EUROGRAPHICS '98 / N. Ferreira and M. Göbel
(Guest Editors)

Volume 17, (1998), Number 3

A Vector Approach for Global Illumination in Ray Tracing

Jacques Zaninetti Xavier Serpaggi Bernard Péroche

Ecole des Mines de Saint-Etienne
Centre SIMADE, Laboratoire LISSE †
158 Cours Fauriel
42023 Saint-Etienne Cedex - France
email: [zaninetti,peroche]@emse.fr

Abstract

This paper presents a method taking global illumination into account in a ray tracing environment. A vector approach is introduced, which allows to deal with all the types of light paths and the directional properties of materials. Three types of vectors are defined: Direct Light Vectors *associated to light sources,* Indirect Light Vectors *which correspond to light having been diffusely reflected at least once and* Caustic Light Vectors *which are associated to light rays emitted by sources and reflected and/or transmitted by specular surfaces. These vectors are estimated at a small number of points in the scene. A weighted interpolation between known values allows to reconstruct these vectors for the other points, with the help of a gradient computation for the indirect component. This approach also allows to take uniform area light sources (spherical, rectangular and circular) into account for all the types of vectors. Computed images are thus more accurate and no discretizing of the geometry of the scene is needed.*

1. Introduction

High quality physically-based rendering algorithms are intended to generate realistic images by accurately and efficiently simulating global illumination effects, i.e. illumination of surfaces in a scene by sources and other surfaces. In order to solve this problem, Heckbert proposed in [1] the concept of *light path*, which allows to characterize the paths followed by light. A grammar is defined by the four symbols L (Light source), E (Eye), D (Diffuse interaction) and S (Specular interaction). Each path can therefore be labelled with some string in the set given by the regular expressions $L(D|S)^*E$. Standard ray tracing simulates only LDS^*E or LS^*E paths, while classic radiosity simulates only LD^*E paths. Though complementary, these two methods don't solve all the cases. In particular, paths such as $LD(D|S)^*E$, corresponding to indirect illumination, and paths such as $LS^*D(D|S)^*E$, corresponding to caustics, are not taken into account.

One of the aims of this work is to propose a method allowing to deal with LDS^*E, LS^*E, LD^*E and $LS^*D(D|S)^*E$ paths. To reach this purpose, we shall introduce a vector approach providing a unifying framework for all the types of paths and allowing to take directional properties of materials into account. We shall define the concept of *Light Vector*, inspired both by Arvo's *Irradiance Vector* [2] and by Jensen's *Photon Map* [3]. To permit an accurate simulation of all luminous phenomena, three types of Light Vectors are introduced: *Direct Light Vectors* describe the energy coming directly from light sources; *Caustic Light Vectors* are associated to light rays emitted by sources and reflected and/or transmitted by specular surfaces; finally, *Indirect Light Vectors* represent light having been diffusely reflected at least once before hitting the current point.

To reach our goal, we chose ray tracing as our basic algorithm. Ray tracing [4] samples light by tracing rays from the eye point into the scene, recursively exploring specularly reflected and transmitted directions, and tracing rays towards point light sources to simulate shadowing. Thus standard ray tracing can generate images with shadows, specular surfaces and transparency. However, the treatment of diffuse interreflections between surfaces is approximated by a constant ambient term, which does not fit any physical law.

† Supported in part by Région Rhône-Alpes in the framework of the ACTIV project.

Cook [5] improved standard ray tracing by distributing rays, either uniformly or stochastically, using any integration-based distribution. This method was later extended to path tracing by Kajiya [6]. Ward et al. [7] improved on this by devising a coherence method based on the fact that the amount of diffusely reflected light received by a surface changes gradually over the surface. Storing sampled values at the first sample points, and reusing them for later samples in the same region of the scene considerably reduces the sampling effort. This method led to the well-known *Radiance* rendering system [8]. In [9], Arvo introduced *Backward ray tracing*, which is based on sending rays from light sources, thus re-creating the natural progression of a light ray. During the first pass, power emitted by light sources is stored on diffuse surfaces, after paths such as LS^*D. This power is collected during the next pass, a standard ray tracing from the eye, which gives paths such as DS^*E. Grouping together these two paths gives paths such as LS^*DS^*E. These paths correspond to the concentration of light by specular or refracting objects on the diffuse component of other surfaces, and create caustics.

The remainder of the paper is organized as follows. Section 2 presents basic definitions in the domain of illumination simulation. Section 3 introduces our fundamental concept, the Light Vector. The direct component of this vector is studied in section 4, the caustic component in section 5 and the indirect component in section 6. Some results are discussed in section 7. A conclusion and several further developments are given in section 8.

2. Illumination Concepts

In this section, we shall briefly present the main concepts used in our work and we shall define the notations.

Radiance $L(x,\vec{\omega})$ [10] (expressed in $Wm^{-2}sr^{-1}$) is, for a given point x, the power per unit projected area perpendicular to vector $\vec{\omega}$ per unit solid angle in direction $\vec{\omega}$.

