The Fast Multipole Method for Global Illumination

Alap Karapurkar
Computer Science & Engg.
IIT Bombay
Mumbai, INDIA 400076
alap_k@iitb.ac.in

Nitin Goel
Dept. Institute of Technology
Banaras Hindu University
Varanasi, INDIA 221005
nitin_cse_itbhu@yahoo.co.in

Sharat Chandran
Computer Science & Engg. Dept.
IIT Bombay
Mumbai, INDIA 400076
sharat@cse.iitb.ac.in

Abstract

Despite its wide applicability in scientific computing, the linear time fast multipole method (FMM) with provable error bounds has not been used extensively in computer graphics. This paper presents – to our knowledge – the first application of FMM to the problem of global illumination. The light transport kernel is broken into multipole and local expansions and required transformations for these expansions are provided. We also provide an adaptive octree based visibility algorithm. We have implemented our algorithm using the adaptive version of FMM and provide empirical results for the same.

1. Introduction

Global illumination – the simulation of the physical process of light transport – is an extremely complicated problem due to the presence of a large number of variables. This problem has been considered for several years with interesting methods like photon mapping, wavelets & hierarchical radiosity, and progressive refinement.

Computational science and engineering is replete with problems which require the evaluation of pairwise interactions in a large collection of particles. Direct evaluation of such interactions results in $O(N^2)$ complexity which places practical limits on the size of problems which can be considered. Techniques that attempt to overcome this limitation are labeled N-body methods. The N-body method is at the core of many computational problems, but simulations of celestial mechanics and coulombic interactions have motivated much of the research into this. Numerous efforts have aimed at reducing the computational complexity of the N-body method, particle-in-cell, particle-particle/particle-mesh[11], and treecodes[1, 2] being notable among these. However, the first numerically-defensible algorithm[5] that succeeded in reducing the N-body complexity to $O(N)$ was the Greengard-Rokhlin Fast Multipole Method (FMM)[6, 7, 3]. The FMM, in a broad sense, enables the product of restricted dense matrices with a vector to be evaluated in $O(N)$ or $O(N \log N)$ operations, when direct multiplication requires $O(N^2)$ operations.

1.1. Contributions

A new algorithm for solving the global illumination problem has been given here.

- We present (Section 2) the mathematical apparatus required to apply the *linear-time adaptive* FMM algorithm to diffuse objects. Five mathematical results with respect to the core radiosity equation under full visibility are shown in this context. If the BRDF [13] is presented as a table lookup (a common practice), then the method can be extended to non-diffuse objects. The linearity parameter N is with respect to a "particle" version of the scene and is related to the amount of complexity in the shading patterns that one might desire.

- The visibility function is highly discontinuous and, like the BRDF, does not easily lend itself to an analytical FMM formulation. Thus the nature of this computation is $\Omega(n^2)$ for n primitives, which depends on the geometry of the scene. We present a new visibility algorithm (Section 5) to handle the unoccluded FMM solution mentioned above.

Qualitative and quantitative results are shown in Section 6.

2. The Radiosity Kernel

For simplicity, we first present the mathematical framework when the radiosity kernel is used (the visibility function is treated later in Section 5).

2.1. Receiver Matrices and Source Matrices

The key task in the FMM method is to *factorize the kernel into terms containing the source and terms containing the receiver.* To this end, we rewrite the rendering equation at a surface point x due to an area A_y as

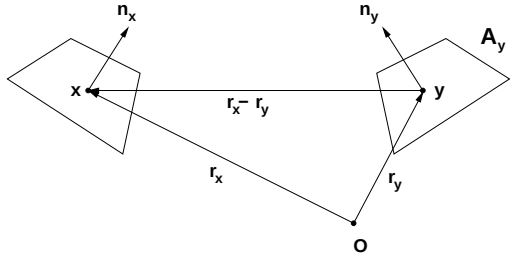


Figure 1. Geometry and notations used in this paper.

$$B(x) - E(x) = \rho(x) \int_{A_y} \frac{[\vec{n}_y.(\vec{r}_x - \vec{r}_y)][\vec{n}_x.(\vec{r}_y - \vec{r}_x)]}{\pi |\vec{r}_y - \vec{r}_x|^4} B(y) dA_y \tag{1}$$

Here, for a surface point x, $B(x), \vec{r}_x$, and $\vec{n}_x$ denote its radiosity, position, and normal to the surface respectively.

