a

[1]Now at Electronic Arts Research, `plalonde@ea.com`
[2]`fournier@cs.ubc.cs`

lamp. Ashdown [1] presents a method for measuring such a data set for arbitrary luminaires, based on using a CCD-based video camera to record the emitted radiance from a number of positions around the luminaire.

1.1 Radiance Representations for Rendering

A related area of work that must be considered in the context of light field rendering are those representations of the light field used implicity or explicitly in global illumination rendering systems. For example, traditional ray tracing [16] samples a light field algorithmically for only those directions required for the current view. At the other extreme, Lucifer, the wavelet radiative transport based renderer of Lewis *et al.* [12] builds explicit representations of the light field at a large number of cell boundaries throughout the environment. A similar representation is implicit in Christensen *et al.*'s wavelet radiosity system[4]. Their representation of radiance distributions uses a non-standard wavelet decomposition over a unit patch and a radially stretched gnomonic projection of the hemisphere of directions. They then solve two point light transport equations by numerical sampling of the reflectance funtion and of the transport integrals. These systems can both use light field objects as light sources as well as generate light field objects representing incident and emitted light on surfaces. Given that the information is available as a by-product of the global illumination computation it would be adventageous to be able to view them directly.

One important area in the use of light field objects is in using them as models in synthetic scenes. Since objects are described as a radiance field it is easy to integrate them into scenes by treating them as light sources. In a ray tracing application this means that we can trivially include emission toward the viewer. Emission into the rest of the scene is more difficult, requiring either fine subdivision or Monte Carlo sampling to account for the light shed onto other objects in the scene [9, 7]. One area of considerable difficulty is in re-shading the objects to account for additional lights in the scene. This topic is beyond the scope of this paper.

2 Wavelets

The wavelet transform takes a signal and decomposes it in terms of the wavelet basis functions. These basis functions have the property that they can localize features in the signal at different resolutions. Unlike the Fourier transform non-stationary signals can be decomposed meaningfully.

Wavelets are built from *scaling functions*, which we define by dilations and translations of a base scaling function $\phi(x)$ of the form:

$$\phi_{lm}(x) = 2^{-l/2}\phi(2^{-l}x - m)$$

each level l corresponds to a function space V_l, which is part of a nested sequence of subspaces $\ldots \subset V_{-1} \subset V_0 \subset V_1 \subset V_2 \ldots$ with these properties:

- the union of all V_l spans L^2
- $f \in V_l \to g \in V_l$ where $g(x) = f(x+k)$
- $f \in V_l \leftrightarrow g \in V_{l+1}$ where $g(x) = f(2^l x)$
- any $f \in V_l$ has a unique representation as a linear combination of ϕ_{lm}'s

A wavelet function space W_l is defined as composed of those functions that need to be added to a given space V_l to span the next finer space $V_{l+1} = V_l \oplus W_l$. The basis functions for W_l are also dilations and translations of a parent wavelet $\psi(x)$:

$$\psi_{lm}(x) = 2^{-l/2}\psi(2^{-l}x - m)$$

Since $\phi(x) \in V_0$ and $V_0 \subset V_1$, we can write $\phi(x)$ as a linear combination of the basis functions $\phi(2x - m)$ for V_1:

$$\phi(x) = \sqrt{2}\sum_m h_m \phi(2x - m)$$

This also holds for ψ:

$$\psi(x) = \sqrt{2}\sum_m g_m \phi(2x - m)$$

These are the *dilation* or *refinement equations*. They are the essence of multi-resolution analysis. Wavelet bases differ principally in their choices of $\{h_m\}$ (which in the case of orthogonal bases determine $\{g_m\}$).

3 Wavelet Light Fields

We propose a wavelet based solution for a number of reasons:

1. The representation allows a good control of the accuracy/space trade-off;
2. The point-wise reconstruction is logarithmic in the number of original light field samples;
3. A wavelet reconstruction can interpolate the original data;
4. Error metrics are easy to compute;
5. Incremental levels of accuracy can be used; and
6. The wavelet transform exploits the spacial and angular coherence present in multiple views of an object.

