A Progressive Algorithm For Three Point Transport

Reynald Dumont [*+], Kadi Bouatouch [+], Philippe Gosselin [*]

[*] CRIL Ingénierie, 5 Sq. Chêne Germain, 35510 Cesson-Sévigné, France

[+] IRISA, Campus Universitaire de Beaulieu, 35042 Rennes Cedex, France

Abstract: When computing global illumination in environments made up of surfaces with general Bidirectional Reflection Distribution Function, a three point formulation of the rendering equation can be used. Brute-force algorithms can lead to a linear system of equations whose matrix is cubic, which is expensive in time and space. The hierarchical approach is more efficient. Aupperle et al. proposed a hierarchical three point algorithm to compute global illumination in presence of glossy reflection. We present in this paper some improvements we brought to this method: shooting, "lazy" push-pull, photometric subdivision criterion... Then we will show how our new method takes into account non-planar surfaces in the hierarchical resolution process.

1 Introduction

Radiosity and ray-tracing techniques make strong assumptions about the characteristics of the used materials. The first one assumes that all surfaces are perfectly diffuse and the second, totally view dependent, replaces the indirect diffuse component by an ambient term. A few years ago, some approaches, called two-pass methods, mixed both techniques to handle at once specular and lambertian surfaces ([15], [18] and [19]). The global diffuse and the global specular components are evaluated at the first and the second pass respectively.

Conversely, the one-pass methods compute in one pass the global diffuse and the global specular components. To compute an image with these methods a rendering step is needed. Two kinds of one-pass method have been devised in the past years to take account of glossy reflection.

The first one relies on the Monte Carlo method such as *random walk*. It is relatively easy to implement and can be used with complex geometry and general reflectance functions but is notoriously slow since too many rays have to be cast to get an accurate result ([7], [10], [11], [13], [20]).

The second technique makes use of wavelets ([5], [12], [14]) and spherical harmonics ([16]) to compute global illumination. Indeed, wavelets or spherical harmonics are used as basis functions to express the radiance of each patch for each direction. In [5], the object surfaces as well as the direction space are adaptively subdivided (two point method). In [12][14], the rendering equation is expressed in terms of three point transport and only the environment surfaces are meshed into patches. In case of constant basis functions, Aupperle and Hanrahan proposed an interesting specific method for three point transport which computes the radiance from a patch to another one. The data structures needed by the wavelet-based methods require an important memory capacity which limits them practically. Regarding the spherical harmonics-based methods, they need a high number of basis functions and make use of a uniform meshing of the object surfaces. Like the wavelet-based methods, Aupperle's method has the advantage to be hierarchical but still requires an important memory storage because of the huge number of links (between interactions) and a too fine meshing due to the non use of a photometry-based subdivision criterion.

One solution to overcome the memory limitation is to use the methods described by Teller and Airey ([17], [1]). These methods relie on a binary space subdivision of the environment into 3D cells. A polygon in a cell C sees only the polygons lying in C as well as those visible through the holes (also termed portal) in its boundary, like windows and doors for building interiors. To gather energy impinging on each patch within a cell C, we need to store in memory only the cells C_i visible from C, which limits the memory storage. In spite of the efficiency of this kind of method, the problem of memory storage still remains in a cell containing many surfaces and many glossy reflectors.

To overcome these difficulties we propose in this paper an improvement as well as an extension of Aupperle's method to non-planar subdivision.

The organization of this paper is as follows. Section 2 is an outline of our work. Section 3 briefly recalls Aupperle's et al. algorithm. Section 4 describes the different improvements we have brought to Aupperle's method such as: Shooting, photometric subdivision criterion, efficient Push/Pull and rendering. During the subdivision process, objects can be subdivided into four subsurfaces. Some difficulties arise when surfaces in the environment are not planar (i.e. the four subsurfaces resulting from the subdivision may not lie on the same plane). Section 5 gives one solution to this problem.

2 Outline

Our approach is based on Aupperles's work which relies on a one pass method as said previously. Recall that with this method, the number of links stil remains high and the meshing is too fine due to the non-use of a photometry based subdivision criterion.

Once an iteration (shooting or gathering) has been completed, we propose to remove all the created links to save memory. Moreover, to avoid a too fine meshing, we make use of a photometry-based subdivision criterion. In our implementation we have opted for shooting rather than gathering because useful images can be produced very early in the calculation process. To improve the efficiency of our algorithm we propose an accelerated push-pull process called "lazy" push-pull from now on. Unlike Aupperle's method our algorithm is totally view independent, say it does not consider the eye as a small patch within the environment.

Another important contribution brought by our work is the handling of curved surfaces. We will see how a patch can be subdivided into four non coplanar sub-patches.

3 Hierarchical Algorithm for Three Point Light Transport

3.1 The Radiance Equation

3.1.1 Three point transport

Global illumination in a general environment may be expressed in terms of three point light transport between a triplet of surfaces A, A' and A" (i.e. flux emitted by A' towards A" when illuminated by A).

The radiance from a point x' towards a point x" is defined as the emitted flux, per unit solid angle, per unit projected area, originating at x' in the direction of x".

We can therefore express the radiance $L(x',x'')$ at a point x' towards a point x" when illuminated by a point x as:

$$L(x',x'') = fr(x,x',x'').L(x,x').G(x,x').dx \qquad \textit{Equation 1}$$

where the geometric term G is given by $G(x,x') = \frac{\cos\theta_i . \cos\theta_A}{\|x-x'\|^2}.V(x,x')$ (See Figure 1).

For simplicity we leave wavelength out of our equations. The term *fr* is called bidirectional reflectance distribution function or BRDF and V is a boolean used to express occlusions between x and x'.

By integrating over A, A' and A" we can easily prove that the total flux emitted by A' towards A" when illuminated by A is equal to:

$$\Phi_{AA'A''} = \int_A \int_{A'} \int_{A''} fr(x,x',x'').L(x,x').G(x,x').G(x',x'').dx''dx'dx \qquad \textit{Equation 2}$$

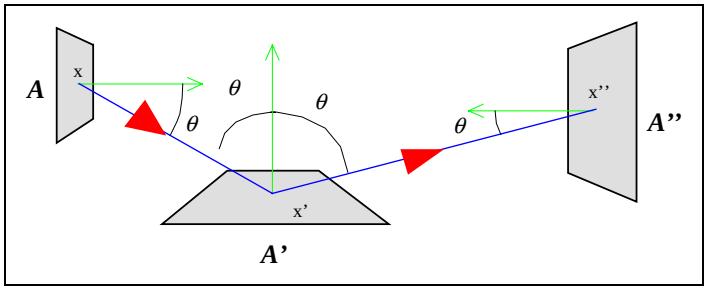

Figure 1: Geometry for three point transport

3.1.2 Discrete form of Three Point Transport

We can rewrite Equation 2 in a discrete form with surface elements A_i, A_j and A_k of surfaces A, A' and A" respectively, so as radiances, BRDF's and geometric terms are nearly constant over each surface element. Then we can obtain a discretized form of the radiance equation:

$$L_{A_jA_k} = \sum_i L_{A_iA_j} R_{A_kA_jA_i} \qquad \textit{Equation 3}$$

where $L_{A_jA_k}$ (resp. $L_{A_iA_j}$) is the radiance emitted by A_j towards A_k (resp. A_i towards A_j).

