MULTIDIMENSIONAL ILLUMINATION FUNCTIONS FOR VISUALISATION OF COMPLEX 3D ENVIRONMENTS*

S.P. Mudur and **S.N. Pattanaik**
National Centre for Software Technology
Gulmohar Cross Road No.9, Juhu
Bombay — 400 049, India

Abstract

This paper presents a new view-independent, energy equilibrium method for determining the light distributed in a complex 3D environment consisting of surfaces with general reflectance properties. The method does not depend on discretisation of directions or discretisation of surfaces to differential elements. Hence, it is a significant improvement over the earlier complete view-indepedent method which is computationally intractable for complex environments or the hybrid methods which include an extended view-dependent ray tracing second pass. The new method is based on an efficient data structure of order $O(N^2)$ named as the spherical cover. The spherical cover elegantly captures the complex multidimensional directional nature of light distributed over surfaces. Subdivision techniques based on range estimation of various parameters using interval arithmetic like methods are next described for efficiently computing the spherical cover for a given 3D environment. Using the spherical cover, light is progressively propagated through the environment until energy equilibrium is reached. Complexity analysis of the propagation step is carried out to show that the method is computationally tractable. The paper also includes a comprehensive review of earlier rendering techniques viewed from the point of capturing the multidimensional nature of light distribution over surfaces.

1 INTRODUCTION

1.1 Three Dimensional Environments

Over the years geometric modeling [1] has evolved enabling us to model the complex object shapes encountered in our real world compactly and with reasonable fidelity.

*The Journal of Visualisation and Computer Animation, vol 1, 49-58, 1990

Typically, objects are represented by their bounding surfaces. The bi-parametric polynomial, in particular, the bi-cubic form of surface representation is most widely used in many computer aided geometric modeling systems [2,3]. A model of a complex three dimensional scene may include a large number of such surfaces. Further processing of such models, say, spatial, visual or other interrogations, require the development of powerful algorithms for handling this complex geometry. The obvious technique of converting the model to an approximate form using simpler geometry such as planar faces is not always acceptable as it may give rise to unwanted artifacts and may also result in an unacceptable increase in combinatorial complexity. Visual analysis requires association of various optical properties with the surfaces in the environment. Examples of optical properties are colour, opacity, transparency, absorbivity, shinyness, surface finish, emissivity, etc. Properly modeling the nature of these optical properties is essential and has been addressed by researchers in other disciplines such as radiometry [4]. For reasons of computational efficiency, simplified models have been in use in computer graphics and computer vision [5,6]. This paper is concerned with the development of algorithms for the visualisation of complex 3D environments with varying optical properties dealing directly with the exact geometrical forms. Visualisation of such scenes involves the synthesis of images depicting or rendering the surfaces of objects as seen from different view points.

Image synthesis [7] is thus, the process of converting a 3-dimensional geometric representation with associated optical properties into a visual representation mostly on a 2D medium such as a CRT display screen or a photographic plate. Realistic image synthesis is a major subclass which has been a subject of keen interest in computer graphics for quite some time. Realistic image synthesis requires simulation of the physical process, namely, interaction of light with matter, as accurately as possible in order to finally generate brightness values for different points of the 2D medium constituting the image. Brightness is an informal term used to refer to at least two different concepts — image brightness and object brightness. In the image, brightness is image irradiance. The blackening of a film in a camera or the glow of phosphor on the CRT is a function of the irradiance. The object brightness is the object radiance. Image irradiance is proportional to object radiance and the constant of proportionally depends on the imaging system used. The definition of the terms radiance and irradiance can be found in the study of radiometry [4]. Irradiance is the amount of light energy falling on a surface. It is the power per unit area incident on the surface and has the unit watts per square metre. Radiance is the amount of light energy irradiated from a surface. It is the power per unit projected area per unit solid angle emitted from the surface. The area is projected in the direction of emission. Radiance has the unit watts per square metre per steradian.

1.2 The Nature of Light Distribution over a Surface

For the purposes of this discussion, let us consider an environment consisting of opaque surfaces only. Consider any point on any surface in the environment. There is a hemisphere of directions around the normal at the point from which it can be said to receive light and through which it can be said to reflect/emit light. The amount of light received/emitted in any particular direction varies.

Let θ denote the polar angle and ϕ the azimuth angle. On the hemisphere θ varies from 0 to $\pi/2$ while ϕ varies from 0 to 2π.

Let us define two functions at any point on the surface, one $E(\lambda, \theta, \phi)$ which we shall term the *Illumination function* and the other $I(\lambda, \theta, \phi)$ which we shall term the *Brightness function*. The Illumination function defines the radiance received by the point from the hemispherical directions. Similarly, the Brightness function defines the radiance emitted by the point through the hemispherical directions. Both E and I have the unit watts per square metre per steradian. As in all earlier image synthesis work we shall consider evaluation of the functions E and I independently for a number of discrete wavelength bands and hence omit λ in all future references. At any time given functions $E(\theta, \phi)$ and $I(\theta, \phi)$ at a point one can determine the light incident from any direction (θ_i, ϕ_i) and the light emitted into any direction (θ_r, ϕ_r) (*See* Fig.1). Without loss of generality, we shall assume that all surfaces in the environment are parametrically defined. Thus, any point P on a surface would be uniquely identified by two parametric values (u, v), which we shall denote by $P(u, v)$. $P(u, v)$ is a vector $[x(u, v), y(u, v), z(u, v)]$. The normal at the point $P(u, v)$ is also a function of (u, v), say $N(u, v)$. θ, ϕ at a point are with respect to the normal. If we now consider the two functions E and I defined over all the points in the surface, they would take the form $E(u, v, \theta, \phi)$ and $I(u, v, \theta, \phi)$ respectively.

The *Brightness function* $I(u, v, \theta, \phi)$ is actually a complex function dependent on :

(i) the *Illumination Function* $E(u, v, \theta, \phi)$.

(ii) the normal at the point $N(u, v)$.

(iii) the reflectance property of the surface.

The reflectance termed bi-directional reflectance is itself a complex function represented by $R(\theta_i, \phi_i, \theta_r, \phi_r)$. For a brief review of various reflectance models in use the reader is referred to [8,9].

The relationship amongst these is expressed by the equation given below:

$I(u, v, \theta_r, \phi_r) = \iint_\Omega R(\theta_i, \phi_i, \theta_r, \phi_r)\ \ E(u, v, \theta_i, \phi_i)\ \ cos\theta_i\ \ d\omega_i$

where Ω denotes the hemisphere.

Thus I is the integral over the hemisphere of the *Illumination Function* convolved with the *reflectance function*. It is important at this juncture to note the following points about this equation:

(i) For a perfect diffuse surface brightness is the same in any (θ, ϕ) direction, as R is a constant independent of direction. Hence, this equation reduces to

$$I(u, v) = R \iint_\Omega E(u, v, \theta_i, \phi_i) cos\theta_i d\omega_i$$

If E is the same for all (θ, ϕ) in the hemisphere and there is no absorption then $I(u, v) = E(u, v)$.

(ii) The incident light $E(u, v, \theta_i, \phi_i)$ is actually the brightness of the surface seen by (u, v) in the direction $(\theta_i^{-1}, \phi_i^{-1})$ and hence has the form $I(s, t, \theta_i^{-1}, \phi_i^{-1})$ for another point $P(s, t)$ in the environment. This means that the brightness of one surface determines, possibly partially, the brightness of another surface which in turn determines, possibly partially, the brightness of yet another surface and so on thus leading to a light energy distribution equilibrium in which the illumination and brightness functions are all in a steady state.

The basic requirement of any rendering algorithm is the determination of the distribution of light over surfaces in the environment. In the last two decades a variety of rendering algorithms have been proposed differing widely in their capabilities to capture the illumination and/or brightness functions as accurately as possible. In the next section, we review some of the well known algorithms for rendering.

2 A REVIEW OF RENDERING TECHNIQUES

In this section, rendering algorithms are compared on the basis of the following:

(i) Approximations made to the illumination/brightness functions.

(ii) Assumptions regarding the geometry in the environment.

(iii) Explicit storage of the illumination/brightness functions over surfaces, if at all.

