Hierarchical Link Structure for Dynamic Scene Radiosity

Chen-Chin Feng
knight@cs.nthu.edu.tw

Su-Yuan Chang
gunman@cs.nthu.edu.tw

Cheng-Tai Shen
p33@cs.nthu.edu.tw

Shi-Nine Yang
snyang@cs.nthu.edu.tw

Department of Computer Science, National Tsing Hua University,
Hsinchu, Taiwan, R.O.C.

Abstract

Radiosity method is a widely used global illumination model for image synthesis. It computes all energy interactions among diffuse elements in a virtual envirunment. One of the major drawbacks is its time consuming computation. Existing radiosity algorithms for static scene is difficult to be applicable to dynamic environments due to excess computation of visibility and energy transfer. In this paper we proposed a hierarchical link indexing scheme and associated hierarchical visibility testing algorithm to speedup the link update computations in the dynamic environments. The proposed indexing scheme not only efficiently updates the links due to geometry change, but also can handle multiple dynamic objects easily. Several empirical tests are given to demonstrate the effectiveness and the efficiency of our improved algorithm.

CR Categories and Subject Descriptors: I.3.7 [Computer Graphics]: Three-Dimensional Graphics and Realism - Radiosity; I.3.6 [Computer Graphics] Methodology and Techniques - Graphics data structures and data types, Interaction techniques.

Additional Keywords: Visibility, Dynamic Scene.

1 Introduction

To create realistic images for virtual environments is important to applications such as industrial and architectural designs, film and video game productions. In 1980s, two major global illumination techniques have been developed for synthesizing virtual environments, namely, the ray tracing method and the radiosity method.

Since the ray tracing method is developed basically for specular reflection, it is difficult to model diffuse reflections among objects in the environment. Besides, the ray tracing method is view dependent that the entire ray tracing process has to go over again whenever the viewpoint is changed. Therefore it is not appropriate to applications involving dynamical change of viewpoint, such as walkthrough in a virtual world. These problems can be remedied by applying the radiosity method [11], which was originally developed for computing radiant interchange between surfaces. According to the energy conservation within a closed space, this method simulates the interaction of diffuse light among objects using a more precise physical model. Since this model assumes that all surfaces in the scene are ideal diffuse, the lighting effects of synthesized world are view-independent. Thus the scene rendered by radiosity method is quite suitable for virtual reality walkthrough.

One of the major limitation of the radiosity method is that it suffers from heavy computation. Many improved algorithms have been proposed and one of the most notable improvements is the hierarchical radiosity [12] which reduces the computation complexity from $O(s^2)$ to $O(s+p)$ without affecting the precision of form factor estimation, where s is the number of subdivided patches in the environment, and p is the number of links for energy transport. However, for a scene with median complexity, the computation time of hierarchical radiosity method is still too long to be applicable to dynamic environments. This is because radiosity computation involves all energy interactions among objects and any change of objects' attribute or geometry may cause a great amount of recomputation.

To resolve the excessive recomputation of dynamic scene radiosity, several researchers proposed link classification methods to avoid the recomputation of unchanged links after dynamic objects being moved [9] [13] [17]. However, most existing improvements dealt with single dynamic object. For example, in [17], no specific link structure is given, and the extension to multiple dynamic objects is mentioned in her future works. Besides, the link structure proposed in [6] can cope with multiple objects, it suffers from switching between objects and link structure, since its link hierarchy is embedded in the in the object hierarchy.

In this paper, a new hierarchical link indexing structure is proposed, and based on this indexing structure, a visibility recomputation algorithm is also devised to accelerate the radiosity computation. Futhermore, the proposed method is suitable for scenes with multiple dynamic objects.

The remainder of this paper is organized as follows: In section 2, we overview the previous work of dynamic scene radiosity. The link updating process for dynamic scene radiosity is given in section 3. Our hierarchical link indexing structure is described in section 4. In section 5, the hierarchical visibility testing algorithm is introduced. Besides, energy updating strategy is described in section 6. The empirical results are given in section 7. Finally, conclusions and future works are given in section 8.

2 Previous Works

The radiosity computation spends too much time to be used in dynamic environments, because it involves all energy interactions among objects and any change of objects' attribute or geometry may cause a great amount of recomputation. To reduce recomputation of visibility and energy transfer is the key to make radiosity applicable to dynamic environments. Recently, there have been numbers of researches devoted to the improvement of dynamic scene radiosity computation. New data structure and algorithms are sought to reduce the amount of recomputation due to the change of the environment.