Taking into account self-emission we thus have a formulation of the light transport between A_j and A_k when A_j is illuminated by all the surface elements A_i in the environment:

$$L_{A_jA_k} = L^{\varepsilon}_{A_jA_k} + \sum_i L_{A_iA_j} R_{A_kA_jA_i} \qquad \textit{Equation 4}$$

3.1.3 Area Reflectance

The quantity $R_{A_k A_j A_i}$ is a formulation of reflection over three surfaces A_i, A_j, A_k and has a physical significance: it is no more than the proportion of energy originating at surface A and reflected by surface A' towards A":

$$R_{A_i A_j A_k} = \frac{\int_{A_i}\int_{A_j}\int_{A_k} fr(x,x',x'')G(x,x')G(x',x'')dx''dx'dx}{\int_{A_i}\int_{A_j} G(x,x')dx'dx} = \frac{Flux\ transported\ from\ A_i\ to\ A_j\ that\ is\ reflected\ towards\ A_k}{Flux\ transported\ from\ A_i\ to\ A_j}$$

This term is called area reflectance and satisfies both energy conservation and symmetry properties. With the assumption that the BRDF and geometric terms are relatively constant over each surface element, we can thus obtain a simplified expression:

$$R_{A_k A_j A_i} = \pi . F_{A_j A_i} . fr_{A_k A_j A_i} \qquad \textit{Equation 5}$$

The term $F_{A_j A_i}$ represents the form factor between A_j and A_i and $fr_{A_k A_j A_i}$ is the discretized value of the BRDF over surfaces A_k, A_j, A_i. The accuracy of the estimates for F and *fr* depends on the size of the patches over which reflectance is computed, relative to the distance between them. As the relative size decreases so do the computation errors leading then to the adaptive refinement strategy as shown in the following.

3.2 The Algorithm

3.2.1 Principle

Equation 4 leads to a linear system of equations whose solution can be obtained by gathering. The unknowns are the radiances L_{AiAj} with i and j ranging from 1 to n, n being the number of patches. The complexity of the system is $O(n^3)$ which makes the resolution problem untractable. That is why Aupperle et al. proposed a hierarchical algorithm to solve this system. While hierarchical radiosity operates on patches and refines the links between them, the three point transport algorithm operates on patch-to-patch interactions and refines the links between them. Figure 2 gives the data structure associated with an interaction. As refinement results, the interactions are subdivided under the form of hierarchy as shown in Figure 3. In this figure the contribution of the interaction IJ to the interaction JK is illustrated by four contribution links: $I_1J \rightarrow JK$, $I_2J \rightarrow JK$, $I_3J \rightarrow JK$ and $I_4J \rightarrow JK$).

```
Structure Interaction
    {
    Patch From;                 /* Emitter */
    Patch To;                   /* Receiver */

    Float L[];                  /* Total radiance */
    Float L^G[];                /* Radiance gathered during one iteration */

    List InteractionsToGather;  /* The interactions which contribute to this one during one iteration */
    Interaction D1, D2, D3, D4; /* Subinteractions produced by adaptive subdivision */
    }
```

Figure 2: Interaction Data Structure

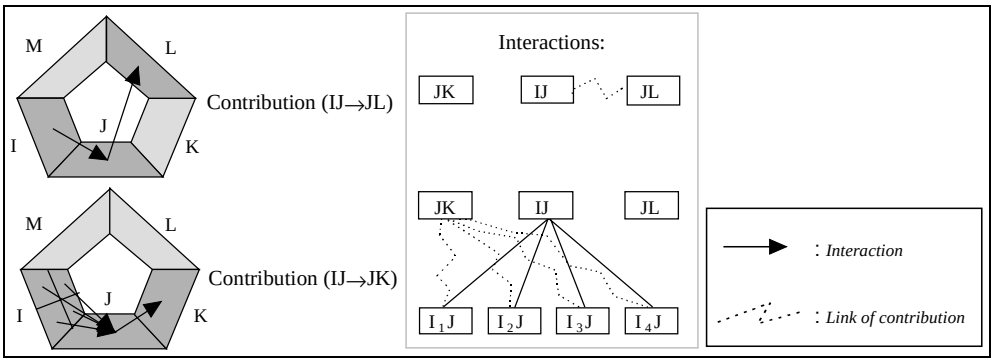

Figure 3: Hierarchical discrete three point transport

3.2.2 Refinement

The refinement procedure `RefineAndLink()` computes pairs of interactions by subdividing and recursively refining links if the error estimates exceed specified error bounds, then links these interactions if these bounds are satisfied or no further subdivision is possible. `FFErr` and `FrErr` are the bounds for the geometric and the reflection errors respectively and `MinArea` is the minimum area for a patch (See Figure 4 and Figure 5).

```
Procedure RefineAndLink(Interaction IJ, Interaction JK, Float FFErr, Float FrErr,Float MinArea)
    {
    If (errors on formfactors and fr are low) Then
        Link(IJ,JK);
    Else
        P=Patch_Inducing_Maximum_Error() ;
        Subdivide P;
        Subdivide IJ and/or JK according to P; (See Figure 5)
        Switch (P)
            {
            case I : RefineAndLink over children of IJ ;
            case J : RefineAndLink over children of IJ and JK ;
            case K : RefineAndLink over children of JK ;
            }
    EndIf
    }
```

Figure 4: RefineAndLink() Procedure.

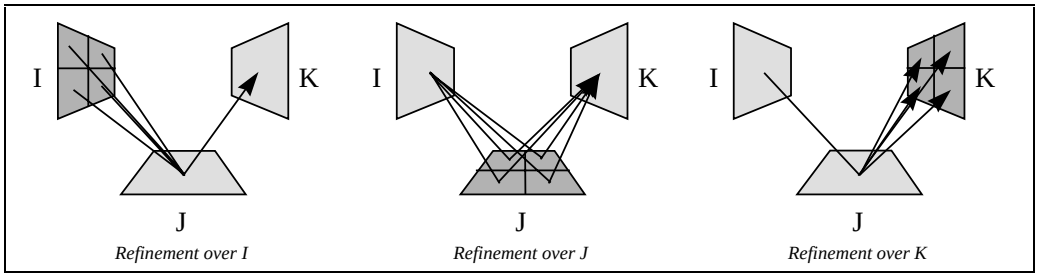

Figure 5: The three potential case of subdivision.

3.2.3 Gathering Radiance

Once the refinement process has been performed, each interaction gathers radiance from the interactions to which it is linked. This process continues till convergence (Figure 6).

```
Procedure Gather(Interaction JK)
    {
    If (JK≠NULL) Then
        JK→L^G=0;
        For each interaction IJ of JK→InteractionsToGather Do
            JK→L^G+=IJ→L*AreaReflectance(IJ,JK);
        EndOfFor
        Gather(JK→D1);
        Gather(JK→D2);
        Gather(JK→D3);
        Gather(JK→D4);
    EndIf
    }
```

Figure 6: Gathering() Procedure

Note that the eye is considered as a small patch causing no occlusions nor reflections.