For notational convenience, we define the *receiver matrices* **RM**, the *source matrices* **SM**, and an operator $\otimes$ as ($\mathbf{r}_x$ denotes the 3x1 matrix form of $\vec{r}_x$):

$$\mathbf{SM}(y) = \begin{bmatrix} \mathbf{n}_y \mathbf{r}_y^t \\ \mathbf{n}_y \\ \mathbf{r}_y^t \mathbf{n}_y \mathbf{r}_y^t \\ \mathbf{r}_y^t \mathbf{n}_y \end{bmatrix} \qquad \mathbf{RM}(x) = \begin{bmatrix} \mathbf{r}_x^t \\ \mathbf{n}_x \\ \mathbf{r}_x^t \mathbf{n}_x \mathbf{r}_x^t \\ \mathbf{r}_x^t \mathbf{n}_x \\ \mathbf{n}_x \mathbf{r}_x^t \end{bmatrix}$$

Notice that

$$\mathbf{RM}(x) \otimes \mathbf{SM}(y) = [\vec{n}_y.(\vec{r}_x - \vec{r}_y)][\vec{n}_x.(\vec{r}_y - \vec{r}_x)] \tag{2}$$

$$\sum_{y=y_1}^{y_k} \mathbf{RM}(x) \otimes \mathbf{SM}(y) = \mathbf{RM}(x) \otimes \sum_{y=y_1}^{y_k} \mathbf{SM}(y) \tag{3}$$

2.2. Multipole Expansion

If we denote the spherical coordinates of $\vec{r}_x$ by (r_x, θ_x, ϕ_x), then our first result makes use of [10] to write (for $r_y < r_x$),

$$\frac{1}{|\vec{r}_y - \vec{r}_x|^4} = \sum_{n=0}^{\infty} \sum_{j=0}^{[n/2]} \sum_{m=-n+2j}^{n-2j} \pi e_n^j \left\{ \frac{1}{r_x^{n+4}} Y_{n-2j}^m(\theta_x, \phi_x) \right\} \left\{ r_y^n \overline{Y_{n-2j}^m(\theta_y, \phi_y)} \right\} \tag{4}$$

where

$$e_n^j = 4 \frac{(n-j+1)!(j+1/2)!}{(n-j+1/2)!j!}$$

and Y_n^m are the normalized spherical harmonics. Substituting (2), and (4) in (1) and rearranging terms, we get the *multipole expansion*

$$B(x) - E(x) = \sum_{n=0}^{\infty} \sum_{j=0}^{[n/2]} \sum_{m=-n+2j}^{n-2j} e_n^j R_{nj}^m(x) \otimes M_{nj}^m(A_y) \tag{5}$$

$$R_{nj}^m(x) = \frac{\rho(x)}{r_x^{n+4}} Y_{n-2j}^m(\theta_x, \phi_x) \mathbf{RM}(x) \tag{6}$$

$$M_{nj}^m(A_y) = \int_{A_y} B(y) r_y^n \overline{Y_{n-2j}^m(\theta_y, \phi_y)} \mathbf{SM}(y) dA_y \tag{7}$$

The intuition for this step is shown in Figure 2.

