Principles and Applications of Pencil Tracing

Mikio Shinya Tokiichiro Takahashi
Seiichiro Naito
NTT Electrical Communications Laboratories
3-9-11, Midori-cho, Musashino-shi
Tokyo 180, Japan

Abstract

Pencil tracing, a new approach to ray tracing, is introduced for faster image synthesis with more physical fidelity. The paraxial approximation theory for efficiently tracing a pencil of rays is described and analysis of its errors is conducted to insure the accuracy required for pencil tracing. The paraxial approximation is formulated from a 4×4 matrix (a system matrix) that provides the basis for pencil tracing and a variety of ray tracing techniques, such as beam tracing, ray tracing with cones, ray-object intersection tolerance, and a lighting model for reflection and refraction. In the error analysis, functions that estimate approximation errors and determine a constraint on the spread angle of a pencil are given.

The theory results in the following fast ray tracing algorithms; ray tracing using a system matrix, ray interpolation, and extended 'beam tracing' using a 'generalized perspective transform'. Some experiments are described to show their advantages. A lighting model is also developed to calculate the illuminance for refracted and reflected light.

CR Categories and Subject Descriptors: I.3.3 [***Computer Graphics***]: Picture/Image Generation; I.3.7 [***Computer Graphics***]: Three-Dimensional Graphics and Realism
Additional Keywords and Phrases: Ray Tracing, Paraxial Theory

1 Introduction

The ray tracing algorithms [1] provide powerful tools for creating realistic images. However, from a practical viewpoint, there have been problems such as high computational cost and aliasing. Many attempts have been made to tackle those problems, and some of them have produced good results by tracing a *pencil* [1] (or bundle) of rays, instead of an individual ray. However, as the methods lack sufficient mathematical bases, they are limited to specific applications.

Heckbert proposed a method called 'beam tracing'[2] which works well for reflecting polygonal objects. His method uses a pencil to be traced by introducing affine transformations in an object space. Unfortunately, the method finds only limited applications because of the way in which it approximates refractions. Moreover, since an error estimation method has not been proposed for guaranteeing the image accuracy, the accuracy cannot be controlled.

Amanatides proposed a 'ray tracing with cones' technique for anti-aliasing, fuzzy shadows, and dull reflections[3], where a conic pencil is traced. However, it failed to present a general equation for characterizing the spread-angle change of a conic pencil through an optical system. Such an equation is also required for the calculation of ray-object intersections proposed by Barr[4], where the calculation tolerance is related to the pencil spread-angle. In a lighting model, the equation will also play an important role, because the illuminance distribution results from the calculation of how the light pencils converge and diverge according to reflections and refractions.

This paper describes the theory of pencil tracing and its applications to provide general mathematical tools for efficiently tracing a pencil and also for conducting error analysis to insure the image accuracy. In the theory, a linear approximation approach is taken, because, in general, the exact behavior of a pencil cannot be analytically obtained. The theory is based on the paraxial approximation theory[5,6], where a pencil transformation through an optical system is formulated from a 4×4 matrix (a system matrix). This formulation is well-known in optical design and electromagnetic analysis[6]. The error analysis provides functions that estimate approximation

[1] The rays that are near to a given axial ray are called paraxial and are said to form a pencil.

errors and determine a constraint on the spread angle of a pencil to insure the required accuracy of generated images.

Applications of the theory result in the following fast ray tracing algorithms: ray tracing using a system matrix, ray interpolation, and extended 'beam tracing' using a 'generalized perspective transform'. Some experiments are described to show their advantages. A lighting model from which to calculate the illuminance for refracted and reflected light is also developed.

2 Paraxial approximation theory

The paraxial approximation theory provides a linear approximation for ray changes due to refraction, reflection, and 'transfer', where 'transfer' means propagation in a homogeneous medium. A linear ray change can be represented by a matrix, and thus, paraxial ray tracing by a matrix product. Since the paraxial approximation theory seems little known in computer graphics today, it will be briefly reviewed here. For details, see [5],[6].

2.1 Definitions

Ray:

A *paraxial ray* is a ray extending along the vicinity of a given axial ray [2]. Thus, it is appropriate to represent a paraxial ray with respect to the axial ray. In the theory, a paraxial ray is represented by a four-dimensional vector (*ray vector*) in a coordinated system formed with respect to the axial ray (*ray coordinate system*).

Ray coordinate system In Figure 1, an orthogonal coordinate system $\hat{x}_1$-$\hat{x}_2$-$\hat{z}$, called a 'ray coordinate system', is used to represent a paraxial ray. The origin O is a point on the axial ray, and the $\hat{z}$-axis is the direction of the axial ray. The $\hat{x}_1$- and $\hat{x}_2$-axes can be arbitrarily chosen, and the $\hat{x}_1$-$\hat{x}_2$ plane is perpendicular to the $\hat{z}$-axis.

Ray vector for a paraxial ray Generally, a ray is uniquely specified by its direction and the position it passes. Thus, referring to $\hat{x}_1$ and $\hat{x}_2$, a paraxial ray can be represented by two kinds of vectors: the position vector $\boldsymbol{x} = (x_1 x_2)^t$ to represent the intersection of the paraxial ray with the $\hat{x}_1$-$\hat{x}_2$ plane relative to the origin O, and the direction vector $\boldsymbol{\xi} = (\xi_1 \xi_2)^t$ which is the projection of the normalized ray direction vector $\boldsymbol{s}$ of the paraxial ray onto the $\hat{x}_1$-$\hat{x}_2$ plane. Combining those two vectors, the paraxial ray is defined by a four-dimensional vector ψ at O by the equality

[2]An axial ray has nothing to do with the axes of optical systems having special physical or mathematical meanings in the systems, such as a lens axis. An adequate ray can be chosen as the axial ray for a pencil to trace, e.g., the center line of a cone in the case of a conic pencil.