3.2.4 PushPull Radiance

Recall that an interaction IJ between two input patches I and J is represented by a hierarchy of interactions between subpatches of I and J. For a reason of coherence, the gathered radiances must be distributed through each hierarchy. Indeed, a radiance is pushed down unchanged from a node to its children and area averaged when it is pulled from a node to its ancestors.

Given an interaction IJ, if patch I is subdivided, the pulled radiance is $L_{IJ} = \sum_{m=1}^{4} \frac{A_{I_m}}{A_I} L_{I_m J}$, or if J is subdivided, it is equal to $L_{IJ} = \sum_{m=1}^{4} \frac{A_{J_m}}{A_J} L_{IJ_m}$.

3.3 Discussion

In spite of the advantages brought by the three point transport algorithm, some problems remain not solved. Indeed, firstly, since the used refinement criterion does not account for photometric quantities, the number of interactions still remains important, which requires a large memory size and an important computation time. Secondly, the algorithm is view dependent since the view point is considered as a small patch within the environment. And thirdly, only planar patches are considered.

4 Our Approach

4.1 Principle

As said before, Aupperle's method is a demanding process in terms of computing time and memory ressources, which limits the method practically. In this section we will describe in turn the different solutions to overcome these difficulties: Shooting, photometric subdivision criterion, efficient Push/Pull and rendering.

Unlike Aupperle's approach, we have opted for a shooting technique instead of gathering in order to get useful images at the earlier steps of the resolution and to reduce the memory size needed for storing the data structures thanks to a dynamic link management strategy. With this latter the spatial complexity is reduced to $O(n+k^2)$ (rather than $O(n+k^3)$), where n is the number of elements at the finest level of subdivision and k is the number of input patches. Figure 7 gives our algorithm.

```
Until converged
  ① Selection of the most powerful interaction (IJ)
  ② PushPullRadiance(IJ);
     /* Radiances are pushed/pulled to descendants and ancestors to maintain the correctness of the hierarchy of IJ */
     /* See Section 4.4 for more details */
  ③ For each interaction (JK) Do
        (A) If (BRDF(IJ,JK)>0) Then
              a. RefineAndLink(IJ,JK);
                   /* Recursive refinement between IJ and JK to obtain the required precision */
                   /* Creation of links between the 2 interactions. */
              b. Shooting(IJ);
                   /* Recursive shooting over links created in (a.) */
                   /* Unshot energy of IJ contributes to the radiance of JK */
              c. Unlink(IJ);
                   /* Links created in (a.) are removed for memory saving */
            EndIf
        (B) SetToZeroδL(IJ);
              /*  (IJ) have just emitted its power; all δL_IJ of its hierarchy are set to zero */
     EndOfFor
```

Figure 7: Main Algorithm

Rendering is performed by a gathering method combining ray-tracing and Monte Carlo method.

4.2 Shooting Radiance

The gathering method proposed by Aupperle needs the storage of all the links between hierarchies of interaction. Since a too large amount of memory is required for this information, we have opted for a different approach based on shooting.

Our shooting algorithm allows a dynamic management of links and then only few links are always kept in memory. To achieve this, links between the emitting and the receiving interactions are first created. Once the shooting operation has been performed these links are deleted. This process allows to save memory at the expense of an extra computing time.

The data structure associated with an interaction becomes now:

```
Structure Interaction
    {
    Patch From;                 /* Emitter */
    Patch To;                   /* Receiver */

    Float L[];                  /* Current estimate of the final radiance */
    Float L^G[];                /* Radiance gathered during one iteration */
    Float δL[];                 /* Unshot radiance of the interaction */

    List InteractionsToShoot;   /* The interactions to which this one contributes during one iteration */
    Interaction D1, D2, D3, D4; /* Subinteractions produced by adaptive subdivision */
    }
```

Figure 8: The New Data Structure

The computation of the shooting contribution consists of three steps:

1. Select the next shooting interaction IJ corresponding to the greatest unshot flux $\delta\phi$:
   $\delta\phi_{IJ} = \pi . A_I . F_{IJ} . \delta L_{IJ}$
2. Compute the contribution of interaction IJ to each other interaction JK:
   $L^G_{JK} = \delta L_{IJ} R_{KJI}$ (*for each node of IJ linked to JK*)
3. Reset the unshot radiance:
   $\delta L_{IJ} = 0$ (*for each node of IJ*)

Note that, to achieve this, this shooting procedure looks for all the existing links between the nodes belonging to the hierarchies associated with the emitting and the receiving interactions (Figure 9).

```
Procedure Shooting(Interaction IJ)          /* IJ is the interaction having the most unshot energy */
    {
    Interaction JK;
    If (IJ→InteractionsToShoot≠NULL) Then
        JK=First(IJ→InteractionsToShoot);   /* First element of the list of interactions */
        While (JK≠NULL) Do
            JK→L^G = JK→ L^G + IJ→δL*AreaReflectance(IJ,JK);
            JK=Next(IJ→InteractionsToShoot); /* Next element of the list of interactions */
        EndOfwhile
    Else
        /* We look for links at lower levels */
        Shooting(IJ→D1);
        Shooting(IJ→D2);
        Shooting(IJ→D3);
        Shooting(IJ→D4);
    EndIf
    }
```

Figure 9: The Shooting Procedure

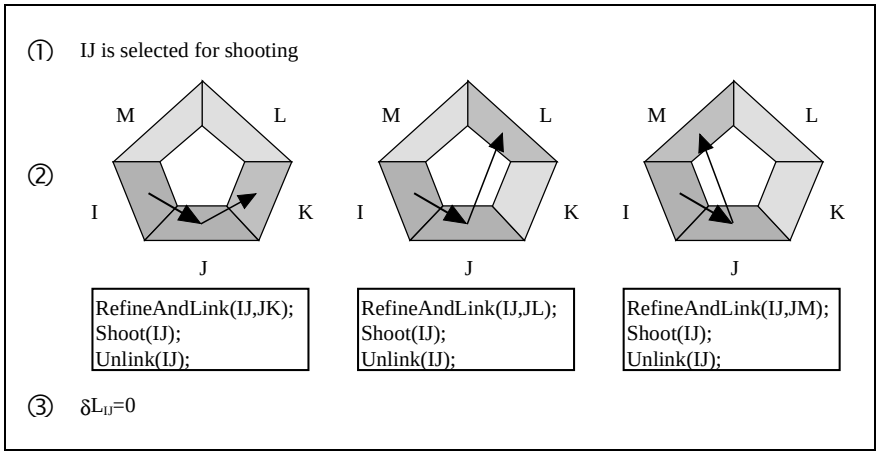

Figure 10: One iteration

4.3 Photometric Subdivision Criterion

Like for hierarchical radiosity (with BF criterion as in [8]), our subdivision criterion is based on the shot flux. In the `RefineAndLink` procedure the subdivision criterion becomes now:

```
If ((φIJK is low) or (errors on formfactors and fr are low)) Then
        Link(IJ,JK);
```

Where ϕ_{IJK} is the unshot flux which will be emitted by the interaction IJ towards the interaction JK.