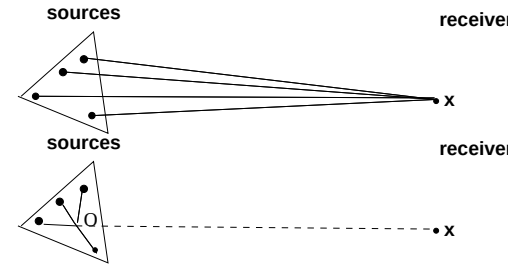


Figure 2. By associating a constant number of coefficients at center O, we can calculate the energy received by x from a number of differential emitters. The value of the coefficients depends upon the location of these emitters on patch.

We now notice that multipole coefficients are additive; this enables us to consider the effect of several patches as shown in Figure 3. For several patches $A_1, A_2, A_3, \ldots A_k$, the *multipole coefficients* $M_{nj}^m(A_y)$ can be accumulated once and (5) can be used to directly evaluate the radiosity at different values of x.

$$\begin{aligned} B(x) - E(x) &= \sum_{y=1}^{k} \left(\sum_{n=0}^{\infty} \sum_{j=0}^{[n/2]} \sum_{m=-n+2j}^{n-2j} e_n^j R_{nj}^m(x) \otimes M_{nj}^m(A_y) \right) \\ &= \sum_{n=0}^{\infty} \sum_{j=0}^{[n/2]} \sum_{m=-n+2j}^{n-2j} e_n^j R_{nj}^m(x) \otimes \left(\sum_{y=1}^{k} M_{nj}^m(A_y) \right) \end{aligned}$$

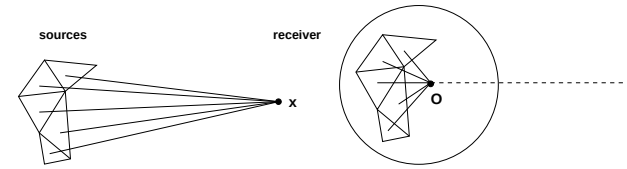


Figure 3. Multipole coefficients are additive.

For practical implementation, the summation to infinity must be truncated to some p. If $B^p(x)$ is the radiosity com-

puted by truncating the outer summation to p terms, the error incurred is of the form (from [10])

$$|B(x) - B^p(x)| \leq \frac{A_y \overline{B(y)} (r_x + \overline{r_y})^2}{r_x^4} \sum_{n=p+1}^{\infty} \left(\frac{\overline{r_y}}{r_x}\right)^n \binom{n+3}{3}$$

where

$$\overline{B(y)} = \max_{A_y} |B(y)|$$
$$\overline{r_y} = \max_{A_y} |r_y|$$

This however is a conservative bound – we provide empirical results (Figure 11) to show that truncating $p = 2$ gives acceptable results.

2.3. Local Expansion

For $r_x < r_y$, we derive [12] an expression similar to the multipole expansion by interchanging the r_x and r_y terms in (4) called the *local expansion*

$$B(x) - E(x) = \sum_{n=0}^{\infty} \sum_{j=0}^{[n/2]} \sum_{m=-n+2j}^{n-2j} e_n^j E_{nj}^m(x) \otimes L_{nj}^m(O) \quad (8)$$

$$E_{nj}^m(x) = \rho(x) r_x^n \overline{Y_{n-2j}^m(\theta_x, \phi_x)} \mathbf{RM}(x) \quad (9)$$

$$L_{nj}^m(A_y) = \int_{A_y} B(y) \frac{1}{r_y^{n+4}} Y_{n-2j}^m(\theta_y, \phi_y) \mathbf{SM}(y) dA_y \quad (10)$$

Our intuition behind this formulation is explained in Figure 4.

