Pergamon

Comput. & Graphics, Vol. 22, No. 2–3, pp. 225–232, 1998

PII: S0097-8493(98)00011-9
Printed in Great Britain
0097-8493/98 $19.00 + 0.00

Technical Note

AN ACCURACY ENHANCEMENT ALGORITHM FOR HIERARCHICAL RADIOSITY

CHIN-CHEN CHANG and ZEN-CHUNG SHIH†

Department of Computer and Information Science, National Chiao Tung University, Hsinchu, Taiwan 30010, Republic of China
e-mail: zcshih@cc.nctu.edu.tw

Abstract—In this research, we propose an algorithm to enhance the accuracy of the hierarchical radiosity. The proposed approach particularly creates links on the hierarchy in a bottom-up fashion according to both a user-specified error tolerance and the minimum patch area. It differs from the original hierarchical method that creates the links in a top-down manner. In our approach, the upper-level patches have more accurate estimates of the kernel. More accurate error bounds can be obtained for the upper-level patches. These error bounds make the creation of links on the hierarchy more accurate. Furthermore, more accurate form factors can also be obtained for the upper-level patches. Thus, we enhance the accuracy of hierarchical radiosity. © 1998 Elsevier Science Ltd. All rights reserved.

Key words: global illumination, radiosity, hierarchical radiosity, form factor, error bound.

1. INTRODUCTION

In computer graphics, radiosity provides a solution of the global illumination problem in realistic image synthesis. In 1984, Goral *et al.* [1] initially proposed the basic radiosity algorithm by meshing the environment and then solving a linear system to approximate the transfer of energy among surface elements. Unfortunately, achieving accurate results required a large number of potential elements and the computational cost was quadratic to the number of elements.

Cohen *et al.* [2] initially proposed a two-level hierarchical radiosity algorithm known as substructuring. All surfaces are subdivided coarsely into patches and then finely into elements. Although the number of interactions are reduced, the computational cost of the algorithm is $O(mn)$, where m is the number of patches and n is the number of elements. Hanrahan *et al.* [3] generalized this idea to a multi-level hierarchy from Appel's algorithm for solving the N-body problem [4]. All surfaces are subdivided recursively into patches and elements and the links on the hierarchy are created according to both a user-specified error tolerance and the minimum patch area. This approach dramatically reduces the computational cost from $O(n^2)$ to $O(s^2+n)$, where n denotes the total number of resulted elements and s represents the number of input surfaces. However, the cost remains quadratic to the number of input surfaces.

Several improvements and extensions have been made for hierarchical radiosity. Teller and Hanrahan [5] proposed global visibility algorithms that preprocess polygon databases to accelerate visibility determination during illumination calculations. However, global visibility algorithms do not suffice for environments involving large collection of objects that are mutually visible. Smits *et al.* [6] proposed an importance-driven radiosity algorithm that reduces the computational cost for global solutions with respect to a particular view. This algorithm is effective for complex environments in which only a small fraction is visible. Kok [7] proposed the radiosity method by adaptively grouping small neighbouring patches into groups. Thus, patches could efficiently transfer energy to groups of patches immediately. Smits *et al.* [8] proposed an approach for accelerating hierarchical radiosity by clustering objects. This approach estimates energy transfers between clusters while maintaining reliable error bounds on each transfer. Hence, the cost of hierarchical radiosity can be reduced, especially for complex environments. Lischinski *et al.* [9] improved hierarchical radiosity to achieve better numerical accuracy and faster convergence by combining discontinuity meshing [10, 11]. Sillion and Drettakis [12] proposed an approach to control error for hierarchical radiosity clustering algorithms. This approach ensures that just enough work is done to meet the user's quality criteria. Chang and Shih [13] proposed an accurate method of form-factor estimation for hierarchical radiosity. This approach can reduce the number of links and computational cost to meet a user-specified error tolerance. Gibson and Hubbold [14] proposed a

† Corresponding author. Tel: 886-3-5731916; Fax: 886-3-5721490.

rapid hierarchical algorithm that enables accurate solutions to be computed quickly and efficiently. The hierarchical radiosity algorithms for more complex environments, with particiating media [15, 16] and glossy environments [17, 18], were also proposed. A more general hierarchical method, that of wavelet radiosity, was also devised [19–21].

Herein, we propose an accuracy enhancement algorithm for hierarchical radiosity. The rest of this work is organized as follows. Section 2 describes the hierarchical radiosity. Section 3 describes our approach. Implementation and results are addressed in Section 4, with conclusions in Section 5.

