Shape Approximation for Low-Cost Refraction Models[1]

Gustavo A. Patow

LIFIA–Laboratorio de Investigación y
Formación en Informática Avanzada
Universidad Nacional de La Plata
c.c.11 – La Plata – ARGENTINA
dagush@ada.info.unlp.edu.ar

Claudio A. Delrieux

Departamento de Ingeniería Eléctrica
Instituto de Ciencias e Ingeniería de Computación
Universidad Nacional del Sur
Alem 1253 – Bahía Blanca – ARGENTINA
usdelrie@criba.edu.ar

Abstract

We present a shape approximation formalism adequate to compute environment-mapped refractions. Translucid objects are approximated as spherical lenses. The geometrical parameters of the lenses are easily derived from the polygonal description of the objects. We then develop a refraction model that leads to a computationally inexpensive procedure to compute accurate refractions from environment maps.

1 Introduction

Successful implementations of refractions are generally based on standard ray tracing implementations [3]. Such implementations achieve optimal results since the exact geometric behavior of light paths is computed. The computational expense of ray tracing provides a motivation to investigate lower cost approximation algorithms based on scan line algorithms [14]. Blinn and Newell approximations [1] provide an overall adequate solution to photorrealistic reflections rendering in terms of cost, efficiency and accuracy. These scan-line approximations are based on environment map implementations, and more specifically, reflection map implementations. The issue of refraction models, however, was not addressed in these approximations. Other simple refraction models, generally based on transparencies and alpha blending, obviously lack realism, since the most important aspect of the problem –*i.e.*, the geometry of the refraction of light paths– is not addressed.

The aim of this work is to extend the general philosophy of environment maps to refraction models. To do this, we develop a shape approximation model, which is both geometrically adequate and allow a computationally inexpensive refraction algorithm. An outline of the paper is as follows: In the next section we present a brief review of current scan-line refraction models, together with a discussion of their major contributions and drawbacks. Then we contend that an obvious alternative to low-cost refraction models is to adapt Blinn-Newell reflection maps to compute refractions. We discuss in section 3 the necessary object shape approximation conditions to compute geometrically adequate refraction maps. This leads to the main results of this work, where the issue of shape approximation for refraction models is addressed. To make this work self-contained, in section 4 we give an outline of refraction geometry. In section 5 some useful approximations and known particular cases are recast in terms of the proposed formalization. In section 6, some environment map implementation issues and examples are presented. In section 7, finally, we discuss the conclusions and future work.

[1]This work was partially supported by a grant of Fundación Antorchas

Figure 1: Kay and Greenberg refraction model

2 Review of scan-line refraction models

Most scan-line rendering schemes can be somehow adapted to support translucid object representation and its associated rendering. The simplest translucid object rendering scheme in priority list implementations is to ignore refraction at all, so that light rays are not refracted when passing through the interface between two optical media [2]. Then, the visibility along an optical ray coincides with the geometric (straight) line between the camera (observer) and the object to be rendered.

There are commonly two methods to compute transparencies. In *interpolated transparency*, the shading value of a pixel p that intersects two polygons P_1 and P_2 of a translucid object, is determined with a linear interpolation between the shading value of the two polygons: $I_f(p) = (1-k)I_{P_1}(p) + kI_{P_2}(p)$, where $0 \geq k \geq 1$ is the transmission coefficient, which depends in general on the wavelength. In *filtered transparency*, polygons are considered as translucid filters: $I_f(p) = I_{P_1}(p) + O.k.I_{P_2}(p)$, where O is the transparency color of P_1 at the considered wavelength. It is generally recommended [2] to interpolate only the ambient and diffuse component of P_1 with the complete shading equation of P_2, and after that add the specular component of P_1.

A further enhancement was proposed by Kay and Greenberg [6], where the transmission coefficient k is a non linear function of z_n, the z component of the normal. Kay and Greenberg sugested an expression $k = k_{min} + [k_{max} - k_{min}].[1 - (1 - z_n)^m]$, where k_{min} and k_{max} are the minimal and maximal transparencies of the object, and m is an arbitrary coefficient, normally between 2 and 3, where a greater m represents a thinner object (see figure 1).