The concept of *bidirectional reflection distribution function* (BRDF, expressed in sr^{-1}) is well known to model reflections on materials:

$$f_r(x,\vec{\omega}_i \rightarrow \vec{\omega}_r) = \frac{dL(x,\vec{\omega}_r)}{L(x,\vec{\omega}_i)\cos\theta_i\, d\omega_i}$$

BRDF's also depend upon wavelength, even if, for writing convenience, this parameter does not explicitly appear in the notation.

In this work, we shall assume that the BRDF can be split into a purely specular component $f_{r,s}$ and another component $f_{r,d}$ which includes a diffuse component and a specular lobe component [11]:

$$f_r(x,\vec{\omega}_i \rightarrow \vec{\omega}_r) = f_{r,s}(x,\vec{\omega}_i \rightarrow \vec{\omega}_r) + f_{r,d}(x,\vec{\omega}_i \rightarrow \vec{\omega}_r) \quad (1)$$

We implemented Schlick's analytic model [12], because it is both physically plausible and inexpensive. But any other model fulfilling the physical laws of reflection may be used, without modification.

Irradiance $E(x)$ (expressed in Wm^{-2}) gathers the whole energy on a given point x, taking the incident angle, but not the BRDF, into account.

$$E(x) = \int_\Omega L_i(x,\vec{\omega}_i)\cos\theta_i\, d\omega_i \quad (2)$$

The *rendering equation* stems from the formulation by Kajiya [6]. This equation provides the best foundation for building a physically-based rendering system.

$$\begin{aligned} L_r(x,\vec{\omega}_r) &= L_e(x,\vec{\omega}_r) \\ &\quad + \int_\Omega f_r(x,\vec{\omega}_i \rightarrow \vec{\omega}_r)\, L_i(x,\vec{\omega}_i)\cos\theta_i\, d\omega_i \\ &= L_e(x,\vec{\omega}_r) + L_{i,r}(x,\vec{\omega}_r) \end{aligned} \quad (3)$$

The obvious simplicity of this formulation obscures the great complexity of simulated phenomena. Recursiveness, the need to evaluate the integral at every surface point and an only partial knowledge of the main terms appearing in the formula, practically forbid any analytic solution, even for very simple scenes.

3. A Vector Approach

3.1. Purpose

The computation of a picture amounts to evaluating rendering equation (3) at each point in the scene, a very lenghty process. To reduce computation time, we propose a method based on a homogeneous principle stating that the probability of having a radiance discontinuity is rather small. Thus, when radiance is known at a given point, there is every chance that radiance at close locations is very similar. Our method reduces complete calculation to a minimum, by reusing already known values in the neighbourhood of the given point. The main problem is to accurately detect zones with discontinuities or with high variations: boundaries between objects, transitions between light and shadow, curvature changes, etc. Indeed, these zones very often carry fundamental information about the meaning of the picture, and must be described with accuracy.

It is not conceivable to keep all the results obtained when equation (3) is evaluated, for memory ocupation reasons as well as for efficient search for neighbourhoods in space. So we propose a representation suited to this problem, the *Light Vector*. This vector puts together all the minimal geometric and radiometric information allowing a correct reconstruction.

3.2. Radiance Components

The reflected radiance may be split into four components, which will be investigated separately because their different behaviours allow suited optimizations.

$$L_{i,r}(x,\vec{\omega_r}) = L_{dir}(x,\vec{\omega_r}) + L_{caust}(x,\vec{\omega_r}) + L_{ind}(x,\vec{\omega_r}) + L_{spec}(x,\vec{\omega_r})$$

$L_{dir}(x,\vec{\omega_r})$ is the direct component. It is associated to light coming directly from sources without having been reflected or transmitted;

$L_{caust}(x,\vec{\omega_r})$ is the caustic component, which has only been subjected to specular reflections or transmissions;

$L_{ind}(x,\vec{\omega_r})$ is the indirect component. It is linked to the light that has been subjected to at least a non specular reflection;

$L_{spec}(x,\vec{\omega_r})$ is the specular component. It is computed by a classical ray tracer, in which reflected and refracted rays are recursively cast when necessary and may themselves appeal to the three other components.

3.3. The Light Vector

The *Light Vector* represents the luminous incident energy at a given point in the scene, while restricting the amount of information to be stored. The knowledge of this vector allows to find the correct value of the reflected radiance again, for each point at which it is computed.

This concept is derived from the *Irradiance Vector* investigated by Arvo [2] to model partially occluded polyhedral sources. However, the material and the sighting direction have to be taken into account for the computation of this Light Vector. Our goal is to restrict reconstruction errors in the reflected direction, i.e. for the light discerned by the eye. Such a description, which is independent of the radiance components, gives a really homogeneous representation for global illumination.

The Light Vector is composed of two fields: an average incident direction $\vec{D}$ and an energy magnitude P (expressed in Wm^{-2}). P is computed with the help of equation:

$$P\, f_{r,d}(x,\vec{D}\rightarrow\vec{\omega_r}) = \int_{\Omega} f_{r,d}(x,\vec{\omega_i}\rightarrow\vec{\omega_r}) L_i(x,\vec{\omega_i}) \cos\theta_i \, d\omega_i \tag{4}$$

The Light Vector has $n+3$ components, where n depends on the representation model for colors ($n = 3$ in the RGB space, much more with a spectral model). This vector may be seen as a virtual point light source simulating the effect of the whole incident energy at point x.