(iv) Method used to determine the illumination/brightness function at any point in the environment.

2.1 Early Rendering Techniques

These techniques, introduced in the early days of computer graphics, made gross assumptions regarding the incident illumination function. Illumination is assumed to be only due to point light sources. If (θ_s, ϕ_s) is the direction of the source with respect to the point and normal at the point (u,v) then $E(u, v, \theta, \phi)$ at any point is assumed to be a delta function as given below:

$$\begin{aligned} E(u, v, \theta, \phi) &= I_s \ \ for \ \ \theta = \theta_s, \phi = \phi_s, \\ & \ 0 \ \ otherwise. \end{aligned}$$

Early work used planar approximations to the geometry. Illumination or brightness functions are never explicitly stored over the surfaces as part of the rendering process. However, using the above equation brightness is computed only at visible points on surfaces in the environment and only in the viewing direction [8,9]. Diffuse as well as specular properties could be associated with surfaces. The methods were extended to directly render curved surfaces [10].

One major problem with these techniques is that the illumination function models are very local. They do not take into account the fact that brightly lit surfaces themselves act like light sources and illuminate others. In these models all contributions in directions other than the source (θ_s, ϕ_s) are clubbed together to form a uniform ambient light source. Thus in these models inter-reflections, refractions and other such global effects were not modeled.

2.2 Ray Tracing Techniques

In the basic ray tracing technique [11], once again, illumination functions are not explicitly stored over any surfaces in the environment. The image rectangle is assumed to be a matrix of pixels and the brightness function is determined for each pixel in the view direction only. Sometimes, the pixel is treated as a small finite region and the brightness computation is carried out for many points in this region. The final brightness assigned to a pixel is an averaged constant, possibly a weighted average.

Brightness at a pixel in a view direction is computed by first evaluating the brightness at the surface point in the environment visible through this pixel. The first approximations, to the illumination model at this surface point were very much like the early models, i.e., illumination from visible point light sources. For better approximation, particularly to take into account specular properties, illumination from a surface point in the reflected view direction is also considered. At the surface point in the reflected view direction, the brightness is computed recursively using a similar formulation. Recursion is carried out to some finite depth. The basic problem in this formulation is that only the reflected view direction is given importance, effectively capturing the specular behaviour of the surfaces. The effect of brightly lit diffuse surfaces is, however, not properly captured.

Extensions to ray tracing [12] came in the form of a careful choice of a bundle of (θ, ϕ) directions to be chosen at any surface point when determining its illumination. In distributed ray tracing, [13] stochastic sampling techniques were used in order to choose the (θ, ϕ) directions distributed over the hemisphere. Recall that in each (θ, ϕ) direction E is formulated as a delta function. With the right number and the right choice of directions ray tracing can capture all types of illumination effects, reflections, refraction, diffuse inter-reflection, shadows, etc. However, for truly capturing all global effects reasonably accurately, a large number of directions have to be chosen. Since the illumination functions are not explicitly stored over the surfaces these can result in excessive computations.

The basic geometric computation in ray tracing is the line-surface intersection for determining the surface point contributing illumination to a point in the direction of the line. Even though early attempts dealt with planar approximations of the environment, geometric computations were extended to deal directly with complex geometry [14,15].

Ward, *et.al* [16] describe an interesting extension of ray tracing for diffuse surfaces. Perhaps, their's is the only ray tracing method which stores the illumination explicitly on the surfaces and yet is a single pass technique. To start with brightness is known only at point light sources. Ray tracing is carried out as usual with the following modifications:

Brightness at any surface point is computed as the weighted average of the brightness values known in the influence neighbourhood. If, however, there are not enough known values in the neighbourhood, then brightness is calculated using distributed ray tracing or any other applicable technique. This brightness is recorded in an oct-tree data structure for the entire scene and is used for future influencing. Ward *et.al* suggest illumination gradient based methods for determining the influence of a neighbourhood point with known brightness.

2.3 Two Pass Techniques

Some of the extensions to ray tracing have been concerned with the explicit storage of the diffuse component of the illumination or brightness functions over the surfaces in the environment. All two pass methods are primarily in this category. In the first pass an approximate estimate of the illumination function is arrived at using a variety of techniques, typically tracing the path of light from the light sources. View direction is usually not specially taken into account in this pass. In the second pass, a more accurate value for the illumination function is obtained in the view direction using the traditional ray tracing formulation, but this time presumably using much fewer sample directions. Two pass methods differ in their method of storing the illumination function on surfaces and in the methods employed in computing the approximate estimates for these. For example, [17] Arvo stores brightness at each one of a rectangular matrix of points on each surface. Bilinear interpolation is chosen for intermediate values. Only diffuse behaviour is considered for this. Brightness is estimated by tracing rays from light sources to all points in the rectangular matrix of points in each surface. Similarly, Chattopadhyay and Fujimoto [18] store the illumination at points in a uniform grid in the total three dimensional environment. This illumination is computed by transferring the brightness from diffuse surfaces whose illumination is approximated as in the early techniques, i.e., direct contribution of point light sources.

There are a number of two pass methods which use the energy equilibrium based radiosity techniques for estimating the illumination functions in the first pass. Radiosity techniques will be reviewed in the next subsection. In these techniques [19,20] the approximate illumination estimated in the first pass is stored as brightness at the centre of a small planar patch. Shao *et.al* [21] store a directionally dependent brightness function using a hemicube centred at each point of a specular surface patch. They present an iterative method of improving these initial estimates, which they call as procedural refinement.

All of these two pass techniques store the brightness emitting from the surface. Using the brightness values at all other surface points visible to any point on a surface in the environment the illumination function can always be computed. Buckalew *et.al* [22] on the other hand, explicitly store both the E and the I functions over each of the surfaces in the environment. This is done by maintaining a patch-link network data structure. Each incoming link is representative of a particular (θ, ϕ) direction of the E function at the patch. Similarly, each outgoing link is representative of a specific direction of the I function. At a patch point a reflectance map, a matrix of weights, is used to compute I for each of the outgoing links using E from each of the incoming links. In a preprocessing stage, the patch-link network data structure is established for a given environment. In the first pass light is distributed through the links in an iterative fashion until there is no appreciable change in illumination value in any of the incoming links. Uniformly distributed link directions are chosen so that incremental computations can be carried out. By choosing different reflection maps at different points varying optical properties may be simulated. As argued by the authors themselves, in the limiting case with infinite links, this can capture all possible illumination effects. In practice, however, the data structures can be prohibitively large.

2.4 Radiosity Techniques

Radiosity techniques are based on the energy equilibrium concept used in radiative heat transfer computations [23]. These are the very first techniques which modeled the illumination function not as delta functions in the (θ_s, ϕ_s) directions but as integrals over finite area light sources [24]. They also could accurately take into account all diffuse inter-reflections. In the basic radiosity technique, the brightness function is stored at the centre of small planar surface patches. Brightness is assumed not to vary over the entire patch. Further only perfectly diffuse surfaces are considered. Hence, brightness is represented independent of direction. Under these assumptions in the brightness equation the incident illumination function can be replaced by the brightness of the surface point visible to this point in that direction. Thus brightness at a surface point, say, I_i is given by

$$I_i = R \iint_{\Omega} I(\theta, \phi) \;\; cos\theta \;\; d\omega$$

Where $I(\theta, \phi)$ denotes the brightness of the other surface point visible to this point in the (θ, ϕ) direction. Recall that R is independent of (θ, ϕ) for perfectly diffuse surfaces.

Assuming surface subpatches of constant intensity the double integral can be reduced to a summation of brightness functions of the visible surface patches.

$$I_i = \frac{R}{A_i} \;\; \sum_j F_{ji} \;\; I_j A_j$$

Where F_{ji} is a geometrical term dependent on the geometry of patches i and j and the distance between them; and A_j is the surface area of the jth patch. $F_{ji}A_j$ accounts for the surface integral computation for the jth patch. Similar brightness equations for all the N surface patches in the environment result in N equations with N unknowns in I and are computed by any of the simultaneous equation solution methods.