3.1 Parameterization

There are a number of decisions to be made in chosing a light field representation. The most basic is the parameterization. Whatever parameterization is chosen it should support those operations we require in a natural and efficient way. We also require a parameterization that is amenable to the wavelet transform and allows reasonable exploitation of angular and spacial coherence.

The topology of the light field's function space is an hemisphere crossed with a surface. The most common wavelet formulations are applied on Cartesian spaces. The surface is usually parameterized by two parameters, u and v. Fortunately there are straightforward embeddings of $S^2 \times R^2$ into R^4. For our experiments we chose to use the *Nusselt* embedding, which is related to the direction cosines [12]. Related work shows that the Nusselt embedding also has advantages when the light field objects are used for shading, simplifying the local shading equation [10, 8].

To express a direction given by a vector $\vec{v}$ in a local coordinate frame given by the normal $\vec{N}$ and surface tangent $\vec{T}$, we let $\mu_x = \vec{v} \cdot \vec{T}$ and $\mu_y = \vec{v} \cdot (\vec{N} \times \vec{T})$ be the direction cosines. We normalize these to the range $(0..1)$, and call them κ and λ:

$$\kappa = (\mu_x + 1.0)/2.0$$
$$\lambda = (\mu_y + 1.0)/2.0$$

κ and λ then index in the unit square. Directions then cover only the circle at the center of the square. Figure 1 illustrates this mapping. We define the value of any function of directions for which $\mu_x^2 + \mu_y^2 > 1$ as zero. Although this might appear to be wasteful of storage the wavelet transform, and our sparse representation of it, is particularly efficient at compressing zero values — none are stored.

To represent our light field object, **L**, we use 4 parameters: u and v define the position on the surface, usually a quadrilateral, and κ and λ are the Nusselt parameters for the given viewer direction. We then write the light field object as $\mathbf{L}(u, v, \kappa, \lambda)$.

3.2 Choice of Decomposition

Although wavelet transforms are easily defined in one dimension, the extension to higher dimensions allows a number of choices of decompositions. The two most common are called the standard decomposition, and the non-standard decomposition [5]. A third option that merits a few words is the spherical wavelets proposed by Schröder and Sweldens [14]

Schröder and Sweldens showed a method for encoding the reflections from one incident direction using a spherical wavelet representation. This restriction to one incident direction is a serious shortcoming. This method does

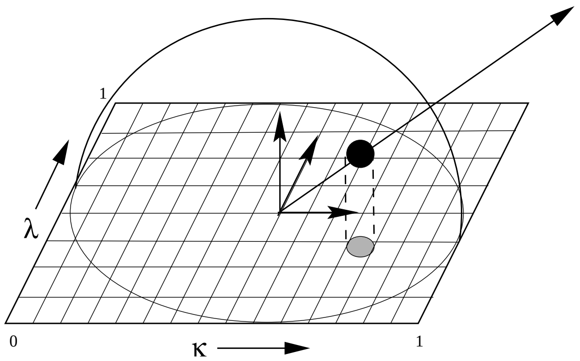

Figure 1: The Nusselt embedding. The given direction is mapped to a particular pair (κ, λ) by projecting the intersection with the hemisphere onto the unit square.

not easily extend to encoding the image information in each direction, as their method is very dependant on the underlying topology of the function space, which does not map to the topology of the space our light field objects are embedded in. The more serious difficulty is that even extending the method to higher dimensions, the basis functions of their transforms are built by subdivision of the input space. The basis functions are no longer self-similar, and cannot be composed of sets of unidimensional functions, which as we shall see is critical to efficient reconstruction of the light field at a given point and direction.

We chose instead to use a multi-dimensional wavelet transform, mapping the parameter space of the light field object (two positional and two directional) to a Cartesian grid. The multi-dimensional wavelet decomposition allows a choice in how the uni-dimensional decomposition is extended to higher dimensions. In the standard case a product of one-dimensional decompositions is used. In the non-standard case a product of the basis functions is used, leading to a multi-dimensional multi-resolution analysis that is analogous to the uni-dimensional case [5]. We will examine each in turn, and the algorithms that follow.

