Rendering Polygonal Scenes with Diffraction Account

Lilian Aveneau, Michel Mériaux

IRCOM SIC, UMR 6615 CNRS
Boulevard 3 - Téléport 2 - BP 179
86960 Futuroscope Cedex, France
aveneau@sic.sp2mi.univ-poitiers.fr

ABSTRACT

Global processing of diffraction phenomena by a half-plane was proposed by Aveneau and Mériaux, using the Geometrical Theory of Diffraction. In this theory, diffraction rays are emitted by diffraction points belonging to the half-plane edge (or dihedron edge). The solution given for finding these points being numerical, it is difficult to implement and leads to low precision solutions with a slow algorithm. In this paper we first propose a new geometric solution for finding the dihedron diffraction points, which leads to an efficient analytic algorithm. Next we present an implementation of a Ray-Tracing software for polygonal scenes with an automatic diffraction treatment. Since this implies diffraction by dihedra, we present the dihedron data structure and the corresponding algorithm which solves two problems : the existence of diffraction paths, and the impossibility for dihedra to share their edge. These elements lead to an implementation of a Ray-Tracing software with first diffraction account for any polygonal scenes.

Keywords: Rendering Techniques, Diffraction, GTD, Ray-Tracing.

1 INTRODUCTION

The quest for realism has always been a major preoccupation among the image rendering community. To achieve this goal, researchers have added physically based effects, from reflection on perfect mirrors with ray-tracing to glare effects [Spenc95].

All observable optic phenomena are described by one or more physical theory. Thus realism in image rendering can be achieved by including these theories in computational models, if possible.

In this way multiple reflections on diffuse surfaces can be compared to thermal exchanges and leads to radiosity computation [Goral84]. Polarization [Wolff90], birefringence [Tanne94] and interferences [Calle94] [Dias94] can be added to a ray-tracer since the phenomena can be modelled by Geometrical Theory of Optics. Caustics can be understood like Fermat's principle effects [Mitch92] and leads to numerical computations or Monte-Carlo methods [Jense97]. Atomic scattering can be taken into account with phase functions and geometrical optics, and is used in Monte-Carlo based processes [Blasi94] [Perez97].

Like in global illumination, local rendering methods intensively use the concept of ray, i.e. geometrical optic theory, for instance the Cook-Torrance's model [Cook81]. Simulation of the subsurface scattering leads to skin or leaf reflections [Hanra93]. Diffraction by a subsurface structure leads to a model based on Kirchhoff's law [He91].

Eye, eyelash, and eyebrow diffraction were also described using a ray-tracer method and a filtering process [Nakam90]. This method has been augmented for lenticular halo, bloom and flare lines [Spenc95].

However, global diffraction models are usually considered as too complex for their use in image rendering, for a small benefit. Nevertheless, powerful light sources imply visible diffraction phenomena (*e.g.* sun through a shutter on a wall).

In this way, the Geometrical Theory of Diffraction [Kelle62] was introduced in computer graphics and leads to diffraction by a half-plane or a slit [Avene97]. Therefore this solution was limited to very simple geometries. The next step for a global treatment of wave propagation effects is to allow diffraction by any polygonal objects. Fortunately, the GTD describes dihedron diffraction as an extension of the half-plane diffraction.

In this paper we present an implementation of a Ray-Tracing software which supports diffraction by dihedron. First, we present a new solution for finding optical paths with one edge diffraction point. This analytical solution is more efficient and much simpler to implement than the previous one [Avene97], which involves numerical computations. Next we introduce the dihedron data structure which allows us to deal with the problem of optical path existence in a polygonal geometry.

We finish with a discussion on the efficiency of our solutions, and conclude with further works.

2 SINGLE DIFFRACTION BY A DIHEDRON

Global processing of diffraction by a half-plane was introduced by L. Aveneau and M. Mériaux in [Avene97], using the Geometrical Theory of Diffraction [Kelle62] [Kouyo74]. This particular theory extends the *principle of Fermat* [Born80] by introducing diffraction points along light paths. These new points are localized on geometric objects responsible for diffraction phenomena. Each geometric shape needs specific treatment. In [Avene97], half-plane diffraction and, by extension, dihedron diffraction, were introduced. In these cases, because surface diffractions are similar to surface reflections, only diffraction by edges needs to be dealt with.

