Werk

Titel: Computability and Complexity of Ray Tracing.

Autor: Reif, J.H.; Tygar, J.D.; Yoshida, A.

Jahr: 1994

PURL: https://resolver.sub.uni-goettingen.de/purl?362609810_0011|log25

Kontakt/Contact

Digizeitschriften e.V.
SUB Göttingen
Platz der Göttinger Sieben 1
37073 Göttingen

info@digizeitschriften.de

Discrete Comput Geom 11:265–287 (1994)

Discrete & Computational
Geometry
© 1994 Springer-Verlag New York Inc.

Computability and Complexity of Ray Tracing*

J. H. Reif,[1] J. D. Tygar,[2] and A. Yoshida[1]

[1] Department of Computer Science, Duke University,
Durham, NC 27706, USA
reif@cs.duke.edu

[2] Department of Computer Science, Carnegie Mellon University,
Pittsburgh, PA 15213, USA
tygar@cs.cmu.edu

Abstract. *The ray-tracing problem* is, given an optical system and the position and direction of an initial light ray, to decide if the light ray reaches some given final position. For many years ray tracing has been used for designing and analyzing optical systems. Ray tracing is now used extensively in computer graphics to render scenes with complex curved objects under global illumination.

We show that ray-tracing problems in some three-dimensional simple optical systems (purely geometrical optics) are undecidable. These systems may consist of either reflective objects that are represented by rational quadratic equations, or refractive objects that are represented by rational linear equations. Some problems in more restricted models are shown to be PSPACE-hard or sometimes in PSPACE.

1. Introduction

We often observe light rays passing through a window or a hole in a cloud, which appear to propagate in straight lines. Since the size of windows and clouds is significantly larger than the wavelength, which is on the order of 10^{-7} m, the wave

* A preliminary version of this paper appeared as "The Computability and Complexity of Optical Beam Tracing" in the *Proceedings of the 31st Annual Symposium on Foundations of Computer Science*, October 1990, Vol. I, pp. 106–114. The research of J. H. Reif and A. Yoshida was supported in part by Air Force Contract No. AFOSR-87-0386, DARPA/ARO Contract No. DAAL03-88-K-0195, DARPA/ISTO Contract No. N00014-88-K-0458, and NASA subcontract 550-63 of primecontract NAS5-30428. J. D. Tygar's research was supported in part by the Defense Advanced Research Projects Agency under Contract No. F33615-87-C-1449 and by a National Science Foundation Presidential Young Investigator Award under Contract No. CCR-8858087.

nature of light becomes negligible. Based on this rectilinear propagation of light rays, the theory of geometrical optics was developed, and the foundation or ray tracing was established.

The theory of geometrical optics is a classical field. In this theory the propagation of light is assumed to be rectilinear in a homogeneous medium, and the ray-tracing technique uses this assumption. Ray tracing is a valuable tool for designing and evaluating optical instruments and for rendering realistic scenes in computer graphics.

The history of ray tracing goes back at least to Euclid (315–250 B.C.) and his student, Archimedes (287–212 B.C.). Portions of Euclid's *Elements* were motivated by Greek interest in the theory of rectilinear propagation of light. In 1730 Newton (1642–1727) published his treatise *Opticks* [21] in which he defined the reflective and refractive laws of light based on his corpuscle theory and first formally defined and investigated some ray-tracing problems.

Given a mathematical model of an optical system, light rays can be projected through it to evaluate its theoretical performance. The rays are traced by applying the reflection or refraction equation at each intersecting surface to locate where the reflected or transmitted ray intersects the next surface. In this way the rays are traced throughout the system.

Ray tracing also has important applications in computer graphics. It is used to render scenes consisting of objects with reflective or refractive surfaces under global illumination [5], [11], [13], [26]. The realism of an image is enhanced by the way in which a scene is illuminated by light sources. A *global illumination model* called ray tracing, which is based on geometrical optics, computes the illumination of any visible point in the scene by following the paths of light rays as they are reflected or refracted through the scene. Not surprisingly, ray tracing has produced some of the most spectacular computer-synthesized images to date. See [4], [8], [10], [12], [15], and [25] for physical theories and a brief history of ray tracing.

Despite a great amount of attention paid to the use of ray tracing, the computability and complexity of various ray-tracing problems has not been well studied. Formalizing these problems by using assumptions from the theory of geometrical optics, their computability and complexity can be investigated. The results are very interesting as they answer the questions in the classical theory of optics, computer graphics, and computational geometry.

2. Formal Description

The *ray-tracing problem* is a decision problem: given an optical system (namely, a finite set of reflective or refractive objects), a light ray's initial position and direction, and some fixed point p, does the ray eventually reach p?

Our optical systems consist of a finite set of optical *objects* that may be *reflective* or *refractive*. The boundaries of these objects may be *totally reflective*, *partially reflective*, or *totally refractive*. In this paper we restrict ourselves to optical systems

constructed out of *homogeneous* reflective or refractive objects whose boundaries are described by *linear* or *quadratic* equations.