This method when directly extended to non-diffuse surfaces cannot make use of the earlier simplification due to the directional nature of the reflectance function and hence the brightness function. The earlier equation with emitting directions becomes

$$I(\theta_r, \phi_r) = \iint_{\Omega} R(\theta, \phi, \theta_r, \phi_r) \;\; I_{\theta,\phi}(\theta^{-1}, \phi^{-1}) \;\; cos\theta \;\; d\omega$$

Where $I_{\theta,\phi}(\theta^{-1}, \phi^{-1})$ is the brightness from the surface along θ, ϕ direction in the reciprocal direction (θ^{-1}, ϕ^{-1}). In an attempt to solve this equation the (θ, ϕ) space was discretised to D directions [25]. The double integration was simplified to summation by discretising the environment to near differential patches. The resulting equation is given as:

$$I_i(\theta_r, \phi_r) = \sum_{j=1}^{N} \sum_{d=1}^{D} R(\theta_d, \phi_d, \theta_r, \phi_r) \;\; I(\theta_d^{-1}, \phi_d^{-1}) \;\; cos\theta_d \;\; \omega_d$$

If one chooses D to be of the order of 15,000 to 20,000, then even for moderately complex environments this technique becomes computationally intractable.

The basic computation that is repeatedly carried out in the diffuse radiosity techniques is the computation of form factor. In all, N^2 form factors are involved. Under the simplifying assumptions of perfectly diffuse surfaces and invariance of brightness over patches the form factor term reduces to being dependent only on the geometry. Form factor computation is expensive mainly because it involves a visible surface computation and a projection of the surface patch onto a hemisphere for computing the solid angle. Extensions to the basic radiosity technique are primarily based on the following:

(i) Increasing the efficiency/accuracy of form factor computation (The ingenious Hemicube technique devised by Cohen *et.al* is one such extension. [26,27,28]).

(ii) Avoiding unnecessary discretisation of the environment so as not to violate the condition of invariant brightness over the surface patch [29].

(iii) Taking directional aspect of illumination into account without making the total computation intractable [19,20,21].

(iv) Incremental techniques for solving the equations [30]. Progressive refinement was the name used by Cohen, *et.al* for this technique. An interesting angle to look at this method is to think in terms of light getting gradually propagated to other surfaces in the environment. A complete solution is not always needed as brightness values reach close to their equilibrium state early in the iteration process.

Because of the stringent assumptions made in the formulation of the energy equilibrium equations it has not been easy to extend radiosity techniques to deal directly with curved surfaces or to differentially handle diffuse and specular surfaces in an environment.

A major advantage of radiosity techniques is view independent representation of illumination functions. Rendering is a simple task given the illumination. Thus interactive walk throughs are possible even through geometrically complex environments [30].

2.5 Main Features of the Proposed Method

It should be quite clear by now that the nature of the illumination and brightness functions over a surface are so complex that continuous algebraic function representations for these is not feasible. Discretisation is unavoidable. All the methods discussed above compute the values of these functions in a discretised form. For example, in ray tracing discrete view directions are chosen subsequently resulting in other choices for discrete directions. When computations are view independent, discretisation of geometry also becomes essential. Hence in all two pass methods as well as in radiosity techniques, the geometry is assumed to be properly discretised. Similarly, when the directional nature of the functions is important then directions too have been discretised. Whenever an environment independent discretisation is chosen (planar approximations, uniform directions, hemicube, etc.) problems of aliasing and other undesirable artifacts show up. Taking into the account the complexity of all the computations involved, discretisation is necessary but careful environment-dependent discretisation can help considerably in reducing these problems.

In the rest of this paper, we describe a technique for determining the illumination functions over surfaces efficiently and accurately in a complex three dimensional environment. The directional aspect of illumination is specially taken into account. Hence, diffuse, specular or even more general reflectance properties may be associated with surfaces in the environment. The main features of this method are :

(i) A data structure called the spherical cover is devised to capture the directional nature of light distribution, keeping energy equilibrium conditions in mind. The spherical cover keeps enough information regarding the projections of any patch on the hemisphere of any other patch avoiding unnecessary discretisation of surfaces or directions. The spherical cover results in a data structure of order $O(N^2)$, storage.

(ii) Next an environment-dependent subdivision strategy is described for discretising the environment. Directions are not discretised. Hence directional aliasing inherent in other methods is not present in this. Recognising the fact that a reasonably accurate approximation to the illumination function is adequate, subdivision methods are devised based on range evaluations using interval arithmetic like methods over entire surface subpatches. Criteria for subdivision are determined by recognising the nature of light propagation from one surface to another.

(iii) An iterative algorithm is then described which uses the spherical cover and propagates light through the surfaces in the environment until equilibrium is reached. A complexity analysis of the propagation step is carried out to show its efficacy.

3 THE SPHERICAL COVER

3.1 Observations

We repeat below some of the important observations regarding illumination and the light propagation processes which have led us to the development of the spherical cover data structure to be described shortly. In what follows we use the term luminance to stand for either incident illumination or emitted brightness and the notation $L(\theta, \phi)$ to denote either of $E(\theta, \phi)$ or $I(\theta, \phi)$.

Observation 1: *In a closed environment equilibrium state illumination can be reached by progressively propagating light from bright emitting surfaces to other surfaces.*

To start with, all surfaces directly facing light sources will have their luminance function updated. This process is continued for surfaces facing other brightly illuminated surfaces until equilibrium is reached. Thus, at any time during this propagation process the luminance function at any point on a surface in the environment is an approximation of the final luminance function for that point in the equilibrium state.

Observation 2: *The luminance function over a surface is far too complex to have an algebraic function form of representation.*

It is best to visualise $L(u, v, \theta, \phi)$ as an implicit surface function, $L_p(\theta, \phi)$ defined over a parametrically defined surface $Q(u, v)$. If $L_p(\theta, \phi)$ functions are defined over discrete points in Q, say $(u_1, v_1), (u_2, v_2), \ldots (u_n, v_n)$ then using a suitable interpolant, $L_p(\theta, \phi)$ may be obtained for any point (u, v) on the surface.

Observation 3: *An accurate representation of $L(\theta, \phi)$ is a tessellated hemisphere with luminance functions associated with each subregion of the tessellation.*

Let P be a point receiving some of its light from or transmitting some of its light to a surface $Q_1(u, v)$ (*See* Fig.2). It should be recognised that some finite portion of $Q_1(u, v)$ is visible to P. This portion of $Q_1(u, v)$ occupies some region of the illumination hemisphere at P. The hemisphere must be tessellated such that this region occupied by Q_1 is distinct from others. If P is receiving light from Q_1 then over this subregion the incident illumination associated with it must be computed as the radiance emitted by the patch Q_1 in the bundle of directions reaching P. On the other hand, if P is transmitting light to Q_1 then over this subregion, the emitted brightness associated with it must be computed as the radiance emitted from P in the bundle of directions reaching Q_1.

Observation 4: *Either one of the two luminance functions, $E(\theta, \phi)$ or $I(\theta, \phi)$ may be explicitly stored at points on surfaces in the environment.*

Given an illumination function $E(\theta, \phi)$ the emitted radiance can be computed in any desired direction as and when needed using the equation below:

$$I(\theta_r, \phi_r) = \iint_{\Omega} R(\theta, \phi, \theta_r, \phi_r) \;\; E(\theta, \phi) \;\; cos\theta \;\; d\omega$$

The reflectance function $R(\theta, \phi, \theta_r, \phi_r)$ is determined by the optical properties of the surface on which the point lies. If it has a simple form like for perfect diffuse surfaces the emitted radiance may be evaluated analytically using a closed form formula. For more complex optical behaviour suitable weighted tables may be used to convolve $E(\theta, \phi)$ and obtain the I. Similarly, given $I(u, v, \theta, \phi)$ at all other surfaces visible to a point P, $E(\theta, \phi)$ may be easily computed.