Standard Decomposition

The standard four-dimensional wavelet decomposition $\mathcal{L}$ of a light field object **L** yields:

$$\mathcal{L}(u, v, \kappa_r, \lambda_r) =$$

$$\sum_{g=0}^{D_g} \sum_{h=0}^{D_h} \sum_{j=0}^{D_j} \sum_{k=0}^{D_k} c_{g,h,j,k} B_g(u) B_h(v) B_j(\kappa) B_k(\lambda) \quad (1)$$

where D_g, D_h, D_j, and D_k are the dimensions of each the dimensions and B_n is defined as

$$B_n(x) = \begin{cases} \phi(x) & \text{if } n = 0 \\ 2^{-l/2}\psi(2^{-l}x - m) & \text{if } n = 2^l + m \\ & \text{for some } l \text{ and } m \geq 0 \end{cases} \tag{2}$$

and ψ is our mother wavelet and ϕ is our smoothing function.

Consider the coefficients and basis functions that must be examined to reconstruct $\mathbf{L}$ at point (u, v, κ, λ) in the standard decomposition. Each coefficient $c_{g,h,j,k}$ for which any of $B_g(u), B_h(v), B_j(\kappa), B_k(\lambda)$ is non-zero needs to be examined. We say that a point x is in the support of B if $x < \delta_l$ or $x > \delta_r$ where δ_l is the smallest value for which $B(\delta) \neq 0$ and where δ_r is the largest value for which $B(\delta) \neq 0$. Recall that $B_n(x) = 2^{-l/2}\psi(2^{-l}x - m)$ if $n = 2^l + m$ for some l and $m \geq 0$ when $n > 0$. We can then, for each l and m, establish which functions $\psi_{l,m}(x)$ are non-zero. If the wavelet has a width of support w then there will be w basis functions to examine at each level l.

Observe then that each term in the summations of equation 1 can reference basis functions $\psi_{l,m}$ at different levels. This interaction means that a reconstruction costs $O(w^4 l^4)$; if there are n^4 original samples in the signal the cost is $O(w^4 \log_2^4 n)$. This cost is too high to use the standard decomposition to evaluate the value of the light field at a point.

3.3 Non-Standard Decomposition

Consider instead the non-standard decomposition. Express the object $\mathbf{L}$ as its projection onto a multi-resolution wavelet space. For a univariate wavelet basis, the basis functions are all derived from two functions by scaling by a factor l (we will us $l = 0$ as the coarsest level of the pyramid, that is a larger l in the discrete case corresponds to a larger number of elements) and translations by m, which can range from 0 to 2^l. One group comes from the *scaling function* ϕ:

$$\phi_{lm}(p) = 2^{l/2}\phi(2^l p - m)$$

and the other from the *mother wavelet function* ψ:

$$\psi_{lm}(p) = 2^{l/2}\psi(2^l p - m)$$

To manage the notation for multidimensional wavelet bases, we will use a four-variable vector $\mathbf{q}$:

$$\mathbf{q} = (u, v, \kappa, \lambda)$$

and a multi-resolution index $\mathbf{j}$:

$$\mathbf{j} = (\nu, l, m_u, m_v, m_\kappa, m_\lambda)$$

Here ν is the *selector index* for the wavelet functions, l is the level in the wavelet subspaces, 0 being the coarsest, and m_p the offset at that level for parameter p. With the non-standard decomposition the level is the same for all variables because the bases themselves are multidimensional [5, 15]. Since the basis functions all have an hypercube support (same extent for all variables at a given level) the data structure is simpler, as are many geometric operations. The selector ν determines the product of one dimensional wavelet and smooth basis functions that together define $B_j(\mathbf{q})$. We introduce a special notation to indicate the selection, where $\nu[h]$ is interpreted as the hth bit of ν:

$$\Gamma^h_{l,m,\nu}(p) = \left\{ \begin{array}{ll} \phi_{l,m}(p) & \text{if } \nu[h] = 0 \\ \psi_{l,m}(p) & \text{if } \nu[h] = 1 \end{array} \right\}$$