When it is unknown, an approximate value of this vector may be deduced from already computed values by a vector interpolation (and not a scalar one, like previous methods). By using additional parameters, the interpolation may be refined and validity bounds may be set. By taking BRDF f_r and a privileged direction $\vec{D}$ into account, errors made are noticeably reduced.

In the following, we shall distinguish between three types of Light Vectors, in accordance with the radiance component used (cf. section 3.2): $\overrightarrow{DLV}$ for direct component $L_{dir}(x,\vec{\omega_r})$, $\overrightarrow{CLV}$ for caustic component $L_{caust}(x,\vec{\omega_r})$, and $\overrightarrow{ILV}$ for indirect component $L_{ind}(x,\vec{\omega_r})$.

3.4. Computation of a Picture

3.4.1. Creating a Seed

Standard row by row computation of a picture could introduce significant errors: interpolation between known values would systematically ignore half the neighbourhood of the current point. To avoid this problem, a seed of randomly distributed points in the scene is chosen. For each such point, direct and indirect components of Light Vectors are computed and stored. The storage of these vectors is made with kd-trees [13], which allow to perform efficient search of space neighbourhoods. Several distinct kd-trees are used: one for the $\overrightarrow{ILV}$s, one for the $\overrightarrow{CLV}$s and, for each light source, one for the corresponding $\overrightarrow{DLV}$s.

3.4.2. Traversing the Set of Pixels

The second step is a usual traversal of the picture, with possible generation of reflected and refracted rays. The main difference in relation to standard ray tracing is the search, in some neighbourhood of the current point, of known values for direct and indirect radiances in their respective kd-trees. When it is possible, the result of the interpolation is used in place of a complete calculation. When a complete calculation is made, the result is stored in vector form and may be used for later interpolations.

$\overrightarrow{DLV}$s, $\overrightarrow{CLV}$s and $\overrightarrow{ILV}$s are used in agreement with the same general principle. But the estimation of the validity of the interpolation is noticeably different because available data are more or less complete. We shall present these various methods in the three next sections.

4. Direct Component

In this section, we present the use of the Light Vector in the case of direct illumination. This method is inspired by Jensen's *photon map* [3] and adapted to our vector model. It is especially beneficial when complex light sources, requiring a costly sampling process, are used.

4.1. The Direct Light Vector

The *Direct Light Vector*, denoted $\overrightarrow{DLV}$, represents, at a given point in the scene, the energy supplied by a given light source. It is composed of two fields:

- An average incident direction $\vec{D}$ computed by weighting the n visible sampling directions $\vec{\omega}_i$ of the light source by incident radiance L_i and by the projected solid angle Ω_i associated to $\vec{\omega}_i$:

$$\vec{D} = \frac{\sum_{i=1}^{n} L_i \Omega_i \vec{\omega}_i}{\sum_{i=1}^{n} L_i \Omega_i} \tag{5}$$

- An energy magnitude P satisfying equation (4).

This computation is easily made when shadow rays are cast towards the light source. In particular, it is reduced to its simplest form when dealing with a point light source. The $\overrightarrow{DLV}$ is stored even when the light source is completely occluded, in order to reduce the number of rays to be generated later.

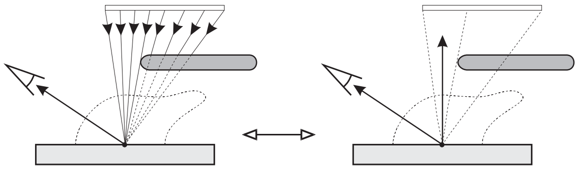

Figure 1: *The concept of direct light vector.*

4.2. Direct Interpolation

4.2.1. Search for Neighbours

A set of usable and already calculated $\overrightarrow{DLV}$s is chosen before making the interpolation. This set is made up of the *nmax* nearest neighbours of the current point, located inside a neighbourhood of radius *rmax*. This set is rejected if the number of $\overrightarrow{DLV}$s found is less than some threshold *nmin*. In this case, a full calculation is started.

The choice of these three parameters is mainly dependent on the size of the scene. In practice, for indoor scenes, $nmin = 5$, $nmax = 50$ and $rmax = 0.08m$ give good results with an initial seed of 5000 $\overrightarrow{DLV}$s.

4.2.2. Validity Test

Dispersion between suitable candidates for interpolation must be beyond a given threshold. As the direct component may have great changes, dispersion is measured by calculating the variance of the set of brightness values ($l = 0.299r + 0.587g + 0.114b$ in the RGB color space) of the samples. In practice, a low threshold of 20% is used, which is weak enough to detect quick changes for direct irradiance.

In fact, this method allows to more or less compare the Y component of the vectors in the XYZ color space. The good solution for our problem would be to work with a spectral model in a color space provided with a perceptual metric. But this problem still is a current research theme.