Observation 5: *The directional variation of radiance functions is more difficult to characterise than spatial variation.*

Any mathematical function defined over a continuous domain may be characterised by a finite number of characteristics defined at discrete values of the domain. For example, a cubic polynomial function is characterised by function value and first derivative at two distinct values while a quintic requires second derivatives as well. Thus, we may choose to associate the luminance function with a subregion of the tessellated hemisphere at a point by its characteristics; say, value and derivatives with respect to u, v, θ and ϕ. As we shall see later, the bidirectional reflectance function for non-ideal surfaces is very complex and has been determined only experimentally for many surfaces. Also, since radiance depends on the integral of the convolution of the reflectance function with the incident illumination it is difficult to determine its characteristics with respect to direction. On the other hand, variation of luminance with respect to spatial distance depends on the surface normal. Surfaces are modeled with normals varying continuously with distance. Hence, we believe that spatial variation of radiance is easier to characterise.

Based on the above observations, we have made the following decisions which are elaborated upon later in this paper.

1. Efficiency of computations involved in light propagation is directly related to the number of discrete surface patches at which the luminance function has to be updated. A careful subdivision of the surfaces in the environment into discrete surface patches is necessary to limit this number (based on Observation 1).

2. The luminance function has to be stored in a discretised manner over the surfaces of the environment. For computational convenience, we store it at the approximate centre of the discrete surface patch and then interpolate between centres or neighbouring patches (based on Observation 2).

3. We use the (θ, ϕ) rectangular domain for representing the tessellated hemisphere. And with each tessellated subregion we store radiance function characteristics. In our present implementation of this concept, we assume that the radiance function varies linearly with respect to u and v on a surface and hence just store the radiance value and no derivatives. (based on Observations 3 and 5).

4. We store incident illumination $E(\theta, \phi)$ at each patch centre rather than emitted brightness $I(\theta, \phi)$. This we believe will result in lesser number of discrete surface patches and also lead to easier criteria for the subdivision process to be described later. (This is based on Observations 4 and 5). However, this is only our belief. We do not have any theoretical or experimental results in support of this. It should be noted that all earlier work stores emitted radiance at the surface patches. Thus, whenever light is propagated from a patch Q_1 to Q_2 the emitted

radiance stored at Q_1 projected towards Q_2 must be used to update the emitted radiance function at Q_2. On the other hand, in our proposed method, since we store incident illumination, we must evaluate the emitted radiance towards Q_2 and update the incident illumination at Q_2.

3.2 Characteristic Points in an Environment

The discrete points on each surface at which the illumination function $E(\theta, \phi)$ is explicitly stored are called as characteristic points. The characteristic points have to be chosen so as to be representative of the illumination of an associated area. Each surface in the environment has one or more characteristic points in it. The associated area is termed as a patch. For computational and representational convenience, we assume that a patch is a rectangle in the (u, v) domain of the surface. The methods for selection of characteristic points and associated surface areas are discussed in Section 4. Illumination functions defined at characteristic points are used along with suitable interpolants to determine the illumination function at any intermediate point (u, v) on the surface.

The notion of characteristic points has always existed in earlier work. In ray tracing the characteristic points vary for each ray. They are basically the ray surface intersection points. In radiosity, the characteristic point is assumed to be at the centre of a patch. It is worth repeating here that an optimal choice of characteristic points can capture the illumination function over surfaces in a complex 3D environment more efficiently and with greater fidelity.

Once an environment has been characterised by identifying all the characteristic points, the light propagation process will take place only amongst characteristic points. The brightest of the characteristic points will be chosen and its light propagated to others depending on its associated area and its projection on the hemisphere of the other characteristic points.

The method used to identify the characteristic points in an environment will depend on a number of factors:

(i) representation of the $E(\theta, \phi)$ function.

(ii) interpolant used for $E(\theta, \phi)$.

(iii) computation techniques used for obtaining $I(\theta, \phi)$ during light propagation.

(iv) representation of the associated surface patches.

(v) the radiance function characteristics characterising the illumination over the associated area.

Of course, ease and efficiency in identifying these characteristic points would also play a crucial role.

3.3 The Data Structure

Consider two characteristic points in the environment denoted by P_1 and P_2. Let $Q_1(u_1, v_1)$ and $Q_2(u_2, v_2)$ be the associated surfaces and E_1 and E_2 be the illumination functions. Assume that light has to be propagated from Q_1 to Q_2. We shall also assume that patch Q_1 is visible to patch Q_2. Let (θ_{21}, ϕ_{21}) denote the direction of P_1 at P_2. There is a bundle of directions around (θ_{21}, ϕ_{21}) forming a connected region on the hemisphere at P_2 say h_{21} in which Q_1 and only Q_1 contributes to E_2. Propagating light, i.e., updating E_2, would, therefore, require one to compute the radiance function coming out of Q_1 towards P_2 and associate it with the region h_{21} of the domain of E_2. Since by definition, P_1 is characteristic of $Q_1(u, v)$ we need to compute radiance function characteristics in the direction (θ_{12}, ϕ_{12}).

Based on the above discussion, we can make the following observation:

If H_i is the hemisphere at P_i, then H_i is of the form $\bigcup_{j=1}^{N} h_{ij}, i \neq j$ where h_{ij} is a connected subregion on the hemisphere at P_i and P_j denotes another characteristic point and N the total number of characteristic points in the environment. If Q_j is the associated patch of characteristic point P_j, then h_{ij} is the projection of Q_j on the hemisphere at P_i (*See* Fig.3). Every characteristic point is a potential candidate for being part of the tessellated hemisphere of every other point, provided it is visible. If we adopt the convention that $h_{ij} \neq$ *null* iff Q_j is visible to P_i, and *null* otherwise, and that a patch is not visible to itself then $H_i = \bigcup_{j=1}^{N} h_{ij}$ for all i ranging from 1 to N.

We call the data structure for storing all the H_is in an environment as the **spherical cover**. Clearly, the spherical cover is order $O(N^2)$ and is easily represented as an $N \times N$ matrix. In order to carry out the light propagation process, illumination function characteristics are associated with every h_{ij} along with other house-keeping variables. Fig.4 shows a cross-section of the tessellated hemisphere at a point representing the $E(\theta, \phi)$ illumination function.

There is a major representational issue to be considered before we end this discussion on the spherical cover — the representation of h_{ij} and hence H_i.

The hemicube introduced by Cohen *et.al* [26], is a potential candidate and has been used to capture directional brightness [21,25]. H_i would be the union of all the cells on the hemicube and h_{ij} would be a connected subset of cells of the hemicube. In this paper we have not used the hemicube for the following reasons:

- Cell boundaries of the hemicube tessellate the hemisphere in a highly irregular fashion. Since we intend to evaluate the integral to obtain I as and when needed, a regular tessellation of the (θ, ϕ) domain of the hemisphere is preferable.

- Equally important is the fact that we do not have immediate access to graphics hardware with a built-in Z buffer visible surface algorithm. Thus, the primary advantage of using the hemicube cannot be availed by us.

We have, therefore, chosen to discretise the (θ, ϕ) rectangular domain of the hemisphere into a rectangular grid of cells very much like a 2D pixel array. A subregion of the hemisphere is thus a connected collection of these grid cells. For compactness of storage, we store the subregion as a sequence of strips ranging from ϕ_1 to ϕ_2 in a row of the (θ, ϕ) cell array.

The precise spherical cover data structure H can now be described. Each element of the 2D matrix stores a pointer. Each row represents a hemisphere. A null value for the pointer $H(i, j)$ indicates that the j^{th} characteristic point does not contribute to the i^{th} hemisphere. Each column j gives all the spherical covers to which the j^{th} characteristic point contributes. The non-null pointer $H(i, j)$ points to a storage structure which carries information as shown in Fig.5.

3.4 Interpolation of the Hemispherical Tessellation

Given any (u, v) we determine the four characteristic points on the surface $(u_0, v_0), (u_0, v_1), (u_1, v_0), (u_1, v$ such that $u_0 \leq u \leq u_1$ and $v_0 \leq v \leq v_1$. We then bilinearly interpolate between $H(u_0, v_0), H(u_0, v_1), H(u_1, v_0)$ and $H(u_1, v_1)$. Since characteristic points are typically centres of patches, we bilinearly extrapolate for points nearer to the boundaries of the surface (*See* Fig.6).