Z-buffer rendering of translucid objects can be quite cumbersome. Objects are rendered in an arbitrary order, and since the buffer does not store enough information to determine which opaque objects are behind translucid objects. An incomplete solution to this is to implement ad-hoc ordering techniques, for example, to render opaque objects first and translucid objects last. All these scan-line approximations, however, obviously lack realism, since the most important problem –*i.e.*, the geometry of the refraction of light paths– is not addressed.

Other proposed techniques are based on sophisticated algorithms and data structures. Mammen, for example [7] considers a back-to-front ordering scheme with a multipass rendering and auxiliary data structures. First, all opaque objects are conventionally scan converted and z-buffered. Then, translucid objects are back-to-front processed with a transparency z-buffer. Every translucid object is scan-converted with an additional buffer that stores, per pixel, a transparency value, color, z value initialized to z_{min}, and a flag bit initialized to false. If the z value of a given pixel is closer to the observer than the corresponding z value in the opaque z-buffer, but is farther than the corresponding z value in the transparency z-buffer, then the last z-buffer is updated with the color and z value of the pixel, and the flag bit of the pixel is set to true. After all translucid objects have been processed, in the transparency z-buffer we have information about every farthest object whose pixel was set to true. Then the information of the these pixels is combined with the opaque z-buffer and frame buffer, and the flags are set

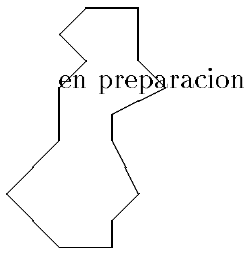

Figure 2: Approximation of a generic object with a lens.

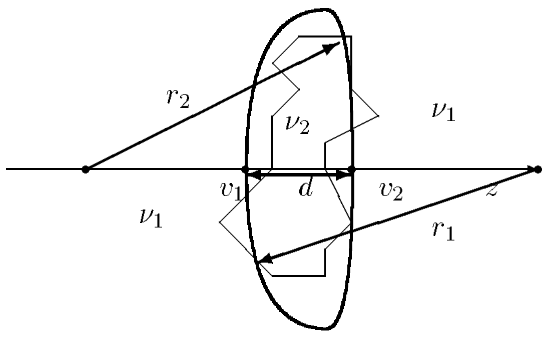

Figure 3: Geometric parameters of the lens.

again to false. This process is repeated back-to-front with all translucid objects.

The solution proposed in this work is to adapt Blinn-Newell environment-maps and other reflection-map techniques to compute refractions. Then, it achieves adequate geometric properties with standard one-pass z-buffer rendering schemes, and thus is far simpler to implement.

3 Object shape approximation

Most frequently, scan line algorithms [14] together with Blinn and Newell approximations [1] provide an overall adequate solution to photorrealistic rendering of reflection models, in terms of cost, efficiency and accuracy. This scan-line approximations are based on environment map implementations, and more specifically, reflection map implementations. The aim of this work is to extend this general philosophy to refraction models. Consider a situation where the shading value of a pixel p that intersects two polygons P_1 and P_2 of a translucid object. The shape of the "generic object" laying between P_1 and P_2 can be represented –in a rough approximation– as a pair of parallel plane faces. This is exactly what is done in the refraction models considered in the previous section.

A better approximation can be obtained considering the the object as a lens. Now the object laying between P_1 and P_2 can be described as an object limited by the two sphere segments: the "input" surface is P_1 (with curvature $\rho_1 = \frac{1}{r_1}$), the "output" surface is P_2 (with curvature $\rho_2 = \frac{1}{r_2}$) (see figure 3). This optical system possesses axial symmetry with respect to an axis that intersects the centers of the spheres, called the *optical axis*.

The intersection of the spheres with the optical axis determines the *vertices* v_i of the lens. The sign of the curvatures ρ_i will be assigned positive if the surface is convex with respect to the incidence direction of the optical ray and negative if it is concave. The *thickness* of the lens is the distance d between the vertices. In this section we will consider first a formal justification of the lens model, and then we will discuss how the parameters of the lens can be extracted from the object data.