4.2.3. Interpolation

The value of a $\overrightarrow{DLV}$ is obtained by a weighted interpolation of the directions $\vec{D}_i$ and of the energy magnitudes P_i of the selected sampled vectors. The weighting function f is a tight polynomial approximation of a Gaussian:

$$f(d_i) = [2(\frac{d_i}{rmax})^3 - 3(\frac{d_i}{rmax})^2 + 1]^2 \tag{6}$$

where d_i is the distance between the current point and sample i.

$$\vec{D} = \frac{\sum_{i=1}^{n} f(d_i)\vec{D}_i}{\sum_{i=1}^{n} f(d_i)} \quad \text{and} \quad P = \frac{\sum_{i=1}^{n} f(d_i)P_i}{\sum_{i=1}^{n} f(d_i)}.$$

For each light source in the scene, the value of the direct component $L_{dir}(x, \vec{\omega}_r)$ of the reflected radiance $L_{i,r}(x, \vec{\omega}_r)$ is given by equation (4). The direct component of the reflected radiance at point x is obtained by adding all the components associated to each light source.

4.3. Uniform Specific Sources

4.3.1. Spherical Sources

Spherical sources are a simple way to simulate bulbs. In relation to point light sources, they produce penumbra zones, increasing the realism of the images. Our vector method reduces the overhead due to the greatest number of generated rays.

Spherical sources are simulated by N point light sources randomly located inside a sphere. This simulation is plausible only when the radius of the sphere is small compared to the other distances in the scene. The solid angle subtented by a sphere with radius R located at distance d from point x is $\Omega = \frac{2\pi R^2}{d^2}$. This angle is equally distributed among the N point sources, so $\Omega_i = \frac{1}{N}\frac{2\pi R^2}{d^2}$.

Furthermore, the energy is supposed to be uniformly distributed in all directions, in order to associate a uniform radiance to the surface when it is hit by a specular ray or when it is directly seen by the observer. Thus $L = \frac{I}{4\pi R^2}$, where I is the intensity (in $W sr^{-1}$) that would have a unique point light source with the same total power. The direction and energy magnitude of the $\overrightarrow{DLV}$ is computed with equations (5) and (4).

4.3.2. Rectangular Area Sources

Area light sources are seldom used in ray tracing, because they require a very dense sampling to avoid excessive aliasing problems (zoning in penumbra, for example). As this sampling problem is irrelevant when the source is completely visible or completely occluded, we propose an adaptive subdivision method which decreases the number of rays cast. Associated to our vector representation, it allows a notable cut of computational time.

The rectangular source is subdivided in relation to a bintree. The choice to whether subdivide or not a rectangular zone is guided by the following heuristic: if the solid angle subtented by the rectangle is greater than a given threshold, then the edge of the rectangle which has the greatest apparent angle is subdivided into two parts by cutting the apparent angle into two equal angles. The rectangle is then subdivided into two sub-rectangles from the new point, and the process is recursively repeated. The condition on the solid angle corresponds to a threshold on the minimal size of small objects we accept not to detect.

The solid angle subtented by a polygon with N vertices may be computed analytically by taking the area of the projection of this polygon on a unit sphere [14]:

$$\Omega = \left[\sum_{i=1}^{N} \cos^{-1}\left(\frac{(\vec{u_i} \wedge \overrightarrow{u_{i+1}}) \cdot (\vec{u_i} \wedge \overrightarrow{u_{i-1}})}{\| \vec{u_i} \wedge \overrightarrow{u_{i+1}} \| \, \| \vec{u_i} \wedge \overrightarrow{u_{i-1}} \|}\right)\right] - (N-2)\pi$$

where $\vec{u_i}$ is the vector joining current point x to the ith vertex of the polygon.

After this subdivision step, all the rectangles are traversed. If all the vertices of a rectangle are visible, then Arvo's *Irradiance Vector* [2] is taken as sampling direction for this rectangle. If the vertices of the rectangle are not all visible, then the rectangle is subdivided into four sub-rectangles with the same size, and the process is recursively repeated until a given recursivity level is reached.

4.3.3. Circular Area Sources

The circular area is included into a rectangular bounding zone. The process described in section 4.3.2 is applied with only one additional test on the vertices of the rectangles. Each vertex is checked to be inside or outside the circular area. If it is outside, then it is marked "occluded".

In fact, the previous method may be applied to any area source. The shape of the area is taken as a visibility mask for the vertices of the rectangles obtained by the subdivision process of section 4.3.2.

5. Caustics

5.1. Introduction

Caustics are due to the concentration of light by specular or refracting objects on the diffuse component of other surfaces, after light paths such as LS^*DE. Starting from the eye point, classical ray tracing is unable to simulate such a kind of path. Actually, it is not possible to determine in advance the set of the paths ending in a source after having been subjected to deviations. The search of Fermat's paths [15] is practically usable only for very simple geometric shapes.

A solution to this problem is *Backward ray tracing* [9], which is based on sending of rays from light sources, thus re-creating the natural progression of a light ray. During the first pass, power emitted by light sources is stored on diffuse surfaces, after paths such as LS^*D. This power is collected during the next pass, a standard ray tracing from the eye, which gives paths such as DS^*E. Grouping together these two paths gives paths such as LS^*DS^*E.