Then we then write the basis function $B_j(\mathbf{q})$ as:

$$\Gamma^0_{l,m_u,\nu}(u) \times \Gamma^1_{l,m_v,\nu}(v) \times \Gamma^2_{l,m_\kappa,\nu}(\kappa) \times \Gamma^3_{l,m_\lambda,\nu}(\lambda) \tag{3}$$

Then the projection of the light field object $\mathbf{L}$ onto the multi-resolution space is given by:

$$\mathcal{L}(u, v, \kappa, \lambda) = \sum_j c_j B_j(u, v, \kappa, \lambda) \tag{4}$$

where B_j are the appropriate basis functions. The c_j are given by

$$\begin{aligned} c_j &= \langle \mathbf{L}(u, v, \kappa, \lambda) | \tilde{B_j}(\kappa_i, \lambda_i, \kappa_r, \lambda_r) \rangle \\ &= \int_0^1\int_0^1\int_0^1\int_0^1 \mathbf{L}(u, v, \kappa, \lambda)\tilde{B_j}(u, v, \kappa, \lambda) du dv d\kappa d\lambda \end{aligned}$$

where $\tilde{B_j}$ is the dual of B_j.

Since our functions are defined as zero when $\mu_x^2 + \mu_y^2 >= 1$ we can safely narrow the bounds of integration from $(-\infty..\infty)$ to $(0..1)$.

If we now examine the time-complexity of the reconstruction at a point we find that since the basis functions B_j used in the analysis are multi-dimensional, each of the terms of the summation in equation 4 has a unique level l for all combinations of ψ and ϕ required to assemble B_j. There are then 16 basis functions with w^4 translations at each level that support the point $\mathbf{q}$ at which we are reconstructing, where w is the width of the basis functions. This means we have only $O(w^4 \log_2 n)$ terms to examine during our traversal to evaluate at a point if the initial dataset had n^4 samples. Figure 2 illustrates the difference in which terms must be examined in the standard vs nonstandand decompositions for a 2-D example. In view of these results, we use the non-standard decomposition for our representation.

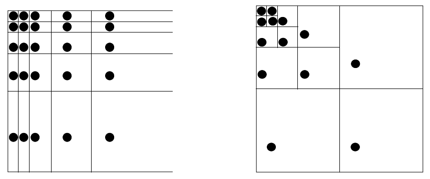

Figure 2: Coefficients examined for a point reconstruction using the standard decomposition (left) and the non-standard decomposition (right).

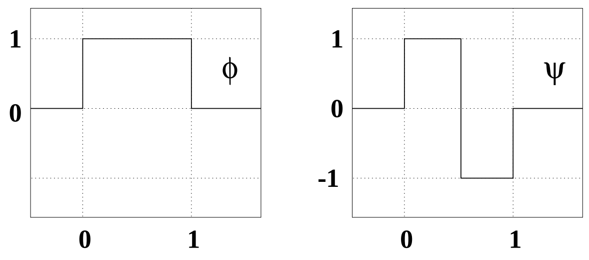

Figure 3: The Haar smooth and wavelet functions.

3.4 Choice of Bases

Our choice of wavelet basis functions is determined by two properties: the width of support of the basis, and the amount of compression a basis can provide. Since one of our goals in using the wavelet representation is to reduce the memory requirements we see immediately why greater compression is a goal. It is not so obvious why the width of support of the basis is important, or how it relates to compression rates. The narrower the width of support of a wavelet, the less computational work is required in the reconstruction, since the scaled and translated wavelets will cover fewer terms. When extending to higher dimensions the amount of work required by wider bases increases with the power of the dimension. The number of terms examined then becomes a dominant factor in the cost of the reconstruction. In the case of bi-orthogonal wavelets all we care about is the width of the reconstruction wavelet. The analysis wavelet can be as wide as necessary, since the wavelet transform is performed off-line using the fast wavelet transform.