Recall that $H_i = \bigcup_{j=1}^{N} h_{ij}$ and that each h_{ij} is either null or a sequence of strips, going from ϕ_1 to ϕ_2. Also one must note that characteristic points on a surface are reasonably closely spaced so that the neighbours of H_i are likely to be very similar. We, therefore, propose a simple algorithm for linearly interpolating H_1 to H_2 row by row.

A row of H_1 is an ordered sequence of strips $h_{1j_1}, h_{1j_2}, \ldots h_{1j_n}$ where $j_1..j_n$ are all in the range 1 to N. The same row of H_2 is another ordered sequence of strips $h_{2k_1}, h_{2k_2}, \ldots h_{2k_n}$ where $k_1..k_n$ are all once again in the range 1 to N. Noting that H_1 and H_2 are similar, we expect a large number of common elements between $j_1..j_n$ and $k_1..k_n$. If for some x, y $j_x = k_y$ then this means that the characteristic point P_{j_x} is present in both the hemispherical tessellations H_1 and H_2 (In H_2 it is denoted as P_{ky}). Linear interpolation of this strip for an in-between value is straight forward. A slight complication arises when the characteristic point P_{j_x} is present in H_1 and absent in H_2 or *vice versa.* The row by row interpolation algorithm for all these situations is best described diagrammatically as shown in Fig. 7.

3.5 Computation of Brightness Radiance

Propagation of light requires computation of radiance emitted from a characteristic point. Similarly final rendering of the scene requires the radiance of the point visible from the view point. The requirement is *Given the illumination function at a point, we have to calculate the radiance.* Evaluation of radiance requires double integration of terms including point illumination function and bi-directional reflectance. Before discussing the computation we will briefly explain the bi-directional reflectance.

Bidirectional Reflectance : $R(\theta_i, \phi_i, \theta_r, \phi_r)$.

The surface radiance and irradiance are related by this bidirectional reflectance. If the irradiance from the direction $(\theta_i \phi_i)$ is $\delta E_{ir}(\theta_i, \phi_i)$ then the radiance $\delta I(\theta_r, \phi_r)$ is given by:

$$\delta I(\theta_r, \phi_r) = R(\theta_i, \phi_i, \theta_r, \phi_r)\delta E_{ir}(\theta_i, \phi_i)$$

The surface radiance as a function of illumination from the whole hemisphere of radiance can be derived as follows:

Let $E(\theta_i, \phi_i)$ be the radiance from a unit solid angle coming from direction (θ_i, ϕ_i). Then the surface irradiance due to the illumination from the differential patch (Fig. 8) on the hemisphere in (θ_i, ϕ_i) direction is given by:

$$E(\theta_i, \phi_i)\ cos\theta_i\ d\omega_i$$

It can be shown that

$$d\omega_i = sin\theta_i\ d\theta_i\ d\phi_i$$

So the radiance is

$$\delta I(\theta_r, \phi_r) = R(\theta_i, \phi_i, \theta_r, \phi_r)\ E(\theta_i, \phi_i)\ sin\theta_i\ cos\theta_i\ d\theta_i\ d\phi_i$$

The radiance due to the illumination from all (θ_i, ϕ_r) directions on the hemisphere results in the familiar equation encountered in Section 1.2 earlier.

$$I(\theta_r, \phi_r) = \int_{\phi=0}^{2\pi} \int_{\theta=0}^{\pi/2} R(\theta_i, \phi_i, \theta_r, \phi_r)\ E(\theta_i, \phi_i)\ sin\theta_i\ cos\theta_i\ d\theta_i\ d\phi_i$$

3.5.1 Surface Reflectance Properties

The bidirectional reflectance is a material property and hence varies from surface to surface. For a Lambertian surface, the reflectance is isotropic. An ideal Lambertian surface is one that appears equally bright in all directions and reflects all incident light absorbing none. For a Lambertian surface $R(\theta_i, \phi_i, \theta_r, \phi_r)$ is a constant independent of incident and reflective direction making I independent of (θ, ϕ). For the ideal Lambertian surface the constant can be calculated by integrating the radiance of the surface over all directions and equating with surface irradiance E_{ir}:

$$I \int_0^{2\pi} \int_0^{\pi/2} sin\theta_r\ cos\phi_r\ d\theta_r\ d\phi_r = E_{ir}$$

or

$$I\ .\ \pi\ = E_{ir}\ \ or\ \ I\ \ = \frac{E_{ir}}{\pi}$$

So $R(\theta_i, \phi_i, \theta_r, \phi_r)$ for an ideal Lambertian surface $= \frac{1}{\pi}$.

For any non-ideal diffuse surface the reflectance may be taken as a constant fraction of $\frac{1}{\pi}$. So for a surface illuminated from direction θ_1 to θ_2 and ϕ_1 to ϕ_2

$$I = \frac{k_{Lambert}}{\pi} \int_{\phi_1}^{\phi_2} \int_{\theta_1}^{\theta_2} E_{\theta_i,\phi_i} sin\theta_i\ cos\theta_i\ d\theta_i\ d\phi_i$$

Since we have assumed E_{θ_i,ϕ_i} to be constant over the associated area of a characteristic point, we have

$$I = \frac{k_{Lambert}}{4\pi} E_{\theta_i,\phi_i}(\phi_2 - \phi_1)(cos2\theta_1 - cos2\theta_2)$$

Such a simple relationship does not hold for surfaces exhibiting anisotropic reflectance. The simplest of such surfaces is an ideal mirror surface which reflects all of the light arriving from the direction (θ_i, ϕ_i) into the direction $(\theta_i, \phi_i + \pi)$. In the case of ideal mirror reflection, the bidirectional reflectance is proportional to the product of two delta factors $\delta(\theta_r - \theta_i)$ and $\delta(\phi_r - \phi_i - \pi)$.

$$I(\theta_r, \phi_r) = \int_0^{2\pi} \int_0^{\pi/2} k \ \delta(\theta_r - \theta_i) \ \delta(\phi_r - \phi_i - \pi) E(\theta_i, \phi_i) \ sin\theta_i \ cos\theta_i \ d\theta_i \ d\phi_i$$

$$I(\theta_r, \phi_r) = E(\theta_r, \phi_r - \pi)$$

requires

$$k \ = \frac{1}{sin\theta_i \ cos\theta_i}$$

So, for a mirror surface with some absorption

$$R(\theta_i, \phi_i, \theta_r, \phi_r) = k_{specular} \ \frac{\delta(\theta_r - \theta_i)\delta(\phi_r - \phi_i - \pi)}{sin\theta_i \ \ cos\theta_i}$$

where $0 \leq k_{specular} \leq 1$ accounts for the absorption in the surface. Shiny mirror like surfaces are modelled with delta factors giving non-zero value for a small angle around (θ_r, ϕ_r) the value falling rapidly from 1 to 0 as a function of the angle spread α by $cos^n\alpha$.

Particular surface reflectances can be determined experimentally by illuminating a flat sample of the material of interest with a lamp mounted on a goniometer and measuring its irradiance using a sensor mounted on another goniometer. However, the experimental determination is quite tedious because of the four variables involved. The other way to obtain reflectance is to model how light is reflected from a surface and to find the corresponding reflectance properties analytically or by numerical simulations. This has been done for shiny surfaces with suitable approximations. Since the analytical forms are quite complex to integrate, we use a weighted table, a reflectance map table.

To summarise, given the illumination function in the form of hemispherical cover and the associated radiance information we will use the following equations:

For A Diffuse Surface

$$I(\theta_r, \phi_r) \quad = \quad \frac{k_{Lambert}}{4\pi} \sum_j e(cos2\theta_{j1} - cos2\theta_{j2})(\phi_{j2} - \phi_{j1})$$

where j ranges over all the strips in a subregion of H and e denotes the incident radiance associated with it.

For A Mirror Surface

$$I(\theta_r, \phi_r) \quad = \quad k_{specular} \ e(\theta_r, \phi_r - \pi)$$

where $e(\theta_r, \phi_r - \pi)$ is the incident radiance associated with the subregion containing the $(\theta_r, \phi_r - \pi)$ direction.