Arvo suggests to store information coming from the first pass in a fixed structure, the *Illumination Map*, which is a regular grid mapped like a texture on diffuse surfaces. This heavy structure requires a parametrization of surfaces, which is difficult to be made compatible with a CSG model for scenes. Furthermore, complete features of BRDF are not taken into account, and caustics only appear on perfectly lambertian surfaces. The *Photon Map*, introduced by Jensen [3], solves these problems by using an adaptive data structure (a kd-tree), independent of the geometry of the scene.

Our reconstruction method uses density estimation with a kernel, like Collins [16], but takes the complete BRDF into account, like Jensen. Furthermore, area light sources may produce caustics in our method.

5.2. The Caustic Light Vector

5.2.1. Computation

The Caustic Light Vector $\overrightarrow{CLV}$ is similar to the concept of *photon* from Jensen. However we keep our notation, because it puts the homogeneity of the vector representation of the various light components forward. Direction $\vec{D}$ of a $\overrightarrow{CLV}$ is the incident direction of the last part of the light path going from the source to the current point. Energy magnitude P is equal to the power of the source divided by the total number of rays cast. If the ray has been subjected to reflections or to refractions, P is softened by component $f_{r,s}$ of the materials met.

5.2.2. Reconstruction

The reconstruction of the caustic component is not an interpolation, but a density estimation. The whole incident energy must be collected in a neighbourhood of the current point. The choice of the size of the research zone determines the smoothing effect that removes the noise due to the discretization. If this zone is too small, some noise persists. But if it is too large, the high frequencies of the caustic disappear. The zone size must be at least equal to the projection of a pixel onto the surface.

An estimation kernel f provides the continuity of the reconstruction of the density function. Like Collins, we use a kernel with a gaussian appearance. Thus the caustic component of the reflected radiance becomes:

$$L_{caust}(x, \vec{\omega_r}) = \sum_{i=1}^{n} f(d_i) f_r(x, \vec{\omega_i} \rightarrow \vec{\omega_r}) P(x_i)$$

with $f(d_i)$ given by equation (6), where d_i is the distance between points x and x_i, and $rmax$ is the size of the research zone. A value of about some millimeters is used for simulations of indoor scenes.

5.3. Adaptive Sampling of Sources

5.3.1. Point Light Sources

Only a ray hitting a specular or a refractive object may lead to the creation of a $\overrightarrow{CLV}$, if the generated path hits a diffuse surface afterwards. In order to limit the total number of generated rays, a first rather rough pass determines the zones of the scene which are visible from the source and which contain specular objects, as done by Shirley [17]. These zones are then recursively refined up to a given accuracy (a threshold on the maximal number of rays). Furthermore, these zones are slightly extended towards neighbouring zones in order to limit the risks of missing small specular objects.

5.3.2. Area Light Sources

Area light sources may also create caustics, which are more difficult to compute. Thanks to our method, a systematic sampling of the sources may be avoided.

The area light source is located inside a bounding sphere, whose center is the center of gravity of the source. This sphere is then sampled as a point light source, and rays are cast from its center in all the directions. However, all the possible intersections between a ray and the scene are taken into account, even when a diffuse object is hit. The visibility of the source is tested each time a specular or refractive object is found in the list of intersection points, even if an opaque object has been hit before. Indeed, diffuse objects may be only partial occluders, a fact that cannot be determined by the only initial ray.

The corresponding $\overrightarrow{DLV}$ is then computed. If its energy magnitude P is non null, the source is at least partly visible and the $\overrightarrow{DLV}$ may be used as a starting point for a classical specular path with initial direction $\vec{D}$, which will give rise to a $\overrightarrow{CLV}$. Let us notice that this method is entirely compatible with the adaptive sampling of area light sources presented in section 4.3.

6. Indirect Component

In this section, we shall explain how the indirect component of the reflected radiance - denoted $L_{ind}(x, \overrightarrow{\omega_r})$ in section 3.2 - is computed.

6.1. General Principle

The indirect component of reflected light is computed with a Monte Carlo method, by sampling an hemisphere centered on the current point. Only light having been reflected at least once is taken into account here. Power coming directly from light sources is evaluated with L_{dir}. Because of the recursive formulation of equation (3) and of the great number of rays to be generated in order to reduce the variance, the computation time is very long.

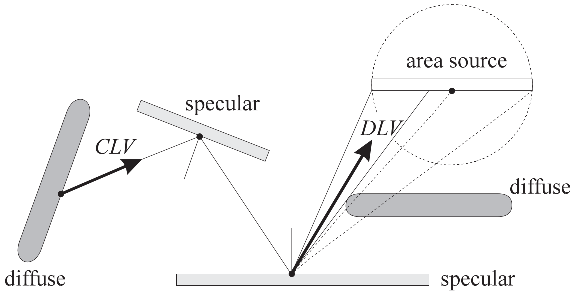

Figure 2: *The handling of area light sources for caustic light vectors.*

The following assumption is usually admitted in computer graphics [8]: *Indirect illumination changes gradually over surfaces.* Accordingly, we may consider that around every point in the scene, there exists a neighbourhood in which the changes of indirect illumination are less than a given threshold. Then, it is possible to reconstruct the value of a given point from the values of near points, provided the valid neighbourhood of the latter contains the former.