In 1986, Baum first extended the radiosity method to dynamic scenes [1]. However, he required that the path of moving object and the viewpoint are given in advance. That is, it is neither interactive nor view-independent. George [10] and Chen [2] independently addressed the notion of energy redistribution in dynamic environments using the progressive refinement method [3]. For hierarchical radiosity of dynamic environment, Forsyth proposed three atomic link editing operations to update hierarchy when an object is moved [9]. Shaw introduced the idea of mesh folding to speed the construction of new linear system after a change in geometry [17]. To reduce the loading of link recomputation, all links related to mesh folding are established and record in advance. This method

can effectively improve the computation time for a single dynamic object. However, it mainly addressed the case when there is only one single moving object. No further extension is reported for multiple dynamic objects.

Since visibility computation consumes an essential portion of radiosity computation, Orti *et al.*[16] and Durand *et al.* [7] introduced the notion of visibility complex to solve the visibility problem in dynamic environments. But their extension to 3D space is still under development. Using a technique similar to shaft volume and recorded the volume information in links, Drettakis *et al.* proposed an efficient method to update links due to the change of visibility [6]. In [8], a spatial visibility structure named visibility skeleton is introduced. This is also a strategy to trade massive memory space for rendering time.

3 Link Update Process in a Dynamic Scene

A simple block diagram for dynamic scene radiosity computation is shown in Figure 1. According to this diagram, a naive approach to compute the dynamic scene radiosity is to recompute the entire energy interaction links after changes of dynamic objects. This is obviously inefficient because those unaffected links do not need to be recomputed. Hence there are two aspects in designing efficient dynamic radiosity algorithm, one is to avoid recomputing unaffected links, and the other is to update affected links efficiently.

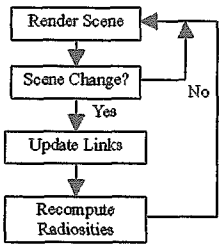

Figure 1: The program architecture of dynamic scene radiosity

In order to meet these two goals, the behavior of interaction links have to be analyzed. For convenience, we divide objects in a scene into dynamic objects - those subjected to change, and environment objects - those cannot be changed. According to [17], all interaction links can be classified into three types: *links between dynamic objects* (O-O links), *links between environment and dynamic objects* (E-O links) and *links between environment objects* (E-E links) (Figure 2).

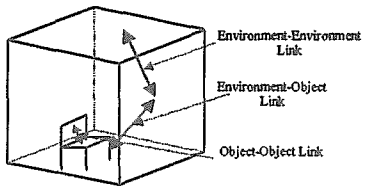

Figure 2: E-E Link, E-O Link and O-O Link.

In general, when an dynamic object changes its geometry, for example, position or orientation, two types of links, namely, E-E links and E-O links, are apt to change. But when an object only changes its lighting attribute, the changing links will mostly be E-O links, which is relatively easy to identify. Thus, in this study we confine ourselves to the geometry change in a dynamic environment.

Now let us consider the case when there is one moving object O. Suppose A_{old} (A_{new}), S_{old} (S_{new}) and G_{old} (G_{new}) denote the set of active links, shadow links and ghost links respectively before (after) geometry change. The whole link update process can be separated into three stages, that is, shadow detection stage, un-refinement stage and refinement stage.