The surface S of any object can be subdivided as a set of n domains (*i.e.*, $S = \cup_n S_d$), in a way such in every domain of the subdivision, the surface S_d can be uniquely characterized as a function $z = f(x, y)$. A multivariate Taylor series around point (x_0, y_0) assumes the form

$$f(x,y) = f(x_0,y_0) + \frac{\partial f}{\partial x}\mid_{x_0,y_0}(x-x_0) + \frac{\partial f}{\partial y}\mid_{x_0,y_0}(y-y_0) + \frac{1}{2!}\frac{\partial^2 f}{\partial x^2}\mid_{x_0,y_0}(x-x_0)^2 + $$
$$+\frac{1}{2!}\frac{\partial^2 f}{\partial y^2}\mid_{x_0,y_0}(y-y_0)^2 + \frac{2}{2!}\frac{\partial^2 f}{\partial x \partial y}\mid_{x_0,y_0}(x-x_0)(y-y_0) + \cdots \qquad (1)$$

The first term is a constant which depends on the particular origin of the coordinate system, and therefore need not be considered. We will now introduce some definitions (further details can be seen in [11]).

DEFINITION **1** *The* **unit normal vector n** *of the surface S at $p = (x, y, z)$ is given by* $\mathbf{n} = \frac{(\nabla f)(p)}{\|(\nabla f)(p)\|}$. *Given a smooth function*[2] *f in an open set $U \subset \Re^3$, and a vector $\mathbf{v} \in \Re^3_p$ (i.e., all tridimensional vectors originated in $p \in U$), the* **derivative** *of f with respect to $\mathbf{v}$ is the scalar $\nabla_{\mathbf{v}} f = (f \circ \vec{\alpha})'(t_0)$, where $\vec{\alpha} : [0,1] \rightarrow U$ is any curve of parameter U such that $\vec{\alpha}'(t_0) = \mathbf{v}$, and $\vec{\alpha}'(t_0)$ represents the derivative of $\vec{\alpha}$ with respect to t evaluated at t_0. It can be shown (using the chain rule) that $\nabla_{\mathbf{v}} f = (\nabla f(p)) \cdot \mathbf{v}$. In turn, the derivative of a vectorial field $\mathbf{X}$ in an open set $U \in \Re^3$ with respect to a vector $\mathbf{v} \in \Re^3_p$, with $p \in U$, is defined as $\nabla_{\mathbf{v}}\mathbf{X} = (\mathbf{X} \circ \vec{\alpha})'(t_0)$, where $(\mathbf{X} \circ \vec{\alpha})'$ is the componentwise derivative of the vectorial function composition.* $\Box$

DEFINITION **2** *The* **Weingarten map** *of surface S in point p is the linear map $\vec{L_p} : S_p \rightarrow S_p$, where $S_p \subset \Re^3_p$ (i.e., the tangent space of S at p) is given by $\vec{L_p}(\mathbf{v}) = -\nabla_{\mathbf{v}}\mathbf{n}$.* $\Box$

This expression -understood more easily if we use definition 1– can be stated as $\nabla_{\mathbf{v}}\mathbf{n} = (\mathbf{n} \circ \vec{\alpha})'(t_0)$, where $\vec{\alpha} : [0,1] \rightarrow S$ is any curve parametrized in S such that $\vec{\alpha}'(t_0) = \mathbf{v}$. It is easy to see that $\vec{L_p}(\mathbf{v})$ is a measure of the variation rate of $\mathbf{n}$ (which, by definition, is of constant lenght) in passing through p following an arbitrary curve $\vec{\alpha}$.

DEFINITION **3** *The* **normal curvature** *of surface S at point p in the direction $\mathbf{v}$ (with $\| \mathbf{v} \| = 1$) is the scalar $k(\mathbf{v}) = \vec{L_p}(\mathbf{v}) \cdot \mathbf{v}$.* $\Box$

It is easy to see that following $\mathbf{v}$, if $k(\mathbf{v}) > 0$, then the surface bends toward $\mathbf{n}$, and if $k(\mathbf{v}) < 0$, then the surface bends away $\mathbf{n}$.

