Irregular, Unknown Light Sources in Dynamic Global Illumination

Sharat Chandran
Computer Science and Engg. Dept.
IIT, Powai
Mumbai, INDIA 400076
sharat@iitb.ac.in

Mayur P. Srivastava
Computer Science and Engg. Dept.
IT BHU
Varanasi, INDIA 221005
mayur_prakash@rediffmail.com

Abstract

The goal in global illumination solutions for dynamic environments is to update a scene based on past scenes. Current state of the art solutions are either not applicable, or unduly complex, when there are large changes in the illumination of unbounded number of objects. Such changes may be caused by the appearance of unexpected (at modeling time), irregular light sources.

We define a subset of dynamic environments in which new light sources may be user introduced, and implement solutions that complement existing schemes.

1. Introduction

View-independent global illumination techniques are invaluable for a gamut of applications in engineering, design, education, advertisement, and entertainment. The importance of effects such as soft shadows and caustics are underscored using these schemes. At the same time, users have increasingly been demanding *interactive* methods to deal with *dynamic* environments.

The general problem of adding, removing, and changing objects in scenes might at first seem to flummox traditional radiosity methods since it violates the basic principle. Despite this, several early researchers [1, 7] show how to compute incremental solutions by shooting corrective (possibly negative) energy to the scene affected by the environmental changes. Older solutions could not satisfy the interactive demands. Recently there has been renewed interest in both technology driven hardware solutions [9], and knowledge-based software solutions [4] using faster wavelet radiosity techniques in an attempt to satisfy these user demands. There are two cases when these solutions are satisfactory.

For instance, there is an important subclass of dynamic problems where the geometry of the scene does not change, but only the lighting *predictably* changes. A very common situation is when users turn on (or off) switches in room. If the sudden change in lighting is caused by predictable light sources, it is possible to anticipate the shape of light source changes, and preprocess the scenes to take care of large lighting changes.

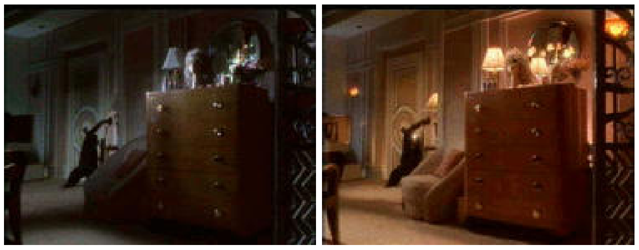
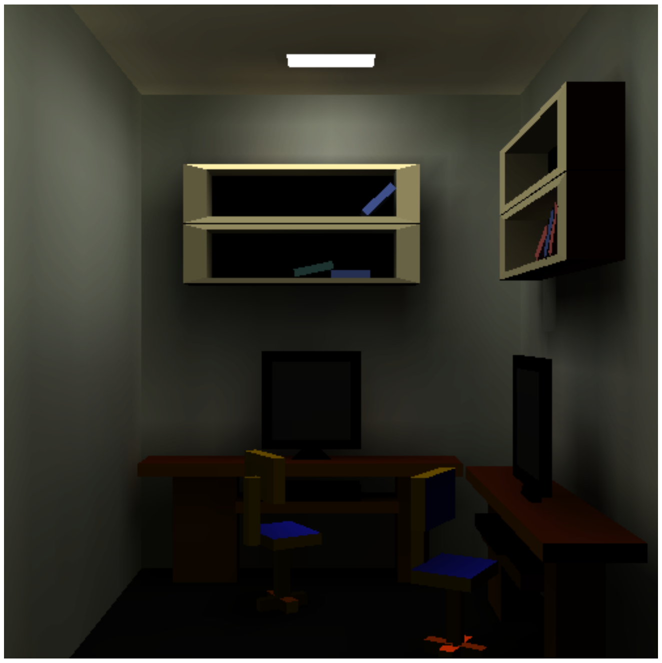


Figure 1: Two snapshots from the movie “Dunston Checks In.” Sudden changes in lighting conditions are common, and can be handled by existing incremental radiosity techniques if the geometry of the light sources are known in advance.

