


• 1100013242
Copia A

**The use of visibility coherence
for radiosity computation**

X. Pueyo

Report LSI-93-3-R

 **UPC**
Facultat d'informàtica
de Barcelona - Biblioteca

The Use of Visibility Coherence for Radiosity Computation

X.Pueyo

Dept. Llenguatges i Sistemes Informàtics, Universitat Politècnica de Catalunya, Av. Diagonal 647 pl.8, E-08028-Barcelona

ABSTRACT

Coherence has been widely used to improved the performance of visibility computation in traditional hidden line/surface algorithms. The appearance of radiosity algorithms has increased the importance of computing efficiently the visibility function given that it is performed a lot of times to obtain a single static image. We develop, in this paper, a characterization of the uses of coherence in radiosity algorithms and we give a brief and schematic description of proposed algorithms using them.

INTRODUCTION

As well known, radiosity was introduced as a new powerful method to model the behavior of light interacting between diffuse reflecting surfaces. This new technique dealt with a problem of illumination models not studied in depth up until that moment.

Derived from thermal engineering, its application to image synthesis was proposed by Goral et al. [1] with the following conclusions:

(1)

$$B_i = E_i + \rho_i \sum_{j=1}^N F_{ij} B_j$$

where

B_i = radiosity in surface i

E_i = emissivity of surface i

ρ_i = reflectivity of surface i

F_{ij} = form-factor between surface i and j

N = number of surfaces in the environment

Equation 1 yields a set of N linear equations with N B_i unknowns where E_i and ρ_i are application data, and F_{ij} must be derived from geometry of the environment:

(2)

$$\begin{bmatrix} 1 - \rho_1 F_{11} & \dots & -\rho_1 F_{1N} \\ \vdots & \ddots & \vdots \\ -\rho_1 F_{N1} & \dots & 1 - \rho_N F_{NN} \end{bmatrix} \begin{bmatrix} B_1 \\ \vdots \\ B_N \end{bmatrix} = \begin{bmatrix} E_1 \\ \vdots \\ E_N \end{bmatrix}$$

The computation of the form factor is obtained by,

(3)

$$F_{A_i A_j} = \frac{1}{A_i} \int_{A_i} \int_{A_j} \frac{\cos(\phi_i) \cos(\phi_j)}{\Pi r^2} dA_j dA_i$$

Where A is the area, r is the distance between patches and ϕ is the angle between r and the normal. Instead of solving this double area integral, Goral et al. used Stoke's theorem and solved a contour integral. In fact, the resolution of the integral in equation (3) is the key to the different implementation methods.

The computation of form factors and the solution of equation (2) for a discretization N of the environment, gives the radiosities (B_i) of the patches' centre points, which are view independent. Finally, the rendering procedure computes the scene's visible points by z-buffering. The intensities are calculated by linear interpolation of B_i 's in object space.

We have seen above that the form-factor is given by the geometric relationship of two surface entities (patch to patch), and represents the ratio of energy leaving one of these entities that lands on the other. In real environment, light leaving a surface and arriving on another depends not only on this relationship, but also on elements in the scene that intercept light and produce shadows. Then the form-factors need only to be computed between surfaces or pieces of surfaces that are visible to each other. So a hidden surface removal process takes place for the computation of each form-factor. Thus, equation (3) must be modified in the following way,

$$F_{A_i A_j} = \frac{1}{A_i} \int_{A_i} \int_{A_j} \frac{\cos(\phi_i) \cos(\phi_j)}{\Pi r^2} h dA_j dA_i$$

where h takes the value one or zero depending on whether dA_j is visible from dA_i .

In order to speed up the computation of the intervisibility (i.e. the h function of equation 4) a number of papers have appeared which try to reduce its computational cost by means of exploiting several types of coherence. We develop here a characterization of them, firstly pointed out in [6]. This characterization focuses on radiosity algorithms for static environments with static observers. Finally, we describe schematically some of the algorithms using the different types of coherence.

We do not consider in the present work those algorithms where speed up is obtained by means of minimising the number of patches into which the environment is divided. We deal with algorithms which reduce the cost of computing the form factors of a "previously fixed" number of patches.