EXAMPLE **1** *Consider the case when S is a (3D) sphere $x^2 + y^2 + z^2 = r^2$, with orientation* $\mathbf{n}(p) = \frac{-p}{\|p\|}$. *Then $\vec{L_p}$ is simply the multiplication by $\frac{1}{r}$. Moreover, $k(\mathbf{v}) = \frac{1}{r}$ assumes the same value following every direction $\mathbf{v}$ and at every point $p \in S$.* $\Box$

As can be seen from the example above, in a smooth, oriented (3D) surface S, the normal curvature $k(\mathbf{v})$ is defined for every vector $\mathbf{v}$ of the tangent space S_p of S at point p. Then, the normal curvature at p is a real function with domain in the unit sphere in S_p. Since this function is continuous and its domain is a compact set, the function must assume its extrema at certain given directions in S_p. Moreover, this extrema are the eigenvalues of the Weingarten map $\vec{L_p}$, and their associated directions are the eigenvectors of the map [11][3]. Conversely, any given eigenvalue of $\vec{L_p}$ is stationary.

[2]Here and in what follows we consider the particular case of three dimensional spaces, but the properties can be considered in any space of dimension n.

[3]In a $n + 1$ space there exist at most n eigenvalues, which are the roots of the characteristic polynomial $det(L - \lambda I)$.

THEOREM 1 *Let V be a finite dimension vectorial space (with inner product $\cdot$), and let $L : V \rightarrow V$ be a linear auto-adjoint transformation in V. Then the eigenvectors of L form an orthonormal base for V (see [11] for further details and the demonstration).* $\square$

The eigenvalues of $\vec{L_p}$ are the *principal curvatures* of S, and their respective eigenvectors are the *principal curvature directions* of S. Every principal curvature is a stationary value of the normal curvature.

THEOREM 2 *Let n be an oriented n-surface in $\Re^{n+1}$, let $p \in S$ and let $\{k_1(p), \cdots, k_n(p)\}$ the principal curvatures of S at p, with associated curvature directions $\{\mathbf{v}_1, \cdots, \mathbf{v}_n\}$. Then the normal curvature $k(\mathbf{v})$ at any direction $\mathbf{v} \in S_p$ (with $\| \mathbf{v} \| = 1$) is given by*

$$k(\mathbf{v}) = \sum_{i=1}^{n} k_i(p)(\mathbf{v} \cdot \mathbf{v}_i)^2$$

(see [11] for further details and the demonstration). $\square$

The Weingarten map has several usefull geometrical properties. A 2-space is a two dimensional manifold, *i.e.*, a three-dimensional surface defined over a two-dimensional parameter space, and henceforth the Weingarten map has two eigenvalues. Moreover, its determinant $K(p) = det(\vec{L_p}) = k_1(p)k_2(p)$ is the *Gaussian curvature* of S at p, and $\frac{1}{n}$ times the trace of $\vec{L_p}$ is the *mean curvature* $H(p) = \frac{1}{n} \sum K_i(p)$.

This formal basis allows to consider a spherical approximation as a higher order approximation with respect to parallel plane approximation. Third-order (bicubic) approximations are left for further analysis. We will now concentrate on how to extract the data needed to characterize the lens from the surface data. The first step is to find the principal plane P of the lens, in a way such that it contains the two axes of greatest geometrical disperssion (see figure 2). A simple way to do this is to consider the coefficients of the plane equation $ax + by + cz + d = 0$ with the additional constraint $a^2 + b^2 + c^2 = 1$, and adjust a, b, c, d with a least square regression.

The following step is to transform the object to a space in which the xy plane coincides with P, and the origin coincides with the center of the object. The final step is to find the mean curvature of the points above and below P [8]. This also can be done with a least square regression, with respect to the functional form $F(u, z) = (z - z_0) - \rho(u + \frac{(z-z_0)^2}{2}) = 0$, where $2u = x^2 + y^2$ is the squared distance to the origin in cylindrical coordinates, ρ is the curvature of the lens, and z_0 is a *vertex* of the lens. The distance d between the vertices is the thickness of the lens. F is biparametric, and so its regression is straightforward.