4.4 Efficient PushPull

As the push/pull operation is time consuming, it is not necessary to perform it each time an interaction gets energy from the other interactions. The energy of an interaction is pushed/pulled only when it is chosen for shooting. However, to select the interaction having the greatest unshot energy, the radiance of each interaction between two input patches must be known. To this end, a variable δL^{Root} is associated with the root node of each hierarchy. Note that whenever an interaction of a hierarchy H gets energy from the others, the variable δL^{Root} associated with H is updated.

Let IJ be a receiving interaction, I_m a subpatch of I and J_n a subpatch of J. It is easy to see that:

$$\delta L^{Root} = \delta L^{Root} + L^{G}_{I_m J_n} \cdot \frac{AreaOf(I_m)}{AreaOf(I)} \cdot \frac{AreaOf(J_n)}{AreaOf(J)} \qquad \textit{Equation 6}$$

4.5 Final Gathering

Recall that, unlike Aupperle's method, our approach is view independent. To compute an image of the environment as seen by an observer, a rendering step combines ray-tracing and the Monte Carlo method. Note that our approach is not a two pass method like in [15], [18] and [19].

Let's recall the radiance emitted at point x in direction $\vec{\omega}_r$ for an incidence $\vec{\omega}_i$:

$$L^{out}(x,\ \vec{\omega}_r) = \int_{2\pi} fr(\vec{\omega}_i, \vec{\omega}_r).L^{in}(x, \vec{\omega}_i).\cos\theta_i.d\omega_i$$

This integral can be estimated with the Monte Carlo method by:

$$L^{out}(x, \vec{\omega}_r) = \rho(x) \cdot \frac{1}{nb\_samples} \cdot \sum_{i=1}^{nb\_samples} L_i^{in}(x, \vec{\omega}_i) \qquad \textit{Equation 7}$$

where $\rho(x) = \int_{2\pi} fr(\vec{\omega}_i, \vec{\omega}_r).\cos\theta_i.d\omega_i$ is the reflectivity of the considered material at point x.

To carry out the Monte Carlo method, a ray originating at the viewpoint is traced through each pixel. This ray intersects the scene at point X_S. From X_S, a number n of secondary rays are traced according to the pdf $fr(\vec{\omega}_i, \vec{\omega}_r).\cos\theta_i / \rho(x)$ (See Figure 11a). Suppose that a secondary ray intersects the scene at point X_D (See Figure 11b). The points X_S and X_D belong to two input patches A and B respectively. The problem now is to find the radiance at X_D in direction of X_S. To achieve this, the hierarchy of interaction AB is traversed from the root to the leaf corresponding to the smallest subpatches containing the points X_S and X_D (See Figure 11c).

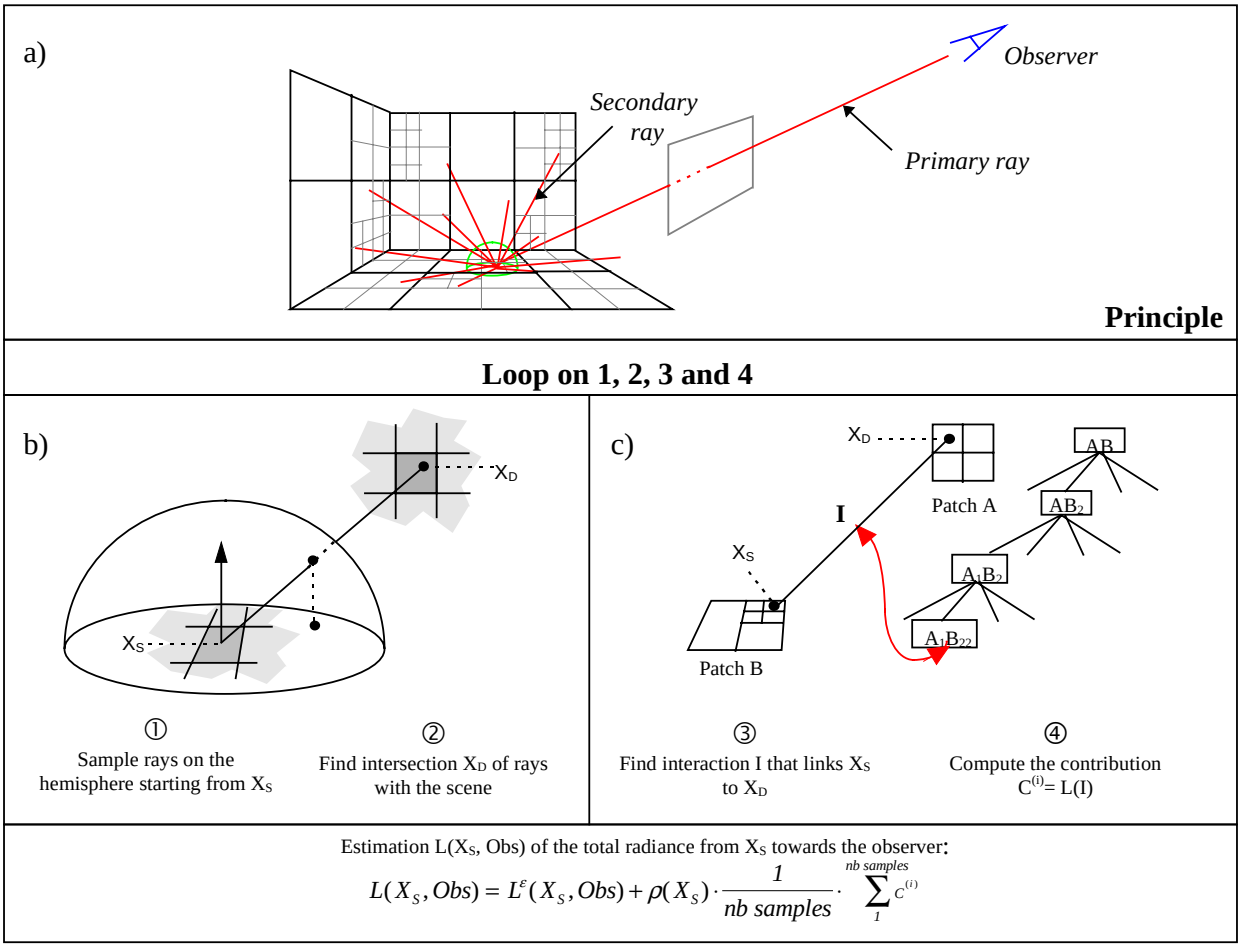

Figure 11: Final Gathering

4.6 Visibility

The method used for visibility calculation was deliberately left out of this paper since we use exactely the one given in [4]. In order to avoid computing it repeatedly, we store it in each interaction, under the form of a form factor multiplied by a visibility term.

5 Non-Planar Subdivision

5.1 Motivation

Let S be a curved surface within an environment for which global illumination computation has to be performed with our hierarchical algorithm. Thus, S must be initially approximated by a collection of polygons fitting the surface S (input patches). Let us call this approximation : "initial meshing".