2. HIERARCHICAL RADIOSITY

This section reviews the basic concepts of the hierarchical radiosity [3]. The kernel of the hierarchical radiosity is a recursive refinement procedure recursively subdividing each surface into patches and elements and creates links on the hierarchy at different levels. For each pair of patches, the procedure initially estimates approximate upper bounds on the error of form factors and uses it as the subdivision criteria. If the estimated bounds are less than the user-specified error tolerance, we establish a link and compute the form factors between the corresponding patches. Otherwise, the patch with larger form factor is subdivided, and the procedure is performed recursively for the smaller subpatches. Besides bounding the error on form factors, a brightness-weighted refinement strategy (BF refinement) [3] also can be employed as the subdivision criteria.

A good criteria of determining whether links a pair of patches or not is essential for hierarchical radiosity. Up to now, several subdivision criterias for improving the hierarchical radiosity are available. Smits *et al.* [6] proposed an importance-weighted refinement strategy. The importance-weighted error bounds can be applied as a subdivision critera to achieve accurate radiosity. Smits *et al.* [8] analysed the bounding error on transfers. The error bounds thus obtained are taken as the errors in form factors. Although the error bounds are less conservative than Hanrahan's, they are generally more accurate [22]. Lischinski *et al.* [22] derived an error-driven adaptive refinement strategy for hierarchical radiosity. This strategy is superior to the BF refinement. We follow the notation of Smits *et al.* [8] in finding the error bounds on form factors.

3. OUR APPROACH

Basically, we focus on the accuracy enhancement of the hierarchical radiosity approach [3]. We discover the error bounds and establish the links among patches in a bottom-up fashion. Our approach increases the accuracy of form factors.

3.1. *Bounding error on form factors*

The form factor from a point x on patch p to another finite-area patch q is given as follows:

$$F_{pq} = \int_q G(x, y)\mathrm{d}y$$

where y is a point of q and $G(x, y)$ is the kernel defined by

$$G(x, y) = \frac{\cos\theta_x \cos\theta_y}{\pi r^2}\, v(x, y)$$

where r denotes the distance between x and y; θ_x is the angle formed by the normal of the differential area at x and the direction given by $y - x$; θ_y is the angle formed by the normal of the differential area at y and the direction given by $x - y$, and $v(x, y)$ is the visibility function.

The error bound on the form factor can be obtained by bounding the difference between the maximum and the minimum form factors of two patches. Let

$$\lceil G\rceil_{p,q} \equiv \max_{x\in p,\, y\in q} G(x, y)$$

and

$$\lfloor G\rfloor_{p,q} \equiv \min_{x\in p,\, y\in q} G(x, y)$$

Then, the error bound on the form factor from p to q is given by $(\lceil G\rceil_{p,\,q} - \lfloor G\rfloor_{p,\,q})A_q$, where A_q is the area of q. Besides, the form factor from p to q is $\langle G\rangle_{p,\,q}A_q$, where $\langle G\rangle_{p,\,q}$ is defined as

$$\langle G\rangle_{p,q} \equiv avg_{x\in p,\, y\in q}\, G(x, y)$$

The maximum, the minimum and the average values of G between two patches can be obtained by using the point sampling technique [8].

3.2. *Basic idea*

Let p and q be the two patches. Consider Fig. 1. Suppose that both p and q involve two levels of hierarchy. Let p_i be a subpatch of p, i = 1, 2, 3, 4, and q_j be a subpatch of q, j = 1, 2, 3, 4. (p_i could be p and q_j could be q). To obtain error bounds on form factors between p and q, we need to find $\lceil G\rceil_{p,\,q}$ and $\lfloor G\rfloor_{p,\,q}$. We initially find the maximum and the minimum values of G between p_i and q_j. Then the maximum and the minimum values of G between p and q can be obtained as follows:

$$\lceil G\rceil_{p,q} = \max_{i,\, j\in 1,\, 2,\, 3,\, 4}\left\{\lceil G\rceil_{p_i,q_j}\right\}$$

and

$$\lfloor G\rfloor_{p,q} = \min_{i,\, j\in 1,\, 2,\, 3,\, 4}\left\{\lfloor G\rfloor_{p_i,q_j}\right\}$$

respectively. The same approach can be applied to the multi-level hierarchy.

Consider two observations. First, if the error bounds between p and q do not satisfy the error tolerance, we can reuse $\lceil G\rceil_{p_i,\,q_j}$ and $\lfloor G\rfloor_{p_i,\,q_j}$ in the computation of error bounds between p_i and q_j.