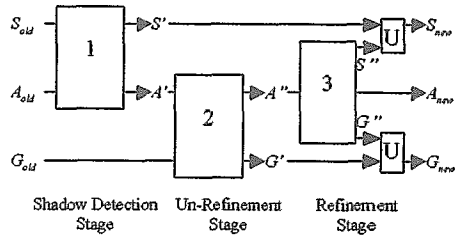

Figure 3: The link update process

In the shadow detection stage, the A_{old} and S_{old} are updated according to the visibility of new scene. Both E-E links and E-O links are most likely affected in this stage. After finding new active links, we un-refine the links in the second stage due to distance change. In the second stage, we un-refined links found in the first stage. This stage deals mostly with E-O links. The third stage refines links according to the new visibility and distance, it involves those E-E links that is activated from shadow links and some E-O links left out in the second stage. We depict the process of link update in Figure 3. The corresponding algorithm is given as follows,

```
// Shadow Detection Stage
for each link L in (Sold + Aold)
    if (occlude(L, Oold) or occlude(L, Onew))
        add(A', L)
    else add(S', L)

// Patch Un-Refinement Stage
G" = Gold
for each links L in A'
    if B(L)*F(L) < eps1
        begin
            drop(L)
            drop(brother(L))
            add(A", L->parent)
            G" = G" - L->parent
        end
    else add(A", L)

// Patch Refinement Stage
for each links L in A"
    if B(L)*F(L) > eps2
        begin
            refine(L)
            add(G", L)
            for each Lc in L->children
                if visible(Lc)
                    add(Anew, Lc)
                else add(Snew, Lc)
        end
Gnew = G" + G'
Snew = S" + S'
```

A further observation shows that almost all E-O links are subject to change in a dynamic environment, and most O-O links are not changed.

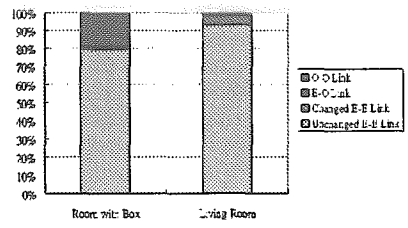

Figure 4: Link Ratio

In general, the number of dynamic objects is much less than the number of static objects (environment). Therefore, the set of all links comprises mostly E-E links. Figure 4 shows the ratio of different types of links or our testing scenes.

It is clear that the percentage of different type of links is scene dependent. However, our testing examples show the portion of E-E links is about 80% $\sim$ 95%, and it goes up when the scene becomes more complicate. Besides, the percentage of O-O links is less than 1% of the entired links. The rest of links are E-O links.

According to previous observation, if one wants to improve the computing time of dynamic scene radiosity, then the keypoint is to speedup the link reclassification and visibility recomputation. We will propose improved link updating and visibility recomputation algorithms in the next section.

4 Hierarchical Link Indexing Scheme

According to our previous analysis, to computes E-E link is the most time-consuming part during the visibility re-computation. For scenes with multiple dynamic objects, when a different dynamic object is selected, it requires a lot of efforts to identify E-E links and to re-compute its visibility. To accelerate the identification and visibility computation, an efficiently data structure is needed for finding out the specific links easily. Furthermore, it should also be helpful in visibility testing process during the dynamic scene radiosity re-computation. The new link indexing scheme and the visibility re-computation algorithm based on the indexing scheme are introduced in the following.

In order to minimize the access time and memory size, we devised the following Hierarchical Link Indexing (HLI) structure. For every patch pairs, a unique link index is given. The hierarchical relation of patches is implicitly encoded in the index. Our indexing scheme is based on patch subdivision with quadtree encoding. For example, a subdivision sequence and the links are depicted in Fig.5, then the corresponding quadcode indexing is given in Fig.6.

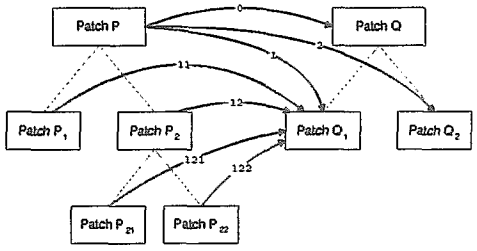

Figure 5: Link Indexing of Fig.5

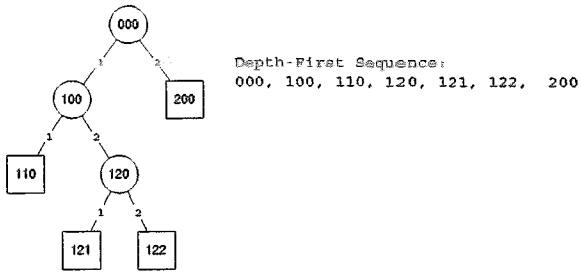

Figure 6: Depth First Sequence

Name	Description
Src	Emit Patch/Cluster
Des	Gathering Patch/Cluster
Ghost	1 for ghost link, 0 for active/shadow link
Vis	Visibility, 0 for shadow link
FF	form factor value of this link
B	Energy Transfer on this link
Next	Pointer to the next record of the same subdivide level

Table 1: Link Data Structure

This indexing scheme has several merits. For example, all links are inherently sorted according to depth-first sequence. Therefore, the search of a specific link requires only $O(logn)$ times. Moreover, the index structure is built along with the construction of scene hierarchy according to the hierarchical radiosity algorithm. The explicit data structure of a link is given in Table 1.