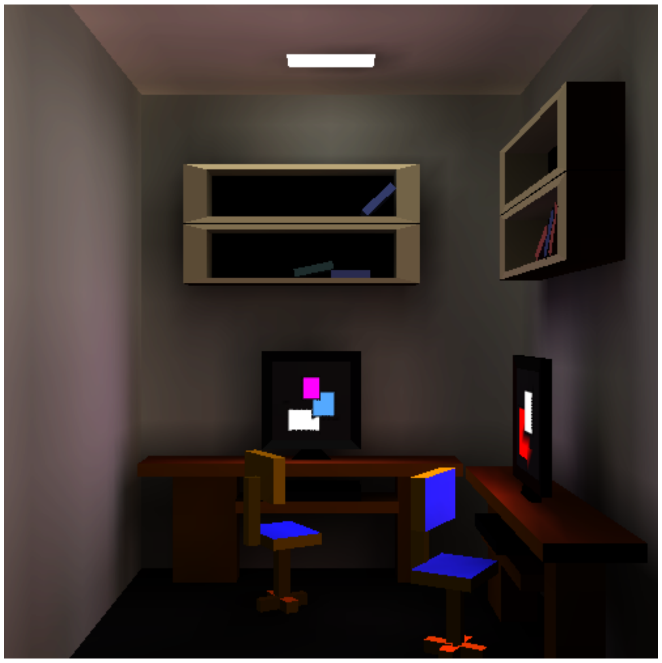
In the second case, only localized marginal lighting changes happen such as in the creation of synthetic movies where there is not much change between frames. Unfortunately, as mentioned in [10], current techniques do not work when the lighting changes are *not* localized.

However, there are a large class of situations where non-localized *abrupt* changes are useful to model, or when the lighting change is unexpected. Here the *shape* and amount of unexpected lighting changes preclude any preprocessing. Examples of these are partial pulling of window blinds, closing of curtains, appearance of daylight in rooms due to the motion of parked cars, bathing of interiors as the full moon appears past overcast clouds, and effects created in operas and theaters. In such situations, recomputing the lighting from scratch is a better [10] choice since the illumination of unbounded number of surfaces change dramatically.

In this work, we leverage upon the fact that the sophisticated methods of dynamic radiosity are either not applicable for reasons cited, or way too complex for rendering unexpected irregular light sources. The key situation modeled in our work is that *users may specify the existence of a light source of any shape* at run time. Here we provide a simple fast solution based on hierarchical radiosity. Figure 2 shows



(a) An office with two flat monitors, presumably with screen savers on. (Results are best viewed in color.)



(b) The appearance changes because mouse motion causes the windows to break out of the screen saver mode. Note that the position of these new 'light sources' is not known apriori.

Figure 2: Two different states of an office. The solution on the right is incrementally computed based on the illumination in the first by Algorithm AR (Section 3.1).

an example.

The rest of the paper is as follows. After a discussion of the previous work in Section 2, we give details of our method in Section 3. In Section 4 we give the results of our approach. We conclude with a discussion of the advantages and limitations of our work in the final section.

2. Previous Work

The simplest situation involving lighting changes is when the application calls for many lighting changes in an otherwise static environment. Since the radiosity equation is linear, this solution can be obtained by anticipating the changes, and preprocessing a solution with unit light sources. In [3], lighting with well-defined finite geometries is preprocessed for design and simulation of opera lighting effects. This method is not applicable if the light sources are not defined at modeling time.

In [1] scenes involving geometry changes are rendered by shooting negative energy in progressive refinement radiosity. This solution may be used for novel light sources, by postulating a new patch of the defined geometry. It forms a basis for one of the three solutions we present in this work. Unfortunately the solution, which predates hierarchical radiosity (HR) is not very efficient since it essentially calls for a recomputation.

The algorithm in [6] partly resembles this effort in that HR is used as a basis, and objects are assumed to be neither created nor destroyed. Links are either "broken," promoted, or demoted based on the motion of objects. If we know a priori which objects move, then the update of these links is possible and provide an incremental solution.