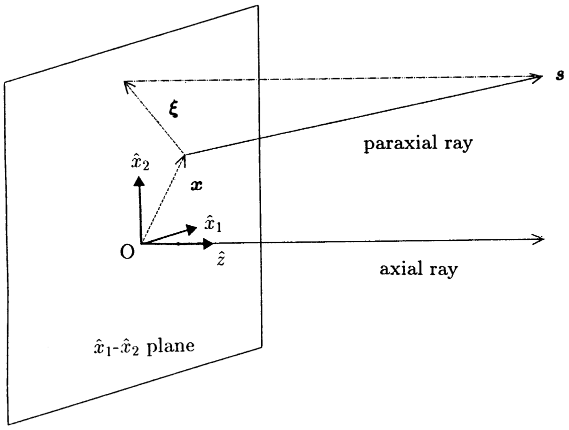

Figure 1 Definition of ray vector

$$\psi = \begin{pmatrix} \boldsymbol{x} \\ \boldsymbol{\xi} \end{pmatrix}.$$

The four-dimensional vector ψ is called a ray vector.

Pencil:

A pencil is made up of an axial ray and a bundle of paraxial rays around it represented by ray vectors. A pencil is mathematically represented by a domain in four-dimensional space, $\boldsymbol{x}$-$\boldsymbol{\xi}$, representing the deviations in positions and directions of its paraxial rays from its axial ray. In image synthesis, all rays to be traced usually start at a common point, or a pin hole. In this case, a pencil can be simply represented by its direction deviation from its axial ray at the pin hole, or the pencil spread angle.

System matrix:

When a ray goes through an optical system, the ray vector changes due to reflections, refractions and transfers. The deviation of a paraxial ray from the axial ray can be chosen small enough that the transformation representing a ray vector change is regarded as linear and can be represented by the matrix

$$\psi' = T\psi,$$

where ψ is the input ray vector and ψ' is the output vector. T is a 4×4 matrix called a system matrix. When a system consists of two sub-systems in cascade and their respective system matrices are known, the overall system matrix is simply the product of the two matrices.

2.2 System matrices

In computer graphics, optical systems usually consist of homogeneous regions separated by smooth surfaces, where any optical phenomenon can be represented by a reflection, a refraction or a transfer. Therefore, an overall system matrix for any system can easily be obtained, if system matrices are given for each *element system*, i.e., a transfer, a reflection, or a refraction. Surface smoothness is an essential condition for the system matrix approach; the surface in a system should have continuous second-order differentials. If discontinuities or edges exist in a system, it should be spatially divided into sub-systems so that no system contains an edge.

Element-system matrices are given analytically by using Snell's law and geometry. They are formulated as follows:

(1) Transfer:
In a homogeneous region, rays are straight. Thus, the propagation along the axial ray from $z = 0$ to z_0 shown in Figure 2 is represented by

$$T = \begin{pmatrix} 1 & z_0 \\ 0 & 1 \end{pmatrix}, \quad (1)$$

where each element is a 2×2 matrix and 1 means a 2×2 identity matrix.

(2) Refraction:
Consider a refraction on the surface Σ in Figure 3. The optical indices of the two media are n and n'. At the origin O, the incident axial ray meets Σ. The orthogonal coordinate systems $\hat{x}_1$-$\hat{x}_2$ and $\hat{x}'_1$-$\hat{x}'_2$ are the incident ray coordinate system and the refracted ray coordinate system, respectively, and both coordinate planes are perpendicular to their respective axial rays. Another orthogonal coordinate system $\hat{u}_1$-$\hat{u}_2$ is perpendicular to the normal of Σ at O and is used to represent Σ. For simplicity, the coordinates are chosen such that $\hat{x}_2 = \hat{u}_2 = \hat{x}'_2$. The formulation of the system matrix for a refraction on Σ is performed by approximating Σ to a paraboloid; an approach somewhat similar to Barr's tangent plane approximation[4]. Thus, the transformation is analytically derived using Snell's law as

$$T = \begin{pmatrix} \Theta'\Theta^{-1} & 1 \\ (\Theta'^t)^{-1} h Q \Theta^{-1} & (n/n')(\Theta'^t)^{-1}\Theta^t \end{pmatrix}, \quad (2)$$

where

$$\Theta = \begin{pmatrix} \hat{\boldsymbol{x}}_1 \cdot \hat{\boldsymbol{u}}_1 & \hat{\boldsymbol{x}}_1 \cdot \hat{\boldsymbol{u}}_2 \\ \hat{\boldsymbol{x}}_2 \cdot \hat{\boldsymbol{u}}_1 & \hat{\boldsymbol{x}}_2 \cdot \hat{\boldsymbol{u}}_2 \end{pmatrix}, \Theta' = \begin{pmatrix} \hat{\boldsymbol{x}}'_1 \cdot \hat{\boldsymbol{u}}_1 & \hat{\boldsymbol{x}}'_1 \cdot \hat{\boldsymbol{u}}_2 \\ \hat{\boldsymbol{x}}'_2 \cdot \hat{\boldsymbol{u}}_1 & \hat{\boldsymbol{x}}'_2 \cdot \hat{\boldsymbol{u}}_2 \end{pmatrix},$$