COHERENCE IN RADIOSITY. CHARACTERIZATION

We develop, in this section, a characterization of the uses of coherence made in radiosity algorithms in order to efficiently compute form factors. Coherence is often used in these algorithms when computing the h function. So most of them are well known strategies widely used in hidden line/surface removal algorithms. Many of the techniques we are going to explain have already been introduced for form factor computation. Other strategies have not yet been used so we just introduce them as ideas to be more deeply studied and eventually implemented. Whether each of the types of coherence we are going to see is adequate or not depends on a number of parameters. Specially on:

- the "complexity" of the environment: number and location of the geometrical elements of the scene.
- the type of implementation: sequential or parallel, software or hardware.

General concepts

Previous to the description of different types of coherence we consider interesting to underline several concepts we are going to use widely below.

- Geometrical elements. When they evaluate the h function, classical radiosity algorithms focus on the patches. The problem to solve is whether patch i may or not see patch j . This is sometimes solved for elements

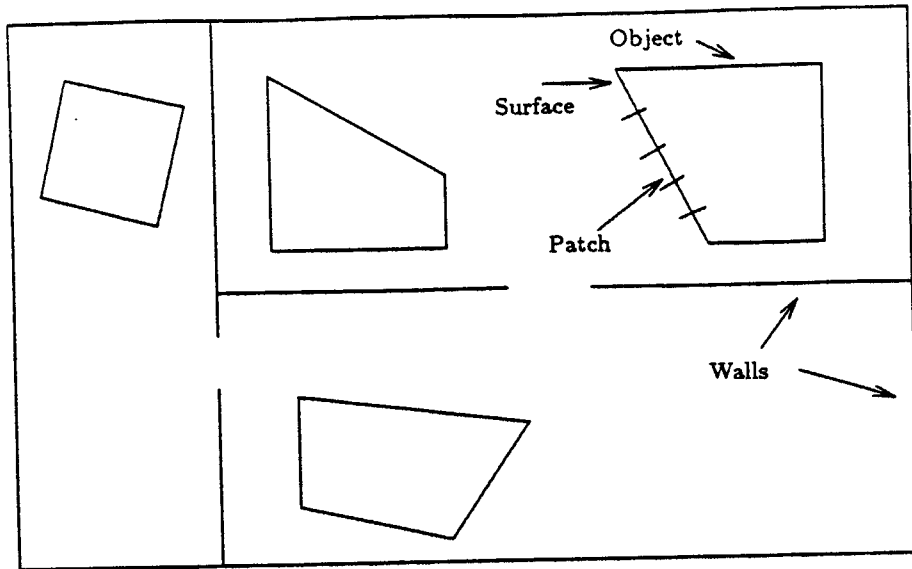


figure 1. Geometrical elements

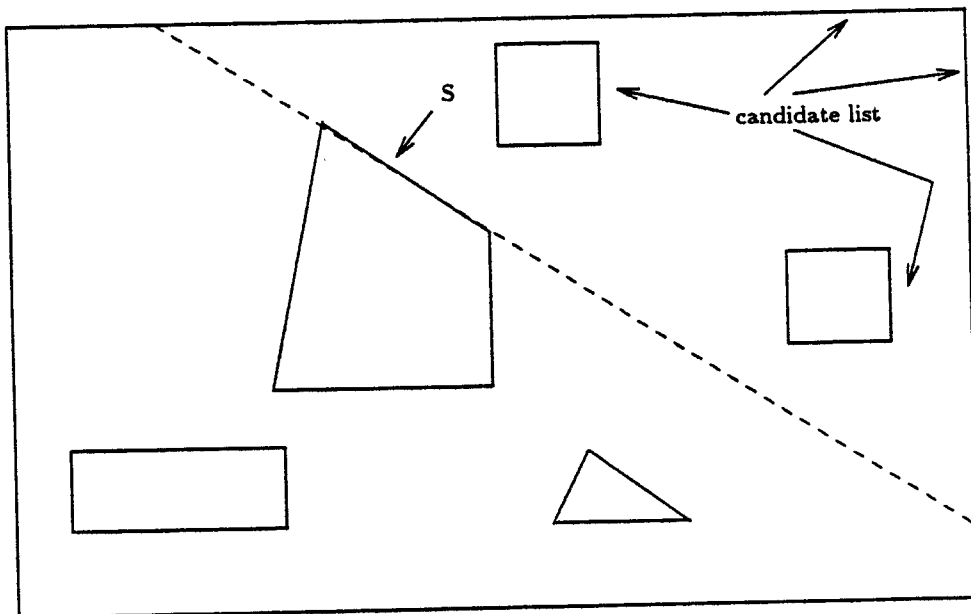


figure 2. Computing the candidate list

which result from the subdivision of patches. Algorithms using coherence to compute form factors often accounts on higher level geometrical elements: surfaces and objects. Finally, we find a special geometrical element, the wall. Sometimes the walls are thought as "big" surfaces, other they are considered as "thin" objects (figure 1).