2.1 Geometric Theory of Diffraction

The theoretical basis of the Geometric Theory of Diffraction is the extended Fermat's principle, or Fermat's principle for edge diffraction : *an edge-diffracted ray from a point P to a point Q is a curve which has stationary optical length among all curves from P to Q with one point on the edge.* This principle implies that diffracted rays, emitted from diffraction points on an edge, are localized on a cone surface with, as axis, the edge itself, and with an opening angle β equal to the incident one (see figure 1).

If we know the radiance of the incident ray, the

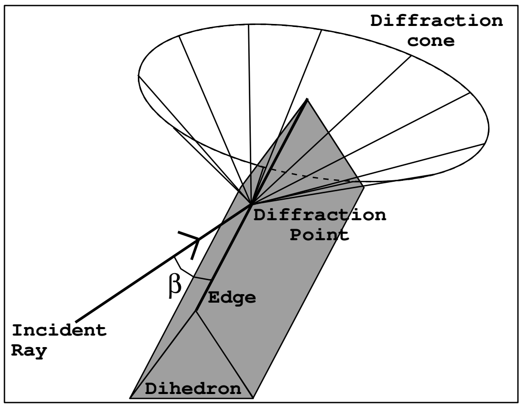

Fig. 1 *Diffraction by a dihedron : diffracted rays are on cone surface*

radiances of the new rays can be deduced easily by use of *diffraction coefficients*. We express the polarized incident radiance with a coherence matrix [Wolff90]

$$J_{inc} = \begin{pmatrix} J_{xx} & J_{xy} \\ J_{yx} & J_{yy} \end{pmatrix} = K.\begin{pmatrix} <\overrightarrow{E_x}.\overrightarrow{E_x^*}> & <\overrightarrow{E_x}.\overrightarrow{E_y^*}> \\ <\overrightarrow{E_y}.\overrightarrow{E_x^*}> & <\overrightarrow{E_y}.\overrightarrow{E_y^*}> \end{pmatrix}$$

where K is a constant due to the quadratic receptor (e.g. eye, camera), the angle brackets denotes values averaged over time, $\overrightarrow{E_x}$ and $\overrightarrow{E_y}$ are respectively the parallel and the perpendicular components of the electric field $\overrightarrow{E}$, and starred superscript denotes the complex conjugate. With these notations, the radiance is equal to the coherence matrix trace. The diffracted coherence matrix is then :

$$J_{dif} = \begin{pmatrix} J_{xx}\cdot D_{\parallel}^2 & J_{xy}\cdot D_{\parallel}D_{\perp} \\ J_{yx}\cdot D_{\parallel}D_{\perp} & J_{yy}\cdot D_{\perp}^2 \end{pmatrix}$$

where $D_{\parallel}$ and $D_{\perp}$ are respectively the parallel and perpendicular diffraction coefficients. For a dihedron of angle $\chi = (2-n)\pi$, J.B. Keller has proposed [Kelle62] :

$$D = \frac{\sqrt{\lambda}\cos\frac{\pi}{4}\sin\frac{\pi}{n}}{2n\pi sin\beta}\left[\frac{1}{\cos\frac{\pi}{n}-\cos\left(\frac{\delta-\alpha}{n}\right)} + C.\frac{1}{\cos\frac{\pi}{n}-\cos\left(\frac{\delta+\alpha+\pi}{n}\right)}\right]$$

where λ is the wavelength ; β (figure 1) is the angle between the incident vector $\overrightarrow{V}$ and the edge unit vector $\overrightarrow{u}$ (figure 2) ; $\overrightarrow{n}$ is the unit vector normal to the dihedron incident face ; δ is the diffraction angle between $\overrightarrow{n}$ and $\overrightarrow{D_p}$, which is the projection of the diffracted vector $\overrightarrow{D}$ into π, a plane perpendicular to the edge ; α is the incident angle[1] between $\overrightarrow{n}$ and

[1] For continuity of diffraction coefficients, and therefore diffracted rays radiance, α can have positive or negative value ; in our implementation we take a negative value for α when $\overrightarrow{P}.\overrightarrow{V} \leq 0$, with $\overrightarrow{P} = \overrightarrow{n} \wedge \overrightarrow{u}$ (see figure 2).