As far as our application is concerned, our principal goal is speed. As such we choose to use the Haar basis [5], which is the narrowest wavelet, having a support of only 2 samples, in contrast to the linear spline wavelet with a support of 4, or the Daubechies 4 basis, with a width of support of 8. Figure 3 shows the Haar smooth and wavelet functions. It should also be noted that the Haar wavelet is not a bad choice for compression results either. Compression ratios of 20:1 frequently result in less than 10% relative RMS error. These results will be expanded upon in section 4.

3.5 Storage

The remaining difficulty is in representing the coefficients. The major advantage of using a wavelet based representation is that the representation is sparse, with many of the wavelet coefficients being zero or small. To increase our compression rate at the cost of accuracy we threshold small coefficients to zero. A property of the wavelet transform is that the sum of these thresholded coefficients gives us the root mean square of the error induced in the reconstruction by ignoring the thresholded terms. If the original light field object is f, the coefficients of its wavelet transformed representation are w_i, and f_A represents the light field object reconstructed by using only the coefficients in the set A then the error is given by:

$$||f - f_A|| \; = \; (\sum_{(i) \notin A} |w_i|^2 \;)^{1/2}. \qquad (5)$$

The difficulty is that there are as many wavelet coefficients as original data points, even though a large number of them may be zero. Thus we want to find a representation that stores only the non-zero coefficients, allowing us considerable economies in storage.

In a one dimensional transform it is possible to use a binary tree structure that at each node records the value of the coefficient and mimics the structure of the wavelet analysis. Then it is possible to prune any subtrees that contain only zero coefficients. There may remain nodes in the tree that record zero values (if any of its descendants are non-zero) but these add no more than a small linear factor to the size of the stored tree. For reconstruction it is sufficient to traverse a path down the tree, following a child pointer if the child's basis function supports the point being evaluated. We let this structure inspire us in higher dimensions.

An efficient option for the representation is as a *Wavelet Coefficient Tree*. This tree is a sparse hexa-decary tree (16 ordered children), where the depth of the nodes indicates the depth of the coefficients in the representation, and where each node stores the non-zero coefficients with the same offset m, indexed by ν, the wavelet basis selector.

We use a structure called a *wavelet node* that stores all non-zero wavelet coefficients with the same indices (l, m_0, m_1, m_2, m_3), and different basis selector ν. The wavelet node also maintains a list of non-empty subtrees rooted at a wavelet node with indices $l + 1, 2m_0 + b\{0\}, 2m_1 + b\{1\}, 2m_2 + b\{2\}, 2m_3 + b\{3\}$ where $b = 0..15$ and $b\{i\}$ selects the value of the ith bit of the binary representation of b. The addition of 2 16-bit masks indicating which coefficients (the value mask) and which chil-

```
function Eval(WCTree lf, vector d)
{
    result = 0;
    for b = 0 .. 15
        if (lf->valueMask & (0x01>>b))
            j = (b,lf->level,lf->m[0..3]);
            result += lf->value[b] * B_j(d);
        if (lf->childMask & (0x01>>b))
            j = (b, lf->level+1,
                     lf->m[0] * 2 + ((b>>0)&0x01),
                     lf->m[1] * 2 + ((b>>1)&0x01),
                     lf->m[2] * 2 + ((b>>2)&0x01),
                     lf->m[3] * 2 + ((b>>3)&0x01));
            // Check the support of B_j()
            if (InSupportOf_B_j(d))
                result += Eval(lf->child[b], d);
    return result;
}
```

Figure 4: Reconstruction of a 4-variable Wavelet compressed function at point **q**

dren (the child mask) are present allow very compact storage of the tree. By storing the child nodes in a contiguous block only a pointer to the first child is required, reducing the overhead of the data structure to one pointer and two 16 bit masks per node. The child pointer and child masks need not be stored for leaf nodes, leading to an amortized cost of half a pointer plus 24 bits per node.

3.6 Reconstruction

Using the Wavelet Coefficient Tree the reconstruction of the light field for a given position and direction becomes a traversal of the wavelet coefficient tree. From the root, the basis functions corresponding to each non-zero coefficient is evaluated and summed, and then each child is visited whose corresponding basis function extents intersect the given point. The most important optimization is that if the a wavelet or smooth basis function at a given level and offset does not support our sample point then neither will any of that basis's children.