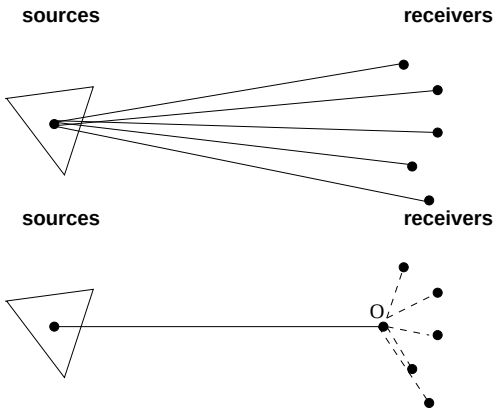


Figure 4. By associating constant number of coefficients stored at center O, we can calculate the energy received by different receivers. The value of coefficients depends upon the location of source patch

Similar to the multipole coefficients, the *local coefficients* L_{nj}^m are also additive as shown in Figure 5.

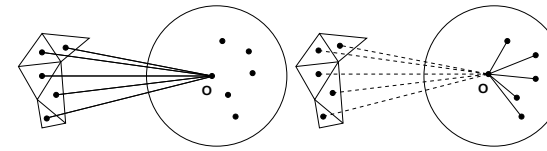


Figure 5. Local coefficients are additive.

2.4. Multipole Translation

We have shown [12] that the multipole coefficients of a set of patches in a coordinate system centered at O', denoted by $M_{nj}^m(O')$ can be expressed in terms of the multipole coefficients of the same set of patches in a coordinate system translated to O, denoted by $M_{nj}^m(O)$. The relation is not presented here for the sake of brevity.

2.5. Local Translation

We have also shown [12] that the local coefficients of a set of patches in a coordinate system centered at O', denoted by $L_{nj}^m(O')$ can be expressed in terms of the local coefficients of the same set of patches in a coordinate system translated to O, denoted by $L_{nj}^m(O)$. The relation is not presented here for the sake of brevity.

2.6. Multipole to Local Translation

A crucial part of FMM is the conversion of multipole coefficients at a given center O into local coefficients at another center O'. We have shown (illustrated in Figure 6) [12] that

$$L_{ksm_1}(O') = \sum_{n=0}^{\infty} \sum_{j=0}^{[n/2]} \sum_{m=-n+2j}^{n-2j} \sum_{l_2=l_{min}}^{l_{max}} e_n^j J(k-2s, m_1, l_2, s, -n-4, m, n-2j) F_{ksm_1 l_2}^{njm}(OO') \otimes M_{nj}^m(O) \quad (11)$$

where the summation over l_2 proceeds in steps of two and

$$l_{max} = n - 2j + k - 2s$$
$$l'_{min} = max(|n-2j-k+2s|, |m-m_1|)$$
$$l_{min} = \begin{cases} l'_{min} & n-2j-k+2s-l'_{min} \quad even \\ l'_{min}+1 & n-2j-k+2s-l'_{min} \quad odd \end{cases}$$
$$F_{ksm_1 l_2}^{njm}(O0') = |OO'|^{-n-4-k} Y_{l_2}^{m-m_1}(OO') \mathbf{TM}(OO')$$

3. Quadrature

Since the terms inside the integral in (7) are polynomial across the surface, we can use Gaussian quadrature to calculate exactly the integral to sufficient accuracy. If the area A_y

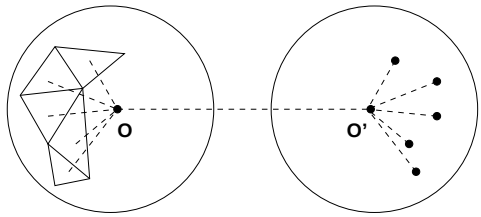


Figure 6. The multipole to local translation converts the multipole coefficients of a set of N patches into local coefficients for a set of M receiver points.

is represented by the set of quadrature points y with weight $w(y)$, we have

$$M_{nj}^{m}(A_y) = \sum_{\text{quad. points}\in y} w(y)B(y)r_y^n\overline{Y_{n-2j}^{m}(\theta_y,\phi_y)}\mathbf{SM}(y) \tag{12}$$

If we define the multipole coefficients (and similarly local coefficients) for a quadrature point y as

$$M_{nj}^{m}(y) \quad = \quad w(y)B(y)r_y^n\overline{Y_{n-2j}^{m}(\theta_y,\phi_y)}\mathbf{SM}(y)$$

we can replace the interaction between surfaces and points (in Equation 7 for example) as between points only. This interaction is termed as a *particle interaction*. Evaluating the multipole expansion of a set of surfaces at a particle y becomes equivalent to evaluating the multipole expansion of all the quadrature points at y. The translation properties shown previously will continue to hold for multipole and local coefficients of particles.