- Attributes of the geometrical elements. Patches (surfaces, objects) are seen from different scopes in radiosity algorithms; as "observers" (visibility is computed from them) or as "observed". As a consequence of this, a same type of coherence will be usually exploited in a different way depending on the attribute "observer / observed".

- Goals of coherence. In form factor computation the targets of the use of coherence are two: to avoid computing form factors and to improve the performance of the computation of form factors taking advantage of the previously computed ones. Of course the former is the most interesting.

Characterization

Here we try to give a hierarchical view of the use of coherence based on the geometrical hierarchy of the scene elements. An environment is composed by a set of objects which are in turn composed by a set of surfaces (we assume here that surfaces are flat polygons). Finally, surfaces are split up into patches. We assume that lower level subdivision supply geometrical elements with the same behavior as patches; so we do not consider it. On the other hand, we consider that the scene may include a higher level of organization. A certain type of structure can be built in order to organize the elements (object and/or surfaces) of the environment. The success of these techniques in accelerating ray tracing makes us to think that it is reasonable to use them.

In an observer-surface the visibility function must be computed for each of his patches. Given one of them, its visibility is presumably very similar to the visibility of his neighbours. Let's take a surface S of the environment. Global clipping, view transformation and depth sort can be, partially or fully, performed for all patches of S in a unique operation. In fact, the plane of S clips the environment providing a candidate set of objects and/or surfaces (figure 2). In this operation the number of form factors to be computed for each patch of S is reduced to the candidate list and this is done once for all the patches of S instead of repeating the operation for each patch. The elements of the candidate list (objects and/surfaces) may be presorted, independently of the specific "observer" (patch) of S, in order to store depth priority information valid for all the patches of the surface. These procedures concerning all the patches of S may be helped using a suitable organization of the environment. So structuring it using well known tools like hierarchies of bounding boxes, octrees, BSP trees.