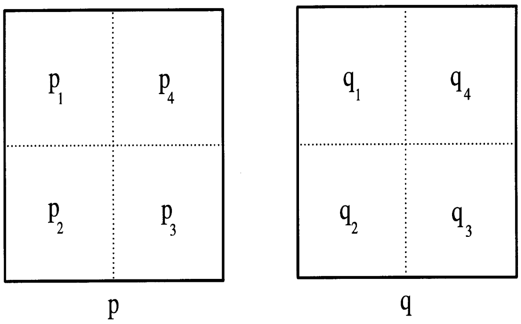

Fig. 1. Both patches p and q have two levels of hierarchy.

Second, we compute $\lceil G \rceil_{p_i, q_j}$ and $\lfloor G \rfloor_{p_i, q_j}$ before finding $\lceil G \rceil_{p, q}$ and $\lfloor G \rfloor_{p, q}$. Once $\lfloor G \rfloor_{p_i, q_j}$ and $\lfloor G \rfloor_{p_i, q_j}$ are obtained, we can employ them directly to compute the error bounds between p_i and q_j and determine whether creating a link between p_i and q_j. From these observations, we can compute error bounds and create the links between patches on the hierarchy in a bottom-up manner.

3.3. *The procedure*

Based on the above discussion, we describe the **BottomUpRefine** procedure. We perform a preprocessing on the surfaces of the input object model. For each surface s, according to the user-specified minimum patch area A_{eps}, we construct a complete quadtree T_s by recursively subdividing the surface s uniformly into patches and elements as far as possible. In our approach, A_{eps} plays a significant role in determining the performance and accuracy. If A_{eps} is too high, it will lose accuracy. If A_{eps} is too low, it will oversample the kernel and the cost will be expensive. It is not easy to determine the adequate value of A_{eps} for an environment. We can initially explore the maximal patch and then set A_{eps} according to the area of the maximal patch. For instance, if we desire to construct complete quadtrees with at most four levels, A_{eps} is set to 1/64 of the area of the maximal patch.

To simplify matters, we introduce the notations employed in the **BottomUpRefine** procedure. Consider Fig. 2. Let l_s be the total number of levels of T_s, n_s be a node of T_s, and $child(n_s)$ be a child node of n_s if n_s is not a leaf node. The level d_s of n_s is defined as

$$d_s \equiv h_s - \text{depth}_s + 1$$

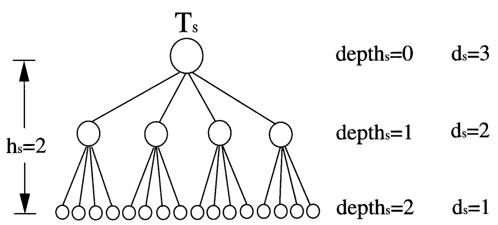

Fig. 2. The complete quadtree T_s with l_s.

where h_s is the height of T_s and depth$_s$ is the depth of n_s. The error bound from n_p to n_q is defined as $E_{n_p n_q} = (\lceil G \rceil_{n_p, n_q} - \lfloor G \rfloor_{n_p, n_q}) A_{n_q}$, where A_{n_q} is the area of n_q. The following is a formal description of the **BottomUpRefine** procedure.

3.3.1. *Procedure* **BottomUpRefine**(**T_p**, **T_q**, **F_{eps}**). /*Quadtrees T_p and T_q and a user-specified error tolerance F_{eps}.*/

Step 1
Let $d_p = 1$ and $d_q = 1$.
Step 2
For each pair of nodes n_p and n_q, where n_p is at the level d_p of T_p and n_q is at the level d_q of T_q.
If it is in the initial iteration, compute $\lceil G \rceil_{n_p, n_g}$ and $\lfloor G \rfloor_{n_p, n_q}$ by the point sampling technique; otherwise, consider three cases.
Case 1: $d_p < d_q$. Compute $\lceil G \rceil_{n_p, n_q}$ and $\lfloor G \rfloor_{n_p, n_q}$ by employing the maximum and the minimum values of G between n_p and each child $child(n_q)$ of n_q obtained in the previous iteration.
Case 2: $d_p = d_q$. Compute $\lceil G \rceil_{n_p, n_q}$ and $\lfloor G \rfloor_{n_p, n_q}$ by employing the maximum and the minimum values of G between each child $child(n_p)$ of n_p and each child $child(n_q)$ of n_q obtained in the previous iteration.
Case 3: $d_p > d_q$. Compute $\lceil G \rceil_{n_p, n_q}$ and $\lfloor G \rfloor_{n_p, n_q}$ by employing the maximum and the minimum values of G between each child $child(n_p)$ of n_p and n_q obtained in the previous iteration.
If **Error**(**n_p**, **n_q**, **F_{eps}**) returns false, do **CreateLink**(**n_p**, **n_q**, **F_{eps}**); otherwise, if $d_p = l_p$ and $d_q = l_q$ hold, link n_p and n_q. If no any **Error**(**n_p**, **n_q**, **F_{eps}**) in this iteration returns true, **STOP**.
Step 3
There are four cases:
Case 1: $d_p < l_p$ and $d_q < l_q$. Let $d_p = d_p + 1$ and $d_q = d_q + 1$, go to Step 2 for the next iteration.
Case 2: $d_p < l_p$ and $d_q = l_q$. Let $d_p = d_p + 1$, go to Step 2 for the next iteration.
Case 3: $d_p = l_p$ and $d_q < l_q$. Let $d_q = d_q + 1$, go to Step 2 for the next iteration.
Case 4: $d_p = l_p$ and $d_q = l_q$. **STOP**.
END.