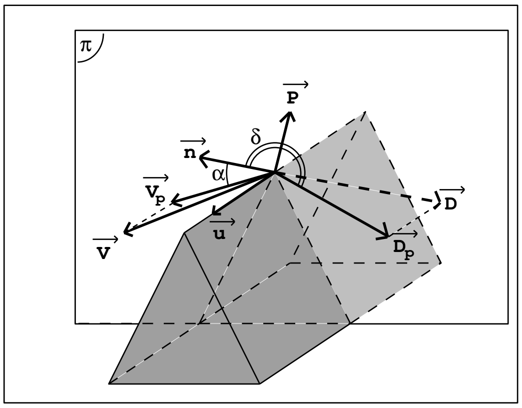

Fig. 2 *Angles of diffraction in a plane perpendicular to the dihedron edge*

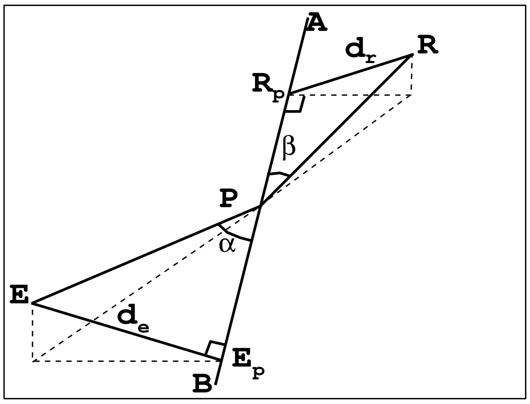

Fig. 3 *Geometric based algorithm*

$\overrightarrow{V_p}$, which is the projection of $\overrightarrow{V}$ into π ; $C = +1$ for $D_{\|}$ or $C = -1$ for $D_{\perp}$.

In an implementation without polarization states, the expression of the diffracted radiance L_{dif} can be simplified as :

$$L_{dif} = L_{inc}.\lambda.\left(\frac{\cos\frac{\pi}{4}\sin\frac{\pi}{n}}{n\pi\sin\beta\left(\cos\frac{\pi}{n} - \cos\left(\frac{\delta-\alpha}{n}\right)\right)}\right)^2$$

where L_{inc} is the incident radiance.

2.2 Diffraction and Ray-Tracing

The main advantage of the GTD is that it is an extension of the Geometrical Optic Theory which is the basis of the Ray-Tracing algorithm [Whitt80] and extended methods [Blasi94] [Jense97] [Perez97]. Including the diffraction treatment into such algorithms implies to extend them in a relatively simple way. The solution presented in [Avene97] is based on a classical Ray-Tracing implementation.

For evaluating the radiance received at the eye, rays are emitted through the image pixels from the eye. These rays can lead to a scene object at a point P. So we must compute the emitted radiance at P in the observer direction. In this way, we firstly cast rays from P to each light source points, and secondly recursively compute the radiance received at P from the reflected and the refracted ways.

This work, which consists in evaluating the emitted radiance at P, was done in the *Shade* function of the Ray-Tracer. Aveneau and Mériaux's solution [Avene97] consists in adding new possibilities to this function. They introduce the *diffraction algorithm*, which consists in finding the diffracted rays, and so diffraction points on edges. For all edges, they search for a diffraction path. If such a path exists the diffracted radiance is computed and added to the received radiance at point P. So they consider that the visible diffraction is mainly due to paths which begin at source points, continue with one edge diffraction point and some reflections.