$$h = \cos\theta' - (n/n')\cos\theta,$$

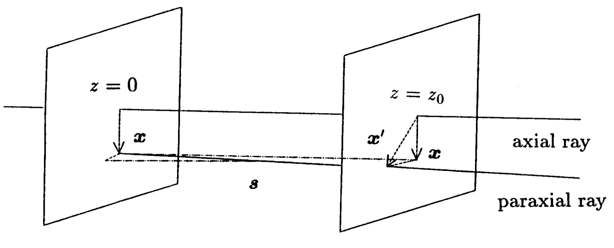

Figure 2 System matrix for transfer

and $\hat{\boldsymbol{x}}_1$ is a unit vector of the $\hat{x}_1$ axis direction. When the coordinate systems are chosen as in the figure, the matrices Θ and Θ' are diagonalized as

$$\Theta = \begin{pmatrix} \cos\theta & 0 \\ 0 & 1 \end{pmatrix}, \Theta' = \begin{pmatrix} \cos\theta' & 0 \\ 0 & 1 \end{pmatrix}.$$

The matrix Q is the curvature matrix of Σ in the $\hat{u}_1$-$\hat{u}_2$ coordinate. For example, when Σ is a sphere of radius r,

$$Q = \begin{pmatrix} 1/r & 0 \\ 0 & 1/r \end{pmatrix}.$$

(3) Reflection:
The system matrix for a reflection is derived mathematically as a special case of a refraction. It is obtained by simply replacing θ' with $(\pi - \theta)$ and n' with n in Eq.(2).

3 Tolerance and error analysis

In this section, pencil tracing approximation errors and tolerances for calculated ray-object intersections are discussed to show that they are given as functions of the system matrices. This leads to a discussion of how to determine a limit on the spread angle of a pencil in order to retain accuracy and fidelity in calculated images.

3.1 Tolerances

The criterion used here is similar to Barr's[4], which is based on pixel width and ray sampling interval. Since the sampling interval limits the resolution of a ray-traced image, an approximation error smaller than the interval has little effect on the resolution. Although Barr's tolerance equation is not applicable to refracted or reflected pencils, it can easily be extended by using a system matrix.

Consider the situation shown in Figure 4, where δ is a four-dimensional vector representing a ray interval, i.e., a sampling interval of position and direction, and

$$T = \begin{pmatrix} t_1 & t_2 \\ t_3 & t_4 \end{pmatrix}$$

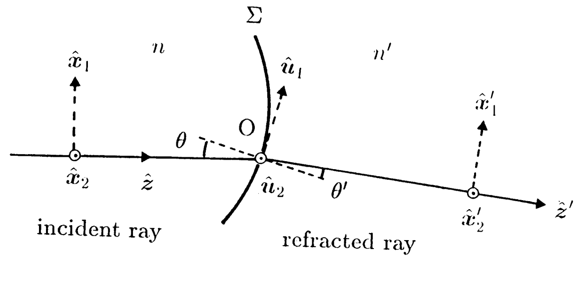

Figure 3
Refraction of a pencil. A circle with a dot denotes a vector emarging perpendicular to the page.

is an overall system matrix. The sampling interval of the intersection points on the $\hat{x}_1$-$\hat{x}_2$ plane becomes $\begin{pmatrix} t_1 & t_2 \end{pmatrix} \delta$, and the tolerance τ on the plane is given in terms of parameter ρ as

$$\tau = \rho \left| \begin{pmatrix} t_1 & t_2 \end{pmatrix} \delta \right|, \qquad (3)$$

where ρ indicates the ratio of the tolerance to the interval, and it represents the image accuracy. When one-pixel-width resolution is required, ρ should be less than 1/2. Equation (3) is also applicable to Barr's intersection calculation method.

3.2 Error estimation function

When a paraxial ray ψ^{in} is changed into ψ^{out} through an optical system, ψ^{out} may be expressed in terms of a power series of ψ^{in}. The i-th component of the vector ψ^{out} is represented by

$$\begin{aligned}(\psi^{out})_i &= \sum_{j=1}^{4} t_{ij}^{(1)}(\psi^{in})_j + \sum_{j,k=1}^{4} t_{ijk}^{(2)}(\psi^{in})_j(\psi^{in})_k + \cdots \\ &= (\psi'^{out})_i + (\varDelta\psi)_i + o((\psi^{in})_j^3),\end{aligned}$$

where $t_{ij}^{(1)}$ is an element of the system matrix, and

$$t_{ijk}^{(2)} = \partial^2(\psi^{out})_i / \partial(\psi^{in})_j \partial(\psi^{in})_k.$$

Since higher-order terms are considered to be negligible for small ψ^{in} value, the second-order term $\varDelta\psi$ alone is enough to estimate the linear approximation error $(\psi^{out} - \psi'^{out})$.

The coefficients $t_{ijk}^{(2)}$ for each element system, e.g., refraction, can be derived analytically and they can be applied to general systems. However, a straightforward computation of $\varDelta\psi$ is rather cumbersome because of the complicated forms of its elements. Furthermore, since the absolute values of error vectors, $\varDelta\boldsymbol{x}$ and $\varDelta\boldsymbol{\xi}$, are far more important than their directions in terms of error estimation, we introduce the more compact error estimation functions, e_x and e_ξ, to estimate the absolute values of errors. This results in the expressions

$$\begin{aligned}e_x(x_0, \xi_0) &= Ax_0^2 + Bx_0\xi_0 + C\xi_0^2 \\ &\geq |\varDelta\boldsymbol{x}|\end{aligned}$$

and

$$\begin{aligned}e_\xi(x_0, \xi_0) &= Dx_0^2 + Ex_0\xi_0 + F\xi_0^2 \\ &\geq |\varDelta\boldsymbol{\xi}|, \end{aligned} \qquad (4)$$

where

$$\begin{pmatrix} \varDelta\boldsymbol{x} \\ \varDelta\boldsymbol{\xi} \end{pmatrix} = \varDelta\psi$$

$$x_0 = |\boldsymbol{x}_0|, \ \xi_0 = |\boldsymbol{\xi}_0|$$

$$\begin{pmatrix} \boldsymbol{x}_0 \\ \boldsymbol{\xi}_0 \end{pmatrix} = \psi^{in}.$$

A - F are constants and depend on the system matrices, axial incident angles, optical indices, and so on. As the derivation is very complicated, the details are omitted here. The results are presented in the appendix.