4 Principles of the geometry of refraction through a lens

Consider a light ray passing through the interface of two media m_1 and m_2 at point p (see figure 3). Let $\mathbf{s}$ be a vector whose direction is the direction of the incident ray, and whose length is the refraction coefficient ν_1 of m_1. Let $\mathbf{o}$ be the (inner) normal[4] of the surface at p. $\mathbf{s}'$ will be the vector in the reflected or refracted direction, with length ν_2. Then $\mathbf{s}' = \mathbf{s} + \Gamma\mathbf{o}$, where the astigmatic constant Γ is $\Gamma = \left(\sqrt{(\nu_2)^2 - (\nu_1)^2 + (\mathbf{o} \cdot \mathbf{s})^2} - \mathbf{o} \cdot \mathbf{s}\right)$ for refractions, and $\Gamma = -2(\mathbf{o} \cdot \mathbf{s})$ for reflections.

It can be shown [5] that a two-dimensional consideration of the problem of tracing rays through a lens is equivalent, since the surface is a revolution surface and possesses axial symmetry. In our convention, where the optical axis coincides with the z axis of the coordinate system,

[4]Opposed to Computer Graphics practice, it is a widespread convention in geometrical optics to consider the normal of a surface as a vector pointing toward the center of the surface. Then we will adopt the standard notation $\mathbf{n}$ to refer to the (outer) normal as in Computer Graphics, and $\mathbf{o}$=$-\mathbf{n}$ to refer to the (inner) normal.

we can abstract axis z and work with the entities projected on the xy plane, with the origin of the coordinate axis coincident with the vertex of the surface. To simplify notation, we will follow the convention that designates the projected entities with the corresponding uppercase letter (*i.e.*, if $\mathbf{s} = (\eta, \xi, \zeta)$, then $\mathbf{S} = (\eta, \xi, 0)$). Moreover, since every ray $\mathbf{a}^\star$ can be regarded as an origin $\mathbf{a}$ plus a parametric direction $l \cdot \mathbf{s}$, we can choose $\mathbf{a}$ on the original ray and in a plane that contains the surface vertex. That means that $\mathbf{a} = (x, y, 0)$.

To compute the final rays, it suffices to know, as we will show below, that the final rays are linear combinations of the initial rays:

$$\begin{aligned} \mathbf{A}' &= \alpha\mathbf{A} + \beta\mathbf{S} \\ \mathbf{S}' &= \gamma\mathbf{A} + \delta\mathbf{S}, \end{aligned} \tag{2}$$

where $\alpha, \beta, \gamma, \delta$ are real scalars such that $\alpha\delta + \beta\gamma = 1$. Then if we denote $\vec{\mathbf{z}}$ the unit versor along z axis we find that

$$\begin{aligned} \mathbf{a}' &= \alpha\mathbf{a} + \beta\mathbf{s} + c_1\vec{\mathbf{z}} \\ \mathbf{s}' &= \gamma\mathbf{a} + \delta\mathbf{s} + c_2\vec{\mathbf{z}}, \end{aligned} \tag{3}$$

with c_1 and c_2 two appropriate constants. Finally, it is easy to see that equation system 2 is the projection of the equation system 3 onto xy plane.

Let $\mathbf{a} = (x, y, z)$, $\mathbf{a}' = (x', y', z')$, $\mathbf{s} = (\eta, \xi, \zeta)$, $\mathbf{s}' = (\eta', \xi', \zeta')$. Then we find, from the third component of equation 3, that

$$\begin{aligned} z' &= \alpha z + \beta\zeta + c_1 \\ \zeta' &= \gamma z + \delta\zeta + c_2. \end{aligned} \tag{4}$$

Since $(\mathbf{s}')^2 = (\zeta')^2 + \mathbf{S}'^2 = (\nu_2)^2$, then

$$\zeta' = \pm\sqrt{(\nu_2)^2 - ((\eta')^2 + (\xi')^2)}, \tag{5}$$

taking positive sign if there were an even number of reflections (including no reflections), and negative sign otherwise[5].

ζ' remains undetermined in equation 4 since we don't know c_2, but (as we will do below) we can compute it with equation 5 . As was stated above, c_1 depends on the particular election of the coordinate system origin. Then it is easy to convert from a given coordinate system to another.