In Step 2 we check if the error bounds between n_p and n_q are less than the user-specified error tolerance F_{eps}. This can be accomplished by the **Error** procedure. This procedure returns a Boolean value and is defined as follows:

3.3.2. *Procedure* **Error**(**n_p**, **n_q**, **F_{eps}**). /*Nodes n_p and n_q and the user-specified error tolerance F_{eps}.*/

Step 1
If $E_{n_p n_q} < F_{\text{eps}}$ and $E_{n_q n_p} < F_{\text{eps}}$ hold, returns true; otherwise, returns false.
END.

If the error bounds $E_{n_p n_q}$ and $E_{n_q n_p}$ are not both less than the user-specified error tolerance F_{eps}, it is

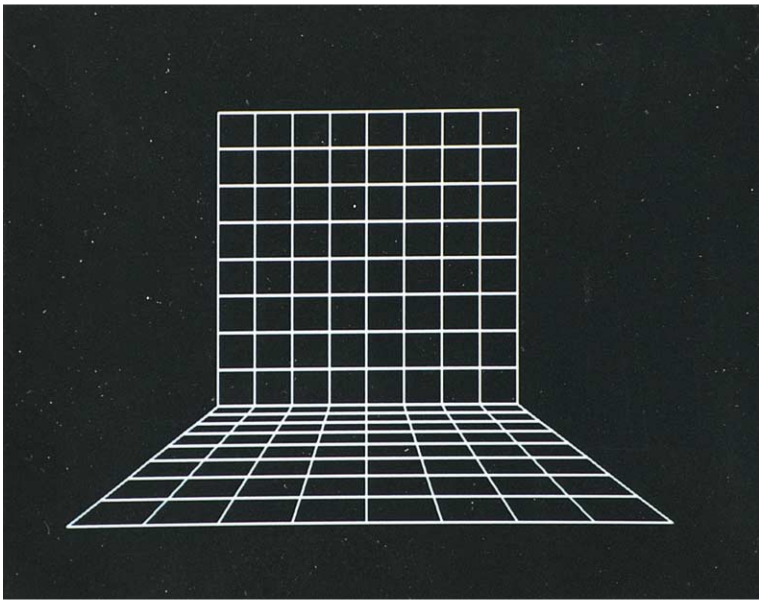

Fig. 3. The complete quadtree subdivision.

possible to set up a link between n_p or *child*(n_p) and n_q or *child*(n_q). This can be accomplished by invoking the **CreateLink** procedure. It is defined as follows:

3.3.3. *Procedure* **CreateLink**($\mathbf{n_p}$, $\mathbf{n_q}$, $\mathbf{F_{eps}}$). /*Nodes n_p and n_q and the user-specified error tolerance F_{eps}.*/

Step 1

If both n_p and n_q are leaf nodes, link n_p and n_q; otherwise, consider the following three cases.

Case 1: $d_p < d_q$. For n_p and each child *child*(n_q) of n_q, compute the maximum and the minimum values of G from the data in the previous iteration. If **Error**($\mathbf{n_p}$, **child**($\mathbf{n_q}$), $\mathbf{F_{eps}}$) returns true, link the corresponding two nodes n_p and *child*(n_q).

Case 2: $d_p = d_q$. There are two subcases.