For A Surface With General Reflectance

$$I(\theta_r, \phi_r) \quad = \quad \sum_{i=0}^{\pi/2} \sum_{j=0}^{2\pi} \; w_{ij} \;\; e_{ij}$$

where w_{ij} gives the weightage and e_{ij} is the incident radiance in the bundle of directions denoted by the grid cell ij. *Note:* $\sum_i \sum_j w_{ij} \leq 1$. w_{ij} is precomputed and set up in tables depending on surface properties.

4 SUBDIVISION TECHNIQUES FOR COMPUTING THE SPHERICAL COVER

Computing the spherical cover of an environment would require

(i) identification of characteristic points on each of the surfaces in the environment

and

(ii) determining the hemispherical tessellation for each of the characteristic points.

For a point to be characteristic of the associated patch let us recall the main requirements:

(i) *Full Visibility:*
If P_i is in the hemispherical tessellation of P_j then the associated areas Q_i and Q_j are fully visible to each other. This implies that there is no self shadowing or hiding by other patches in the environment.

(ii) *Hemispherical Containment:*
If P_i is in the hemispherical tessellation of P_j then the patch Q_i must be contained within a hemisphere around the normal at P_j.

(iii) *Spherical Shape:*
If light is distributed from P_i to P_j then for this to be representative of the associated area all points on Q_i must be roughly at the same distance with the proper normal orientation. This is equivalent to saying that the shape of Q_i must be as spherical as possible with P_j as the centre.

(iv) *Characterisability:*
If P_j is receiving light from Q_i then P_j should be characteristic of the associated region Q_j with regard to receiving light from P_i. This implies that the normal over Q_j should not vary very significantly with respect to the direction $P_i\ P_j$.

Determining the hemispherical tessellation at a characteristic point is equivalent to a complete visible surface algorithm being performed with that point as the eye point. This is an extremely computation-intensive operation if one considers that the number of characteristic points in an environment may be in thousands. However, radiance values needed at any surface point are required only as accurately as the resolution of intensity in the imaging medium. Hence, reasonably correct estimates are all that are sought. This means that one need not be very strict regarding the criteria listed above. All conditions may be satisfied within acceptable tolerance values. For example, a patch shape may be said to be spherical if normal is always nearly, but not necessarily exactly, facing the receiving characteristic point. Based on this observation, in this section, we describe environment dependent subdivision techniques for computing the spherical cover. The subdivision techniques use interval arithmetic like range estimation methods to check if entire patches satisfy the criteria listed above.

4.1 A Brief Review of Interval Methods

The effective use of interval arithmetic-based [31] subdivision techniques for processing of geometric objects has already been demonstrated [32]. Here, we briefly review the main concepts.

An interval is a set of real numbers, defined by an ordered pair as below:

$$[a, b] = \{x \mid a \leq x \leq b\}$$

If @ represents one of $+, -, *, /$ then we can define interval arithmetic as follows:

$$[a, b]@[c, d] = \{x@y \mid x\epsilon[a, b] \wedge y\epsilon[c, d]\}$$

except that we do not define $[a, b]/[c, d]$ in case $0\epsilon[c, d]$. An equivalent set of definitions in terms of end values is given below:

$$\begin{aligned}
[a, b] + [c, d] &= [a + c, b + d] \\
[a, b] - [c, d] &= [a - c, b - d] \\
[a, b] * [c, d] &= min[(a * c, a * d, b * c, b * d), \\
&\quad max(a * c, a * d, b * c, b * d)] \\
[a, b]/[c, d] &= [a, b] * [1/d, 1/c]
\end{aligned}$$

provided 0 *does not belong to* $[c, d]$.

Note that a real number may be denoted by the interval $[a, a]$. The use of interval arithmetic gives a direct and simple method of calculating bounds to the variation of any well defined rational function over a certain set of intervals within which the arguments of the function vary. For example, given a parametric surface defined by three polynomials, say,

$$x(u, v), y(u, v), z(u, v), \quad u, v\epsilon\, [0, 1]$$

then interval evaluation of these polynomials

$$x([0, 1], [0, 1]), \quad y([0, 1], [0, 1]), \quad z([0, 1], [0, 1])$$

yields three intervals $[x_1, x_2], [y_1, y_2]$ and $[z_1, z_2]$ which define a bounding box for the surface patch. The bounding box is not exact. However, methods exist to get better range estimates. Note that interval methods are one way of getting bounds to functions. Other methods if suitable may be used.

4.2 Formulation of Subdivision Criteria

4.2.1 Self-Shadowing Criterion

We know that there is no self-shadowing in patch Q_j viewed from a point P_i if the normal at any point on Q_j is always facing P_i. Let $Q_j(u_j, v_j)$, with $u_l < u_j < u_h, v_l < v_j < v_h$ be the vector function defining any point on the patch Q_j. Let $N_j(u_j, v_j)$ be the normal at the point (u_j, v_j), where $N_j(u_j, v_j) \equiv \frac{\delta\, Q_j}{\delta\, u_j} \times \frac{\delta\, Q_j}{\delta\, v_j}$. Then, $N_j(u_j, v_j)$ will be facing P_i for all (u_j, v_j) iff
$f = N_j(u_j, v_j).(Q_j(u_j, v_j) - P_i)$ is always > 0.

Similarly, if no point on Q_j is facing towards P_i, then $f \leq 0$ for all (u_j, v_j). Let the range of all values of the above expression namely, $N_j(u_j, v_j)\ (Q_j(u_j, v_j) - P_i)$ be, say, $[f_{low}, f_{high}]$. If Q_j has no self-shadow with respect to P_i then $f_{high} > f_{low} > 0$. Similarly, if Q_j is completely facing the other way, then $f_{low} < f_{high} \leq 0$. Using some efficient method for determining good bounds for the function, one can determine whether an entire patch is fully facing front or fully facing back or part front, part back. In the last case, subdivision is carried out. A point to be noted here is that approximate self-shadowing boundaries within reasonable tolerance are acceptable for the light distribution process. Hence, one does not need to carry out the subdivision to the level of detecting the silhouette curves of Q_j with respect to P_i. Fig. 9 shows a doubly curved patch being subdivided using this criteria. Interval methods proposed in [33] have been used. As can be seen, many large visible/shadowed regions get identified very early in the subdivision process.

Note that we have formulated the self-shadowing criteria only with respect to the characteristic point P_i and not with respect to all points of the patch Q_i. Since Q_i will later get subdivided when it forms part of the spherical cover of other characteristic points, we have made this simplifying assumption to avoid too much of subdivision.

4.2.2 Hemispherical Containment Criterion

This means that all points of Q_j lie within a hemisphere around N_i. This can be formulated as saying that

$$N_i.(Q_j(u_j, v_j) - P_j) > 0 \text{ for all } (u_j, v_j)$$

4.2.3 Spherical Shape Criterion

The spherical shape requirement would imply that the normal $N_j(u_j, v_j)$ is along the same direction as $Q_j(u_j, v_j) - P_i$.

This is a very stringent condition and if applied, even within reasonable tolerance limits may result in excessive subdivision of surfaces. Since emitted radiance is usually independent of direction for a Lambertian surface, we do not think that the normal condition should be strictly checked if Q_j is a diffuse surface. We use it only for non-isotropic surfaces.

Once again this can be formulated as a range evaluation criterion:
$\widehat{N_j}(u_j, v_j).(Q_j(u_j, v_j) - P_i) \approx \mid P_j - P_i \mid$

For diffuse surfaces we use a less stringent condition which says that the distance of any point on Q_j from P_i is nearly constant. The distance constant condition becomes

$$(Q_j(u_j, v_j) - P_i).(Q_j(u_j, v_j) - P_i) \approx \mid P_j - P_i \mid^2$$

4.2.4 Characterisability Criterion

This is the condition which says that P_i is characteristic of its associated area for receiving light from P_j. This can be formulated by saying that $N_i(u_i, v_i).(Q_i(u_i, v_i) - P_j) \approx$ constant.

4.2.5 The R Buffer Criterion

All the criteria listed above are local to the two patches under consideration. Full visibility, however, depends on being not hidden by any other patch. For this a Z-buffer like algorithm is used. Projection in (θ, ϕ) space is used to determine hiding/visibility. This is done by the use of an R-buffer in which the distance R in spherical coordinates is used in place of Z. Any patch which is considerably hidden is subdivided.