Suppose that during the refinement process, an input patch of S is subdivided into 4 coplanar subpatches. If this curved surface has important curvatures then its initial meshing must be fine to closely approximate the surface S. The consequence of this is the increase of the number of initial interactions which affects the performance of our hierarchical algorithm. To overcome this problem, we enable each input patch (and derived subpatches) to be subdivided into 4 non-coplanar subpatches in order to fit at best the curved surface. With this approach the initial meshing may be coarse, which reduces drastically the number of initial interactions, the number of links, and consequently the computation time as well as the required memory capacity.

In the following, interactions between non-coplanar subpatches of the same patch will be called *secondary* interactions while the others will be named *primary* (See Figure 12).

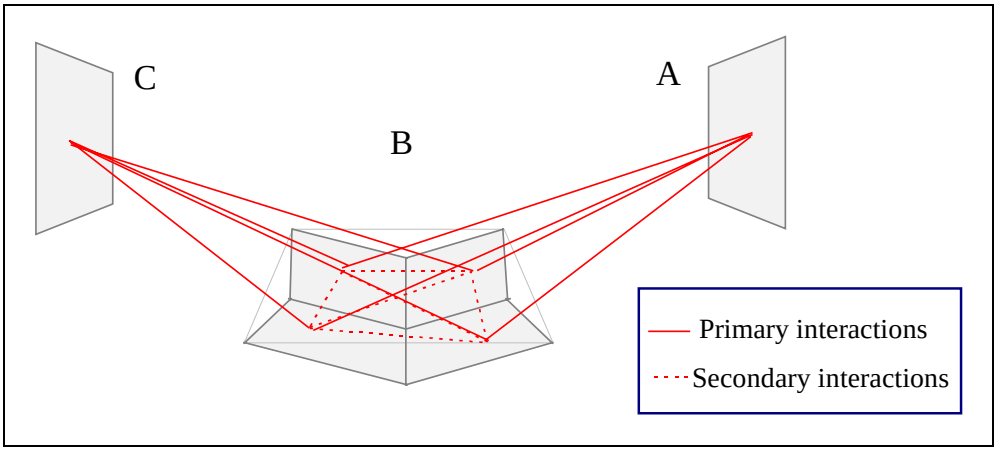

Figure 12 : Non-Planar subdivision

5.2 Outline of the method

It is obvious that the huge number of secondary interactions limits the hierarchical method practically. For this reason, we propose an extended hierarchical algorithm aiming at reducing these interactions as well as the associated links.

Let A and C be any two patches, and B an input patch of a curved surface. Suppose that the refinement process requires the subdivision of B into four non-coplanar subpatches B_1, B_2, B_3, B_4. In our method, the energy transfer from A to C through B is performed directly through the links ($AB_i \rightarrow B_iC$) and indirectly through only the links ($AB_i \rightarrow B_iB_j \rightarrow B_jC$). The other transfers like $AB_i \rightarrow B_iB_j \rightarrow ... \rightarrow B_lB_m \rightarrow B_mC$ are ignored by our method to make the problem tractable. Note that B_iB_j are secondary interactions which are not necessarily saved in memory (See Figure 13a).

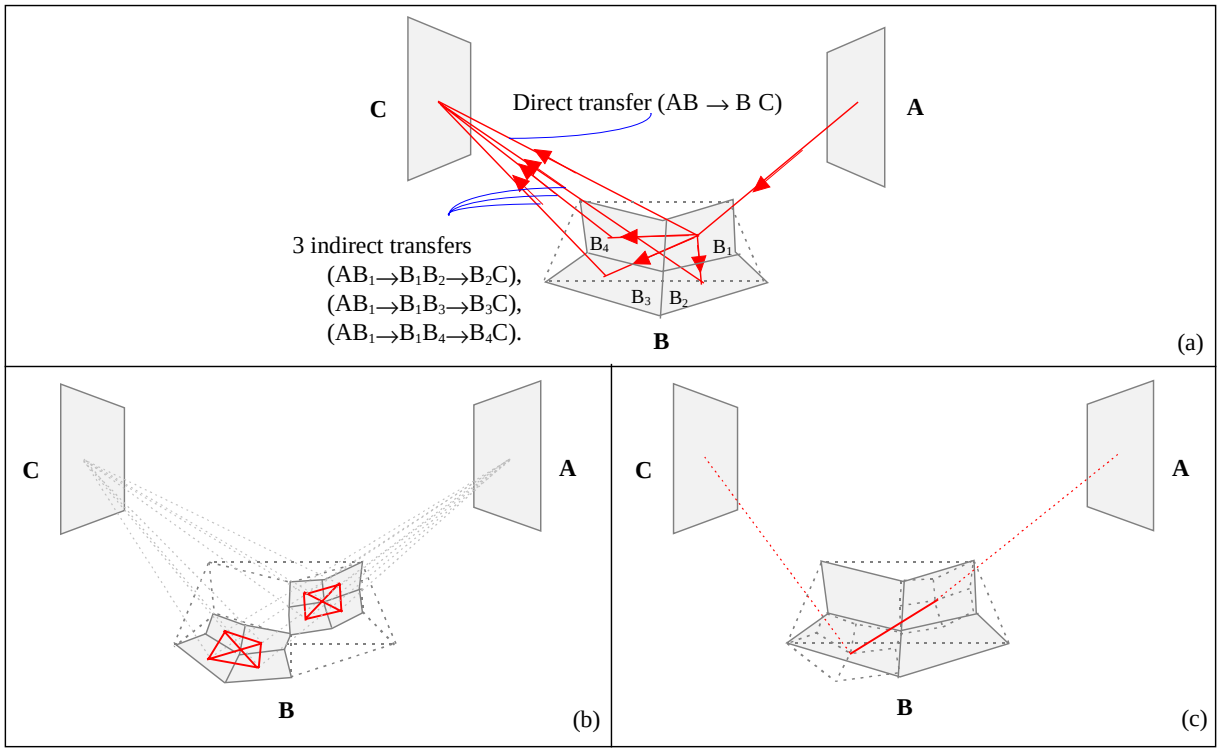

Figure 13 : Extended three point transport

Suppose that the patches B_1 and B_3 have to be subdivided in their turn into four subpatches B_{11}, B_{12}, B_{13}, B_{14} and B_{31}, B_{32}, B_{33}, B_{34} respectively (See Figure 13b).

The energy transfer between A and C through B_1 is performed with the help of the direct links ($AB_{1i} \rightarrow B_{1i}C$) and of the indirect links ($AB_{1i} \rightarrow B_{1i}B_{1j} \rightarrow B_{1j}C$). The energy transfer between A and C through B_3 is performed similarly.

Note that the secondary interactions like $B_{1i}B_{3j}$ are not accounted for to make our method tractable. However, their contributions are considered at a higher level, say when computing the contribution of the secondary interaction B_1B_3 (See Figure 13c).