Case 2.1: $E_{n_p n_q} > E_{n_q n_p}$. For n_p and each child *child*(n_q) of n_q, compute the maximum and the minimum values of G from the data in the previous iteration. If **Error**($\mathbf{n_p}$, **child**($\mathbf{n_q}$), $\mathbf{F_{eps}}$) returns true, link the corresponding two nodes n_p and *child*(n_q); otherwise, for each child *child*(n_p) of n_p and each child *child*(n_q) of n_q, compute the maxi-

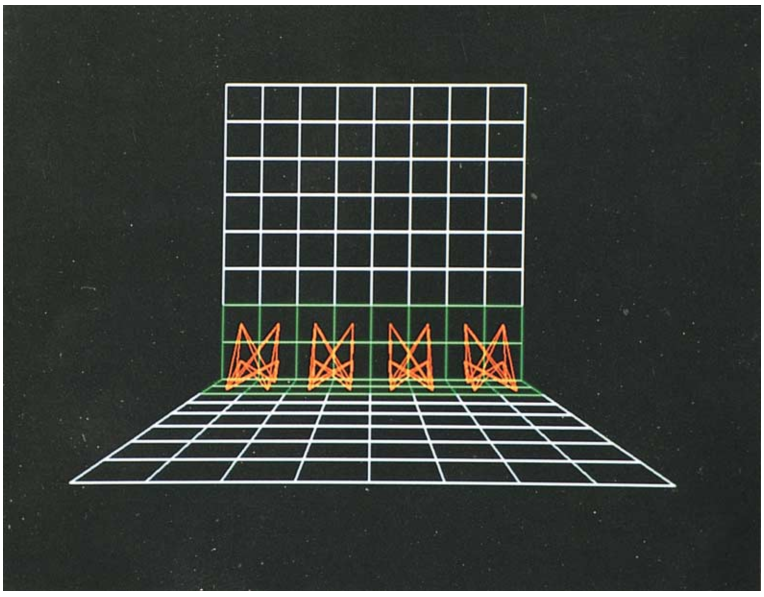

Fig. 4. The links at the first level on the hierarchy.

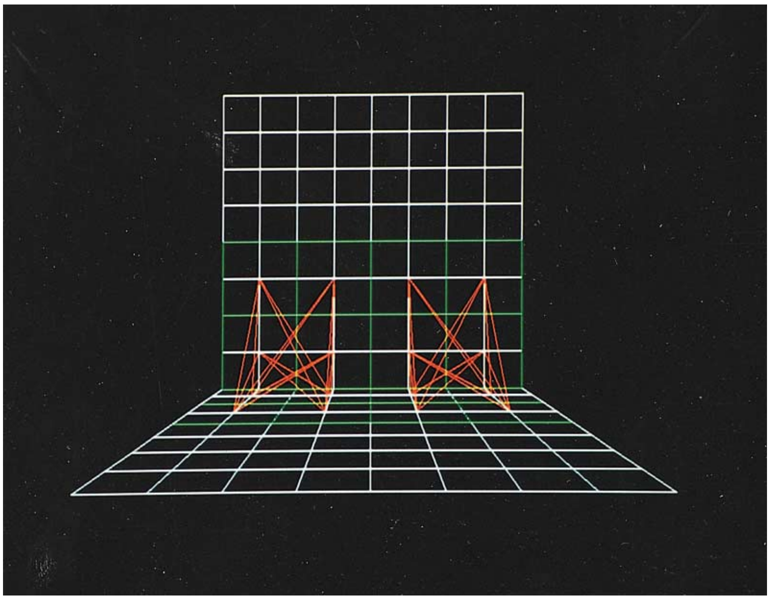

Fig. 5. The links at the second level on the hierarchy.

mum and the minimum values of G from the data in the previous iteration. If **Error(child(n_p), child(n_q), F_{eps})** returns true, link the corresponding two nodes *child*(n_p) and *child*(n_q).

Case 2.2: $E_{n_p n_q} \leq E_{n_q n_p}$. For each child *child*(n_p) of n_p and n_q, compute the maximum and the minimum values of G from the data in the previous iteration. If **Error(child(n_p), n_q, F_{eps})** returns true, link the corresponding two nodes *child*(n_p) and n_q; otherwise, for each child *child*(n_p) of n_p and each child *child*(n_q) of n_q, compute the maximum and the minimum values of G from the data in the previous iteration. If **Error(child(n_p), child(n_q), F_{eps})** returns true, link the corresponding two nodes *child*(n_p) and *child*(n_q).

Case 3: $d_p > d_q$. For each child *child*(n_p) of n_p and n_q, compute the maximum and the minimum values of G from the data in the previous iteration. If **Error(child(n_p), n_q, F_{eps})** returns true, link the corresponding two nodes *child*(n_p) and n_q.

END.

Figures 3–6 depict the quadtree subdivision and the links at each level on the hierarchy in a bottom-up fashion between a pair of perpendicular patches.