In the next section, we will introduce our link updating algorithm based on this link structure.

5 Link Update Algorithm

It is known that the visibility test is the most time consuming process in the radiosity computation. In this section we will intruduce the Hierarchical Visibility Testing (HVT) strategy to speedup the visibility test process.

According to the previous observation, the visibility of E-O link must be recomputed. As for E-E links, although they comprise 90% of links, only a small portion of links require visibility recomputation. The basic idea of HVT is to prune those links with insignificant visibility changes.

Instead of computing all interactions among terminal elements in the object hierarchy, Hanrahan proposed an approximation, called the hierarchical radiosity method, by computing higher level interaction among object hierarchy [12]. This new approach greatly reduces the computing time needed in radiosity implementation. In order to take advantage of hierarchical radiosity, we introduce the Hierarchical Visibility Testing method which can be incorporate into hierarchical radiosity algorithm naturally.

To illustrate our basic idea of HVT, consider the case in Figure 7 (a). If dynamic object does not affect the visibility of P-Q link, then all decent of P-Q link can be ignored. For the case in Figure 7 (b), if the dynamic object fully occludes the P-Q link, then the visibility of all decent of P-Q links are set to 0. If the dynamic object partially occludes the P-Q link, then the visibility test steps down to their children links. (see Fig. 8)

The pseudocode of HVT algorithm is given in the following.

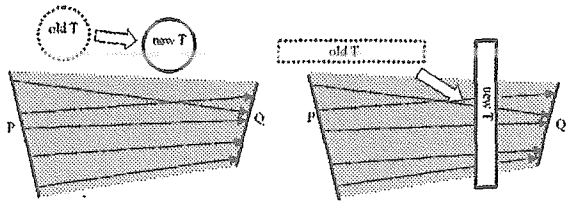

Figure 7: (a) Full Visible and (b) Full Occlude

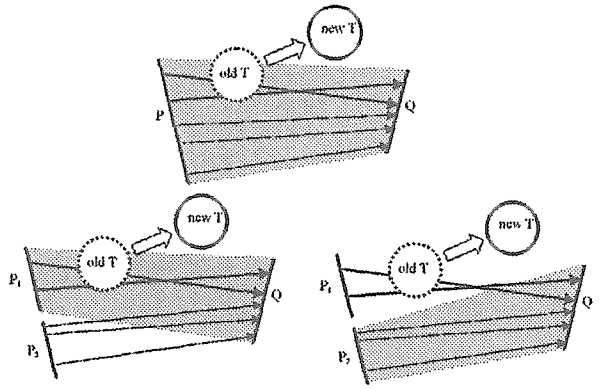

Figure 8: Partial Occlude

```
HVTonEELink (k: Link, obj: Object)
begin
    if (ShaftVolumeVisibility(k, obj) = 1.0)
                                  // if full-visible
        return
    if (ShaftVolumeVisibility(k, obj) = 0.0)
                                  // if full-occlude
        for each link j belong to k
            UpdateLink(j)
        return
    if (k is not a ghost-link)
        UpdateLink(k)
    for each child-link j belong to k
        HVTonEELink(j, obj)
end
```

6 Energy Updating Algorithm

In section 5, we have introduced HVT algorithm to speedup the link update process. However, the entire radiosity computation involves further energy exchange process. In this section we will introduce the Progressive Energy Update (PEU) algorithm to improve the performance of subsequence energy exchange.

6.1 Progressive Energy Update (PEU) Algorithm

For energy exchange, a straightforward approach is to sweep over the entire set of active links after link update. However, there may be some links whose energy exchange is insignificant. Therefore we can use progressive refinement strategy to future speedup the radiosity computation.

Recursively we choose the patch with the greatest unshoot energy, and use the HVT algorithm to find and compute the visibility of correspond E-E links and E-O links. Then it shoots the energy difference along these changed links.

In fact, in this process we only shoot energy along links with significant energy difference. To estimate the significancy, consider the case when the move of object results in an increment of visibility. It is clear that maximum energy difference is $(1-v)BF$ where v is the old visibility. If its value is less than a given threshold, then we do not have to bother about visibility and energy update. Similarly, if a link is occluded due to the move of objects, then the significancy of the link can be estimated by its maximum energy difference ($(v-0)BF$).