2.3 Diffraction path search

Previous analytical solution. The main problem here is to find the single diffraction point on the dihedron edge. In [Avene97], a first algorithm was proposed, based on the extended Fermat's principle. This solution is based on the search of the shortest optical path between two fixed points E and R, with one unknown point P on that edge. This distance and its first derivative are expressed with the parametric coordinates of P. Therefore a recursive interpolation gives the solution for P, if such a solution exists. Unfortunately, this algorithm has two flaws : firstly, no time computation control can be done, and, secondly, it is relatively complex to code. For these reasons, a new algorithm is proposed here, based on a geometrical approach.

New geometric solution. Let E and R be two points in 3D space, and P a point on an edge AB (see figure 3) so that (EPR) is a good path in accordance with extended Fermat's principle (angles α and β seem different, but it is only because of perspective distortion). If R_p and E_p are the projections of E and R on edge AB respectively, it is obvious that P lies between them. So if t_e and t_r are the parametric coordinates of these points on the edge, the angle equality gives

$$\begin{aligned}\tan\alpha = \tan\beta \quad &\Rightarrow \quad \frac{d_r}{(t_r - t)} = \frac{d_e}{(t - t_e)} \\ &\Rightarrow \quad t = \frac{t_e d_r + t_r d_e}{d_e + d_r}\end{aligned}$$

A simple algorithm for determining the diffraction point on a dihedron edge can be derived from this equality. All we need to do is to compute the projected points R_p and E_p (with dot products), and the length of vectors $\overrightarrow{RR_p}$ and $\overrightarrow{EE_p}$. The computed

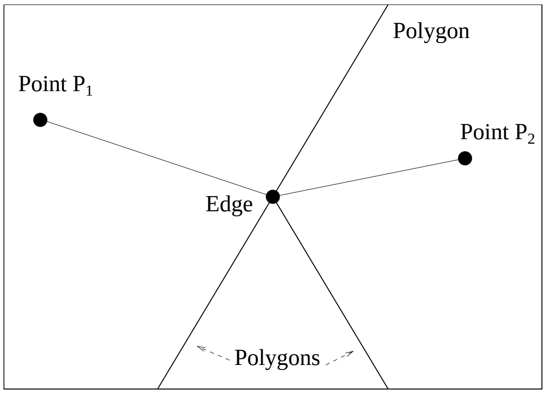

Fig. 4 *A 2D view of the adjacency problem and for the topological partionment one.*

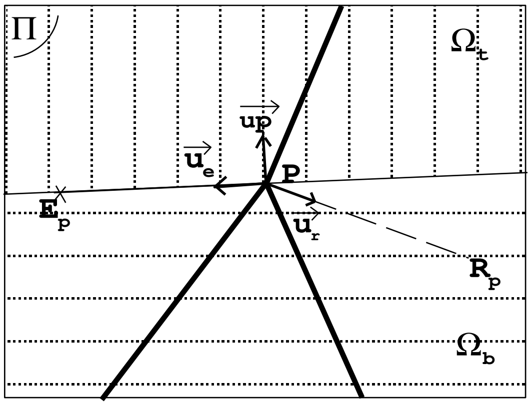

Fig. 5 *Projected points and definitions in the plane* Π

solution P is a good one if it is between the two edge extremities A and B. So a small optimization consist in testing if the projected points E_p and R_p are both before A or after B. Indeed, if it is the case, then P is not on the edge.

The pseudo-code for this function is :

```
Function SolveDiff (AB as edge;
                    E, R, &P as point) as Bool
  Compute :
    te as the projected point Ep coordinate
    tr as the projected point Rp coordinate
  If both Ep and Rp are before A or after B
    Return False
  Compute :
    t = (te*dr + tr*de) / (de + dr)
  If P is not between A and B
    Return False
  Return True
End
```

3 DIHEDRON DIFFRACTION IN IMAGE RENDERING

Now, the goal is to allow image computation with diffraction treatment for any polygonal scene. Since the final user cannot find *a priori* which dihedron edges produce visible diffraction effects, we have searched for an algorithm which automatically finds all the solutions. The dihedron data structure has first to be defined. A naive structure cannot be used since there are two problems which, then, cannot be solved.