Equations (3) and (4) provide the condition for pencil tracing with the required accuracy. If all paraxial rays in a pencil satisfy the inequality

$$e_x(x_0, \xi_0) \leq \tau, \qquad (5)$$

the generated image is accurate enough.

In the case where a pencil emerges from a pin hole, x_0 is 0 and Eq. (5) is simplified as

$$C\xi_0^2 \leq \tau, \qquad (6)$$

to provide the maximum pencil spread angle for the required accuracy.

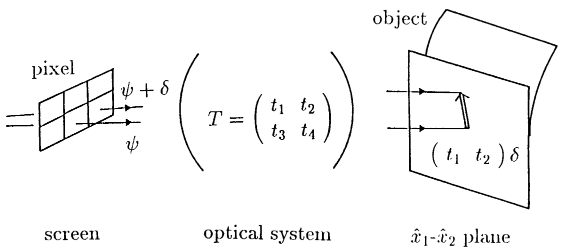

Figure 4 Sampling interval and tolerance

4 Applications

Since the theory describes general pencil behaviors, it can be used in many places in computer graphics. In this section, three fast ray tracing methods are proposed, and a general illuminance formula for refracting and reflecting environments is introduced that demonstrates actual applications of the theory.

4.1 Pencil tracer as a fast ray tracer

(1) Ray tracing with system matrix:

This method is a straight-forward installation of the theory, and accelerates image systhesis of smooth refracting and reflecting objects by replacing conventional refraction and reflection calculations with matrix-vector manipulations.

Paraxial rays are traced by a system matrix, calculated by Eqs. (1) and (2). The pencil spread angle is controlled to satisfy tolerance condition (6). Since ray tracing with a system matrix is not applicable in the neighborhoods of object edges because of the smoothness requirement, individual rays in those regions are traced with a conventional ray tracer.

The procedure is as follows:

1) Divide the screen into $n \times m$ initial domains of a certain number of pixels[3]. Do the following process for each domain:
2) Set the axial ray at the center of a domain, and trace it with the conventional ray tracer. Calculate the system matrix.
3) Check the smoothness condition (to be discussed below). If an edge exists in the domain, trace all the rays with the conventional ray tracer.
4) If there is no edge, calculate the tolerance and the maximum pencil-spread angle by using Eqs. (4) and (6). Then, according to the maximum pencil-spread angle, do the following:
   a) Trace all paraxial rays in the domain with the system matrix if the maximum spread angle of the domain is smaller than that of the pencil.
   b) Trace all paraxial rays with the ray tracer, if the maximum pencil spans an area less than one pixel wide.
   c) Otherwise, divide the domain into sub-domains so that their maximum spread angles are less than that of the pencil. Repeat 2)- 4) for each sub-domain.

Anti-aliasing by subpixel sampling can be achieved in the same way by using the system matrix, except in the neighborhood of an edge.

In our preliminary implementation, the smoothness condition is roughly checked by comparing the ray trees among the neighboring domains, and the condition is assumed to be satisfied when no difference is detected among the ray trees. It is possible for an object smaller than the initial domain area to vanish from the image. This problem can be solved by estimating the distances between an axial ray and object surfaces, as in the case of ray tracing with cones [3]. Amanatides' method can deal with the estimation for simple objects such as spheres and polygons. It is considered that the bounding volume techniques([7] among others) will be effective for this estimation. However, further investigation is required to solve the problem.

(2) Ray interpolation:

In this method, the intersection point and the direction of an interior paraxial ray are linearly interpolated without a system matrix calculation in order to provide further computational saving.

Consider the situation shown in Figure 5, where two rays, ψ_0 and $\psi_0 + \delta$, are traced and x'_0 and x'_1 give their intersections with the object. A paraxial ray represented by $\psi' = \psi_0 + a\delta$ can be considered to intersect with the object at the point that can be linearly interpolated by the expression

$$x' = a x'_1 + (1-a) x'_0.$$

The direction vector of the paraxial ray can also be interpolated in the same manner.

A precise analysis of the interpolation error is not an easy task. However, if a second order approximation is good enough to estimate the true x', the error can be estimated by evaluating first and second order interpolations. For this, one more intersection of another ray is necessary. For example, when the intersection x'_{-1} of the ray $\psi_0 - \delta$ is given, the error is estimated by

$$e_{int} = |(\text{second order interpolation}) - (\text{liner interpolation})|,$$

where

$$(\text{second order interpolation}) = a(a-1)x'_{-1}/2 + (1-a)(1+a)x'_0 + a(1+a)x'_1/2$$

In case that the second-order approximation is not good, e_{int} simply checks the linearity of x with respect to a.