An irradiance gradient is evaluated at the same time as the indirect component, at almost no additional cost. This gradient is not accurate enough to be used outside of a small neighbourhood of the given point, because of the too many unknown data. But it is a very good gauge of the degree of perturbation of the current zone. Thus, it is useful to refine the computation of the validity of the zone.

Before doing a complete computation, we must check if there exist already computed valid values. In this case, the current value is estimated by a simple weighted interpolation of known values. The weight assigned to each known value is inversely proportional to the error that may be introduced. This error estimation is mainly dependent on the distance between the samples, the change in surface orientation, and the gradient. In the opposite case, a Monte Carlo computation is done and the obtained value is stored, in order to be re-used, if necessary.

6.2. Monte Carlo Method

6.2.1. Discretization

To make our computation, the hemisphere is discretized into cells. Each cell of the hemisphere has a constant solid angle. In this way, we may access explicitly to the directional information for the gradient computation. Angles θ_j and ϕ_k are chosen in such a way that they define a random direction inside each cell (j, k) of the hemisphere. The solid angle subtended by each cell is $\omega_{j,k} = \frac{2\pi}{MN}$, and $\theta_j = cos^{-1}\left(\frac{M-j-X}{M}\right)$, $\phi_k = \frac{2\pi}{N}(k+Y)$, where X and Y are uniform random variables in $[0,1[$, M is the number of samples according to the polar angle and N the number of samples according to the azimuthal angle.

So, after discretization of the hemisphere, rendering equation (3) becomes:

$$L_{ind}(x,\vec{\omega_r}) \approx \frac{2\pi}{MN}\sum_{j=0}^{M-1}\sum_{k=0}^{N-1} f_{r,d}(x,\vec{\omega_{j,k}}\rightarrow\vec{\omega_r})L_i(x,\theta_j,\phi_k)\cos\theta_j \qquad (7)$$

This sampling method is recursively used, but with a small predefined level of recursiveness.

6.2.2. Average Incident Direction

Vector $\vec{D}$ shows the average incident direction at point x: $\vec{D} = \int_\Omega L_i(x,\vec{\omega_i})\,\vec{\omega_i}\cos\theta_i\,d\omega_i$.

The computation of $\vec{D}$ may be done in the same loop iteration than (7), for a small cost. To facilitate the interpolation, this vector is normalized before being stored, only its direction being useful for us.

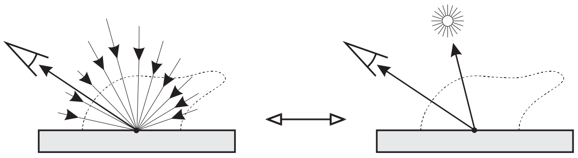

Figure 3: *The concept of indirect light vector.*

6.2.3. Irradiance Gradient

A gradient computation is made when rendering equation (7) is evaluated. As these calculations are done during the same iteration of the loop, the extra computation time is negligible. The gradient is different from the one introduced by Ward and Heckbert [18], because we do not use the same sampling of the hemisphere.

The gradient shows indirect illumination variations in a neighbourhood of the given point. Its numerical value gives a good indication of the degree of pertubation of the zone taken into account. It is used to associate, to each $\overrightarrow{ILV}$, a validity zone inside which interpolation is possible.

The gradient is obtained by deriving irradiance expression (2) according to displacements in a (x,y,z) coordinate system tangent to the surface:

$$\begin{aligned}\frac{\partial E}{\partial a} &= \frac{2\pi}{MN}\sum_{j=0}^{M-1}\sum_{k=0}^{N-1}\frac{\partial(L_i(\theta_j,\phi_k)\cos\theta_j)}{\partial a}\\ &= \frac{2\pi}{MN}\sum_{j=0}^{M-1}\sum_{k=0}^{N-1}\left[\frac{\partial L_i(\theta_j,\phi_k)}{\partial a}\cos\theta_j\right.\\ &\qquad\left. +\ L_i(\theta_j,\phi_k)\frac{\partial\cos\theta_j}{\partial a}\right]\end{aligned}$$

with $a \in (x,y,z)$.

However, $\frac{\partial L_i(\theta_j,\phi_k)}{\partial a}$ is unavailable. Thus, we must assume that displacements in the neighbourhood of point x are small enough to admit $\frac{\partial L_i(\theta_j,\phi_k)}{\partial a} = 0$. With this assumption:

$$\frac{\partial E}{\partial a} \approx \frac{2\pi}{MN}\sum_{j=0}^{M-1}\sum_{k=0}^{N-1} -L_i(\theta_j,\phi_k)\sin\theta_j\frac{\partial\theta_j}{\partial a}$$

$\frac{\partial\theta_j}{\partial a}$ may be computed by examining the changes of incident angle for small displacements of the center of the hemisphere: $\frac{\partial\theta_j}{\partial x} = \frac{-\cos\theta_j\cos\phi_k}{R}$, $\frac{\partial\theta_j}{\partial y} = \frac{-\cos\theta_j\sin\phi_k}{R}$, and $\frac{\partial\theta_j}{\partial z} = \frac{\sin\theta_j}{R}$, where R is the distance between the current point and the first object hit in direction (θ_j,ϕ_k).