3.1 What are the difficulties ?

The first problem appears when three or more polygons are incident to the same edge. This is the case for instance when an object is put down under another one, or, in general, when two objects are adjacent. In this case it seems difficult to construct the corresponding dihedron. The figure 4 allows us to clarify this problem. With these three polygons adjacent to the same edge and a naive dihedron data structure, three different dihedra have to be constructed. So, since a diffraction path exists between the points P_1 and P_2, the diffraction algorithm implies firstly to verify this point three times, and secondly to compute the diffraction point three times. It is obvious that two of these computations should be omitted, since only one is necessary. But these cases cannot be easily detected with a naive data structure.

The second problem is very easy to understand with the same figure 4. We have to verify if the computed path between the points P_1 and P_2 really exists. Indeed, like in the figure, sometimes the diffraction path crosses the diffracted dihedron. In other terms, a dihedron cuts the 3D-space into two topological volumes. So we must check that the optical path fully belongs to one of them.

3.2 Dihedron Data Structure

Since the naive dihedron data structure does not allow us to solve the two previous problems, we have defined a more complex structure : it contains a 3D-straight line for the edges support, and a list of dihedron faces.

The construction of such objects can be done easily if the scene is made of polygons and if it is corefined[2] (or subdivided) [Rossi90].

In order to solve the two previous problems with this data structure, we need some new definitions. Let E and R be two fixed points, and let P be the diffracted point on edge e so that EPR is a correct extended optical path. Let Π (see figure 5) be the plane which contains P and is normal to the edge unit vector $\overrightarrow{u}$ (which points outward the page). Let E_p (resp. R_p) be the projection of point E (resp. R) onto the plane Π. Let $\overrightarrow{u_e}$ (resp. $\overrightarrow{u_r}$) be the unit vector $\overrightarrow{PE_p}/||\overrightarrow{PE_p}||$ (resp. $\overrightarrow{PR_p}/||\overrightarrow{PR_p}||$). Let $\overrightarrow{up}$ be the

[2] makes two objects "compatible" by subdividing the cells of each object at their intersections with cells of the other objects

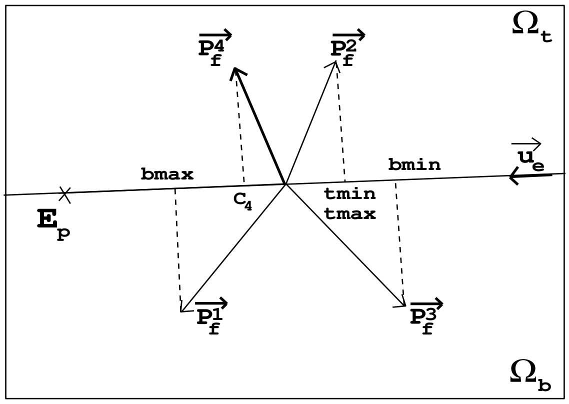

Fig. 6 *Example of the dihedron computation at the fourth iteration*

unit vector so that $\overrightarrow{up} = \overrightarrow{u_e} \wedge \overrightarrow{u}$ where $\wedge$ denotes the vector product. Let Ω_t (resp. Ω_b) be the half-space so that P' is in Ω_t (resp. Ω_b) if and only if $\overrightarrow{up}.\overrightarrow{PP'} \geq 0$ (resp. $\overrightarrow{up}.\overrightarrow{PP'} < 0$). With these definitions and the dihedron data structure previously defined, we can write an algorithm which finds the true dihedron on which diffraction appears, and determines if the points E, P and R belong to the same topological volume.

3.3 Validity of the extended optical path

The key idea of this algorithm is the iterative construction of the true dihedron ; this one must be composed by the two faces which are the limits of the topological volume which encloses the points E and R. Each face belongs to one half-space Ω_t or Ω_b, and is described by a coefficient c_i which is equal to the dot product $\overrightarrow{u_e}.\overrightarrow{P_f^i}$, where $\overrightarrow{P_f^i}$ is the outward unit vector perpendicular to the edge e and parallel to the face plane.