Comparing with the system matrix method, the ray interpolation has the advantage in computation speed, but a disadvantage in the precision of the error estimation. Thus, it is considered that the method is effective for tracing a thin pencil, e.g., in the case of subpixel sampling for anti-aliasing. This will be shown in the section 5.

Note that if linearity is assumed for brightness $I(x')$, the brightness can also be interpolated by

$$I(x') = aI(x'_1) + (1-a)I(x'_0).$$

(3) Generalized perspective transformation:

This method is a modification of Heckbert's beam tracing[2], and it is effective for refracting and reflecting polygonal environments. Although the beam tracing

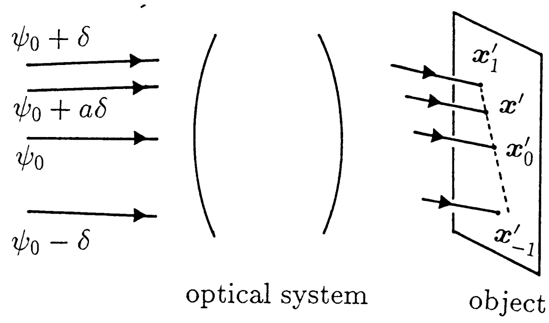

Figure 5 Ray interpolation

[3]Larger initial domains do not necessarily lead to faster image synthesis. There is a certain point in a domain area (5×5 pixels in our experiment) beyond where the speed of image synthesis no longer improves.

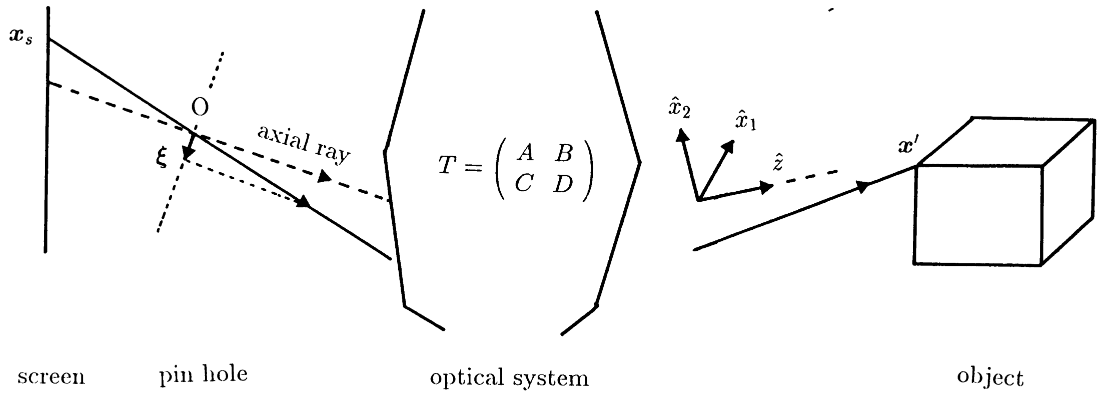

Figure 6 Generalized perspective transformation

method works well for reflections, it provides a rather poor approximation for refraction. It assumes either that incident rays are nearly perpendicular to a surface, or that all rays are parallel.

The system matrix provides a better approximation using local linearity. Consider the situation shown in Figure 6. The ray $\psi = (0, \boldsymbol{\xi})^t$ goes from the pin hole to the object point $\boldsymbol{x}'$ through the optical system, where $\boldsymbol{x}'$ is represented in the ray coordinate system. Using the system matrix, it is expressed as,

$$\boldsymbol{x}' = A\boldsymbol{\xi},$$

where A is a 2×2 sub-matrix of the system matrix. The screen point $\boldsymbol{x}_s$ is directly calculated by $\boldsymbol{\xi}$, wherein the linear approximation is represented by

$$\boldsymbol{x}_s = S\boldsymbol{\xi},$$

where S is a 2×2 matrix. Thus, the transformation from an object point to a screen point is given by

$$\boldsymbol{x}_s = SA^{-1}\boldsymbol{x}' = P\boldsymbol{x}'. \qquad (7)$$

Here, P is considered as a 'local' perspective transformation for refracting and reflecting systems. The transformation P coincides with the usual perspective transformation for a transfer, and with the ones of the beam tracing for a reflection and the perpendicular incident refraction.

The transformation can be implemented in almost the same way as the beam tracing: a pencil formed by a polygon boundary is approximated by a pyramidal pencil that is represented also by a polygon on the $\boldsymbol{\xi}$-plane or on the screen. The system matrix is calculated by Eqs. (1) and (2), and the polygons in the object space are mapped onto the screen by Eq. (7) to allow searching for visible polygons through a polygon clipping technique and a hidden surface technique by referring to z-values in the ray coordinate system.

Anti-aliasing can be performed by the techniques for the scan line algorithm. Errors can be estimated by using Eq. (4), and dividing the pencil can assure image accuracy though this has not been implemented yet. This method is free from the edge problem of the system matrix.

4.2 Illuminance formula

A light is converged or diverged by refractions, and reflections on curved surfaces. This makes a variety of shadow patterns caused by light concentration. Conventional illumination models fail to simulate this phenomena and they create unnatural sharp shadows for transparent objects. Kajiya succeeded in creating realistic shadows of transparent objects by using his powerful rendering equation and a Monte Carlo method[8]. However, there are two problems with his method. First, the equation he

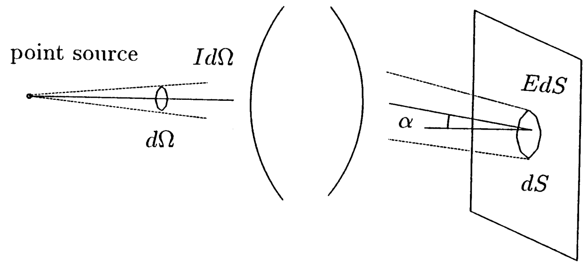

Figure 7 Light pencil emitted from a point source

used is based on the inverse square law of light intensity which is not valid for refracted or reflected light. Strictly speaking, the calculated illuminance is theoretically incorrect. Second, it requires a tremendous amount of computation time, even for a very simple situation.