The traversal then is comprised of a number of bit-mask operations and tests for coverage by the relevant child's basis function. This coverage operation can frequently be implemented much more efficiently than a basis function evaluation. For efficiency the non-Haar wavelet and scaling functions $\Gamma(x)$ are tabulated off line, and basis function evaluation becomes a table lookup after scaling and translating the parameter x. Haar bases are coded algorithmically because of their simplicity.

Fig. 4 shows the traversal pseudo-code for a point-evaluation.

3.7 Level of Detail and Partial Rendering

The above algorithm is suitable for reconstructing a light field at one point. This could be used to generate each pixel in a view of an object, but much of the computation would be redundant. Consider that the many of the basis functions evaluated have wide support, and that the Haar basis is piecewise constant over this support. This means that for a given view direction we could re-use many of the basis function evaluations for neighbouring pixels. At the extreme the level 0 basis function need only be evaluated once for the entire patch. The finest level leaf nodes are the only ones that need to be seperately evaluated at every pixel.

We can modify our implicit pixel at a time depth first traversal into a breadth-first traversal that generates an entire view each pass. By halting the computation at each level partial results can be made available for display, although the breadth-first traversal is fast enough to display 64 by 64 pixel images in real time. The modified algorithm is showin in Figure 5. A quick analysis shows that this algorithm is $O(n)$, where n is the number of leaf nodes that contribute to the image seen from one direction. In the extreme this can be one leaf for each pixel in the reconstruction. There is one quarter less work at the next level up, one sixteenth above, and so on, leading to an overall cost of $O(n)$, in contrast with the $O(\log n)$ work required at each pixel in the pixel-by pixel reconstruction.

This algorithm also allows us to perform level-of detail culling, when the method is used to generate texture maps in an interactive application. By truncating the evaluation at a depth in the tree at which the width of a basis function spans a single pixel we can realize considerable computational savings. In addition the resulting image is appropriately filtered by the smoothing function used in our decomposition. In the case of the Haar basis this smoothing is equivalent to the mip-map averaging scheme that is in wide circulation. The resulting visual artifacts fall in line with the artifacts caused by using mip-maps.

4 Results

4.1 Sampling Light Field Objects

Ashdown has shown a system based on CCD video cameras and a moving gantry for measuring the emissivity of a luminaire over an encircling sphere [1]. His method records video images from a number of directions, and these then need to be transformed into positional and directional samples. The camera registration and lens distortion effects can cause serious artifacts. These problems may also appear in the Gortler *et al.*'s Lumigraph [6] as their data are collected with a hand-held video camera. Levoy and Hanrahan [11] use a modified Cyberware scanner to aquire their data. This option is not feasible on our

```
// We use a queue to implement the breadth-first search.
// The queue stores the wavelet tree node
// and the u,v position at that level
// The TextureMap structure is indexed by level and u,v.
function Eval(float kappa, float lambda, WCTree *lf)
{
  TextureMap[] = 0; // initialize the results
  AddToQueue( lf, 0, 0 );
  while (QueueNotEmpty)
    Current = DeQueue();
    EvalNode( kappa, lambda, Current->lf)
}
function EvalNode( float kappa, float lambda, WCTree *lf)
{
  resul= 0;
  for q = 0..3 // for each quadrant
    u = lf->m[0] + (q&0x01);
    v = lf->m[1] + ((q>>1)&0x01);
    result = 0;
    for b = 0..15
      // only evaluate if we have a non-zero value
      if (lf->valueMask & (0x01 >> b))
        j = ( b, lf->level, lf->m[0..3] );
        result[q] += lf->value[b] * B_j(u,v,kappa,lambda);
    Sum result into texture map at u,v corresponding to j
  // Add any necessary children to the evaluation queue
  for b = 0..15
    // check the child supports
    j = (b, lf->level+1,lf->m[0] * 2 + ((b>>0)&0x01),
      lf->m[1] * 2 + ((b>>1)&0x01),
      lf->m[2] * 2 + ((b>>2)&0x01),
      lf->m[3] * 2 + ((b>>3)&0x01));
    u_c = u*2 + ((b>>0)&0x01);
    v_c = v*2 + ((b>>1)&0x01);
    if (InSupportOf(B_j,u,v,kappa,lambda))
      AddToQueue( lf->child[b] );
}
```

Figure 5: Reconstruction of a 4-variable Wavelet compressed function over the unit square in direction (kappa,lambda)

budget.