In [4] a sophisticated HR based solution (with clustering) is proposed. The solution replaces links in HR with shafts so that when an object moves, it is possible to identify which portions of the scene needs to be updated. More recent work on shafts include [11, 2] that reduce the high memory requirements intrinsic to the method. However, all these solutions work well only when illumination changes are local. Recently an object space approach of caching prior rendering has been proposed in [12] for situations that allow for large movement of light sources. A run-time user defined light source would however require the use of a new patch which did not exist in earlier scenes, thereby defeating the cache.

3. The Algorithm

For simplicity, the methods in this section assume that the source of light appear on one of the preexisting patches. As mentioned earlier, this enables simpler, interactive algorithms, and at the same time, provides solutions for a reasonable subset of situations in dynamic environments in global illumination. Our solutions work in the context of the hierarchical radiosity (HR) algorithm using "BF refinement" [8]. Since patches corespond to roots of quadtrees, we term the patch corresponding to the (undetermined till user specified) light source as the source quadtree SQ. As is well known, the algorithm refines user patches into sub patches and sets up a linear number of links.

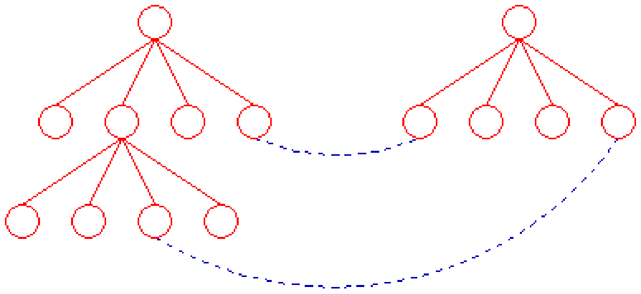


Figure 3: The basic HR algorithm refines patches into sub patches and sets up links between nodes at various levels.

There are a couple of straightforward ways of dealing with a new light source. The naive solution would be to introduce a new patch modeling the light source, and rerun the hierarchical solution, thereby introducing new links. We implemented this solution to contrast with two other solution since this algorithm provides us with a measure of ground truth.

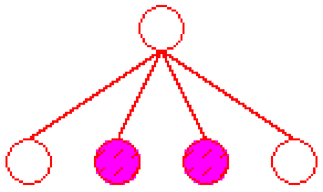
3.1 Adaptive refine

The intuition behind the second solution, termed AR, becomes clear if the light source matches exactly two nodes as shown in Figure 4(a).

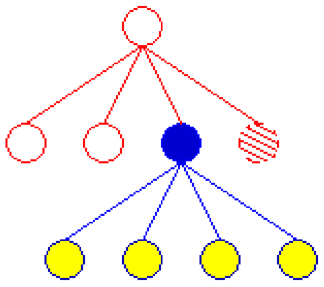
In the general case, the light source will intersect with some of the preexisting nodes of the HR algorithm. Since we use BF refinement, we need to further refine nodes that would not have been expanded in the original algorithm as shown in Fig 4(b).

The pseudocode (Figure 5) below determines precise those nodes that need to be expanded, and an example is shown.

Application of `Intersection()` results in, general speaking, the source quadtree Q becoming deeper. However, patches other than the one on which the light source is presumed to lie on are not affected in the sense that the quadtree corresponding to these do not change. This is the reason why the overall algorithm runs fast.



(a) A contrived light source maps to exactly two nodes (shown filled) in the HR quadtree of height two. Computing the new illumination levels is straightforward.

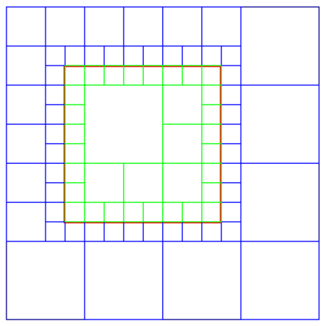


(b) The original HR algorithm terminates at height 2. Two shaded nodes intersect the light source but only one node is completely contained in the light source. Therefore the other node is further expanded.