It is easy to extend the intensity law by using a system matrix[6], and the illuminance is analytically calculated for point light sources and parallel light beams when using it. Consider a pencil emitted from a point source and passing through an optical system onto an object surface, as shown in Figure 7. For simplicity, assume that the same medium, e.g. air, surrounds both the source and the object. Let the luminous intensity of the source be I, the illuminance on the surface E, and the transmittance of the system t_r. Then, energy preservation is represented by

$$t_r I d\Omega = E dS,$$

where $d\Omega$ is the emitted pencil solid angle, and dS is the illuminated area on the surface. Using the system matrix and the ray coordinate systems gives

$$\begin{pmatrix} \boldsymbol{x}' \\ \boldsymbol{\xi}' \end{pmatrix} = \begin{pmatrix} A & B \\ C & D \end{pmatrix} \begin{pmatrix} \boldsymbol{x} \\ \boldsymbol{\xi} \end{pmatrix},$$

and

$$\boldsymbol{\xi} = \begin{pmatrix} \xi_1 \\ \xi_2 \end{pmatrix}, \boldsymbol{x}' = \begin{pmatrix} x'_1 \\ x'_2 \end{pmatrix},$$

and $d\Omega$ and dS are given by

$$d\Omega = |d\xi_1 d\xi_2|, \text{ and } dS = |dx_1 dx_2 / \cos\alpha|,$$

where α is the angle between the axial ray and the surface normal. Thus, the illuminance is given by

$$E = t_r I \cos\alpha |d\xi_1 d\xi_2 / dx'_1 dx'_2| = t_r I \cos\alpha |1/det(B)|. \quad (8)$$

In the case of a transfer, $det(B)$ is z^2, where z is the distance between the source and the illuminated point, and thus, Eq. (8) represents the inverse square law. However, for a refracted light pencil, the inverse square law is not valid, because in general $det(B) \neq z^2$.

Likewise, for a parallel light source, it is derived that

$$E = t_r I \cos\alpha / |1/det(A)|. \quad (9)$$

Using Eq. (8) instead of the inverse square law, Kajiya's equation becomes perfectly correct. However, for a simple situation where point sources illuminate transparent objects, the pencil tracer traces light pencils from the sources, simulating caustics and shadows more efficiently. This becomes more distinctive in the case of the proposed generalized perspective transformation for polygonal objects, as will be shown in Section 5. Furthermore, the illuminance formula (8) and the generalized perspective transformation can be applied to Nishita and Nakamae's radiosity method[9] for simulating interreflection between both refracting and diffusive polyhedra.

5 Experimental Results

Figure 8 shows an image generated through ray tracing with a system matrix and anti-aliased by nine rays per pixel subpixel-sampling using the ray interpolation technique. The computation time is about 230 seconds on a VAX11/780 for 256×256 pixels, which is 7.6 times faster than our conventional ray tracing program. Since ray interpolation is a computationally inexpensive process, the improvement in speed becomes more significant as the subpixel sampling rate increases.

Figure 9 shows the ratio of the CPU time of the pencil tracer to the time of the conventional one, with respect to the number of ray samples per pixel. The time ratio decreases to less than 1/10 as the sampling ratio increases to 49. This suggests that the method is particularly efficient in creating 'high quality' anti-aliased images. The tolerance used here is one-half pixel width, or $\rho = 1/2$.

Figure 10 shows the error distribution of the image, where the error is measured by the distance between ray-object (checkerboard) intersections obtained by the pencil tracer and by the ray tracer normalized by one pixel-interval on the checkerboard. Errors are less than the specified tolerance, 1/2, in over 99.8% of the image area, and the largest error is only 0.66 pixel width. This suggests that errors are estimated strictly enough. Since the error estimation is a worst case estimation, the actual errors are considered to be much smaller than the tolerance in most areas, as shown in the figure.

Figure 11 shows an image of a transparent polyhedron consisting of 100 polygons generated by the generalized perspective transformation without error estimation and anti-aliasing. The illuminance on the three-colored rectangle is calculated by illuminance formula (8), simulating the light pencil concentration effect caused by refractions. The shadows and the image are created individually, and each computation time is about 200 seconds and 74 seconds, respectively, on a VAX11/780 for 512×512 pixels. The ray tracing program takes about 49 minutes and it creates only a non-shadowed image using a rectangular solid bounding volume. It is estimated that it would take several tens of times more to calculate precise shadows like in Fig. 11 by the ray tracer, because, from Kajiya's experiment[8] and our experiences, it is believed that several tens of rays per pixel sampling might be necessary for a good approximation.