The positions of surfaces can be either *rational* or *irrational* relative to a coordinate system. However, we always assume that the position and tangent of the initial ray are represented with rational coordinates. If the optical system is irrational, the path of rays obviously will include irrational coordinates. In a restricted rational system with only reflective objects whose boundaries are described by linear equations, the trajectory of rays can be traced in rational coordinates—that is, we can define each ray's path as a union of line segments bounded by rational coordinates. However, the path of rays in more general, rational optical systems may not be represented in rational coordinates. If there are quadratic reflective surfaces, the ray may fail to intersect the surface at a rational position, and the tangent of the reflected ray may be irrational. If there are refractive objects, neither the position nor the tangent of the refracted ray may stay rational. We say that the ray-tracing problem is *rational* if the path of the ray is represented as the union of line segments bounded by rational coordinates throughout the system. Otherwise, we say that the problem is *irrational.* It seems that rational ray-tracing problems are not so difficult in terms of complexity. In this paper we show the surprising result that some rational problems are undecidable. To keep some ray-tracing problems rational, we may put some restrictions on the general use of refractive objects and quadratic objects so that the trajectory of rays can be traced in rational coordinates.

The ray-tracing problem is a decision problem. We are interested in determining if the ray will reach a given final position, and not in the intensity of the ray at that position. Throughout this paper we assume the theory of geometrical optics [4], [12]. The paths taken by rays are rectilinear inside a homogeneous medium and the directions of reflected or refracted rays are determined by the laws of *reflection* and *refraction.* The law of reflection states that the incident angle and the reflected angle are equal, and the law of refraction states that the angle of refraction depends on the incident angle and the indices of refraction of the materials as depicted in Fig. 1. Rays are assumed to have infinitesimal wavelength and are treated as lines with zero width. This implies that there is no diffraction caused by the wave nature of light. All surfaces are

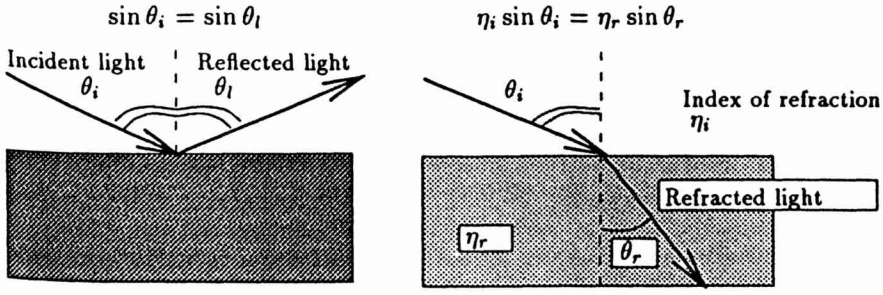

Fig. 1. Reflection and refraction.

perfectly smooth and do not cause the scattering of rays upon reflection or refraction.

Our results in various optical models are summarized as follows:

1. Ray tracing in three-dimensional optical systems consisting of a finite set of reflective or refractive objects represented by a system of rational quadratic inequalities is undecidable.
2. Ray tracing in three-dimensional optical systems consisting of a finite set of refractive objects represented by a system of rational linear inequalities is undecidable.
3. Ray tracing in three-dimensional optical systems consisting of a finite set of *rectangular* reflective and refractive objects is undecidable.
4. Ray tracing in three-dimensional optical systems consisting of a finite set of reflective and partially reflective surfaces represented by a system of linear inequalities, where some of the inequalities are allowed to be irrational, is undecidable.
5. Ray tracing in three-dimensional optical systems consisting of a finite set of reflective and partially reflective surfaces represented by a system of rational linear inequalities is PSPACE-hard.
6. For any $d \geq 2$, ray tracing of d-dimensional optical systems consisting of a finite set of parallel and perpendicular reflective surfaces represented by a system of rational linear equalities is in PSPACE.

An interesting application of the undecidability results is that geometrical optics can, in principle, simulate any computable computation. Some chaotic systems can be computed by finite programs [6], [19]. Thus, as a corollary to our undecidability results, geometrical optical systems can be chaotic in some cases.

Theoretically, these optical systems can be viewed as general optical computing machines, if our constructions could be carried out with infinite precision, or perfect accuracy. However, these systems are not practical, since the above assumptions do not hold in the physical world. Specifically, since the wavelength of light is finite, the wave property of light, namely diffraction, makes the theory of geometrical optics fail at the wavelength level of distances [7].

3. Related Work

Some of the results presented here were shown by us in an earlier paper [23]. There are several new results presented in this paper, which were obtained by considering refraction and by using a reversible Turing machine.