4. The Algorithm

The algorithm presented below assumes a fixed number of mutually visible triangles of area sufficient in order to approximate the integral in (5) by Gaussian quadrature to high degree. (When occlusion is present in the scene, the only change is that interaction list is modified as in Section 5.)

Step 1 Setup

All triangles are converted into Gaussian quadrature points. All points are arranged in an adaptive octree such that no leaf contains more than s points. s is called the *grouping factor*. With each box, we associate two set of disjoint boxes:

- *near neighbors* of a box b are boxes that share a common boundary point. Points in these boxes do not satisfy the distance constraint in (5).
- *interaction list* of a box b are the children of the near neighbors of the parent of b — children who are *not* near neighbors of b itself.

In addition, leaf boxes have the following set associated:

- *local interaction list* of a box b are the leaves in the descendants of the near neighbors of b, which are not near neighbors of b themselves. These are boxes which are too close to interact via a multipole-local translation (Section 2.6) but the local expansion coefficients due to individual points in these leaves can be evaluated and accumulated at the box as shown in Figure 4
- *multipole interaction list* of a box b is the inverse of the local interactions list. These boxes are close enough to prevent their multipole expansion from being converted into a valid local expansion. However, their multipole expansion can be evaluated at each particle in b as shown in Figure 2.

Step 2 Multipole Coefficients

For each leaf box in the octree, we calculate the multipole coefficients of all points contained in the box about its center using (12). Then, for each level, starting from the penultimate level, we calculate the multipole coefficients at each box in that level by translating and accumulating the multipole coefficients of its children (Section 2.4).

Step 3 Downward Pass

For each level, starting from the second, the local coefficients at each box b are calculated by converting the multipole coefficients of boxes in the interaction list of b into local coefficients about b's center using (11). Additionally, the local expansion coefficients obtained from the individual points contained in the local interaction list are aggregated.

Step 4 Evaluation

For each leaf b in the octree, for each evaluation point $y \in b$, the local expansion about the center of b is evaluated at y using (8). Additionally, the multipole expansion due to boxes in the multipole interaction list are evaluated at y using (5). Direct computations if necessary, are also aggregated for points in b.

We iterate over these steps as in shooting [4] till a sufficient convergence is reached. Note that in the final step of the final iteration, the evaluation points can be separate from the quadrature points.

5. Handling Occlusion

Visibility in FMM is complicated by the fact that it is a point to point based phenomenon and not from box to box. When visibility is introduced in (1), the transfer of energy should be handled differently from the method in Section 4. Specifically, each particle receives energy from every other particle either directly, or through the surfaces in the interaction list of its ancestors. The key idea is to modify the interaction list much the way the adaptive version of the

```
Procedure Modify(Box A) {
visible_interactionlist(A)=Null
    for each box B ∈ old_interationlist(A){
       state=visibility(A,B)
       if equals(state,valid) then
          visible_interactionlist(A).Include(B)
       else if equals(state,partial) then
           { if(notLeaf(A))
                 for each a ∈ child(A)
                 for each b ∈ child(B)
                  interactionlist(a).Include(b)
           }}}

Procedure Generate(Box A){
      Modify(A)
      for each a ∈ child(A)
       Generate(a)}
```

Figure 7. This is the algorithm that generates the visible_interaction list for each node in the octree

```
Procedure visiblity(Box A, Box B) {
    visible=0;
    for each cell a ∈ leafcell(A)
     for each cell b ∈ leafcell(B)
     {
      c1=center(a)
      c2=center(b)
      if Positive(DotProduct(c2 − c1,av_normal(a)))
         and Positive(DotProduct(c1 − c2,av_normal(b)))
          then do
          { result=shootandDetect(a,b)
           if Equals(result,0) then do, Increment(visible,1)
          }
    }
    if Equals(visible,0) return(nonvisible)
    else if Equals(visible,leafcell(A).size*leafcell(B).size)
          return(visible)
        else return(partial)
}
```

Figure 8. Finding the state of visiblity between two big boxes A and B.