According to our experiments, this filtering process results in a great improvement of the computation times without significant degrading the quality of the image (see Table 3, 5 and Figure 9, 10).

6.2 Progressive Energy Updating Algorithm

The algorithm using both HVT and PEU is given as follows,

```
HVTonEELinkandShoot
  (k: Link, B: energy, obj: Object)
begin
    if (ShaftVolumeVisibility(k, obj) = 1.0)
                                  // if full-visible
        return
    if (ShaftVolumeVisibility(k, obj) = 0.0)
                                  // if full-occlude
        for each link j belong to k
            if Vis(k)*B(k)*F(j) > Beps
            UpdateAndShootEnergy(j)
        return
    if (k is not a ghost-link)
        UpdateAndShootEnergy(k)
    for each child-link j belong to k
        if dynamic object moved away
                        // dynamic object moved away
            if ((1-Vis(j))*B(j)*F(j) > Beps)
                HVTonEELinkandShoot(j, obj)
        else
                        // dynamic object moved in
            if (Vis(j)*B(j)*F(j) > Beps)
                HVTonEELinkandShoot(j, obj)
end

PEU (obj: Object)
begin
    find shooting patch p with biggest radiosity
        pushpull(p)
        for (all gathering patch q of obj)
                                        // E-O link
            Re-calculate visibility Vis'
              and New Form Factor F'
            k = Link(p, q)
            for all active-link j of k
            if (Vis'(j)-Vis(j))*B(p)*(F'(j)-F(j))
              > Beps
                ShootEnergyDifference(j)
        else
                                        // E-E link
            k = Link(p, q)
            HVTonEELinkandShoot(k, obj)
end
```

7 Experiment Results

Several empirical test have been implemented in a Pentium-II 300, 512MB Ram personal computer running Windows NT 4.0 Worksta-

tion. The testing scene box is generated by placing a box in a room. The next testing scene is a living room. The radiosity computimg times by using conventional hierarchical radiosity algorithm, hierarchical radiosity with HVT algorithm and HVT + PEU algorithm are summerized in Table.2 to Table.5. Figure 9 and Figure 10 show the picture our testing scenes.

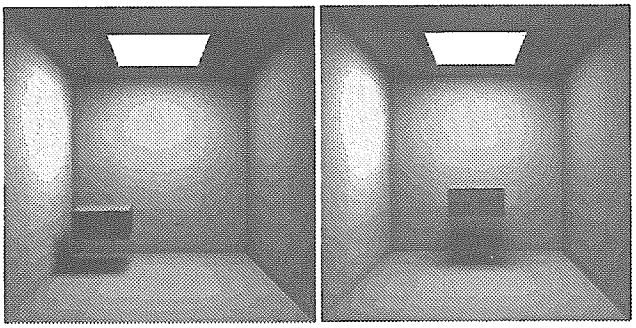

Figure 9: Scene Box Before and After

Patch No.	SubPatch No.	Link No.	E-E Link	E-O Link	O-O Link
13	2625	169593	131244	38349	0
			77.39%	22.61%	0

Table 2: Dynamic Scene (box) statistic datas

	Static Scene	Direct Update	HVT	HVT + BEU
Time (sec)				
Create Link	26.778	23.292	1.772	1.231
FF Computation	4.115	4.223	0.911	0.530
Visibility Test	21.16	18.99	0.861	0.701
Energy Exchange	0.41	0.369	0.41	0.22
HVT			2.163	1.912
Total Time	28.05	25.144	2.944	1.457
Link Update No.	169593	169593	13568	3528
FF Compute No.	678372	588679	134544	78274
Visibility Test No.	8112010	7171017	236290	192380

Table 3: Dynamic Scene (box)

8 Conclusion and Future Works

In this study, we analyze the behavior of link change of dynamic scene radiosity and propose an algorithm to reduce link recomputation. A hierarchical spatial subdivision scheme, together with the correspond object encoding, is proposed to speedup the visibility computation. Several experiments are given to demonstrate the efficiency of the proposed algorithm. For future study, we will incorporate clustering techniques and perceptual measure to improve our performance.