Abstract ray-tracing problems have also been investigated by Fiume [8]. Fiume's work does not describe actual optical systems, but rather considers symbolic systems that transform the intensity of rays without any consideration of the geometry of the optical systems. He showed that if amplification of light

intensity was allowed, his abstract ray-tracing problem was PSPACE-complete. If amplification is not allowed, then he showed that the problem became NP-hard. However, his transformations on the intensity of optical rays would require active optical components with nonlinear transfer functions such as electro-optical devices or photorefractive crystals, which are not purely optical devices.

In contrast, our results do not depend on any intensity of the rays. Instead, we encode problems by the position of optical rays. Manipulation of the positions is carried out by use of pure geometrical optics. Thus, our models give results more appropriate to computational geometry. In particular, they address the ray-tracing problems in the models that are described by Newton in his *Opticks*, and that form the backbone of modern ray-tracing theory.

A similar undecidability proof has been independently obtained by Moore [20] who investigated unpredictability in dynamical systems. He showed that motion with as few as three degrees of freedom in smooth dynamics can simulate universal computation.

4. Optical Systems and Ray Tracing

Optical systems may consist of both reflective and refractive objects. Some objects may be represented by quadratic inequalities. Some boundaries may be described by equations with irrational coefficients. The trajectory of rays in some systems can be represented by rational coordinates and a rational tangent.

The objects in an optical system are described by a finite system of inequalities and equalities. We restrict ourselves to systems including only linear and quadratic inequalities and equalities. If the system can be described by a set of equalities and inequalities where all coefficients in the system are rational, then the optical system is said to be *rational*. Otherwise, the optical system is classified as *irrational*. The degree of representation, namely, linear or quadratic, is also used to describe the optical systems. The optical system is *linear* if it can be represented by linear inequalities and equalities; the system is *quadratic* if it is represented by quadratic inequalities and equalities.

If the path of rays in an optical system can be described in rational coordinates (that is, as a union of line segments with endpoints in rational coordinates), the ray-tracing problem is said to be *rational*. If the path requires irrational representations, the ray-tracing problem is said to be *rational*. The rational ray-tracing problems are recursively enumerable in a strong sense, since each path from one surface to another can be rationally represented, and each subsequent path can be computed by rational representation. The irrational ray-tracing problems could be recursively enumerable, if the computation on irrational values by some symbolic manipulation is considered recursively enumerable. Since computing irrational representations seems difficult, we focused most of our attention on rational problems. Still, we found that even these problems are undecidable.

5. Computability and Complexity

5.1. *Three-Dimensional Rational Quadratic Optical Systems with Reflective or Refractive Objects*

Optical System. In our first optical model we assume that each optical system consists of a finite set of reflective objects with linear and parabolic surfaces, or refractive objects with linear and hyperbolic surfaces. In the first subcase we do not require refractive objects at all. In the second subcase we may not need reflective objects, since reflective surfaces may be implemented by using (total) reflection from refractive objects. We restrict the use of quadratic surfaces so that the positions and tangents of the reflected rays can be traced in rational coordinates as in Fig. 2.

Turing Machines. We show that ray tracing in the system is undecidable by simulating a universal Turing machine as an optical system. In particular, this implies that some optical system simulates some universal Turing machine. The model of a turing machine that we use has a finite control, a tape which contains cells and input, and a tape head that reads one cell on the tape at a time. The tape is infinite in both directions. For a more detailed description of Turing machines, see [14].

The set of tape symbols Γ is $\{0, 1\}$. We assume that the set of states in the finite control is $Q = \{q_1, q_2, \ldots, q_s\}$, that δ is the transition function, that q_1 is the initial state, and that q_s is the (only) final state.

Our discussion assumes reversible Turing machines. A reversible computation consists of a sequence of 1:1 deterministic operations that are also deterministic backward. A Turing machine is said to be *reversible* if its transition function is 1:1, so that there is exactly one predecessor for each whole machine state. Bennett has demonstrated that a reversible Turing machine can be built to perform any possible recursively enumerable computation [1].

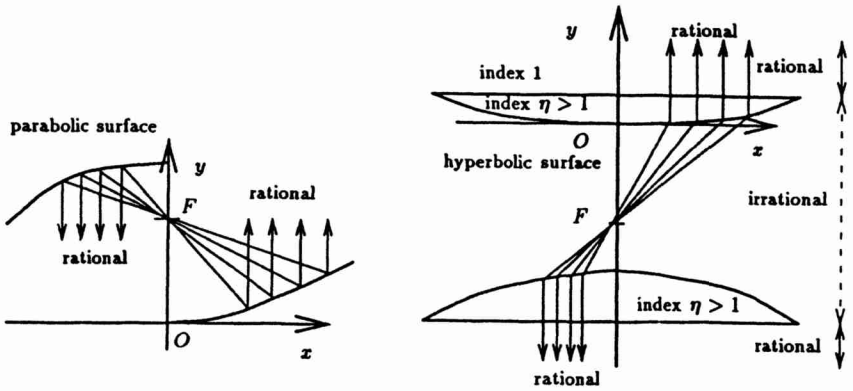

Fig. 2. Rational ray tracing with parabolic and hyperbolic surfaces.