5 Paraxial approximation and reflecction maps

In a paraxial approximation, light paths are supposed to be close to the optical axis of the system. Using paraxial approximation, equations 2 and following can be recast with the following constraints:

$$\begin{aligned} \alpha &= \beta_1\gamma_1 + 1 \\ \beta &= \beta_1 \\ \gamma &= \gamma_1\beta_1\gamma_2 + \gamma_1 + \gamma_2 \\ \delta &= \beta_1\gamma_2 + 1, \end{aligned} \tag{6}$$

[5]Here we adopt as a convention that the direction of the incident ray has a possitive third component. If it is not the case, then the sign of the third component of the resulting reflected or refracted ray must be changed accordingly, *i.e.*, multiplied by the sign of ζ.

Figure 4: Example of refraction approximation through reflection maps of resolutions 50×50, 100×100, 200×200, and 400×400,.

and if we suppose that the first medium is air (*i.e.*, $\nu_1 = 1$) and replacing ν for ν_2, we find

$$\begin{aligned} \beta_1 &= \frac{d}{\nu} \\ \gamma_1 &= (1-\nu)\rho_1 \\ \gamma_2 &= (\nu-1)\rho_2, \end{aligned} \tag{7}$$

Given equation system 2, replacing with constraints 7 and approximations 8, a paraxial approximation of the equations system is

$$\begin{aligned} \mathbf{A}' &= \left[\frac{d}{\nu}(1-\nu)\rho_1 + 1\right]\mathbf{A} + \left[\frac{d}{\nu}\right]\mathbf{S} \\ \mathbf{S}' &= \left[(1-\nu)(\rho_1-\rho_2) - (\nu-1)^2\rho_1\rho_2\frac{d}{\nu}\right]\mathbf{A} + \\ &\quad \left[1 + \frac{d}{\nu}(\nu-1)\rho_2\right]\mathbf{S}. \end{aligned} \tag{8}$$

These expressions can be used to inexpensively compute refracted rays, for instance, by means of a z-buffered environment map as the one presented in [9]. Moreover, if the approximations correspond to small objects or large distances (as in Blinn and Newell [1]), then $\mathbf{S}'$ suffices to find the refracted ray, neglecting the value of $\mathbf{A}'$. Then, we can claim that the framework proposed here allows to compute environment map approximation of refracted rays with the original philosophy of Blinn and Newell.

A reflection map is a projection of the complete environment as seen from a particular viewpoint (perhaps the center of the reflective/refractive object). Normally the reflection map is computed by rendering six views of the sceene from the viewpoint, each with a 90 degree view angle and each looking down an axis of a coordinate system with its origin in the viewpoint. Light refracted by a refractive object is computed with the reflection map and then this information is combined with the diffuse refraction and specular illumination components (as in the standard Phong model):

$$rc = dc.D + sc.S, \tag{9}$$

where rc is the refracted color of the pixel, dc is the refraction coefficient, D is the diffuse illumination, sc is the Snell refraction coefficient (exactly computed, as stated in section 3), and S is the specular illumination. Note that ambient light is accounted for in the diffuse illumination component. Figure 4 was obtained with this procedure, for different resolutions of the reflection map.

Figure 5: Kay and Greenberg model versus refraction approximation through reflection map preprocessing.

6 Computing refractions with standard environment map preprocessing

We will now discuss how to reuse standard environment map calculations to compute reflections. Equations 8 allow the computation of refracted rays as $\mathbf{s}' \sim \mathbf{M_{xy}} \mathbf{s}'_{\mathbf{r}}$. This suggests the possibility of a preprocessing of the environment map in such a way that we can compute refractions with the same algorithmic machinery for reflections (see for example [13] and [12]). The essential operation, then, is to invert the environment map. For instance, in a cubic environment map the *up* and *down* faces must be swapped, and the six faces must undergo a vertical inversion (see figure 5).

The operation $\mathbf{M_{xy}}$, however, mirrors with respect to the xy plane, which is perpendicular to the z axis (defined as the optical axis of the system). Then, the optical axis must be parallel to the z axis of the world coordinate system, otherwise further preprocessing will be needed. This preprocessing can be described mathematically as a coordinate system change, where the z axis of the new system is aligned so that $\mathbf{s}'_{\mathbf{r}}$ is given by equations 3, but computing it with vectors $\mathbf{a}$ and $\mathbf{s}$ previously rotated with a rotation transformation $\mathbf{R}_{\theta,\varphi,\sigma}$.