6.3. The Indirect Light Vector

As the interpolation process is a little more sophisticated for Indirect Light Vectors than for Direct Light Vectors, the Indirect Light Vector has more additional fields: the gradient, the tangent coordinate system to the surface, and the arithmetic average distance to the other objects in the scene.

These data are used to control interpolation errors, and thus allow to reduce the total number of $\overrightarrow{ILV}$s computed with the expensive Monte Carlo method.

6.4. Validity Zone

Let B be a point in which $\overrightarrow{ILV}(B)$ must be computed. To be accepted as usable at point B, $\overrightarrow{ILV}(A)$ computed at point A must satisfy several conditions:

- A and B must belong to objects having the same BRDF. This condition is due to the fact that the $\overrightarrow{ILV}$ is used through the BRDF of the material;
- Distance $\parallel\overrightarrow{AB}\parallel$ must be less than or equal to the mean arithmetic distance *dmean* between A and the other objects in the scene. We impose this condition in order to prevent errors between close objects;
- The relative change of irradiance due to the gradient must be less than or equal to some user-specified accuracy tolerance S_E: $\frac{\vec{AB}.\vec{\Delta E}}{E_A} \leq S_E$;
- The surface curvature must be less than or equal to some predefined accuracy tolerance S_{curv}: $\cos(\vec{N_A},\vec{N_B}) \geq S_{curv}$. This condition is warranted by the fact that $\vec{D}$ shows the average incident direction at point x coming from a hemisphere, and not from a sphere.

To fulfill the conditions above, $\overrightarrow{ILV}$s are stored in kd-trees, with one kd-tree associated to all the objects in the scene having the same BRDF.

6.5. Weighted Interpolation

The number n of $\overrightarrow{ILV}$s used for interpolation must range between two predefined bounds, *nmin* and *nmax*. The choice of these bounds influences the filtering level induced by the interpolation. In practice, empirical values $nmin = 3$ and $nmax = 16$ give good results.

The goal of the weighting function is to give less importance to the samples liable to introduce accuracy errors, following the criteria defined in section 6.4. A weight w is associated to each $\overrightarrow{ILV}$; its value is one when point B coincides with point A, less than one otherwise:

$$w = \frac{1}{1 + \frac{\|\overrightarrow{AB}\|}{dmean}} \cdot \frac{1}{1 + \frac{\overrightarrow{AB}.\overrightarrow{\Delta E}}{E_A}} \cdot \frac{\cos(\overrightarrow{N_A}, \overrightarrow{N_B}) - S_{curv}}{1 - S_{curv}} \quad (8)$$

A standard interpolation scheme is used, with the weights computed in equation (8):

$$\overrightarrow{ILV}(B) = \frac{\sum_{i=1}^{n} w_i \, \overrightarrow{ILV}(A_i)}{\sum_{i=1}^{n} w_i}$$

As the maximum number, *nmax*, of samples is small, this computation is fast.

7. Some Results

This method was implemented in *YART*, the ray tracing software developed at Ecole des Mines de Saint Etienne by Marc Roelens [19].

Figures 4, 5 and 6 show the results of applying the method described in this paper.

Figure 4 shows a fluted glass ring (with a refractive index of 1.2) with many caustic effects. For a 400×400 pixels picture size, the computing time is about 90 minutes on a 200 Mhz MIPS R10000 processor, with about 30000 $\overrightarrow{CLV}$s, corresponding to 1 MB of memory.

Figure 5 is a "Cornell box" with a spherical light source. The cube on the left is specular, while the right one is diffuse. The transparent sphere produces a first caustic which is reflected in the specular cube. The reflection of the light source on the specular cube also produces a caustic after refraction through the sphere. For a 512×512 pixels picture size, the computing time is about 13 minutes, with about 2800 $\overrightarrow{ILV}$s, 15000 $\overrightarrow{DLV}$s and 20000 $\overrightarrow{CLV}$s. For this picture, we use approximately 0.3 MB of memory for the $\overrightarrow{ILV}$s, 0.7 MB for the $\overrightarrow{DLV}$s, and 0.7 MB for the $\overrightarrow{CLV}$s.

Figure 6 shows an indoor scene, with 3D (wood and marble) and 2D (the painting) textures. There is a mirror hung up on a wall and many shiny surfaces, and a rectangular area light source. This scene includes about 500 objects. The picture size is 720×540 pixels and the computing time is about 300 minutes, with an adaptive antialiasing process and the computation of about 20000 $\overrightarrow{DLV}$s, 10000 $\overrightarrow{ILV}$s, and 24000 $\overrightarrow{CLV}$s.