Simulation. We show that a Turing machine can be simulated by an optical system. We view this optical system as a set of complex optical boxes. Each complex box consists of a set of basic boxes with linear mirrors, and either parabolic mirrors or hyperbolic lenses. Each complex box has a unit square through which the ray enters (always entering perpendicular to this surface), and one or two unit squares from which the ray exits (always exiting perpendicular from this surface). These unit squares are called the *entrance widnows* and the *exit windows*, respectively. The tape is encoded by the (X, Y) coordinates of the ray relative to the unit square windows. We organize these complex boxes so that the whole system simulates the Turing machine.

Each complex box corresponds to one state of the Turing machine's finite control, and implements the transition function defined for that state. The ray enters the entrance window and exits out of one of the two exit windows depending on which state the Turing machine may enter next. The system then projects the ray onto the next complex box while preserving the coordinates of the ray relative to the window, thus simulating the transition of states defined on the Turing machine. This is the general idea of how to simulate the Turing machine with this optical model. Next we describe this idea in more detail.

Representation of Operations. We represent the storage tape of M by using two binary fractions U and V. Let u_0 be the symbol at the tape head. Let $u_1, u_2, u_3, \ldots$ be the successive symbols to the left of u_0, and let $v_0, v_1, v_2, \ldots$ be the successive symbols to the right of u_0. This is shown in Fig. 3. Then we can represent the storage tape by using two numbers U and V:

$$U = \sum_{i=0}^{\infty} u_i/2^{i+1}, \tag{1}$$

$$V = \sum_{i=0}^{\infty} v_i/2^{i+1}. \tag{2}$$

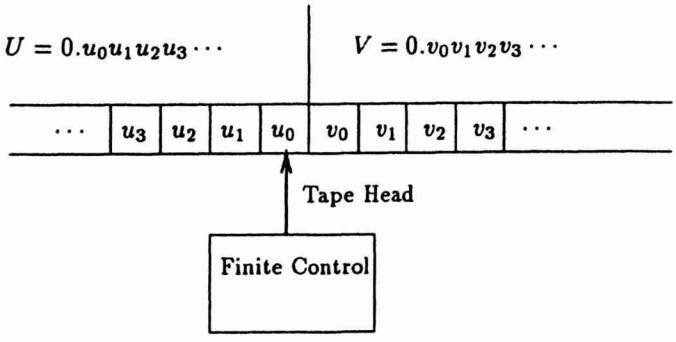

Fig. 3. A Turing machine and two numbers U and V which represent the tape

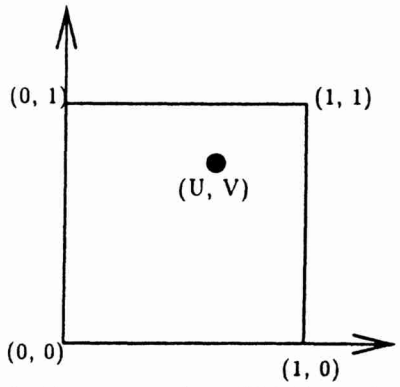

Fig 4. The representation of (U, V) in a unit square

Since $0 \leq U < 1$ and $0 \leq V < 1$, U and V form the coordinates of a point lying in the unit square where the left-bottom corner is its origin as depicted in Fig. 4.

Consider the mapping from the transition function δ of M into the transition operation of this optical system. The transitions can be divided into two cases, *left moves* as in a form $\delta(q, c) = (q', c', L)$ and *right moves* as in a form $\delta(q, c) = (q', c', R)$. Here, q is the current state, q' is the next state, $c \in \{0, 1\}$ is the symbol which the tape head has scanned, and $c' \in \{0, 1\}$ is the symbol which the tape head writes on the tape. L denotes a left move, and R denotes a right move. These transitions are depicted in Fig. 5.

Operations on U and V.

1. : (Left move) $\delta(q, c) = (q', c', L)$.
   Let U', V' be the values of U and V, respectively, after this transition. Then U' and V' can be written as