Instead of building a physical instrument we have chosen to instrument our test-bed ray tracer to sample detailed geometries. Similar measurements can be made with Lucifer, a global illumination software package that uses a wavelet representation of light, with the added benefit of obtaining our data already in the wavelet domain [12]. Both these methods make measurements completely repeatable and lets us ignore calibration issues allowing us to concentrate on representational issues.

Using our system we sampled 3 light field objects at a resolution of 32 samples in each of u, v, κ, and λ:

- A set of spheres, illuminated from within;
- A ceiling mounted fluorescent lamp with sharp baffles; and

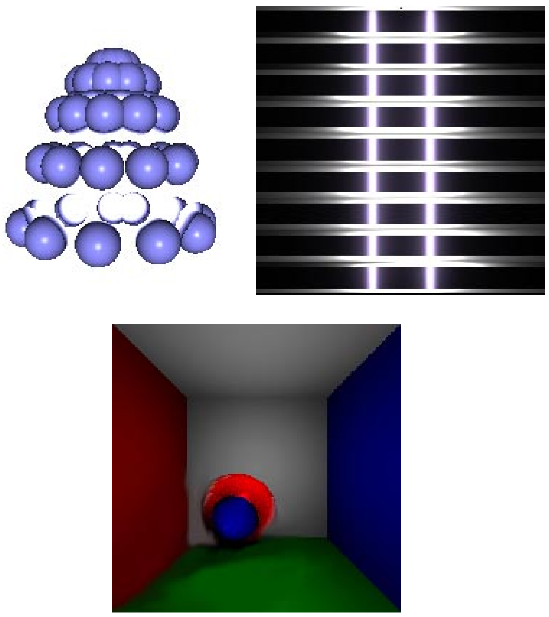

Figure 6: The three objects used in our experiments.

- A simple room environment with a red wall, a blue wall, a green floor, reflective white walls and ceiling and a blue sphere.

A view of each is show in Figure 6.

Table 1 sumarizes the performance of the algorithm. The timings are for the breadth-first version of the reconstruction, as presented in Section 3.7, running on an R10000 SGI workstation. The most recent implementation is included on the acompanying CD, and runs on any PC running Windows. Its run time typically improve on the SGI timings by a factor of 5 times. The most important observations is that thresholding the data set can, as expected, reduce the running times, although not as much as might have been originally expected. The other observation is that the data sets are reduced in size by approximately half during the wavelet transform, without thresholding. Thresholding can then drop the size considerably more for most data sets without unduly affecting drawing quality up to about ratios of 20:1, as seen in the lamp and room examples.

5 Conclusions and Future Work

We have shown that the non-standard wavelet decomposition provides a powerful tool for compressing light field data. Using sparse trees to store the data leads to efficient reconstruction algorithms that provide real-time reconstruction of views of an object. In particular it is possible to reconstruct 2-D slices from a 4-D volume in $O(n^2)$ time, where there are n^2 samples in the slice. The reconstruction is sensitive to level-of-detail considerations and can be built incrementally.

This paper has only examined compression by thresholding. Another important technique is that should be examined is quantization. By quantizing the wavelet coefficients to an 8 or 12 bit representation we should be able to cut the storage requirements by another factor of 8. Likewise, we currently store red, green, and blue channels seperately. Instead of using the RGB colour space we should use a luminance based space, such as YIQ, to apportion more of our precision to the luminance channel, which could lead to visually better reconstructions.