5.3 Data Structures

Before giving details on the used data structure representing secondary interactions, let us recall that the number of secondary interactions entailed by the subdivision of a patch (belonging to a curved surface) is equal to sixteen (See Figure 14): B_iB_j, i and j ranging from 1 to 4.

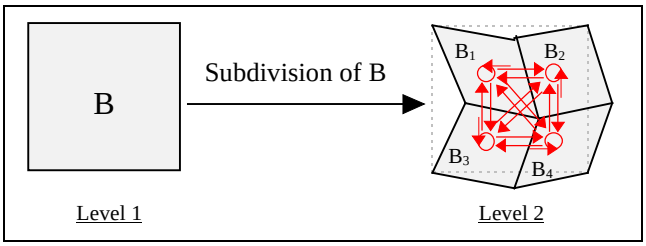

Figure 14 : Number of secondary interactions

The data structure associated with these interactions is a particular 16-Tree as shown in Figure 17. Indeed, as mentioned in the previous subsection, only the nodes B_iB_i (at subdivision level one), $B_{ij}B_{ij}$ (at subdivision level two) and so on, have descendants.

5.4 Refinement

The new refinement algorithm is the following:

```
Procedure RefineAndLink(Interaction IJ, Interaction JK, Float FFErr, Float FrErr,Float MinArea)
    {
    If ((φIJK is low) or (errors on formfactors and fr are low)) Then
        Link(IJ,JK);
    Else
        P=Patch_Inducing_Maximum_Error() ;
        Subdivide P;
        Subdivide IJ and/or JK according to P; (See Figure 5)
        If (P==J and P is a curved surface) Then CreateIndirectTransfer(IJ);
        Switch (P)
            {
            case I : RefineAndLink over children of IJ ;
            case J : RefineAndLink over children of IJ and JK ;
            case K : RefineAndLink over children of JK ;
            }
    EndIf
    }
```

Figure 15 : Refinement with extended three point transport

`CreateIndirectTransfer()` is the procedure which is in charge of creating the secondary interactions and the associated links (Figure 16).

```
CreateIndirectTransfer(IJ)
    {
    Let IJ be the initial primary interaction and JK the final primary interaction.
    Let J1, J2, J3 and J4 be the sub-patches of J.
    For each IJn Do
        For each Jm with m≠n Do
            /* Creation of the indirect transfer */
            Link IJn to JnJm.
            Link JnJm to JmK.
        EndOfFor
    EndOfFor
    }
```

Figure 16 : Creation of the indirect transfer

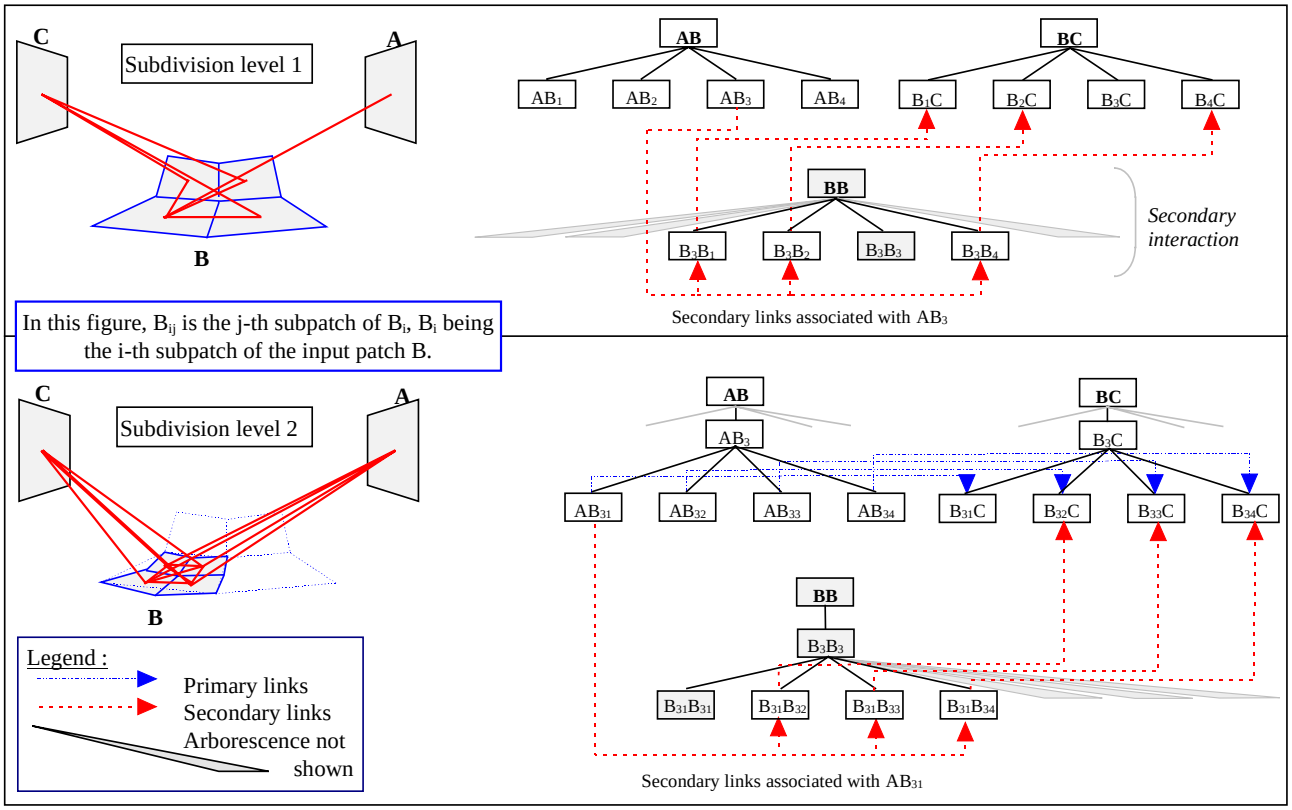

Figure 17 : Links between primary and secondary interactions after refinement.

5.5 Indirect Shooting

In case of secondary interactions the shooting algorithm is different from the one given in Figure 9.

To explain how the new shooting algorithm operates, let us consider Figure 18a. In this figure there are four patches A, B, C and D, B being a patch of a curved surface. The objective is to compute the energy transfers AB→BC and AB→BD, through the direct links (Figure 18b) and the indirect links due to B.

This amounts to compute the following indirect transfers: $AB_i \rightarrow B_iB_j \rightarrow B_jC$ and $AB_i \rightarrow B_iB_j \rightarrow B_jD$. We can remark that these two transfers have in common the transfers $AB_i \rightarrow B_iB_j$. This is why, for a reason of efficiency, we split the indirect shooting process into two steps: $AB_i \rightarrow B_iB_j$ (Figure 18c) and $B_iB_j \rightarrow B_jC$ (resp. $B_iB_j \rightarrow B_jD$) (Figure 18d).