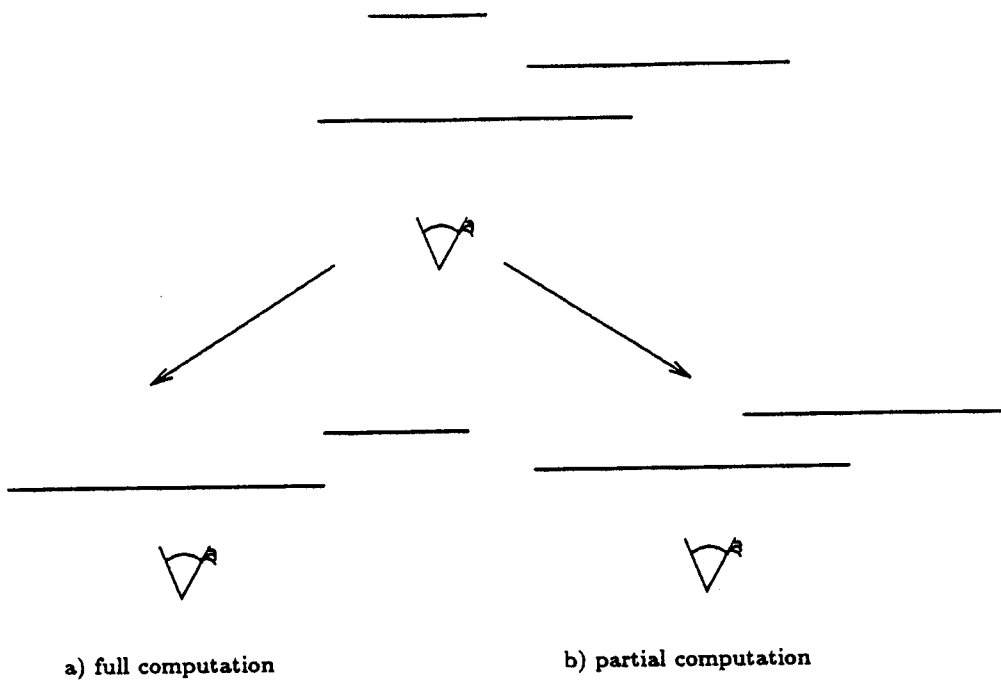


figure 3. Visibility computation at surface level

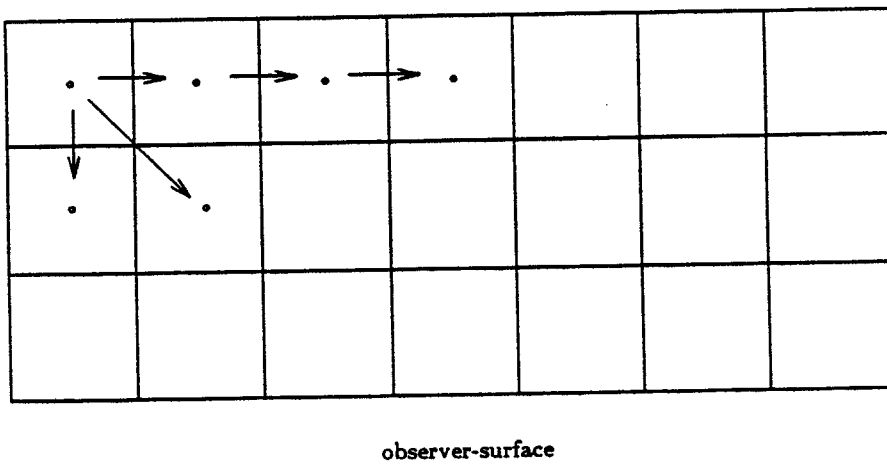


figure 4. Moving from patch to patch

If we focus on an observer-patch P of S , it happens something similar. Now we must solve visibility for several frustrums (five if we use the hemicube technique). For each of them, visibility will probably be very similar to the visibility of his neighbours. So we may take advantage of this in order to reduce the computational cost. For example, to clip a surface S may be done once for two consecutive frustrums. Visibility relationships may also be transmitted from a frustrum to its neighbours.

Once the list of observed-surfaces (or objects or patches) has been built, the algorithm must compute the visibility from each patch P of S ; or, more accurately, from each frustrum F of P . We recall that the list may or not contain depth information. Anyhow, the goal of this step is to determine which observed-patches are seen in the current frustrum. Doing this without account on the fact that patches are grouped on surfaces is a loss of efficiency. Actually, a good strategy may first compute visibility at surface level. This could imply an important reduction of the number of patches to deal with, so of the number of form factors to be computed. Two ways may be envisaged:

- Compute visible surfaces. This means to obtain a set of fully visible surfaces and a set of fully occluded surfaces. Then the surfaces of the visible set must be divided into patches which are fully visible. This technique has the drawback that the original surfaces are clipped by those which are in front of them. So the resulting surfaces are not regular and their discretization into patches will be more complex and must be performed for each view point. A more feasible approach is to compute patches only once; but, in this case, we must clip some patches in addition to the surfaces and these patches will not be regular (figure 3.a).

- Filter non visible surfaces. Now the idea is to remove all those original surfaces which are fully hidden. This means that surfaces partially hidden will be kept in the candidate list. So this list is bigger and must be reprocessed in order to compute the visibility of partially occluded surfaces, at patch level (figure 3.b).

Once the form factors have been computed for a given patch, the visibility computation from neighbouring patches may be accelerated accounting on the fact that this visibility will probably be very similar for neighbours belonging to the same surface (assumed flat here). In fact, this kind of coherence is very similar to frame-to-frame coherence used in animation. Actually, when the algorithm changes from a patch to one of its neighbours, its behaviour is the same as the one of an observer moving on the surface with a linear uniform movement. Something similar happens when moving from a given frustrum of a patch to a neighbour frustrum of the same patch; but, now with a circular movement (figure 4).