A loop, on the dihedron list faces which contain the diffraction point P, builds the true dihedron iteratively. At the step i, the current dihedron is known by its two faces (if they exist). The method consists in computing, at each iteration i, the extrema of the set $(c_j)_{1 \leq j \leq i}$ in Ω_t and in Ω_b, and the corresponding faces. Let tmax and tmin be the extrema into Ω_t, and let bmax and bmin be the extrema into Ω_b. An example of such a process is given in figure 6.

At the end of the loop, it is easy to know if the path is fully in a given topological volume. We have a dihedron diffraction if at least one face edge contains diffraction point P, and if one of the following assertions is verified :

- R is contained in Ω_b (resp. Ω_t), and Ω_b (resp. Ω_t) contains no face,
- Ω_b (resp. Ω_t) does not contain any face, R is contained in Ω_t (resp. Ω_b), and $(c < tmin)$ or $(c > tmax)$ (resp. $(c < bmin)$ or $(c > bmax)$),
- Ω_t and Ω_b contain faces and either R is contained in Ω_b and $(c > bmax)$, or R is contained in Ω_t and $(c > tmax)$,

where $c = \overrightarrow{u_e}.\overrightarrow{u_r}$.

The pseudo-code of this algorithm is :

```
Function FindDihedron(D as Dihedron;
                      E, R, P as Point;
                      F[2] as Face) as Bool
  Initialisation:
    tmin, tmax, tFmin, tFmax,
    bmin, bmax, bFmin, bFmax
  Compute:
   Vectors $\overrightarrow{u_e}$ and $\overrightarrow{up}$
  For All Faces containing P Do
    Compute : $C_i = \overrightarrow{u_e} \cdot \overrightarrow{P_f^i}$
    If $P_f^i$ is in $\Omega_b$³ Then
      If $C_i <$ bmin Then
        bmin  = $C_i$, bFmin = i
      ElseIf $C_i >$ bmax then
        bmin  = $C_i$, bFmin = i
      EndIf
    ElseIf $C_i <$ tmin Then
        tmin  = $C_i$, tFmin = i
      Else If $C_i >$ tmax then
        tmin  = $C_i$, tFmin = i
      EndIf
    EndIf
  EndDo
  If at least one Face contains P Then
    If the solution is a good one
      Initialise F[1] and F[2]
      Return True
  Endif
  Return False
End
```

This treatment may appear costly, but in fact there are rarely more than three or four faces which contain the same diffraction point. So the problem is which faces contain this point, and it is easy to solve.

4 DISCUSSION

The new algorithm presented in the second section for efficiently searching the dihedron diffraction points has been implemented and compared with the previous one. The results of this comparison (see figure 7) show that the new version is more efficient than the previous one, with an acceleration factor between 2.6 and 5.8. We have taken 10,000 random points E and R, and computed each solution 1,000 times. Computing a solution with the new algorithm requires less than $1.15\mu s$ on a SGI-Origin 200 (monoprocessor R10000). Thus we ensure the fact that this second method is always more efficient than the previous one. Furthermore the computation time has an upper limit which is $1.15\mu s$ (computation time is constant

[3] *i.e.* $\overrightarrow{up} \cdot \overrightarrow{P_f^i} < 0$.

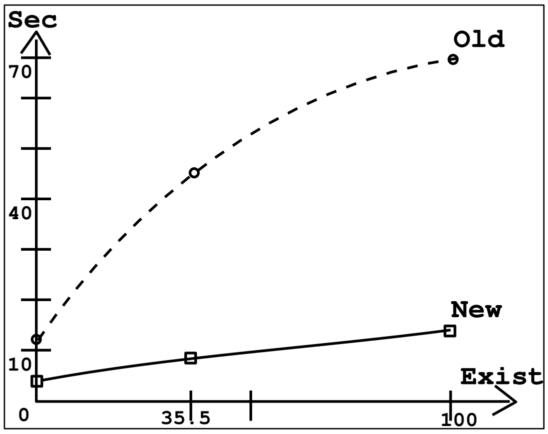

Fig. 7 *Experimental comparison of the two search algorithms*

if one removes the conditional at line 6 of the pseudo-Code). Therefore the most important improvement with this new analytic algorithm is that we always find the good solution, which was not true with the old numerical algorithm.