FMM works. If the surfaces in a box c in the interaction list of box b are completely visible from *every* particle in b, then the *visibility state* of the pair (b,c) is said to be *valid*. If, on the other hand, no surface in c is visible from any particle in b, the visibility state of the pair (b,c) is said to be *invalid*. The box c is dropped from the interaction list since no exchange of energy is permissible. Finally, when the visibility state is *partial*, we *postpone* the interaction. Algorithm `Generate` (Fig. 7) which calls `Modify` ensures that the postponed interaction happens at the lowest possible depth (the root is at depth 0) for maximum efficiency. The complexity of these routines depends on the visibility function discussed in Section 5.1.

The justification in the use of the visible interaction list in the FMM algorithm presented in Section 4 follows from

Lemma 5.1 *If B is in interaction list of A then* $b \in child(B)$ *is eligible to be in interaction list of* $a \in child(A)$

Proof: Omitted in this version. □

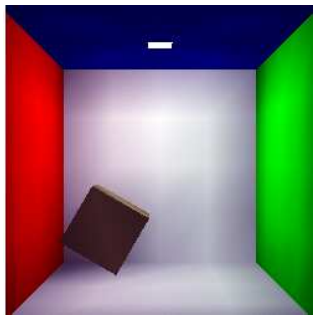
5.1. Computing Box to Box Visiblity

Finding the *state of visibility* between two large sized boxes containing some particles requires knowing the visibility state between the constituent particles. As in the FMM, this becomes tractable when the box pair becomes so tiny that there are very few particles in each box. The visibility state can then be computed by `shootAndDetect()` in $O(r)$ time for these tiny boxes termed *cells*. $O(r)$ is empirically determined to be $O(1)$ since the precise way we do this uses the undivided input polygons, a quantity that is negligibly small compared to N, the number of quadrature points. This information is propagated to the larger sized boxes in procedure visiblity(), shown in Figure 8.

5.2. Computation Complexity: Visibility

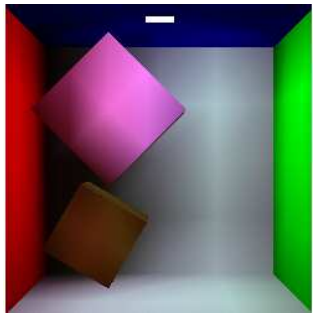
Consider the call `visibility(A,B)` for boxes A and B which results in a call to `shootAndDetect(a,b)` for cells a and b. If there was no postponement of the interaction between A and B, then clearly `shootAndDetect(a,b)` will not be called again. On the other hand, if there *was* partial visibility, shootAndDetect(a,b) will be called again. Therefore the number of calls to shootAndDetect() for children of A is proportional to the total number of boxes postponed in the interaction of list of A.

Let P_i be the expected number of boxes at level i postponed by a box A at level i. Thus the total number of boxes in the interaction list is $189 + 8P_{i-1}$. For box A, the number of calls to shootAndDetect() is $(189 + 8P_{i-1})\frac{n^2}{8^{2i}}$. For purpose of analysis, it is useful to define α to be the fraction in the equation below

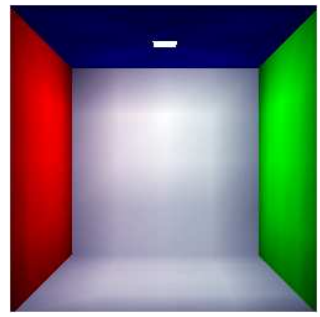


(a) The Cornell room with one block.