As we have pointed out above, a way to help the use of coherence is

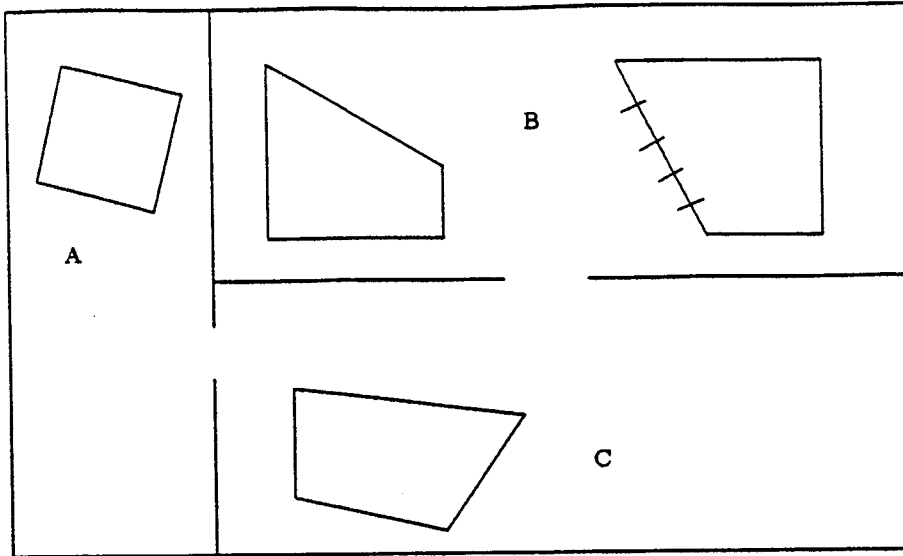


figure 5. Using walls

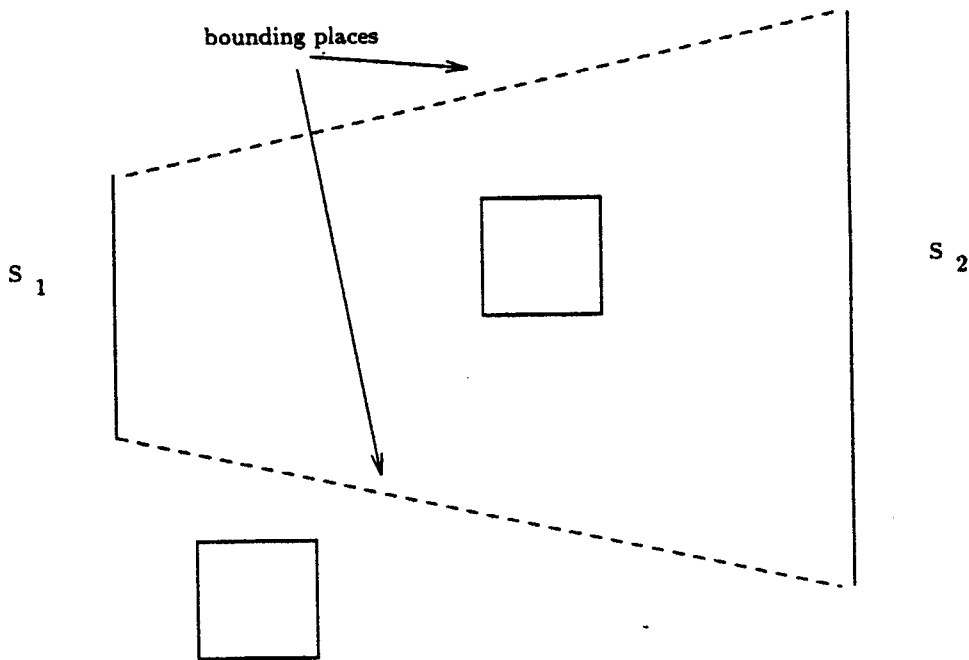


figure 6. Selection of occluding objects for a pair of surfaces

to introduce a suitable data structure of the scene. In some cases, these structures only contribute to quickly select candidate elements (objects, surfaces) by means of grouping them into bounding boxes like clusters. In other cases, the data structure also includes visibility information. It seems clear that the rear should be more suitable given the target.

Finally, we want to underline the fact that walls, very frequent in radiosity environments, may be very helpful to exploit coherence. Really, walls often divide the environment in such a way that they occlude a very big number of surfaces. So the visibility function would preprocess the environment (the candidate list) testing it against the walls of the current "subenvironment". In this way, the list of candidates to be submitted to the full process could be widely reduced. In figure 5, for example, objects in the subenvironment *A* have a low probability to see objects in the subenvironment *B*.

PROPOSED ALGORITHMS

In this section we give some references of papers where radiosity is computed using some kind of coherence and we match them with the characterization introduced above.