6 Conclusion

In this paper, the theory and applications of pencil tracing have been described. In the theory, we introduced a system matrix approximation and analyzed the approximation errors. The system matrix describes general pencil behavior, and it provides the basis for pencil tracing. From approximation error analysis, we derived a parabolic error estimation function that enables pencil

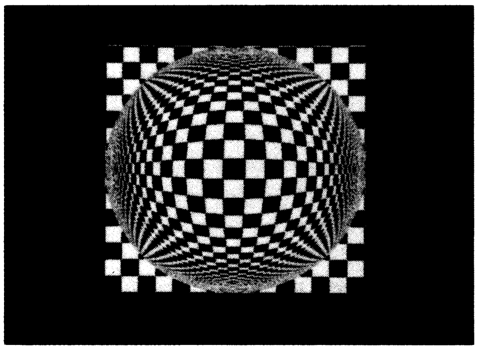

Figure 8
A transparent sphere generated by ray tracing with system matrix

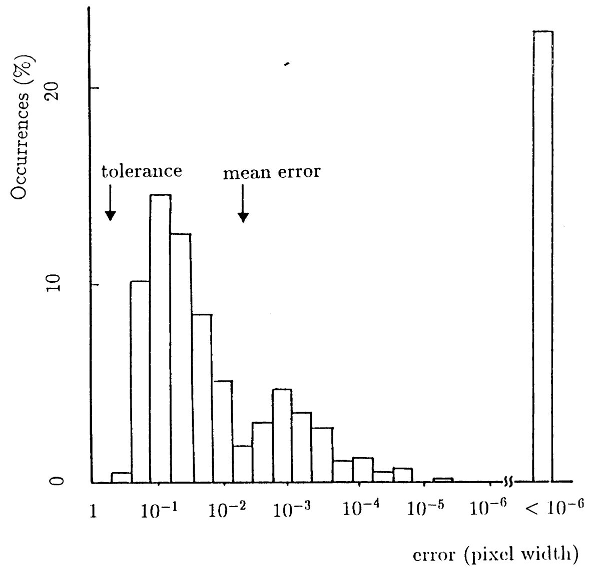

Figure 10 Error distribution of the image

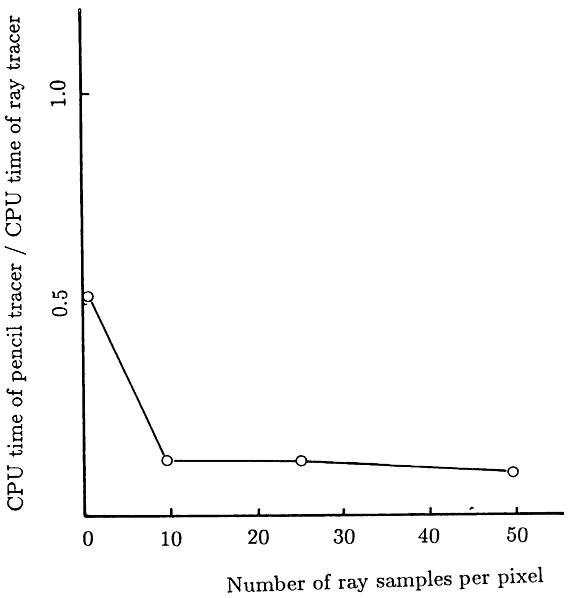

Figure 9
Ratio of CPU time of the pencil tracer to time of the ray tracer, with respect to the number of ray samples per pixel

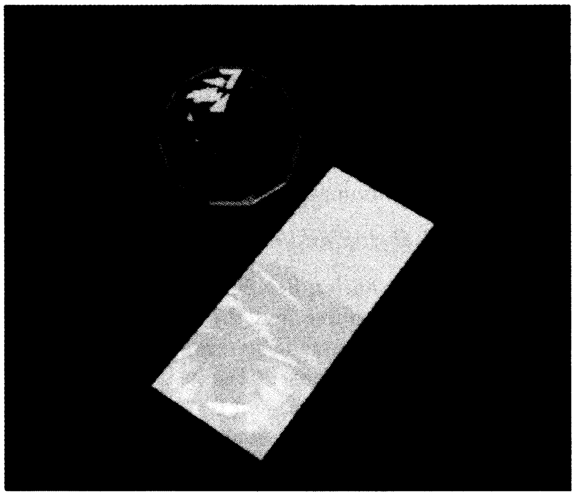

Figure 11
Transparent polyhedron and its shadow using generalized perspective transformation and the illuminance formula (8).

tracers to assure a required image accuracy.

The theory solves a variety of ray tracing problems in the following ways:

- The system matrix provides a better approximation of refraction and error estimation tools for beam tracing[2].
- The system matrix provides a general calculation method to obtain a pencil-spread angle which is necessary for ray tracing with cones[3].
- The system matrix provides a general method for tolerance calculation which is essential for Barr's ray tracing method[4].
- The system matrix provides a general illuminance formula that describes the light concentration effect.
- The system matrix provides fast ray tracing and anti-aliasing methods for smooth refracting and reflecting objects. Image accuracy is assured due to the error estimation function.

As applications, we proposed three fast ray tracing algorithms, ray tracing with a system matrix, ray interpolation, and generalized perspective transformation. An illuminance formula that describes the intensity law for general situations is also presented. Using the formula, pencil tracers can analytically calculate illuminances for refracted and reflected light emitted by point sources. Experiments confirmed their efficiency in speed, accuracy and reality for smooth transparent objects. The edge problem still remains in ray tracing with the system matrix, and further research is required.

As the theory provides general tools for pencil tracing, it is considered that pencil tracing is applicable to many situations that require tracing many close rays. These situations include, for example, when creating a motion-blurred picture, demonstrating the dispersion effect of transparent objects, making many pictures from a continuously moving view point, and distributed ray tracing[10].

Acknowledgment

We would like to thank Dr. S. Shimada, for his continuous support. We would also like to thank Dr. K. Tsukamoto, Dr. I. Masuda, Mr. K. Ueno, Dr. T. Mashiko, Mr. T. Naruse, Mr. M. Yoshida, and Mr. H.Murase for their invaluable advice and encouragement. We are greatly indebted to Dr. N. Osumi for preparing the paper, and to the reviewers for their useful comments.