Next, we have implemented the dihedron data structure and the last algorithm presented here. They allow us to produce some images with dihedron diffractions (with one diffraction between two fixed points). For example figure 8 shows an interior scene which is mainly illuminated by the sun through a shutter. The camera looks at a wall which is in front of the window with the shutter. So we can directly observe the diffraction effect.

This image computation costs 13 minutes, with a resolution of 700x500 pixels, with 16 jittered rays per pixel on a SGI-Origin 200 (monoprocessor R10000). As everyone can notice, this image is not realistic : indeed, the sun through the shutter must produces some ellipses, but with a classical Ray-Tracing software they look like rectangles.

In figure 9 the same scene has been computed with dihedron diffractions. This second image looks really more realistic than the previous one. Everybody can verify at home or at work that this solution is more in accordance with the physical reality. There is a little problem with the rendering of the ellipses border. This is due to the absence of corner diffraction treatment : indeed, with rectangles, we have diffraction on the edges. But this implies a discontinuity between two edges treatment. The solution for solving this problem is to add diffracted rays emerging from the intersection of these two edges that we call corner. Yet this is not implemented at the moment, since the diffraction coefficients for the corner remain unknown.

The computation of this second image costs 46 times more than without diffraction computation. This scene contains 270 polygons, and 378 dihedra have been created from 1080 half-plane edges. This method is still expensive, and needs to be optimized. This expensive computation time is directly connected to the *diffraction algorithm* (section 2.2), and so to the edge number. In [Avene99], we present a solution to improve the computation times with a complexity study.

The main difference with the previous algorithm [Avene97] is that we obtain diffraction effects only where it is necessary, even though the old solution produces diffraction even where it is impossible. This comes from the fact that we consider now the scene topology, thus eliminating all the bad diffraction effects.

5 CONCLUSION AND FUTURE WORK

A new algorithm was presented which solves the search of the diffraction point between two fixed points. Its complexity is always better than the one of the previous algorithm. Moreover it is easily implementable. Furthermore we present the dihedron data structure which allows us to solve the difficult problem of diffraction path existence. This structure allows an easy implementation of the global diffraction by dihedra in a ray-tracer based system.

We have implemented these algorithms in our library and used them for image synthesis and wave propagation purposes. Therefore, as it can be observed on the image shown in figure 9, our method must be extended with the introduction of corner diffraction : indeed, when two diffraction edges share a vertex and so make a corner, the edge diffraction treatment induces discontinuity effects (for instance the discontinuous illumination at the left of the white strips). The solution of this problem is to include corner diffraction. Nevertheless the difficulty is to find the expression of the diffraction coefficient for such a geometrical shape.

Future works include also the optimization of the diffraction path search. First investigation leads to very efficient results [Avene99], with, for the diffraction process, an acceleration factor up to 44300 on our test scenes. Finally we would extend this method to other geometrical shapes, like cylinders, spheres, quadrics, etc. This requires the search for comprehensive diffraction coefficients for each new shape.

Fig. 8 *Interior scene mainly illuminated by the sun through a shutter without diffraction*

Fig. 9 *Same scene with dihedron diffractions*

References

Avene97. Lilian Aveneau and Michel Mériaux. Phénomènes ondulatoires en synthèse d'images. *Revue internationale de CFAO et d'informatique graphique*, 12(4):405–425, 1997 (in french).

Avene99. Lilian Aveneau and Michel Mériaux. The Discrete Tube : A Spatial Acceleration Technique for Efficient Diffraction Computation. *Proceedings of DGCI'99*, Noisy le Grand, March 1999, France.