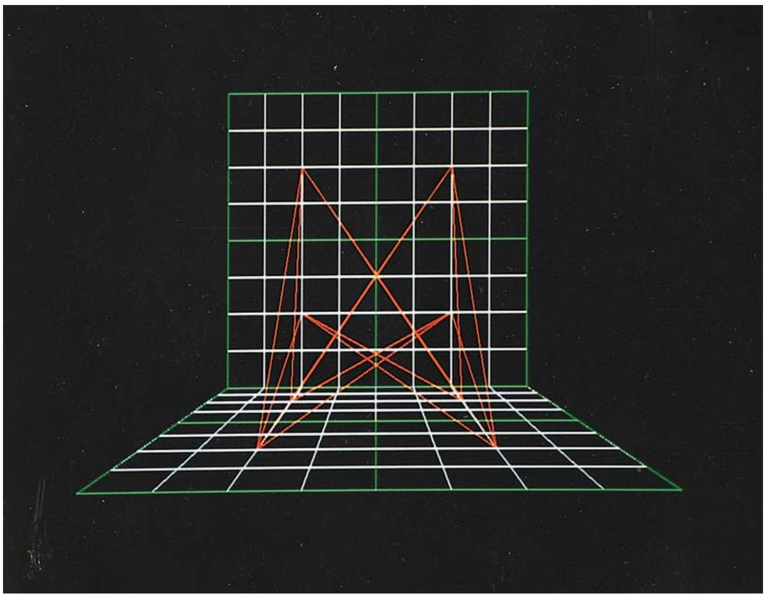

Fig. 6. The links at the third level on the hierarchy.

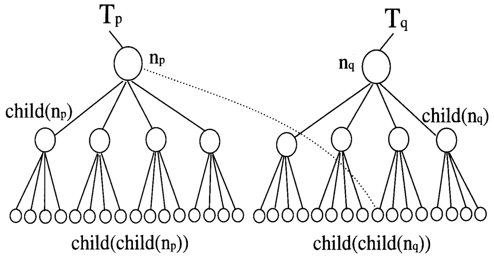

Fig. 7. The link between n_p and *child*(*child*(n_q)) does not exist.

3.4. *Verification*

Without loss of generality, as shown in Fig. 7, let n_p and n_q be at the same level and *child*(*child*(n_p)) and *child*(*child*(n_q)) exist. If **Error**($\mathbf{n_p}$, $\mathbf{n_q}$, $\mathbf{F_{eps}}$) returns false, in the **CreateLink** procedure, it only needs to determine whether creating a link between n_p and *child*(n_q) or *child*(n_q) and n_p or *child*(n_p) and *child*(n_q). Since **Error**($\mathbf{n_p}$, **child**($\mathbf{n_q}$), $\mathbf{F_{eps}}$) returns false, there are two possible cases.

Case 1: $E_{n_p n_q} > E_{n_q n_p}$, we should determine whether creating a link between n_p and *child*(n_q). If **Error**($\mathbf{n_p}$, **child**($\mathbf{n_q}$), $\mathbf{F_{eps}}$) returns true, we link the corresponding two nodes n_p and *child*(n_q). Otherwise, by uniform subdivision, the area of n_p is larger than the area of *child*(n_q). This fact implies that $E_{n_p child(n_q)} < E_{child(n_q) n_p}$ holds. We should determine whether creating a link between *child*(n_p) and *child*(n_q). If **Error**(**child**($\mathbf{n_p}$), **child**($\mathbf{n_q}$), $\mathbf{F_{eps}}$) returns true, link the corresponding two nodes *child*(n_p) and *child*(n_q); otherwise, do nothing because **CreateLink**(**child**($\mathbf{n_p}$), **child**($\mathbf{n_q}$), $\mathbf{F_{eps}}$) is invoked in the previous iteration.

Case 2: $E_{n_p n_q} \leq E_{n_q n_p}$, this case is the same as case 1 except exchanging the roles of n_p and n_q.

In our approach, the upper-level patches have a more accurate estimate of the kernel. The error in computing the maximum, the minimum and the average values of G for the upper-level patches can be reduced. Hence, we can obtain more accurate error bounds for the upper-level patches as the criteria of interaction. The links on the hierarchy can be created more accurately. In addition, we can obtain more accurate form factors for the upper-level patches. This fact enables the energy to be transferred more accurately. Hence, our approach does increase the accuracy of the hierarchical radiosity algorithm.

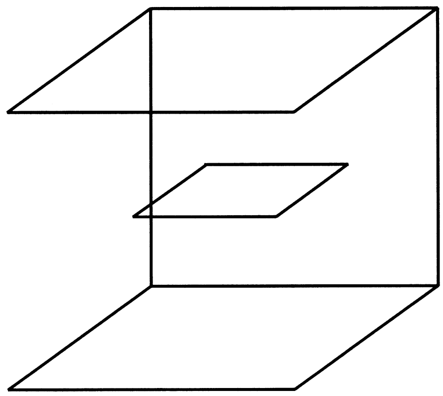

Fig. 8. A testing environment.