Figure 4: *A fluted glass ring with caustic effects.*

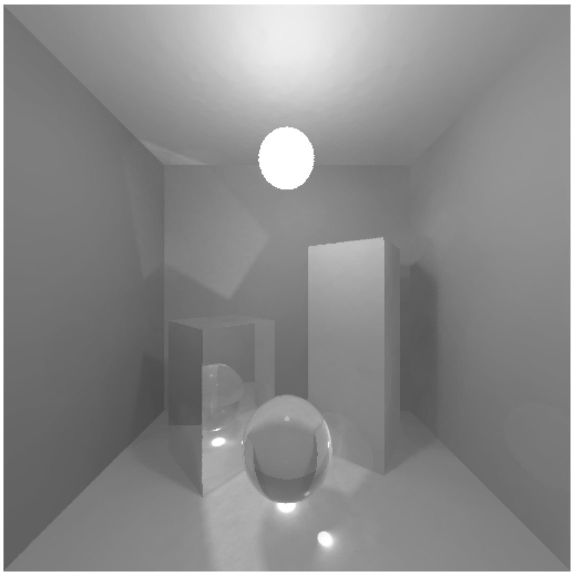

Figure 5: *"Cornell box" with caustic effects.*

Figure 6: *An indoor simulation.*

8. Conclusions and Future Work

In this paper, we have presented a vector approach supplying a unifying framework that allows to process the various types of light paths that must be taken into account to solve the rendering equation. Our method, which may be seen as a two-pass method, has been developed in a ray tracing environment.

The pictures obtained appear to be realistic, with reasonable computation times. However, we plan to work on the following points, to improve our results:

- replacement of the Monte Carlo method used for the computation of the Indirect Light Vectors by an importance sampling, in order to obtain an unbiased method, and to reduce the variance of the computation;
- use of a spectral model for colors when the RGB model is too rough for the requested accuracy;
- estimation of the error made during the computation;
- incremental and adaptive improvement of the accuracy, without losing already computed data [20].

References

1. P. S. Heckbert, "Adaptative radiosity textures for bidirectional ray tracing", *Computer Graphics*, **24**(4), pp. 145–154 (1990).

2. J. Arvo, "The irradiance jacobien for partially occluded polyhedral sources", *Computer Graphics*, **28**, pp. 343–350 (1994).

3. H. W. Jensen, "Global illumination using photon maps", *Seventh Eurographics Workshop on Rendering, Porto*, pp. 22–31 (1996).

4. T. Whitted, "An improved illumination model for shaded display", *Communications of the ACM*, **23**, pp. 343–349 (1980).

5. R. L. Cook, T. Porter, and L. Carpenter, "Distributed ray tracing", *Computer Graphics*, **18**(3), pp. 137–144 (1984).

6. J. T. Kajiya, "The rendering equation", *Computer Graphics*, **20**(4), pp. 143–150 (1986).

7. G. J. Ward, F. M. Rubinstein, and R. D. Clear, "A ray tracing solution for diffuse interreflection", *Computer Graphics*, **22**(4), pp. 85–92 (1988).

8. G. J. Ward, "The radiance lighting simulation and rendering system", *Computer Graphics*, pp. 459–472 (1994).

9. J. Arvo, "Backward ray tracing", *Developements in Ray Tracing. ACM SIGGRAPH '85 Course Notes*, pp. 259–263 (1985).

10. F. Nicodemus, J. Richmond, J. Hsia, I. Ginsberg, and T. Limperis, *Geometric Considerations and Nomenclature for Reflectance*. Monograph 160, National Bureau of Standards, (1977).

11. X. D. He, K. E. Torrance, F. X. Sillion, and D. P. Greenberg, "A comprehensive physical model for light reflection", *Computer Graphics*, **25**(4), pp. 175–186 (1991).

12. C. Schlick, "An inexpensive brdf model for physically-based rendering", *Computer Graphics Forum*, **13**(3), pp. 149–162 (1994).

13. J. L. Bentley, "Multidimensional binary search trees used for associative searching", *Communications of the ACM*, **18**(9), pp. 509–517 (1975).

14. M. Berger, *Geometry, Volume 2*. New York: Springer-Verlag, (1987). translated by M. Cole and S. Levy.

15. D. P. Mitchell and P. Hanrahan, "Illumination from curved reflectors", *Computer Graphics SIGGRAPH'92*, **26**, pp. 283–291 (1992).

16. S. Collins, "Adaptative splatting for specular to diffuse light transport", *Fifth Eurographics Workshop on Rendering*, pp. 119–135 (1994).

17. P. Shirley, "A ray tracing method for illumination calculation in diffuse-specular scenes", *Proceedings of Graphics Interface'90*, pp. 205–212 (1990).

18. G. J. Ward and P. S. Heckbert, "Irradiance gradients", *Third Eurographics Workshop on Rendering, Bristol, England*, pp. 85–98 (1992).

19. M. Roelens, *Un Environnement pour le Tracé de Rayons utilisant une Modélisation par Arbre de Construction*. PhD thesis, Ecole des Mines de Saint-Etienne, (Feb. 1993).

20. J.-L. Maillot, L. Carraro, and B. Péroche, "Progressive ray tracing", *Third Eurographics Workshop on Rendering, Bristol, England*, pp. 9–19 (1992).