4.3 The Subdivision Process

The primary task of the subdivision process is to compute the spherical cover. In the process characteristic points will also be identified. Let $S \equiv S_1, S_2, \ldots S_n$ be the n surfaces constituting the environment. Each is defined parametrically with its own parametric space say (u_i, v_i) ranging over intervals. To start with we may assume all parameters to be ranging over $[0, 1]$. The subdivision is carried out as described procedurally below:

Assume that $Q \equiv Q_1, Q_2, \ldots Q_N$ is the patch list which finally forms the subdivided environment. Initially, Q is set to be equal to S. In the procedure described below steps 1 to 5 are carried out for all patches in Q including new ones added as part of the subdivision process. Assuming Q_i is the patch under consideration:

Step 0: for all Q_i do
{identify the parametric centre as the characteristic point P_i of Q_i and the total parametric range rectangle as the associated area}

Step 1: Set up $T = T_1, T_2, \ldots T_{n-1} = Q - Q_i$ as unprocessed list

Step 2: While unprocessed list not empty do
{ Pick a T_j from this list for processing. Let P_j be its parametric centre. Check for hemispherical containment and self-shadowing between T_j and P_i
If *yes*, subdivide T_j, add relevant subpatches to T and repeat check
Check if T_j is nearly spherical
If not, subdivide T_j, add to T and repeat check
Check if P_i properly characterises the associated area Q_i for receiving light from P_j.
If not subdivide Q_i, add to Q and to T and repeat check.
Compute the polar projection of T_j on the hemisphere of P_i and encode in the form of strips in the (θ, ϕ) space of P_i. This can be done by converting the boundary curves of T_j into 3D polar coordinates and then seed filling the region in the R-buffer. Using R the distance between P_i and P_j as the Z value paint this into the R buffer as one would do for the Z buffer. That is, those (θ, ϕ) grid cells, which have larger R in the R buffer are replaced by the grid cells with smaller R in T_j's polar projection. The characteristic point identifier j is also remembered with each (θ, ϕ) grid cell in the R-buffer.
}

Step 3: for all $T_j \neq Q_i$ do
{ By comparing polar projection of T_j with its image, if any, in the R buffer, determine
If T_j is fully visible, fully hidden or partially visible to P_i.
If partially visible then check connectedness of region and extent of hidden part.
If disconnected or hidden substantially then subdivide T_j, add to T, compute polar projections of sub-patches. Repeat subdivision until the polar projection of a subpatch of T_j matches in the R-buffer. }

Step 4: Extract h_{ij} for each j as a collection of strips from the R buffer. If j not in R buffer, then set h_{ij} to null.

Step 5: Modify Q by suitably replacing all subdivided patches which are not completely hidden. Wherever applicable interpolating the hemispherical tessellation is interpolated.

Note :

1. Patches are checked for satisfying the above conditions only within certain prescribed tolerances. Low tolerance values would result in more subdivision. Tolerance limits can be set depending on the imaging system and on the method used for range evaluation.

2. Subdivision is carried out only for a few levels. In any case, subdivision is not carried out beyond the level where a patch projects to a small region in the (θ, ϕ) grid space, in the limit just a single (θ, ϕ) grid cell.

3. A patch is subdivided into three pieces along u or v as shown in the figure Fig. 10. The division is chosen so as to maintain relatively square shaped subpatches in Euclidean space.

4. Step 5 ensures that all patches completely hidden from the point do not get subdivided.

5. Wherever applicable, the hemispherical tessellation is interpolated at the centre of a newly formed subpatch. The hemispherical tessellation computation is expensive and this way it is avoided whenever possible. This, however, results in the following situation:

   A patch Q_j appears as a whole over the hemisphere of one characteristic point, say P_i but may be subdivided when appearing over another characteristic point say P_k. This is quite acceptable as what it means is that propagation of light from Q_j to P_i is characterised by a single direction while Q_j to P_k is not.

5 Propagation of Light

Light is progressively propagated starting from the brightest characteristic points and continued with less bright points. Brightest points are selected using a brightness measure calculated from the illumination functions of the points and emission function characteristics of the surfaces to which the characteristic points belong. The quantitative measure of the brightness is given by integrating the incident function taking the attenuation due to absorption and adding the emittance function (ϵ).

$$
\begin{aligned}
B &= (k_d + k_s) \iint E(\theta, \phi)\ sin\theta\ cos\theta\ d\theta\ d\phi \\
&+ \iint \epsilon\,(\theta, \phi)\ sin\theta\ cos\theta\ d\theta\ d\phi
\end{aligned}
$$

This quantitative measure of brightness of the surfaces is precomputed and is updated during the light propagation process.

5.1 Algorithm

do {

```
process_set = set of all characteristic points
no_equilibrium = 0.
do {
     Remove the brightest characteristic
     point from process set and let it
     be P_i.
     For all hemispherical tessellations H_j, with
     P_i ϵ H_j {
             Compute I(θ, φ) in direction P_i − P_j
             Update the radiance function entry
             e(θ, φ) associated with h_ji.
             If there is any appreciable change
             then update no_equilibrium.
             }
     } while process_set not empty.
} while (no_equilibrium)
```

5.2 Complexity Analysis

Let N be the number of characteristic points in the environment. As can be seen from the above algorithm the basic step is the propagation of light from one characteristic point to other characteristic points in the environment. Let η be the cost of this propagation. Progressively propagating light through the environment until equilibrium is reached is equivalent to solving N simultaneous equations [30]. Hence in the worst case, the computational complexity is $O(\eta N^2)$. However, Cohen, *et.al* [30] have shown that the number of iterations necessary is much less than N^2 on the average. We shall, therefore, analyse only the cost of η. In our method the basic step of propagating light from a characteristic point to another requires

(i) evaluation of $I(\theta_d, \phi_d)$ from $E(\theta, \phi)$.

(ii) updating $E(\theta_d^{-1}, \phi_d^{-1})$ at the receiving characteristic points.

The total number of updates in the worst case is N, for a convex environment. Updating E is simple in our method and has a constant time, independent of N, say c.

- Let τ be the cost of evaluating the integral.
- The number of times I has to be evaluated depends on the surface type.
  - For a diffuse surface I is independent of direction. Hence I can be computed only once and then used for all updates. η in this case, now becomes $O(N)$.
  - For a perfect mirror, I is non-zero only for the reflected direction. Once again η is $O(N)$.

– For a surface with general reflectance properties, computation of I is dependent on E from all directions.

  Let D be the number of grid cells in the total (θ, ϕ) space at a point. Then τ, the cost of evaluating the integral I is $O(D)$. The cost of propagation, η is $O(ND)$.

Two points need to be noted here:

(i) On the average a point is not likely to affect all the other characteristic points but some fraction of N, as, many are likely to be invisible.

(ii) Depending on the optical nature of the surface, there may be large number of grid cells with 0 weights in the reflectance map table. Hence, by suitable choice of an access data structure the cost of τ can be made much less than D.

In summary for completely diffuse plus perfect mirror environments, the complexity of propagation is same as the basic radiosity technique. For general environments, the worst case complexity is $O(ND)$. But the average complexity will be lower. In comparison in the earlier view independent method [25], when it is modeled not as a set of ND linear equations but as progressive propagation, the worst case complexity of the propagation step is $O(D^2)$, where D is the number of discretised directions. Note that, in general, $D > N$. In our method, we do not have to break up the environment into differential elements. Also, using the subdivision strategy described in Section 4, N can be reasonably contained. It is clear, therefore, that this method is a significant improvement over the earlier method.

5.3 Scene Rendering

Progressive propagation will eventually result in an equilibrium state illumination function at each of the characteristic points. Any standard visible surface algorithm may be used to carry out the final rendering process. The brightness assigned to a visible point is dependent on the illumination function of the surface on which this point lies. For any intermediate visible point, the illumination function has to be obtained by interpolation/extrapolation of neighbouring characteristic points.