4. IMPLEMENTATION AND RESULTS

As discussed in Ref. [8], the lower bound on visibility between two patches is hard to find. Therefore, we set the minimum value of kernel between two patches to zero. Consequently, the error bound on the form factor from one patch to the other patch becomes the maximum form factor between the corresponding two patches. The user-specified minimum patch area A_{eps} is set to 1/256 of the area of the maximal patch in the input surfaces. This implies that there are at most five levels among complete quadtrees for a given environment. All experiments are running on the SGI workstation with an R4400/200 MHz CPU.

Figure 8 depicts a testing environment consisting of four polygons. 832 potential elements and 345,696 potential links are available. First, we use our approach to investigate the relationship between the accuracy of form-factor estimates used in the link formation and the number of links produced. The testing results are shown in Table 1. For each error tolerance, the links produced are increased when the number of samples used in each bottom-most patch is increased. Increasing the number of samples used in each bottom-most patch can reduce the error in computing the maximum and the average values of the kernel. That is, we can increase the accuracy of form-factor estimates used in the link formation. This implies that the links produced are increased when the accuracy of

Table 1. Testing results

	Links			
F_{eps}	One sample	Two samples	Three samples	Four samples
0.030	5512	27 457	28 312	29 122
0.025	5632	31 222	31 912	33 832
0.020	7762	36 652	37 027	40 402
0.015	17 572	55 012	55 912	60 832
0.010	24 202	87 922	88 777	93 907

Table 2. Comparing our approach with the top–down method

	Top–down		Our approach	
F_{eps}	Links	Refinement time (sec)	Links	Refinement time (sec)
0.030	21 532	32.25	29 122	51.02
0.025	26 287	33.17	33 832	51.07
0.020	31 627	34.58	40 402	51.19
0.015	46 552	36.74	60 832	51.36
0.010	78 487	42.07	93 907	51.76

Table 3. The experimental results

		Time (sec)		
F_{eps}	Links	Model preprocessing	Refinement	Gathering
0.030	82 287	28.4	1136.0	5.4
0.025	89 184	28.4	1137.0	5.4
0.020	101 658	28.4	1138.1	5.5
0.015	128 682	28.4	1139.8	5.5
0.010	178 215	28.4	1143.2	5.6

form-factor estimates used in link formation is increased.

Next, we compare our approach with the top–down method. In the top–down method, we take four jittered samples on each level patch. In our approach, we take four jittered samples for each bottom-most patch. Table 2 lists the experimental results. For each error tolerance, our approach creates more links on the hierarchy than the top–down method does. The reason is that, for the upper-level patches, our approach has more accurate estimates on the kernel and, thus, the error bounds obtained by our approach is generally larger than that obtained by the top–down method. This leads our approach to create more links on the hierarchy to meet the error tolerance. This can increase the accuracy of radiosity solutions. For each error tolerance, the refinement time (the time of kernel sampling and creating links) of our appraoch is larger than that of the top–down method. The reason is that our approach takes more cost on the kernel sampling than the top–down method does. Moreover, for different error tolerances, the refinement time of our appraoch is similar, while the refinement time of the top–down method is increased. This fact shows that the refinement time of our approach is dominated by the time of kernel sampling on the bottom-most patches. But the refinement time of the top–down method is dominated by the time of kernel sampling across the produced links.

Fig. 9. The results by applying our approach to a living room.

Applying our approach for a environment consisting of 246 input surfaces. The entire environment are subdivided into 2270 elements totally, with 2,575,315 potential interactions. The number of samples on the patch of each level is 4^{d-1}, where d is the level of the corresponding quadtree node. Table 3 depicts the links produced and the time distributed amongest each part of the algorithm. Figure 9 displays a resulting image of our approach.

5. CONCLUSIONS

We propose an accuracy enhancing approach for hierarchical radiosity by creating the links on the hierarchy in a bottom-up fashion. By our approach, the kernel between the upper-level patches obtains more accurate estimations. Thus, we can obtain more accurate error bounds as the criteria of interaction. In addition, the form factors for the upper-level patches can be computed more accurately. We successfully improve the accuracy of the hierarchical radiosity. Future research is aimed at trying to speed up the performance of our approach.

Acknowledgements—This work was supported partially by the National Science Council, Republic of China, under grant NSC87-2213-E-009-079.