PROCEDURE **1** *To compute a refracted ray* $\mathbf{s}'$ *as a by-product of a reflected ray* $\mathbf{s}'_{\mathbf{r}}$ *computation,*

1. *Rotate vectors* $\mathbf{a}$ *and* $\mathbf{s}$ *with a rotation transformation* $\mathbf{R}_{\theta,\varphi,\sigma}$*, where* θ, φ, σ *are the Euler angles of the rotation [4].*

2. *Compute reflected ray* $\mathbf{s}'_{\mathbf{r}}$ *from rotated* $\mathbf{a}$ *and* $\mathbf{s}$*, following equations 3.*

3. *Compute refracted ray* $\mathbf{s}'$ *as*
$$\mathbf{s}' = \mathbf{R}^{-1}_{\theta,\varphi,\sigma} \cdot \mathbf{M_{xy}} \cdot \mathbf{s}'_{\mathbf{r}},$$
where $\mathbf{R}^{-1}_{\theta,\varphi,\sigma} = \mathbf{R}_{-\theta,-\varphi,-\sigma}$ *is the inverse rotation transformation.*

□

It can be shown that procedure 1 defines a mirror operation $\mathbf{M}_{\vec{\mathbf{u}}}$ about a generic plane. This plane is specified by its unary normal $\vec{\mathbf{u}}$, which is equivalent to determining the rotation with its

Figure 6: Complex object approximation as multiple lenses.

Euler angles [10]. This means that complex objects can be approximated as a superpositon of lenses, and that the associated refraction calculations can be easily computed as a composition of several $\mathbf{M}_{\vec{u}}$ opperations (see figure 6).

7 Conclusions and further work

A new and inexpensive model for shape approximations of generic objects was presented. The model approximates the shape of a generic object as a lens. Then, the optical (exact) expressions were derived. Since these resulted in costly computations, a Gaussian (paraxial) approximation was developed. Blinn-Newell approximation for accelerated computation was also discussed, leading to an overall adequate model, which is computationally inexpensive.

References

[1] J. F. Blinn and M. E. Newell. Texture and Reflection in Computer Generated Images. *Communicatinos of the ACM*, 19(10):542–547, October 1976.

[2] J. Foley, A. Van Dam, S. Feiner, and J. Hughes. *Fundamentals of Interactive Computer Graphics*. Addison-Wesley, Reading, Massachussetts, second edition, 1990.

[3] A. Glassner. *An Introduction to Ray Tracing*. Academic Press, Cambridge, Massachussets, 1991.

[4] H. Goldstein. *Classical Mechanics*. Addison-Wesley, New York, 1981.

[5] Max Herzberger. *Modern Geometrical Optics*. Interscience Publishing, 1958.

[6] T. L. Kay and D. Greenberg. Transparency for Computer Sinthetized Images. *ACM Computer Graphics (Proc. SIGGRAPH '79)*, 13(3):158–164, 1979.

[7] Abraham Mammen. Transparency and Antialiasing Algorithms Implemented with the Virtual Pixel Maps Technique. *IEEE Computer Graphics and Applications*, 9:43–55, 1989.

[8] Don Mitchell and Pat Hanrahan. Illumination from Curved Surfaces. *ACM Computer Graphics (Proc. SIGGRAPH '92)*, 26(2):283–291, 1992.

[9] Gustavo A. Patow. Accurate Reflections Through Z-Buffered Environment Maps. In *Proceedings of the XV International Conference of the Chilean Computer Society*, pages 385–392, Arica, Chile, 1995. SCCC.

[10] D. Rogers and J. Adams. *Mathematical Elements for Computer Graphics.* McGraw-Hill, second edition, 1976.

[11] J. A. Thorpe. *Elementary Topics in Differential Geometry.* Springer-Verlag, New York, 1979.

[12] S. Upstill. *The RenderMan Companion.* Addison-Wesley, NJ, 1992.

[13] D. Voorhies and J. Foran. Reflection Vector Shading Hardware. *ACM Computer Graphics (proc. SIGGRAPH '94)*, pages 163–166, 1994.

[14] Alan Watt and Mark Watt. *Advanced Animation and Rendering Techniques.* Addison-Wesley, London, 1992.