Marks et al. [2] and Haines et al. [3] have used a similar idea in order to account on the fact that patches of a same surface must have similar visibilities. More specifically, they give a tool to analyse whether there is or not any occlusion between two surfaces. In this way, they select surfaces or objects which occlude partially or fully one surface to the other. Afterwards, this reduced list of occluding elements is used to determine patches visibility. That selection is made by means of defining a box bounding both surfaces and using the faces of the box as clipping planes (figure 6). Actually, this idea was first exploited by Nishita and Nakamae in [4].

For a given observer-patch, visibility is computed at surface level in [5] and [6]. The former compute visible surfaces and is based on a BSP structure of the surfaces of each object. The rear discards the fully occluded surfaces using a z-buffer. Only the surfaces present at the buffer and the end of the process are taken into account for a second visibility process at patch level.

The technique exposed above, may be improved when the surfaces have been depth sorted like in the first algorithm of the previous paragraph. In this case, the candidate list may be processed in front to back order, so if we use a z-buffer, when a surface has been assigned to each of its pixels the process stops because all visible surfaces are already known. In addition to Campbell et al. this has been used by Wang et al. [7].

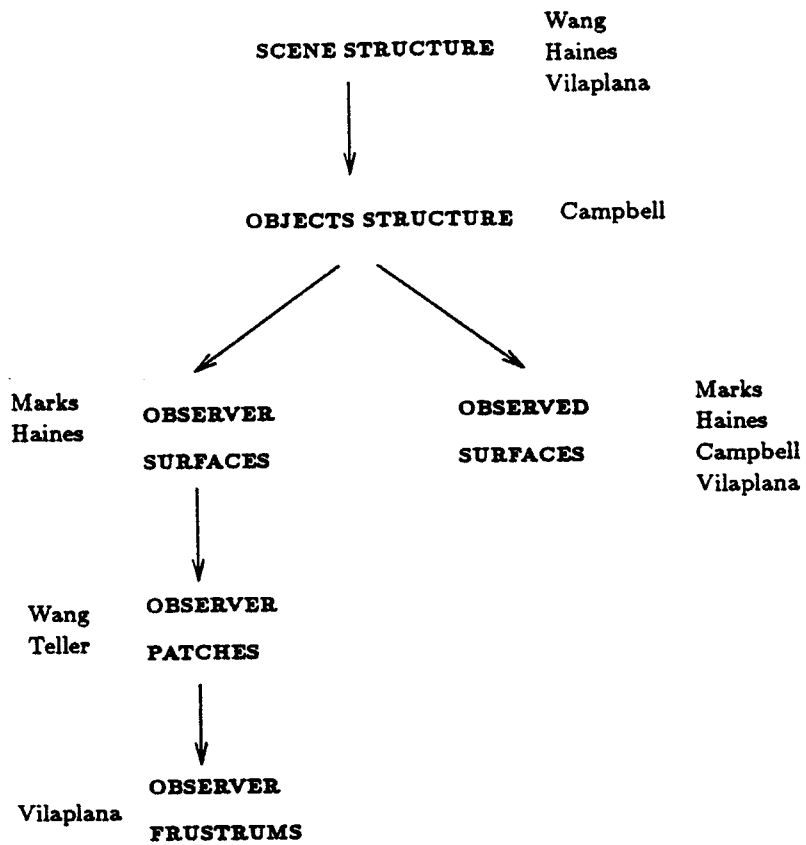


figure 7. Scheme of the levels of coherence used in proposed algorithms

The acceleration of form factor computation of a patch based on the previous computation of form factors of neighbouring patches have been used by Tellier et al. [8] in a ray tracing based algorithm. In this paper, the visible element for a patch in a given direction (so, the delta form factor) is derived from the results of the neighbour patch in the same direction. In [9] is proposed the use of coherence between neighbouring patches and frustrums for clipping and view transformations.

In order to improve the efficiency of radiosity algorithms three types of data structures have been proposed which help the implementation of some coherence concepts.

- Wang et.al.[7] proposed the used of an octtree to organise the radiosity environments. As well know, this struture will give depth priority information which is very helpful; but must clip surfaces.
- Haines et al. [3] use a hierarchy of bounding boxes which offers an efficient pruning tool. Nevertheless, this structure do not include useful visibility information.
- Vilaplana et al.[6] propose the use of a BSP tree structure of the scene. The nodes of the structure are planes separating the objects of the scene and the leaves are the objects. The advantages of this solution are that the structure includes information of potential visibility (as well know for BSP trees) and that surfaces are not clipped.