Figure 4: The intuition behind our algorithms.

```
Algorithm Intersection(LightSource R, Quadtree Q) {
    if Disjoint(R,Q) then
        return NULL;
    if Contained(R,Q) then /* Q is contained in R*/
        return Q;
    else for each $child_i$ of Q do
        Intersection (R, Q.$child_i$);
}
```

Figure 5: Determination of nodes that need to be expanded is obtained by a fast intersection algorithm.



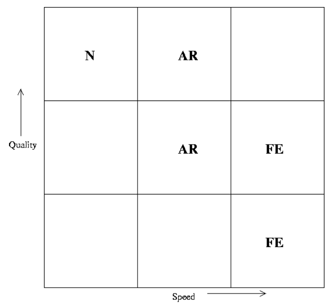


Figure 6: Qualitative comparison of the three algorithms implemented in this work. ‘N” denotes the naive algorithm, ‘AR’ refers to the adaptive algorithm of Section 3.1, and ’FE’ denotes the algorithm of Section 3.2.

3.2 Fractional Emissivity

Although adaptive refine is fast, it can take a large amount of time if the initial scene is dense, and the input light source is highly irregular and not aligned with the patch boundaries. An even faster algorithm is possible if one accepts a coarser approximation of the light source. The idea here is to mark those portions of the HR nodes that get affected by the light source as in adaptive refine (AR). The departure with AR arises when AR decided to further refine a node n that the HR algorithm does not refine. At this point, in the third algorithm, fractional emissivity (FE) is assigned to n based on the percentage of overlap between the light source and n. This overlap is computed using an algorithm such as Hodgman-Sutherland [5].

The three algorithms are qualitatively compared in Figure 6. In general, we found that FE runs slightly faster than AR and is therefore more suitable in highly interactive situations.

4. Results

We first test our method on the Cornell room with the usual light source on the ceiling as shown in Fig. 7. With the top view in mind, let us refer to the (blue) wall closest to the taller block as the front (northern) wall.

The room is now lit with a quadrilateral light source (possibly through an open window on the red wall) on the front wall. The results of rendering the image through the three different methods are shown in Figure 8. Note the illumination changes (as compared to Figure 7) in the north-east portion of the room, and also the changes in the shadow in the south-west portion. Fig. 8(a) shows the image generated by the naive method which took 126.28 seconds, whereas the AR approach, Fig. 8(b), takes only 3.08 seconds. The resulting quality is comparable. Fig. 8(c) shows the image rendered by the FE approach which took 0.82 seconds. However, the quality of the image is lower than the other two approaches. All timings quoted are on a Pentium III workstation with 256MB of RAM running Linux.

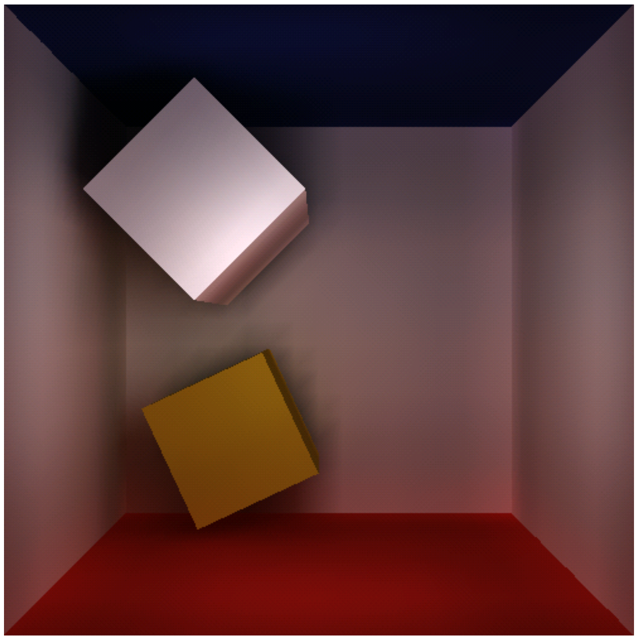


Figure 7: A pigeon’s view of the Cornell room. Only portions of the red wall appear in this view.