$$U' = 2 \sum_{i=1}^{\infty} u_i/2^{i+1} = 2(U - u_0/2), \tag{3}$$

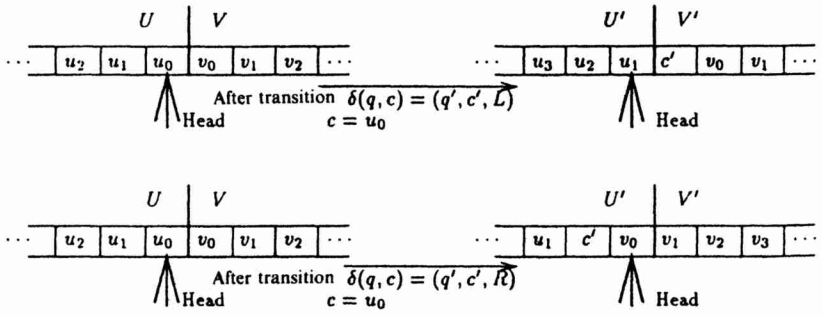

Fig. 5. State transition and tape

$$V' = c'/2 + \tfrac{1}{2} \sum_{i=0}^{\infty} v_i/2^{i+1}$$

$$= c'/2 + V/2. \tag{4}$$

2. : (Right move) $\delta(q, c) = (q', c', \mathrm{R})$.
In this case U' and V' can be written as

$$U' = v_0/2 + c'/4 + \tfrac{1}{2} \sum_{i=1}^{\infty} u_i/2^{i+1}$$

$$= v_0/2 + c'/4 + (U - u_0/2)/2, \tag{5}$$

$$V' = 2 \sum_{i=1}^{\infty} v_i/2^{i+1} = 2(V - v_0/2). \tag{6}$$

If we can implement complex optical boxes that can "read" the ray entering the window, perform the transformations listed above, and then redirect the ray to a complex optical box corresponding to the next state, we will have succeeded in simulating the Turing machine. Next we describe how to build these complex boxes.

Basic Boxes. First we describe basic boxes that can be used by each complex box.

- Readout box.
We assume that U and V are represented as a position on a unit square lying on the X–Y plane. The readout box uses two flat mirrors (reflective), or two prisms (totally reflective), making an angle $\pi/2$ at $x = 0.5$ as illustrated in Fig. 6.

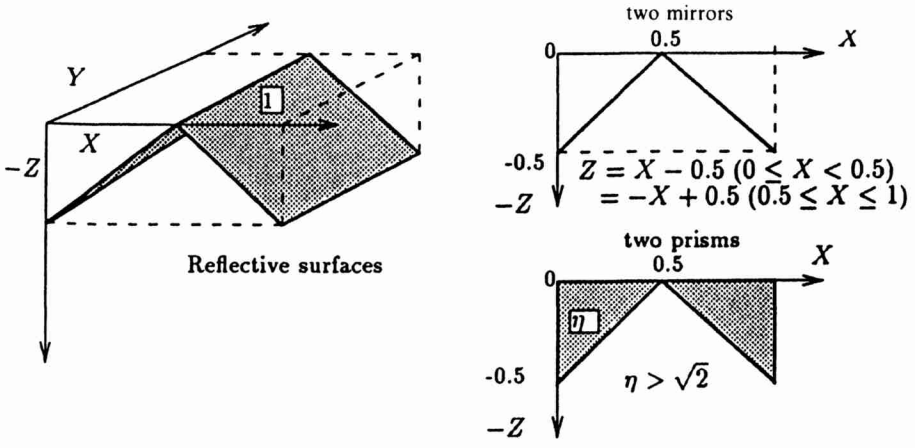

Fig. 6. Readout box

The figure shows several views from different angles. The ray entering a box from the unit square on the X–Y plane hits one of two flat reflective surfaces in the box. We call the unit square through which the ray enters the *entrance window*, and a window of size 1×0.5 from which the ray exits the box an *exit window*. There are two exit windows. If $U < 0.5$, the ray will be reflected through the left exit window. If $U \geq 0.5$, the ray will be reflected through the right exit window. We observe that V maintains its value along the Y axis from the entrance window to one of the exit windows. On the other hand, U loses its value along the X axis, but the value $U - u_0/2$ is obtained along the Z axis at one of the exit windows. Thus, no information is lost.

- Multiply2 (divide2) box.
  We need a box which performs the multiplication by two (division by two) operations used in the equations. This can be done by using a pair of cylindrical parabolic mirrors (reflective surfaces) or plano-convex hyperbolic lenses (refractive surfaces) placed with rational endpoints as in Fig. 7. (Note that conventional spherical lenses or mirrors introduce aberration [4] and should not be used here. Use of such objects would only make the decision problem more difficult, since it would introduce irrationality.)

  Similarly, we use the terms, *entrance window* and *exit window* to denote the window through which the ray enters, and the window from which the ray exits.
- Beam turner box.
  We may need this box in order to change the direction of the ray by $\pi/2$. This box is merely a mirror or a prism oriented at a $\pi/4$ angle. If using a prism, we assume the index of refraction is greater than $\sqrt{2}$ to cause total internal reflection at an incident angle $\pi/4$.

Next we combine these basic optical boxes to construct complex boxes for implementing the transition function δ. We retain the notation introduced above.