Instead of or in addition to structuring the environment, these algorithms may use a suitable structure for the surfaces of each object. Campbell et al. [5] proposed to use a BSP tree of surfaces for each object.

In figure 7 we try to summarize the different types of coherence used by the algorithms briefly described above. This scheme clearly suggest the use of a top down analysis of the radiosity algorithm in order to optimize the use of coherence.

CONCLUSIONS AND FUTURE WORK

We have developed a characterization of the uses of coherence to compute visibility for form factor evaluation. This study shows that radiosity algorithms offers a hierarchical use of coherence like in traditional visibility algorithms; but adding some new types of coherence which are specific for the determination of form factors. We have also presented several algorithms which use coherence strategies that match with the previous characterization.

From this work we may derive two main tasks to be developed. First we could extent this work in order to introduce algorithms which works for

dynamic environments and environments with moving observers. Secondly, we have observed that published algorithms very rarely present comparisons of the proposed methods with others. So it would be interesting to compare the efficiency of uses of coherence which belong to the same class.

ACKNOWLEDGEMENTS

The author wishes to thank J.Vilaplana for his helpful observations and B.Garcia for her valuable aid during the edition of the paper.

REFERENCES

- 1 Goral,C.M., Torrance,K.E., Greenberg,D.P. and Battaile, B. 'Modeling the Interaction of Light Between Diffuse Surfaces' in *Proc. ACM SIGGRAPH* 1984, pp.213-222.
- 2 Marks,J., Walsh,R., Christense,J. and Friedell,M. 'Image and Intervisibility Coherence in Rendering' in *Proc. of Graphics Interface'90*, pp. 17-30, Canadian Information Processing Society, 1990.
- 3 Haines,E.A. and Wallace,J.R. 'Shaft Culling for Efficient Ray-Traced Radiosity' in *Proc. of the Second EUROGRAPHICS Workshop on Rendering*, May 1990.
- 4 Nishita,T. and Nakamae,E. 'Continuous Tone Representation of Three Dimensional Objects Taking Account of Shadows and Interreflection' in *Proc. ACM SIGGRAPH* 1985, pp. 23-30.
- 5 Campbell,A. and Fussell,D.S. 'Adaptative Mesh Generation for Global Diffuse Illumination' in *Proc. ACM SIGGRAPH* 1990, pp. 155-164.
- 6 Vilaplana,J. and Pueyo,X. 'Multilevel Use of Coherence for Complex Radiosity Environments' (submitted), *Research Report of LSI Dept., Universitat Politècnica de Catalunya*, 1993.
- 7 Wang,Y. and Davis,W.A. 'Octant Priority for Radiosity Image Rendering' in *Proc. Graphics Interface'90*, pp. 83-91, Canadian Information Processing Society, 1990.
- 8 Tellier,P., Maiselle,E., Bouatouch,K. and Languenou, E. 'Using Coherence to Accelerate Radiosity' to appear in *The Visual Computer*, IRISA Publication Interne 616, 1991.
- 9 Vilaplana,J. and Pueyo,X. 'Exploiting Coherence for Clipping and View Transformation in Radiosity Algorithms' *Photorealism in Computer Graphics*, pp. 137-149, Eurographics Seminars Series, Springer Verlag, 1991.

Departament de Llenguatges i Sistemes Informàtics
Universitat Politècnica de Catalunya

List of research reports (1993).

LSI-93-1-R "A methodology for semantically enriching interoperable databases", Malú Castellanos.

LSI-93-2-R "extraction of data dependencies", Malú Castellanos and Fèlix Saltor.

LSI-93-3-R "The use of visibility coherence for radiosity computation", X. Pueyo.

LSI-93-4-R "An integral geometry based method for fast form-factor computation", Mateu Sbert.

LSI-93-5-R "Temporal coherence in progressive radiosity", D. Tost and X. Pueyo.

Internal reports can be ordered from:

Nuria Sánchez
Departament de Llenguatges i Sistemes Informàtics (U.P.C.)
Pau Gargallo 5
08028 Barcelona, Spain
secrelsi@lsi.upc.es