To see the flexibility of our system, we now add another light source on the right, eastern wall, and then “turn off” the first light source. The resulting pictures (Figure 9 and Figure 10) using the three methods are shown first in Fig. 9(a), Fig. 9(b), and Fig. 9(c), and later in Fig. 10(a), Fig. 10(b) and Fig. 10(c). These results are best seen in color.

It is instructive to see how the refine process works. For this we include (Figure 11) a wireframe version of Figure 7 and Figure 9(b). The incremental effect is clear. Finally, the incremental solution of Figure 2 took about 3.34 seconds.

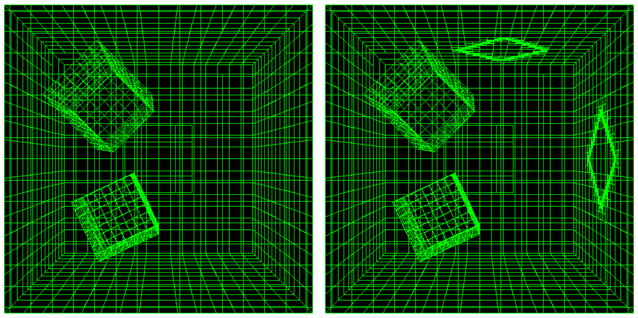
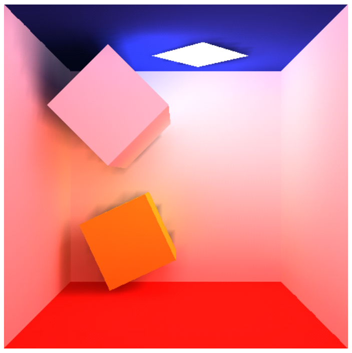
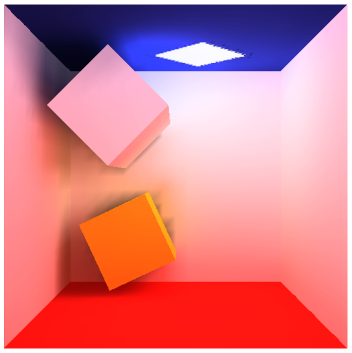


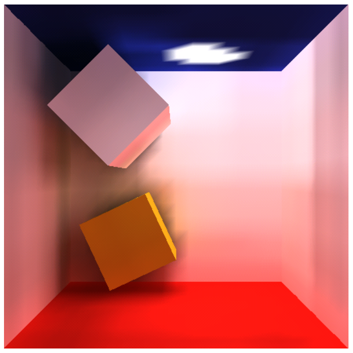
Figure 11: A wireframe is shown to visualize how adaptive refine (appearing on the right) further refines HR.



(a) The naive method (126 seconds)

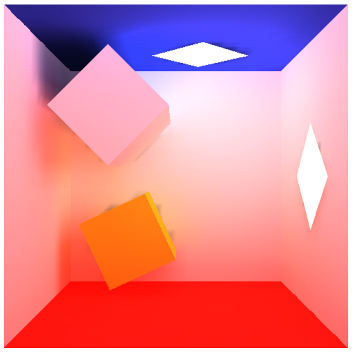


(b) Rendering using AR (3 seconds)

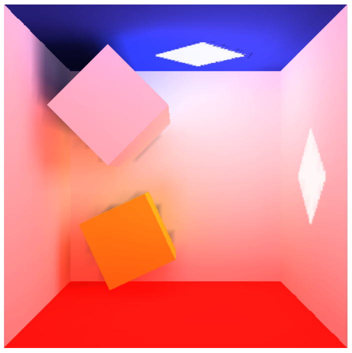


(c) Rendering using FE (0.8 seconds)

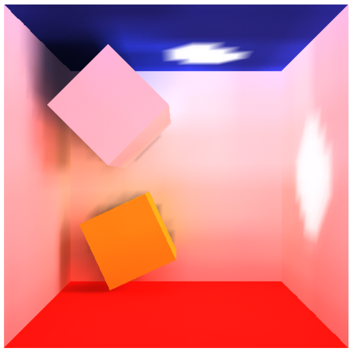
Figure 8: Light filters through the southern wall of Figure 7 onto the blue wall. The resulting scene is rendered incrementally in the two methods on the right.