Given $E(u_1, v_1, \theta, \phi), E(u_2, v_2, \theta, \phi), E(u_3, v_3, \theta, \phi), \ldots E(u_n, v_n, \theta, \phi)$ over a surface the task is to define $E(u, v, \theta, \phi)$ and/or $I(u, v, \theta, \phi)$ at any point on the surface. Throughout in the formulation of our solution to the illumination problem, we have assumed that a complete $E(u, v, \theta, \phi)$ is stored and $I(u, v, \theta, \phi)$ in any desired direction is computed as and when needed. Thus, there are two possibilities :

- $E(u, v, \theta, \phi)$ to be determined.
- $I(u, v, \theta_d, \phi_d)$ to be determined for a specific direction (θ_d, ϕ_d).

5.3.1 Interpolation of the Illumination Function E

Basically, given hemispherical tessellations with associated radiance function characteristics at some discrete points, we must compute the hemispherical tessellation with associated radiance function characteristics at all intermediate points. We treat this in three steps:

1. Goven any (u, v) we determine the four characteristic points on the surface $(u_0, v_0), (u_0, v_1), (u_1, v_0), (u_1, v_1)$ such that $u_0 \leq u \leq u_1$ and $v_0 \leq v \leq v_1$.

2. Interpolate the hemispherical tessellation as described in Sec.3.4.

3. Interpolate the associated radiance function characteristics. This can be done once again using bilinear interpolation.

5.3.2 Interpolation of the Brightness

Since for the final rendering phase, we only need the brightness at a visible point in the view direction, instead of interpolating $E(u, v, \theta, \phi)$ at that point and then obtaining brightness by integration, we can also directly obtain it by bilinearly interpolating the I's in that direction as shown in Fig. 11. However, for textured surfaces, we recommend that $E(\theta, \phi)$ be interpolated and then brightness computed in the view direction using the reflectance map.

5.4 Concluding Remarks

Even though the method was presented assuming only opaque surfaces, it is quite simple to extend it to deal with transparent surfaces as well. For this, with characteristic points of a transparent surface, we must have an hemispherical tessellation in a direction opposite to the normal as well. Textured surfaces can also be dealt with by convolving the texture maps of contributing surfaces with the surface reflectance map and using it during light propagation. Currently we are implementing this method in C on a VAX 8600 system running the ULTRIX Operating System and we hope to provide more experimental results in the near future. We are also extending this method to deal with participating media.

6 ACKNOWLEDGEMENTS

We are grateful to Dr. S. Ramani, Director, NCST for his encouragement and support. We thank Prof. G.R. Shevare, Dr. S. Gopalsamy, Dr. P.A. Koparkar and Mr. D.R. Khandekar for fruitful discussions and help on various topics of this work. We deeply acknowledge the Herculean efforts put in by Mr. V. Padmanabhan in LaTeXing this manuscript in record time.

7 REFERENCES

1. Arvo James, "Backward Ray Tracing", Developments in Ray Tracing, *SIGGRAPH'86 Course Notes*, 1986.

2. Baum Daniel R., Holly E. Rushmeier, James M. Winget, "Improving Radiosity Solutions Through the Use of Analytically Determined Form-Factors", *SIGGRAPH'89 Conference Proceedings*, 325-334 (1989).

3. Blinn James, "Models of Light Reflection for Computer Synthesised Pictures", *SIGGRAPH'77 Conference Proceedings*, 192-198, (1977).

4. Buckalew Chris, Donald Fussell, "Illumination Networks: Fast Realistic Rendering with General Reflectance Functions", *SIGGRAPH'89 Conference Proceedings*, 89-98 (1989).

5. Catmull E., "A Subdivision Algorithm for Computer Display of Curved Surfaces", Ph.D. Thesis, Computer Science Dept., Univ. of Utah, Salt Lake City, 1974.

6. Chottopadhyay Sudeb, Akira Fujimoto, "Bidirectional Ray Tracing", *Computer Graphics 1987*, 335-343 (1987).

7. Cohen Michael F., Donald Greenberg, "The Hemi-Cube: A Radiosity Solution for Complex Environments", *SIGGRAPH'85 Conference Proceedings*, 31-40 (1985).

8. Cohen Michael F., Shenchang Eric Chen, John R. Wallace, Donald Greenberg, "A Progressive Refinement Approach to Fast Radiosity Image Generation", *SIGGRAPH'88 Conference Proceedings*, 75-84 (1988).

9. Cohen Michael F., Donald Greenberg, David S. Immel, Philip J. Brock, "An Efficient Radiosity Approach for Realistic Image Synthesis", *IEEE Computer Graphics and Applications*, 6(2), 26-35 (1986).

10. Cook Robert, Thomas Porter, Loren Carpenter, "Distributed Ray Tracing", *SIGGRAPH'84 Conference Proceedings*, 137-145 (1984).

11. Farin Gerald, *Curves and Surfaces for Computer Aided Geometric Design, A Practical Guide*, Academic Press, 1988.

12. Faux Ivor, Michael Pratt, *Computational Geometry for Design and Manufacture*, John Wiley & Sons, 1989.

13. Glassner Andrew, "An Overview of Ray Tracing", Introduction to Ray Tracing, *SIGGRAPH'88 Course Notes*, 1988.

14. Goral Cindy, K.E.Torrance, Donald Greenberg, Bennett, "Modeling the Interaction of Light Between Diffuse Surfaces", *SIGGRAPH'84 Conference Proceedings*, 213-222 (1984).

15. Horn Berthold K.P., *Robot Vision*, The MIT Press, 1986.

16. Immel David, Michael F. Cohen, Donald Greenberg, "A Radiosity Method for Non-Diffuse Environments", *SIGGRAPH'86 Conference Proceedings*, 133-142 (1986).

17. Kajiya James T., "Ray Tracing Parametric Patches", *SIGGRAPH'82 Conference Proceedings*, 245-254, (1982).

18. Moon P., D.E.Spencer, *The Photic Field*, The MIT Press, 1981.

19. Moore R.E., *Interval Analysis*, Prentice Hall, 1966.

20. Mortenson Michael E., *Geometric Modeling*, Wiley, 1985.

21. Mudur S.P., "A General Schema for Handling Curves and Surfaces in Geometric Modeling", *NICOGRAPH'83 Conference Proceedings*, 213-247 (1983).

22. Mudur S.P., P.A.Koparkar, "Interval Methods for Processing Geometric Objects", *IEEE Computer Graphics and Applications*, 4(2), 7-16 (1984).

23. Phong Bui_Tuong, "Illumination for Computer Generated Pictures", *Communications of the ACM*, 18(6), 311-317 (1975).

24. Rogers David F., *Procedural Elements for Computer Graphics*, McGraw-Hill, 1985.

25. Roth S.D., "Ray Casting for Modeling Solids", *Comp. Graphics and Image Processing*, 18, 109-144 (1982).

26. Shao Min-Zhi, Qun-Sheng Peng, Yon-Dong Liang, "A New Radiosity Approach by Procedural Refinements for Realistic Image Synthesis", *SIGGRAPH'88 Conference Proceedings*, 93-101 (1988).

27. Siegel Robert, John R. Howell, *Thermal Radiation Heat Transfer*, Hemisphere Publishing Corporation, 1981.

28. Sillion Francois, Claude Puech, "A General Two-pass Method Integrating Specular and Diffuse Reflection", *SIGGRAPH'89 Conference Proceedings*, 335-344 (1989).

29. Thalmann Nadia Magnetat, Daniel Thalmann, *Image Synthesis, Theory and Practice*, Springer-Verlag, 1987.

30. Wallace John, Kells A. Elmquist, Eric A. Haines, "A Ray Tracing Algorithm for Progressive Radiosity", *SIGGRAPH'89 Conference Proceedings*, 315-324 (1989).

31. Wallace John, Michael Cohen, Donald Greenberg, "A Two-Pass Solution to the Rendering Equation: A Synthesis of Ray Tracing and Radiosity Methods", *SIGGRAPH'87 Conference Proceedings*, 311-320 (1987).

32. Ward Gregory R., Francois M. Rubinstein, Robert D. Clear, "A Ray Tracing Solution for Diffuse Interreflection", *SIGGRAPH'88 Conference Proceedings*, 85-92 (1988).

33. Whitted Turner, "An Improved Illumination Model for Shaded Display", *Communications of the ACM*, 23(6), 343-349 (1980).