6 References

[1] Ian Ashdown. Near-field photometry: The helios approach. In *Graphics Interface '92 Workshop on Local Illumination*, pages 1–14, May 1992.

[2] Shenchang Eric Chen. Quicktime VR - an image-based approach to virtual environment navigation. In *SIGGRAPH 95 Conference Proceedings*, pages 29–38, August 1995.

[3] Shenchang Eric Chen and Lance Williams. View interpolation for image synthesis. In *Computer Graphics (SIGGRAPH '93 Proceedings)*, volume 27, pages 279–288, August 1993.

[4] Per H. Christensen, Eric J. Stollnitz, David H. Salesin, and Tony D. DeRose. Global illumination of glossy environments using wavelets and importance. *ACM Transactions on Graphics*, 15(1):37–71, January 1996. ISSN 0730-0301.

[5] Ingrid Daubechies. *Ten Lectures on Wavelets*, volume 61 of *CBMS-NSF Regional Conference Series in Applied Mathematics*. SIAM, Philadelphia, PA, 1992.

[6] Steven J. Gortler, Radek Grzeszczuk, Richard Szeliski, and Michael F. Cohen. The lumigraph. In *SIGGRAPH 96 Conference Proceedings*, pages 43–54. August 1996.

[7] Wolfgang Heidrich, Jan Kautz, Philipp Slusallek, and Hans-Peter Seidel. Canned lightsources. In *Rendering Techniques '98 (Proc. of Eurographics Workshop on Rendering*, pages 293–300, June 1998.

[8] P. Lalonde and A. Fournier. Generating reflected directionsfrom BRDF data. *Computer Graphics Forum*, 16(3):293–300, August 1997. Proceedings of Eurographics '97. ISSN 1067-7055.

[9] Paul Lalonde. *Representations and Uses of Light Distribution Functions*. Ph.d. thesis, Department of Computer Science, University of Brisith Columbia, July 1997.

[10] Paul Lalonde and Alain Fournier. Filtered local shading in the wavelet domain. In *Eurographics Rendering Workshop 1997*, pages 163–174, June 1997.

[11] Marc Levoy and Pat Hanrahan. Light field rendering. In *SIGGRAPH 96 Conference Proceedings*, pages 31–42. August 1996.

[12] Robert R. Lewis and Alain Fournier. Light-driven global illumination with a wavelet representation of light transport. In *Eurographics Rendering Workshop 1996*, pages 11–20, June 1996.

[13] Leonard McMillan and Gary Bishop. Plenoptic modeling: An image-based rendering system. In *SIGGRAPH 95 Conference Proceedings*, pages 39–46, August 1995.

[14] Peter Schröder and Wim Sweldens. Spherical wavelets: Efficiently representing functions on the sphere. In *SIGGRAPH 95 Conference Proceedings*, pages 161–172, August 1995.

[15] Eric J. Stollnitz, Tony D. DeRose, and David H. Salesin. Wavelets for computer graphics: A primer. *IEEE Computer Graphics and Applications*, 15(3), 1995.

[16] Turner Whitted. An improved illumination model for shaded display. *Communications of the ACM*, 23(6):343–349, June 1980.

	# Coef.	% Size	RMS Err.	% Err.	Time
Lamp					
Initial Size	1048756				
	89476	0.085	0.0	0.0	12.9
	26841	0.025	17.59	0.15	9.8
	17894	0.017	24.96	0.21	
	8946	0.0085	37.37	0.32	3.1
Dataset RMS			116.62		
Spheres					
# Coefs.	1048756				
	488833	0.47	0.0	0.0	
	97764	0.093	83.36	0.24	
	47276	0.045	120.87	0.35	
	4887	0.0046	205.42	0.59	
RMS			342.57		
Room					
# Coefs.	1048756				
	541234	0.51	0.0	0.0	22
	54122	0.051	24.82	0.09	9.5
	27060	0.026	40.06	0.14	
	5411	0.0051	75.54	0.27	3.7
RMS			281.34		

Table 1: Performance figures. The timings (in milliseconds) are for the algorithm presented in Section 3.7.