(b) An alternative Scene.



(c) The original Cornell room.



(d) The unoccluded Cornell room.

Figure 9. The FMM in action.

$$P_i = \alpha(189 + 8P_{i-1}) \tag{13}$$

Note that $P_0 = P_1 = 0$ and $P_2 = 189\alpha$. Solving the recursion in (13) we have

$$P_{i-1} = 189\alpha((8\alpha)^{i-2})/(8\alpha - 1)$$

The overall complexity of `Generate(root)` (by amortized analysis) is

$$\Sigma_{i=3}^{\log n}(189 + 8P_{i-1})8^i * n^2/8^{2i}$$

The complexity of the construction of the visibility interaction list is thus between $O(n^2)$ and $O(n^2 \log(n))$.

5.3. Computational Complexity: Light calculations

For the case with occlusion, the algorithm in Section 4 is run using the visible interaction list instead of the interaction list. The upward pass is unaffected.

For the full visibility case, the postponement factor α is zero and thus the time complexity will remain $O(N+n)$, where N is the number of particles and n is the number of

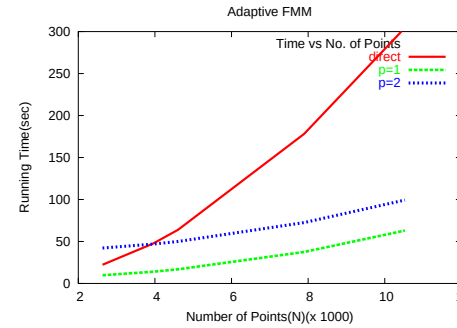


Figure 10. The running time for adaptive FMM for different truncation values (p) and brute force method.

cells. If the postponement factor α is 1, it means that no cell is visible to any other cell at *any* level. No transfer of energy is required, and hence complexity will remain $O(N+n)$ (traversing the tree once). In the worst case, where there is complete postponement at upper levels and only some rejections due to non-visibility at leaf levels the complexity is about $O(N+n^2)$. So, the complexity will fluctuate from $O(N+n)$ to $O(N+n^2)$, depending upon the postponement factor. This postponement factor is completely determined by the geometry of the scene and octree construction,i.e., subdivision of the scene into subparts.

6. Results

Various scenes have been rendered in Figure 9 using our FMM algorithm. For purposes of timing, we tested the algorithm for the Cornell room shown in Fig. 9(d) varying the number of input triangles. Fig. 10 plots the time taken by the algorithm at several truncation numbers vs. the time taken by the brute force method and show the linear time nature of our method. Fig. 11 shows the average relative error incurred for varying truncation numbers.

Fig 12 shows the running time observed for different scenes having different values of the postponement factor α. It also includes the visibility computation time.

7. Conclusion

The FMM method is elegant because it trades of error with quality in a disciplined quantitative way. In this paper, the kernel of the energy balance in the rendering equation has been made conformant to the FMM by deriving the near and far field expansions. We have also presented derivations for the translation operators of the expansion coefficients. The radiosity problem over surfaces is thus reduced to a solution over points, resembling point based rendering. The

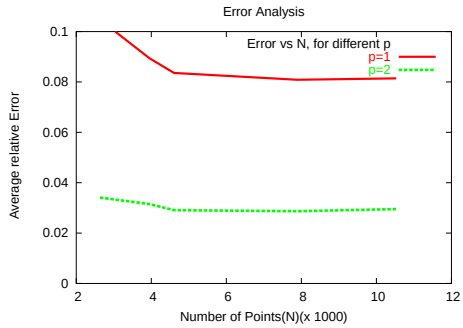


Figure 11. Average relative error for different trancation values (p).