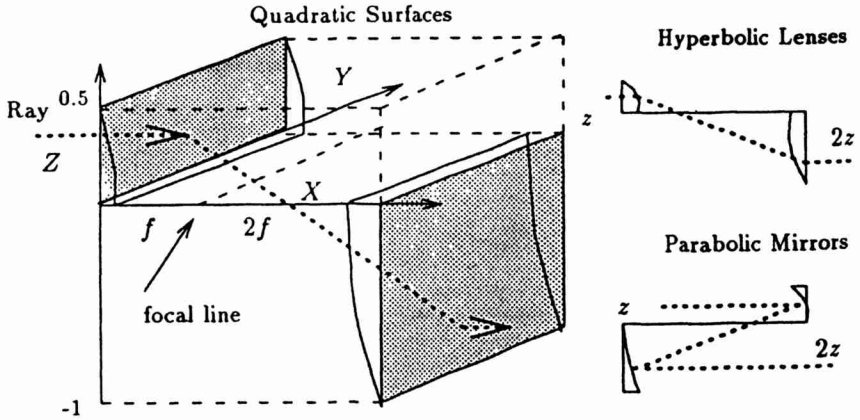

Fig. 7. Multiply2 (divide2) box using quadratic surfaces.

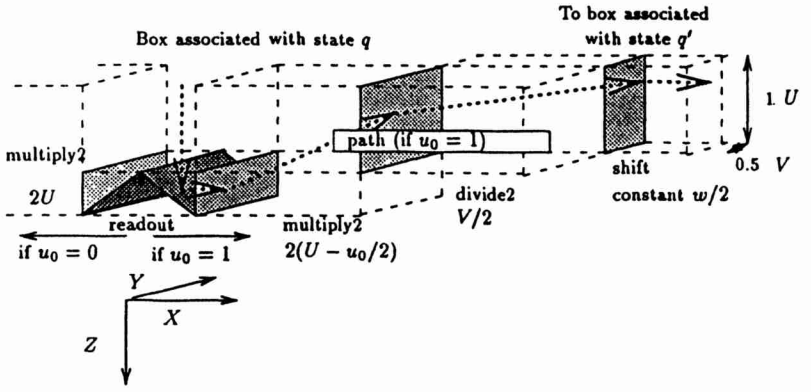

Fig. 8. Implementing $\delta(q, c) = (q', c', L)$.

Implementing (Left Move) $\delta(q, c) = (q', c', L)$. We must configure boxes to project the ray at (U', V'):

$$U' = 2(U - u_0/2),$$

$$V' = c'/2 + V/2.$$

Figure 8 shows a physical implementation of this function.

For each state, a separate complex box, such as shown in Fig. 8, is constructed. The box in the middle is a readout box that reads u_0. We call the unit square at the readout box the *entrance window* of the state. Now, suppose $u_0 = 1$, then the ray enters the multiply2 box on the right. At this point, V has not changed its value. The ray comes out of the plane parallel to the Z–Y plane, and the Z coordinate on the plane represents the value of U'. The ray which exits the multiply2 box enters the divide2 box. This halves the value of V. The ray which exits the divide2 box enters the entrance window of state q' with shifting by the value $c'/2$. The path that would be taken if $u_0 = 0$ can be similarly organized, and the ray can enter the entrance widow of the next state.

Implementing (Right Move) $\delta(q, c) = (q', c', R)$. In this case we must project the ray at (U', V'):

$$U' = v_0/2 + c'/4 + (U - u_0/2)/2,$$

$$V' = 2(V - v_0/2).$$

Figure 9 shows an implementation of this transition.

Undecidability. By simulating each δ function as a complex box and routing the rays between the boxes (using beam turner boxes), we can simulate any reversible

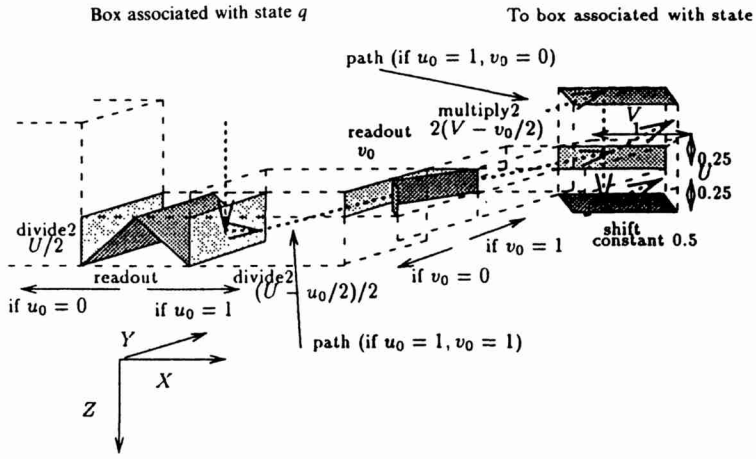

Fig. 9. Implementing $\delta(q, c) = (q', c', R)$.