This work is supported by National Science Council (NSC)under the grant NSC-87-2213-E-007-032.

Figure 10: Scene Living Room Before and After

Patch No.	SubPatch No.	Link No.	E-E Link	E-O Link	O-O Link
445	8149	404225	392078	12103	44
			96.99%	2.99%	0.01%

Table 4: Dynamic Scene (Hall) statistic datas

References

[1] Baum, D. R. *et al.*, "The Back-Buffer Algorithm: An Extension of the Radiosity Method to Dynamic Environments", *The Visual Computer*, 2(5) pp. 298–308, Sept. 1986.

[2] Chen, S. E., "Incremental Radiosity: An Extension of Progressive Radiosity to an Interactive Image Synthesis System", *Computer Graphics (SIGGRAPH '90 Proceedings)*, 24, pp. 135–144, Aug. 1990.

[3] Cohen, M. *et al.*, "A Progressive Refinement Approach to Fast Radiosity Image Generation", *Computer Graphics (SIGGRAPH '88 Proceedings)*, 22(4), pp.75–84, Aug. 1988.

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

[5] Cohen, D., "Voxel Traversal along a 3D Line", *Graphics Gems IV*, Academic Press, 1994.

[6] Drettakis G. and Sillion F., "Interactive Update Of Global Illumination Using A Line-Space Hierarchy", *Computer Graphics (SIGGRAPH '97 Proceedings)*, pp. 57–64, Aug. 1997.

[7] Durand, F., Drettakis, G and Puech, C. "The 3D Visibility Complex: a new approach to the problem s of accurate visibility", *Eurographics Workshop on Rendering*, 1996.

	Static Scene	Direct Update	HVT	HVT + BEU
Create Link (sec)	1364.131	1274.23	10.925	4.386
FF Computation	8.051	7.11	0.260	0
Visibility Test	1352.324	1266.72	10.665	4.386
Energy Exchnage	3.324	2.99	3.334	2.12
HVT Testing			82.007	7.751
Total Rendering	1370.03	1282.35	87.035	9.877
Link Update No.	404225	404225	9578	5560
FF Compute No.	1616900	1505980	45648	0
Visibility Test No.	16777200	16777200	4013250	1650456

Table 5: Dynamic Scene (Hall)

[8] Durand, F., Drettakis, G. and Puech, C., "The Visibility Skeleton: A Powerful And Efficient Multi-Purpose Global Visibility Tool", *Computer Graphics (SIGGRAPH '97 Proceedings)*, pp.89-100, Aug. 1997.

[9] Forsyth, D., Yang, C. and Teo, K., "Efficient Radiosity in Dynamic Environments", *5th Eurograph Workshop on Rendering*, 1994.

[10] George, D., Sillion, F., and Greenberg, D.P., "Radiosity redistribution for dynamic environments", *IEEE Computer Graphics and Applications*, 10(4), pp. 26–35, July 1990.

[11] Goral, C., Torrance, K.E., and Greenberg, D. P., "Modeling the Interaction of Light between Diffuse Surfaces", *Computer Graphics (SIGGRAPH '84 Proceesings)*, 18(3), pp. 212–222, 1984.

[12] Hanrahan, P., Salzman, D., and Aupperle, L., "A Rapid Hierarchical Radiosity Algorithm", *Computer Graphics (SIGGRAPH '91 Proceedings)*, 25(4), pp. 197–206, July 1991.

[13] Muller, S. and Schoffel, F., "Fast radiosity repropagation for interactive virtual environments using a shadow-form-factor-list", *5th Eurograph workshop on Rendering*, 1994.

[14] Ng, A., "A comparison of four visibility acceleration techniques for radiosity", *The Visual Computer*, 12 (3), pp. 307–316, 1996.

[15] Nishita, T. and Nakamae, E., "Continuous Tone Representation of Three-dimensional Objects Taking Account of Shadows and Interreflection" *Computer Graphics (SIGGRAPH '85 Proceedings)*, 19(3), pp. 23–30, 1985.

[16] Orti R., Riviere S., Durand, F. and Puech C., "Radiosity for dynamic scenes in flatland with the visibility complex", *Computer Graphics Forum*, 15(3), pp. C237–C248, 1996.

[17] Shaw, E., "Hierarchical Radiosity for Dynamic Environments ", *Computer Graphics Forum*, 16(2), 1997.