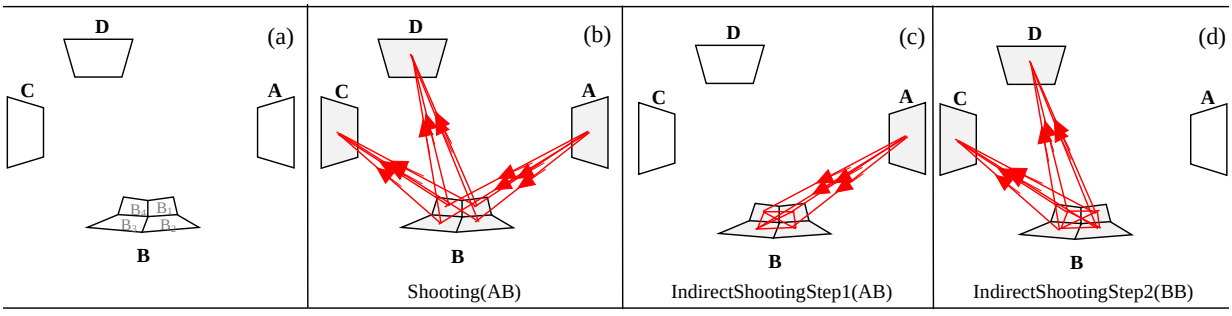

Figure 18 : Indirect shooting

The modified shooting is the following:

```
Shooting(IJ);

IndirectShootingStep1(IJ);

IndirectShootingStep2(JJ);
```

Figure 19: Shooting in presence of curved surfaces

5.6 Push/Pull

The pushing operation is the same as for planar subdivision, while the pulling is slightly different. Indeed, unlike Aupperle's method (Section 3.2.4), the form factor terms must appear in the expression of the pulled radiances so as to account for the orientations of the non-coplanar subpatches belonging to the same patch. In other words, the pulled radiances are given by:

$$L_{IJ} = \sum_{k=1}^{4} \frac{A_{I_k} . F_{JI_k}}{A_I . F_{JI}} L_{I_k J} \text{ or } L_{IJ} = \sum_{k=1}^{4} \frac{A_{J_k} . F_{J_k I}}{A_J . F_{JI}} L_{IJ_k}$$, depending on the subdivided patch, where IJ is a node of an interaction hierarchy and I_kJ (or IJ_k) are its descendants.

6 Results and Discussion

Our algorithm has been implemented and tested with three kinds of scenes and with a Cook's and Torrance's reflection model. Each photometric quantity, like flux and radiance, is defined for ten wavelengths (spectral approach). The rendering process computes for each pixel a spectral radiance, projects this latter into the XYZ color space, and finally converts the resulting components into RGB components.

Scene 1 (see Figure 21) is a room containing a specular cube. The front facing wall is glossy while the other walls as well as the ceiling and the floor are diffuse. Scene 2 (see Figure 20) is a glossy curved surface illuminated by a small light source placed above it. Scene 3 is a more complex scene containing a curved surface (curtain) and planar surfaces (see Figure 22). Figure 23 and Figure 24 correspond to images of scene 3 for different view parameters. Figure 23 shows the curved shape of the curtain while Figure 24 put an emphasis on glossy reflection effects. Note that there are no pure specular reflections but only diffuse and glossy reflections.

The characteristics of these scenes are summarized in Table 1. We call *potential elements* the elements resulting from the uniform subdivision of each input patch at the finest resolution. Such a fine meshing induces a set of interactions (between its elements) which will be called *potential interactions.*

Table 2 gives for each scene the number of interactions and links. Recall that our shooting algorithm maintains in memory only the links created at the current iteration. We can remark (Table 2) that the ratio of the maximum number of these links to the total number of created links (obtained after convergence) is equal to 5.77%, 0.84%, 6.44% for scene 1, 2 and 3 respectively. In addition, the number of interactions created by our hierarchical algorithm is far smaller than the number of potential interactions, even when a high precision is required (scene 2). To sum up, the results given by Table 2 demonstrates the efficiency of our algorithm in terms of memory saving.

Table 3 provides some results in terms of computing time. Before discussing these results let us define the convergence ratio (CR) as: $CR = \frac{IF - RF}{IF}$, where the initial total flux IF is the sum of the light powers emitted by all the light sources and RF the total residual flux after convergence. Even with a high convergence ratio (95%) the resolution times remain reasonable, which proves that our hierarchical algorithm is efficient.

To check the validity of the assumptions made by our algorithm when handling non-planar subdivisions, we have computed two images of scene 2 (with the same view parameters). The first one (Figure 20) has been computed by considering a coarse initial meshing (65 input patches) and non-planar subdivision, while the second has been generated with a very fine initial meshing (1025 input patches) and planar subdivision. The mean squared differences between the two normalized images for each component R, G, B and for the luminance Y are given in Table 4. We can see that these differences are insignificant even though the resolution process took 132 s for the first image and many hours for the second on a silicon graphics INDY R4600 (175 MHz).

	Number of input patches	Number of potential elements	Number of light sources
Scene 1	11	2 816	1
Scene 2	65	4 160	1
Scene 3	243	15 552	7

Table 1

	Number of interactions after convergence	Number of potential interactions	Maximum number of links stored in memory	Total number of links created after convergence
Scene 1	922	7 932 672	118	2 044
Scene 2	146 764	17 305 600	2 791	160 553
Scene 3	107 028	241 849 100	29 000	450 000

Table 2

	Resolution time	Rendering time (image 256 x 256)	Convergence ratio	Number of shooting iterations
Scene 1	63 s	407 s	95.11 %	50
Scene 2	132 s	448 s	94.70 %	100
Scene 3	758 s	780 s	90.20 %	150