Turing machine, and, in particular, some deterministic reversible universal Turing machine. The number of complex boxes which we require is equal to the number of finite states of the Turing machine. Each complex box consists of a finite number of basic boxes that can be represented in rational coordinates. To show our model can indeed simulate a universal computation, we employ Bennett's result and simulate a deterministic reversible Turing machine.

We partition the exit window into eight zones as in Fig. 10. If there is more than one exit window which must be routed to the entrance window of a particular state q', each exit window must be partitioned into several zones, and these zones must be interleaved and routed to the entrance window of q'. There cannot be any overlapping of zones from two distinct exit windows when simulating a reversible Turing machine, which executes 1:1 transitions.

The initial transition can be invoked by projecting a ray at the position that represents the input of the Turing machine. A transition to the final state is

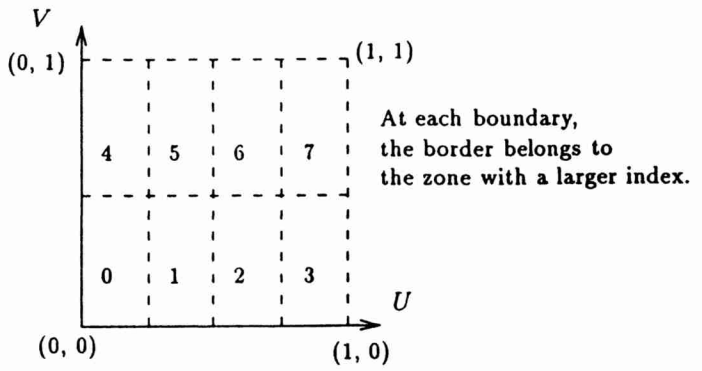

Fig. 10. Exit window partitioned into eight zones.

simulated by the ray entering a state box that is designated for the final state. By induction, all reversible transitions that are defined by the Turing machine can be implemented by our optical system. We conclude that our optical system can simulate some reversible universal Turing machine. Thus, we have:

Theorem 5.1. *The ray tracing of three-dimensional optical systems consisting of quadratic reflective or refractive objects is undecidable, even if all the objects are presented by a system of rational quadratic inequalities.*

5.2. *Three-Dimensional Rational Linear Optical Systems with Refractive Objects*

Optical System. Next we restrict ourselves to objects with linear surfaces. We assume the index of refraction of the refractive objects to be $\eta = 2$. This choice maintains the trajectory of rays in rational representations, but ray-tracing problems in this domain remain undecidable, giving us an interesting scenario. We note that some other value such as $\eta = 7$ also gives the same scenario. Since reflective surfaces can be obtained by using (total) reflection from refractive objects, we have no need for reflective objects.

Basic Boxes.

- Readout box.
  Use the same configuration as in Section 5.1.
- Multiply2 (divide2) box.
  We construct the multiply2 (divide2) box from a prism whose boundaries are represented by a set of rational linear equations as illustrated in Fig. 11. The prism is appropriately oriented so that rays hit the surface at angle θ_i. The refracted rays travel through the prism at angle θ_r and emerge from the other surface of the prism. The transmitted rays are bent from the initial angle by

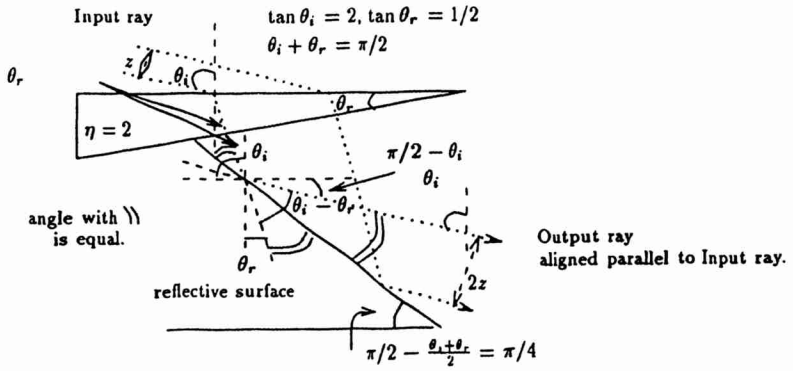

Fig. 11. Multiply2 (divide2) box implemented by a prism.

$\theta_i - \theta_r$. Setting $\eta = 2$ determines the incident angle θ_i' and the refracted angle θ_r. To implement the multiply2 (divide2) box by a single prism, we need $\theta_i = 2$ and $\theta_r = \frac{1}{2}$. In order to route rays between optical boxes, the output rays must be aligned parallel to the input rays. This can be done by using a reflective surface oriented at angle θ_a:

$$\theta_a = \pi/2 - (\theta_i + \theta_r)/2. \tag{7}$$

In our case we have $\theta_a = \pi/4$. Thus, all the surfaces can be represented by a set of rational linear equations, and the trajectory of rays can be traced in rational coordinates. (Note: If we use $\eta = 7$, we have $\tan \theta_i = \frac{7}{4}$, $\tan \theta_r = \frac{1}{8}$, and $\tan \theta_a = \frac{3}{2}$).