REFERENCES

1. Goral, C. M., Torrance, K. E., Greenberg, D. P. and Battaile, B., Modeling the interaction of light between diffuse surfaces. *Computer Graphics*, 1984, **18**(3), 213–222.
2. Cohen, M. F., Greenberg, D. P., Immel, D. S. and Brock, P. J., An efficient radiosity approach for realistic image synthesis. *IEEE Computer Graphics and Applications*, 1986, **6**(3), 26–35.
3. Hanrahan, P., Salzman, D. and Aupperle, L., A rapid hierarchical radiosity algorithm. *Computer Graphics*, 1991, **25**(4), 197–206.
4. Appel, A. A., An efficient program for many-body simulation. *SIAM J. Sci. Stat. Computing*, 1985, **6**(1), 85–103.
5. Teller, S. and Hanrahan, P., Global visibility algorithms for illumination computations. In *Computer Graphics Proceedings*, Annual Conference Series, ACM SIGGRAPH, 1993, pp. 239–246.
6. Smits, B. E., Arvo, J. R. and Salesin, D. H., An importance-driven radiosity algorithm. *Computer Graphics*, 1992, **26**(4), 273–282.
7. Kok, A. J., Grouping of patches in progressive radiosity. In *Proceedings of the Fourth Eurographics Workshop on Rendering*, 1993, pp. 221–231.
8. Smits, B., Arvo, J. and Greenberg, D., A clustering algorithm for radiosity in complexity environments. In *Computer Graphics Proceedings*, Annual Conference Series, ACM SIGGRAPH, Addison-Wesley, Reading, MA, 1994, pp. 435–442.
9. Lischinski, D., Tampieri, F. and Greenberg, D. P., Combining hierarchical radiosity and discontinuity meshing. In *Computer Graphics Proceedings*, Annual Conference Series, ACM SIGGRAPH, Addison-Wesley, Reading, MA, 1993, pp. 199–208.
10. Heckbert, P. S., Discontinuity meshing for radiosity. In *Third Eurographics Workshop on Rendering*, Springer Verlag, Beijing, 1992, pp. 203–216.
11. Lischinski, D., Tampieri, F. and Greenberg, D. P., Discontinuity meshing for accurate radiosity. *IEEE Computer Graphics and Applications*, 1992, **12**(6), 25–39.
12. Sillion, F. and Drettakis, G., Feature-based control of visibility error: a multi-resolution clustering algorithm for global illumination. In *Computer Graphics Proceedings*, Annual Conference Series, ACM SIGGRAPH, 1995, pp. 145–152.
13. Chang, C. C. and Shih, Z. C., Tighter error bounds on form factors for hierarchical radiosity. In *Proceedings of the Fourth Pacific Conference on Computer Graphics and Applications Pacific Graphics '96*, National Chiao Tung University, Hsinchu, Taiwan). 1996, pp. 147–158.
14. Gibson, S. and Hubbold, R. J., Efficient hierarchical refinement and clustering for radiosity in complex environments. *Computer Graphics Forum*, 1996, **15**(5), 297–310.
15. Sillion, F. X., Clustering and volume scattering for hierarchical radiosity calculations. In *Proceedings of Fifth Eurographics Workshop on Rendering*, Springer Verlag, Beijing, 1994, pp. 105–117.
16. Sillion, F. X., A unified hierarchical algorithm for global illumination with scattering volumes and object clusters. *IEEE Transactions on Visualization and Computer Graphics*, 1995, **1**(3), 240–254.
17. Aupperle, L. and Hanrahan, P., A hierarchical illumination algorithm for surfaces with glossy reflection. In *Computer Graphics Proceedings*, Annual Conference Series, ACM SIGGRAPH, Addison-Wesley, Reading, MA, 1993, pp. 155–162.
18. Christensen, P. Lischinski, D., Stollnitz, E. and Salesin, D., Clustering for glossy global illumination. Technical Report UW-CSE-95-01-07, University of Washington, Seattle, 1995.
19. Gortler, S. J., Schroder, P., Cohen, M. F. and Hanrahan, P., Wavelet radiosity. In *Computer Graphics Proceedings*, Annual Conference Series, ACM SIGGRAPH, Addison-Wesley, Reading, MA, 1993, pp. 221–230.
20. Stollnitz, E., Derose, T. and Salesin, D., Wavelets for computer graphics: a primer, part 1. *IEEE Computer Graphics and Applications*, 1995, **15**(3), 76–84.
21. Stollnitz, E., Derose, T. and Salesin, D., Wavelets for computer graphics: a primer, part 2. *IEEE Computer Graphics and Applications*, 1995, **15**(4), 75–85.
22. Lischinski, D., Smits, B. and Greenberg, D. P., Bounds and error estimates for radiosity. In *Computer Graphics Proceedings*, Annual Conference Series, ACM SIGGRAPH, Addison-Wesley, Reading, MA, 1994, pp. 67–74.