(a) The naive method (172 seconds)

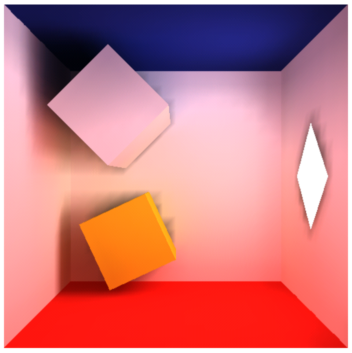


(b) Rendering using AR (4 seconds)

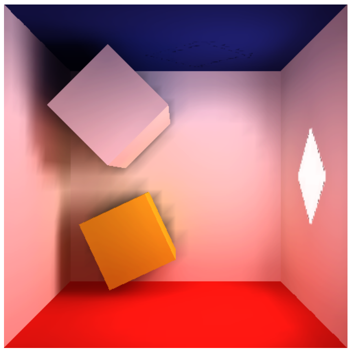


(c) Rendering using FE (0.8 seconds)

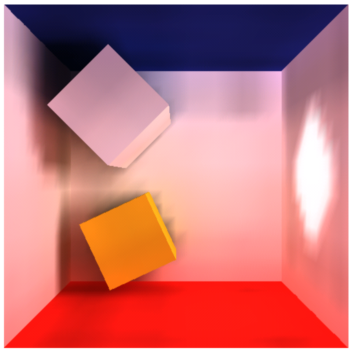
Figure 9: A second light appears on the eastern wall.



(a) The naive method (92 seconds)



(b) Rendering using AR (1.5 seconds)



(c) Rendering using FE (0.8 seconds)

Figure 10: The light on the blue wall disappears to reflect the passage of time.

5 Concluding remarks

Modeling scenes where lighting sources are irregular and unanticipated form a class of dynamic environments. The location and number of objects whose lighting changes are large may be unbounded. For example, revolving police search beams during dusk and dawn may pose challenges to current sophisticated algorithms in the sense that the significant lighting change may preclude their use. Of course, lighting changes need not be sudden; we simply might be interested in generating pictures at various points of the day, possibly in computer vision experiments. Figure 12 show the Cornell room (presumably with a window) bathed in *irregular* light appearing through trees.

In this paper we have considered this subclass of dynamic global illumination environments in an interactive setting (Figure 13). We have provided a simple fast solution based on hierarchical radiosity. Our solutions are computed incrementally and take a very small fraction of time compared to an algorithm that starts from scratch. We anticipate that our solution can be "buried" in more sophisticated schemes (for example, [12, 4]) that handle general dynamic environments.

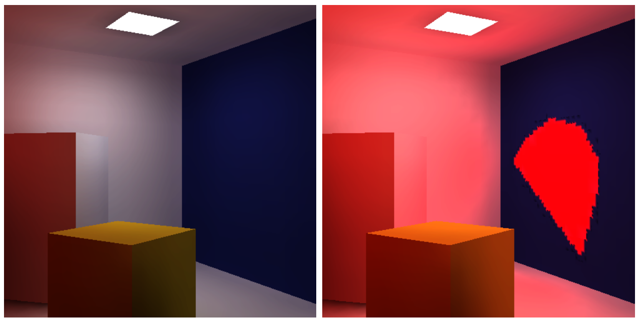


Figure 12: The Cornell room, assumed to have a window. The figure on the left shows the standard room lit by an interior light source. At a different point in the day, the room is lit by an unexpected light source with *arbitrary* shape through the window. We would like to build a solution based on the first one, instead of starting a new solution based on a new patch.

References

[1] S. E. Chen. Incremental radiosity: An extension of progressive radiosity to an interactive image synthesis system. In *Computer Graphics (Proceedings of SIGGRAPH 90)*, volume 24, pages 135–144, Dallas, Texas, August 1990. ISBN 0-201-50933-4.