Table 3

	R	G	B	Y
Mean squared difference	0.032776	0.009426	0.001416	0.016500

Table 4

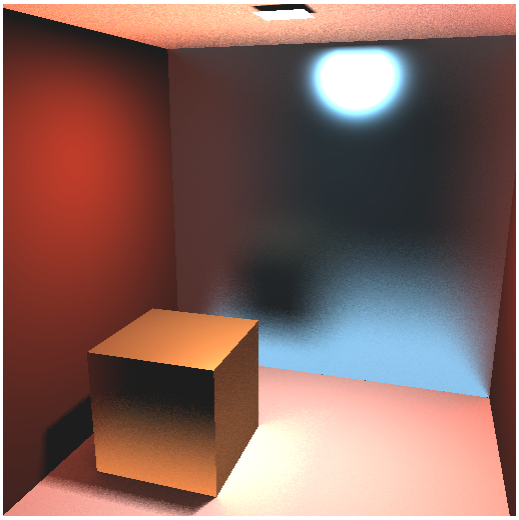

Figure 21: Scene 1

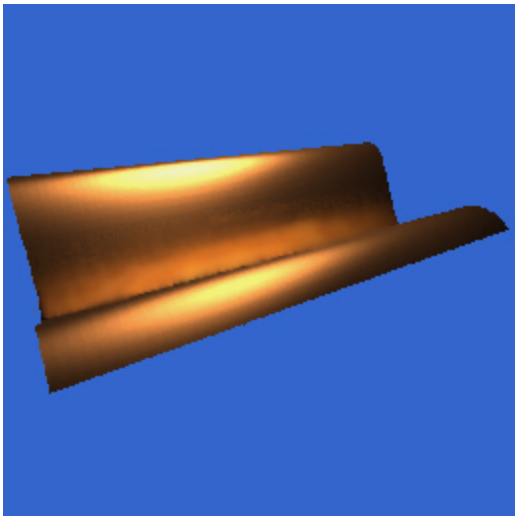

Figure 20: Scene 2

Figure 22: Scene 3

Figure 23: Scene 3 - Zoom 1

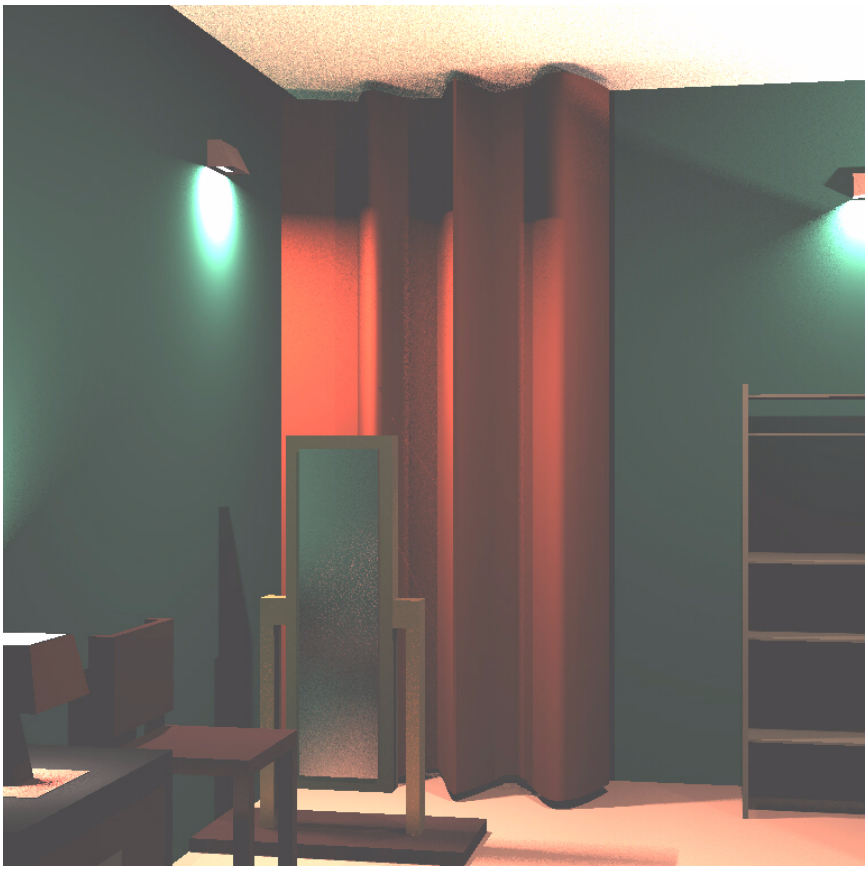

Figure 24: Scene 3 - Zoom 2

7 Conclusion

The global illumination algorithm presented in this paper, at the same time, extends the three point transport algorithm ([2],[3]) to non-planar surfaces, and brings improvements regarding the computation time and memory size required. The used shooting process reduces the spatial complexity to $O(n+k^2)$ where n is the number of elements at the finest level of subdivision and k the number of input patches. The other improvements concern the refinement criterion, the push/pull operation and the view independence of the algorithm.

As extension, our algorithm can handle non-planar subdivisions with a few extra computation and memory storage thanks to the new interaction data structure (with direct and indirect links) and the associated management mechanism.

References

[1] Airey J. M. (1990) Increasing Update Rates in the Building Walkthrough System with Automatic Model-Space Subdivision And Potentially Visible Set Calculations. *PhD Thesis of University of North Carolina.*

[2] Aupperle L. (1993) Hierarchical algorithms for illumination. *PhD thesis of Princeton university*.

[3] Aupperle L., Hanrahan P. (1993) A hierarchical illumination algorithm for surfaces with glossy reflection. *SIGGRAPH '93 Proceedings, 155-162*.

[4] Aupperle L., Hanrahan P. (1993) Importance and discrete three point transport. *Proceedings of 4th Eurographics Workshop on Rendering, 85-94*.

[5] Christensen P.H., Stollnitz E.J., Salesin D.H. DeRose T.D. (1994) Wavelet radiance. *Proceedings of 5th Eurographics Workshop on Rendering, 287-302*.

[6] Cohen M., Chen S.E., Wallace J.R., Greenberg D.P. (1988) A progressive refinement approach to fast radiosity image generation. *SIGGRAPH '88 Proceedings, 75-84*.

[7] Cook R., Porter T., Carpenter L. (1984) Distributed Ray tracing. *SIGGRAPH '84 Proceedings, 137-145*.

[8] Hanrahan P., Salzman D., Aupperle L. (1991) A rapid hierarchical radiosity algorithm. *SIGGRAPH '91 Proceedings, 197-206*.

[9] Immel D.S., Cohen M., Greenberg D.P. (1986) A radiosity method for non-diffuse environments. *SIGGRAPH '86 Proceedings, 133-142*.

[10] Kajiya J.T. (1986) The rendering equation. *Computer Graphics 20, 143-150.*

[11] Lafortune E.P., Willems D.Y. (1995) A 5D tree to reduce the variance of Monte Carlo ray tracing. *Proceedings of 6th Eurographics Workshop on Rendering, 147-162.*

[12] Pattanaik S., Bouatouch K. (1994) Haar wavelet : a solution to global illumination with general surface properties. *Proceedings of 5th Eurographics Workshop on Rendering, 273-286.*

[13] Pattanaik S., Mudur S. (1992) Computation of global illumination by Monte Carlo simulation of the particle model of light. *Proceedings of 3rd Eurographics Workshop on Rendering, 71-83.*

[14] Schröder P., Hanrahan P. (1994) Wavelet methods for radiance computations. *Proceedings of 5th Eurographics Workshop on Rendering, 303-311*.

[15] Sillion F., Puech C. (1989) A general two-pass method integrating specular and diffuse reflection. *SIGGRAPH '89 Proceedings, 335-344.*

[16] Sillion F.X., Arvo J.R., Westin S.H. Greenberg D.P. (1991) A global illumination solution for general reflectance distributions. *SIGGRAPH '91 Proceedings, 187-196.*

[17] Teller S., Fowler C., Funkhouser T., Hanrahan P. (1994) Partitioning and ordering large radiosity computations. *Computer Graphics Proceedings, Annual Conference Series, 443-450.*

[18] Tellier P., Bouatouch K. (1991) Global illumination models: implementation issues. *Proceedings of 2nd Eurographics Workshop on Rendering.*

[19] Wallace J.R., Cohen M. F., Greenberg D.P. (1987) A two pass solution to the rendering equation: A synthesis of ray-tracing and radiosity methods. *SIGGRAPH '87 Proceedings, 311-320.*

[20] Ward G.J., Rubinstein F.M., Clear R.D. (1988) A ray tracing solution for diffuse interreflection. *SIGGRAPH '88 Proceedings, 85-92.*