References

[1] T.Whitted, 'An improved illumination model for shaded display', Comm. ACM, *23*, No.6, pp.343-349 (1980)

[2] P.S.Heckbert and P. Hanrahan, 'Beam tracing polygonal object', Computer Graphics, *18*, No.3, pp.119-127 (1984)

[3] J.Amanatides, 'Ray tracing with cones', Computer Graphics, *18*, No.3, pp.129-135 (1984)

[4] A.H.Barr, 'Ray tracing deformed surfaces', Computer Graphics, *20*, No.4, pp.287-296 (1986)

[5] M.Born and E.Wolf, Principles of Optics, pp.190-196, New York: Pergamon, 1959

[6] G.A.Deschamps, 'Ray techniques in electromagnetics', Proceedings of the IEEE, *60*, No.9, pp.1022-1035 (1972)

[7] S.M.Rubin and T.Whitted, 'A three dimensional representation for fast rendering of complex scenes', Computer Graphics, *14*, No.3, pp.110-116 (1980)

[8] J.T.Kajiya, 'The rendering equation', Computer Graphics, *20*, No.4, pp.143-150 (1986)

[9] T.Nishita and E.Nakamae, 'Continuous tone representation of three-dimensional objects taking account of shadows and interreflection', Computer Graphics, *19*, No.3, pp.23-30 (1985)

[10] R.L.Cook, T.Porter, and L.Carpenter, 'Distributed ray tracing', Computer Graphics, *18*, No.3, pp.137-145 (1984)

Appendix

Error estimation functions, e_x and e_ξ

(1) Transfer:

Since the second term of a ray change $\Delta\psi$ is zero, the error estimation function is equal to zero.

(2) Refraction or reflection:

$$\begin{aligned} e_x &= ax^2 + bx\xi + c\xi^2 \\ e_\xi &= dx^2 + ex\xi + f\xi^2 \end{aligned} \qquad \text{(A-1)}$$

where

$$\begin{aligned} a &= (\cos\theta'/2R\cos^2\theta)|\alpha + \beta\tan\theta'|, \\ b &= (|\alpha|/\gamma)(n/n'), \\ c &= 0, \\ d &= (|\beta|/\cos^2\theta)[(1/4R^3)\{\tan\theta + (2+|\beta|)\tan\theta'\} \\ &\quad + \sqrt{2}d_{3m}], \\ e &= |\gamma\tan\theta' - (1/\gamma)\tan\theta|/(R\cos\theta)(n/n'), \\ f &= (\gamma|\alpha|/2)(n/n'), \\ \alpha &= \tan\theta - \gamma\tan\theta', \\ \beta &= 1-\gamma, \\ \gamma &= n\cos\theta/n'\cos\theta' \end{aligned}$$

and R is the maximum curvature of the surface at the origin, θ and θ' are the incident and refracted axial ray angles, and d_{3m} is the maximum third differential value at the point, defined by

$$d_{3m} = \max_{i,j,k=1,3}(|\partial^3 f/\partial u_i \partial u_j \partial u_k|).$$

(3) General system

Consider a system composed of n element systems, wherein the system matrix is

$$T = T_n T_{n-1} \cdots T_1,$$

where T_i is the system matrix of an i element system. For simplicity, let the 4×4 matrices T_i^a and T_i^b be

$$\begin{aligned} T_i^a &= \begin{cases} T_n T_{n-1} \cdots T_{i+1} & \text{for } i = 1, 2, \cdots, n-1 \\ 1 & \text{for } i = n \end{cases} \\ &= \begin{pmatrix} A_{1i}^a & A_{2i}^a \\ A_{3i}^a & A_{4i}^a \end{pmatrix}, \\ T_i^b &= \begin{cases} 1 & \text{for } i = 1 \\ T_{i-1} T_{i-2} \cdots T_1 & \text{for } i = 2, 3, \cdots, n \end{cases} \\ &= \begin{pmatrix} A_{1i}^b & A_{2i}^b \\ A_{3i}^b & A_{4i}^b \end{pmatrix}, \end{aligned}$$

where A_{ji}^a and A_{ji}^b are 2×2 matrices. The error estimation functions are given by

$$e_x(\zeta) = \zeta^t \{ \sum_{i=1}^{n} M_i^t (\lambda_{1i} P_i + \lambda_{2i} Q_i) M_i \} \zeta,$$

and

$$e_\xi(\zeta) = \zeta^t \{ \sum_{i=1}^{n} M_i^t (\lambda_{3i} P_i + \lambda_{4i} Q_i) M_i \} \zeta,$$

where

$$\begin{aligned} \zeta &= \begin{pmatrix} x \\ \xi \end{pmatrix}, \\ M_i &= \begin{pmatrix} \mu_{i1} & \mu_{i2} \\ \mu_{i3} & \mu_{i4} \end{pmatrix}, \\ P_i &= \begin{pmatrix} a_i & b_i/2 \\ b_i/2 & c_i \end{pmatrix}, \\ Q_i &= \begin{pmatrix} d_i & e_i/2 \\ e_i/2 & f_i \end{pmatrix}. \end{aligned}$$

The values μ_{ij} and λ_{ij} are square roots of the larger eigen values of $A_{ij}^b (A_{ij}^b)^t$ and $A_{ij}^a (A_{ij}^a)^t$, respectively. The value a_i to f_i is a coefficient of the error estimation function for the i element system, given by Eq. (A-1).