Blasi94. Philippe Blasi, Bertrand Le Saec, and Christophe Schlick. An Importance Driven Monte-Carlo Solution to the Global Illumination Problem. In *Fifth Eurographics Workshop on Rendering*, pages 173–183, Darmstadt, Germany, June 1994.

Born80. M. Born and E. Wolf. *Principles of Optics*. Pergammon Press, New York, 6th edition, 1980.

Calle94. Patrick Callet. Interférences, couches minces et peintures métallisées. *Revue internationale de CFAO*, 9(1):251–264, 1994. (in french)

Cook81. R. L. Cook and K. E. Torrance. A reflectance model for computer graphics. *Computer graphics, Aug 1981*, 15(3):307–316, 1981.

Dias94. Maria Lurdes Dias. Ray tracing interference color. *IEEE Computer Graphics and Applications*, 14(3):17–20, May 1994.

Goral84. Cindy M. Goral, Kenneth E. Torrance, Donald P. Greenberg, and Bennett Battaile. Modelling the interaction of light between diffuse surfaces. In *Computer Graphics (SIGGRAPH '84 Proceedings)*, volume 18, pages 212–222, July 1984.

Hanra93. Pat Hanrahan and Wolfgang Krueger. Reflection from layered surfaces due to subsurface scattering. In *Computer Graphics (SIGGRAPH '93 Proceedings)*, volume 27, pages 165–174, August 1993.

He91. Xiao D. He, Kenneth E. Torrance, François X. Sillion, and Donald P. Greenberg. A comprehensive physical model for light reflection. *Computer Graphics*, 25(4):175–186, July 1991.

Jense97. Henrik Wann Jensen. Rendering caustics on non-lambertian surfaces. *Computer Graphics Forum*, 16(1):57–64, 1997. ISSN 0167-7055.

Kelle62. Joseph B. Keller. Geometrical theory of diffraction. *Journal of the Optical Society of America*, 52(2):116–130, February 1962.

Kouyo74. Robert G. Kouyoumjian and Prabhakar H. Pathak. A uniform geometrical theory of diffraction for an edge in a perfectly conducting surface. *Proceedings of the IEEE*, 62(11):1449–1461, November 1974.

Mitch92. Don P. Mitchell and Pat Hanrahan. Illumination from curved reflectors. In Edwin E. Catmull, editor, *Computer Graphics (SIGGRAPH '92 Proceedings)*, volume 26, pages 283–291, July 1992.

Nakam90. Eihachiro Nakamae, Kazufumi Kaneda, Takashi Okamoto, and Tomoyuki Nishita. A lighting model aiming at drive simulators. *Computer Graphics*, 24(4):395–404, August 1990.

Perez97. Frederic Pérez, Xavier Pueyo, and François X. Sillion. Global illumination techniques for the simulation of participating media. In *Eurographics Rendering Workshop 1997*, pages 309–320, New York City, NY, June 1997. Springer Wein.

Rossi90. J. R. Rossignac and M. A. O'Connor. SGC: A dimensional-independent model for pointsets with internal structures and incomplete boundaries. In *Geometric Modeling for Product Engineering*, pages 145–180. Elsevier Science Publishers B.V., North Holland, 1990.

Spenc95. Greg Spencer, Peter Shirley, Kurt Zimmerman, and Donald Greenberg. Physically-based glare effects for digital images. In *SIGGRAPH 95 Conference Proceedings*, pages 325–334. Addison Wesley, August 1995.

Tanne94. David C. Tannenbaum, Peter Tannenbaum, and Michael J. Wozny. Polarization and birefringency considerations in rendering. In *Proceedings of SIGGRAPH '94 (Orlando, Florida, July 24–29, 1994)*, pages 221–222. ACM Press, July 1994.

Whitt80. Turner Whitted. An improved illumination model for shaded display. *Communication of the ACM*, 23(6):343–349, June 1980.

Wolff90. Lawrence B. Wolff and David J. Kurlander. Ray tracing with polarization parameters. *IEEE Computer Graphics and Applications*, 10(6):44–55, November 1990.