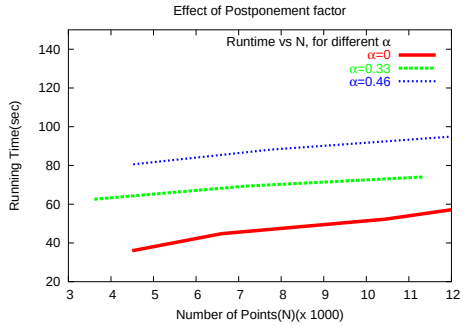


Figure 12. Running time for different scenes, having different postponement factor α in an octree of level 3.

use of Gaussian quadrature rules enables us to use the standard FMM. The new algorithm is expected to be useful for scenes containing highly tessellated models (a huge value of N) since a large number of polygons in a cluster can be represented by a constant number of coefficients thereby giving a $O(N)$ light computation algorithm. In particular, we have implemented both adaptive and non-adaptive versions of the FMM.

We have also given a new visibility algorithm whose complexity is independent on the number of FMM particles N. The visiblity calculation can be done as a 'preprocessing' step. When done this way, changing the lighting effects in the room can be done easier than other methods. BF refinement of hierarchical radiosity[9] requires the modification of the link structure after the upward and downward pass is run.

Several improvements could be possibly realized in the current technique. The multipole expansion is derived by expanding the numerator and denominator of the kernel separately. The large number of terms in the expansion of the numerator explode the number of terms needed to accurately represent the kernel. There could possibly be an expansion which expands the kernel as a whole and not in separate parts. A possible reduction in the number of terms for the same accuracy will greatly increase the efficiency of this method.

In our implementation of FMM, we have taken near neighbors to be two boxes who share a boundary point. There are generalizations of this definition to two boxes separated by k boxes[8]. These schemes will give very high accuracy but may cause a performance hit.

Acknowledgements

We are grateful to Vikas Jain for his help in the implementation and other members of ViGIL, IIT Bombay for their support.

References

[1] A. W. Appel. An efficient program for many-body simulation. *SIAM Journal of Scientific and Statistical Computation*, 6:85–103, 1985.

[2] J. Barnes and P. Hut. A hierarchical O($N \log N$) force-calculation algorithm. *Nature*, 324:446–449, 1986.

[3] R. Beatson and L. Greengard. A short course on fast multipole methods.

[4] M. F. Cohen and J. R. Wallace. *Radiosity and Realistic Image Synthesis*. Academic Press, 1993.

[5] J. J. Dongarra and F. Sullivan. The top ten algorithms. *Computing in Science and Engineering*, 2:22–23, 2000.

[6] L. Greengard. *The rapid evaluation of potential fields in particle systems*. MIT Press, Cambridge, Massachusetts, 1988.

[7] L. Greengard and V. Rokhlin. A fast algorithm for particle simulations. *Journal of Computational Physics*, 73:325–348, 1987.

[8] N. A. Gumerov, R. Duraiswami, and E. A. Borikov. Data structures, optimal choice of parameters, and complexity results for generalized multilevel fast multipole methods in d dimensions. Technical report, Perceptual Interfaces and Reality Laboratory, Institute for Advanced Computer Studies, University of Maryland, College Park, 2003.

[9] P. Hanrahan, D. Salzman, and L. Aupperle. A rapid hierarchical radiosity algorithm. *Computer Graphics*, 25(4):197–206, July 1991.

[10] A. Haunsner. Multipole expansion of the light vector. *IEEE Transactions on Visualization and Computer Graphics*, 3(1):12–22, Jan-Mar 1997.

[11] R. W. Hockney and J. W. Eastwood. *Computer Simulation using Particles*. McGraw-Hill, New York, 1981.

[12] A. Karapurkar. The Fast Multiple Method for global illumination. Master's thesis, Indian Institute of Technology, Bombay, 2003.

[13] S. M. Rusinkiewics. A new change of variables for efficient brdf representation. In *"Rendering Techniques"(proceedings of the Eurographics Workshop on Rendering*, jun 1998.