[2] C. Damez and F. Sillion. Space-time hierarchical radiosity for high-quality animations. In *Eurographics Rendering Workshop 1999*, pages 235–246. Eurographics / Springer Wien, 1999.

[3] J. O. Dorsey, F. X. Sillion, and D. P. Greenberg. Design and simulation of opera lighting and projection effects. In *Computer Graphics (Proceedings of SIGGRAPH 91)*, volume 25, pages 41–50, Las Vegas, Nevada, July 1991. ISBN 0-201-56291-X.

[4] G. Drettakis and F. X. Sillion. Interactive update of global illumination using a line-space hierarchy. In *Proceedings of SIGGRAPH 97*, Computer Graphics Proceedings, Annual Conference Series, pages 57–64, Los Angeles, California, August 1997. ACM SIGGRAPH / Addison Wesley. ISBN 0-89791-896-7.

[5] J. D. Foley, A. van Dam, S. K. Feiner, J. Hughes, and R. L. Phillips. *Introduction to Computer Graphics*. Addison-Wesley, 1994.

[6] D. Forsyth, C. Yang, and K. Teo. "efficient radiosity in dynamic environments. In *Fifth Eurographics workshop on rendering*, 1994.

[7] D. W. George, F. X. Sillion, and D. P. Greenberg. Radiosity redistribution for dynamic environments. *IEEE Computer Graphics & Applications*, 10(4):26–34, July 1990.

[8] P. Hanrahan, D. Salzman, and L. Aupperle. A rapid hierarchical radiosity algorithm. In *Computer Graphics (Proceedings of SIGGRAPH 91)*, volume 25, pages 197–206, Las Vegas, Nevada, July 1991. ISBN 0-201-56291-X.

[9] E. Lindholm, M. J. Kilgard, and H. Moreton. A user-programmable vertex engine. In *Proceedings of ACM SIGGRAPH 2001*, Computer Graphics Proceedings, Annual Conference Series, pages 149–158. ACM Press / ACM SIGGRAPH, August 2001. ISBN 1-58113-292-1.

[10] K. Myszkowski, T. Tawara, H. Akamine, and H.-P. Seidel. Perception-guided global illumination solution for animation rendering. In *Proceedings of ACM SIGGRAPH 2001*, Computer Graphics Proceedings, Annual Conference Series, pages 221–230. ACM Press / ACM SIGGRAPH, August 2001. ISBN 1-58113-292-1.

[11] F. Schoeffel and P. Pomi. Reducing memory requirements for interactive radiosity using movement prediction. In *Eurographics Rendering Workshop 1999*, pages 225–234. Eurographics / Springer Wien, 1999.

[12] P. Tole, F. Pellacini, B. Walter, and D. P. Greenberg. Interactive global illumination in dynamic scenes. In *Proceedings of SIGGRAPH 2002. To Appear.*, Computer Graphics Proceedings, Annual Conference Series, San Antonio, July 2002. ACM SIGGRAPH / ACM Press.

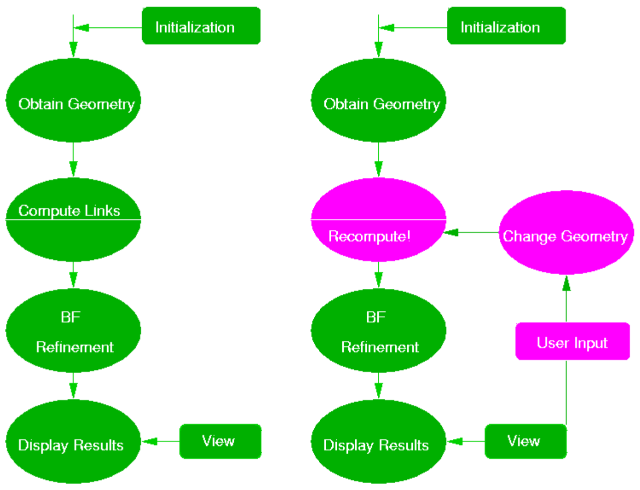


Figure 13: The figure on the left corresponds to traditional radiosity. This context of our algorithm appears in the figure on the right.