- Beam turner box.
  Use the same configuration as in Section 5.1.

As we have done with the previous model, we combine these basic optical boxes to construct complex boxes that implement the transition function δ.

Undecidability. We have demonstrated that our rational linear optical system can implement the multiply2 (divide2) box. Thus, we have:

Theorem 5.2. *The ray tracing of three-dimensional optical systems consisting of linear refractive (and reflective) objects is undecidable, even if all the objects are represented by a system of rational linear inequalities.*

5.3. *Three-Dimensional Rational Linear Optical Systems with Rectangular Reflective and Refractive Objects*

Optical System. The previous two models use quadratic mirrors, lenses, or prisms. We now eleminate these objects from our model. Each optical system in this model consists of a finite set of rectangular reflective and refractive objects whose boundaries are represented by a set of linear rational equations. All rectangular objects have their surfaces oriented either parallel or perpendicular to one another. The trajectory of rays in this model cannot be represented in rational coordinates. The ray-racing decision problem in this model is undecidable.

2-Counter Machine. In order to show that ray tracing in this model is undecidable, we simulate a 2-counter machine with this optical model. Since a 2-counter machine can simulate an arbitrary Turing machine, we can conclude that ray-tracing problems in this model are undecidable. A 2-counter machine has a finite control, and two counters that can store arbitrarily large numbers. The numbers can be incremented, decremented, or tested for zero.

Simulation. First we show the relationship between an optical system and a 2-counter machine. The optical system has a set of optical boxes that is capable of shifting rays by an irrational distance along either the X or Y axis by using rectangular refractive objects. Each box has a unit square through which the ray enters normally, and has a unit square from which the ray exits normally. Similarly, we call the unit square through which the ray enters the *entrance window*, and the unit square from which the ray exits the *exit window*. Two numbers U and V can be encoded as a position ($U\xi \bmod 1$, $V\xi \bmod 1$) relative to the unit square, where ξ is some irrational number. The 2-counter machine has a finite number of states, and each state can be represented by a box. Both incrementing and decrementing a number can be done by shifting the ray by ξ in modulo 1 space. Checking if a number is zero can be done by testing to see if the ray is incident with the rim of a unit square.

Representation of Operations. Use U and V to denote numbers stored by two counters. In this optical system U and V are encoded as a position on a unit square. Let ξ be an irrational number such that $0 < \xi < 1$. The operations for a 2-counter machine are:

- Operation 1: increment (or decrement) a number by 1.
  Shift the ray along the X or Y axis by ξ (or $-\xi$) in modulo 1 space. The shift operations can be implemented by rectangular refractive object surfaces as in Fig. 12. ξ becomes mostly irrational for rational τ and $\tan\theta_i$. The modulo operation can be implemented using a reflective surface and a partially reflective surface that are placed in rational coordinates as in Fig. 13. We note that it is possible to add a modulo 1 operation to the shifting operation by using only reflective surfaces at irrational coordinates. We demonstrate this in the next model.
- Operation 2: check if a number is 0.
  We test if U is zero by checking whether the ray passing through $X = 0$. We can do this by using the fact that we have mirrors with open edges. The case for V is handled similarly.

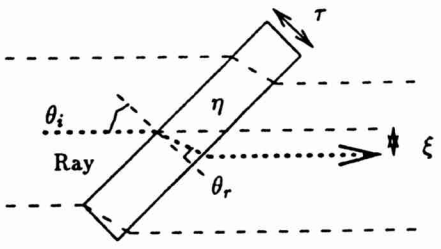

Fig. 12. Shift ξ operation by a refractive slab.

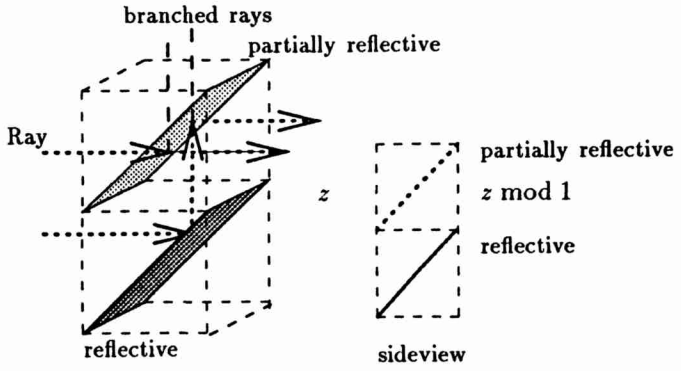

Fig. 13. Modulo 1 box.

- Operation 3: transition from one state to another.
  We must implement a way to go from one state to another. Unlike the previous two models that simulate Turing machines, this counter machine does not achieve reversibility by partitioning the unit square. Thus, we need some way to combine two or more exit windows to a single window. This can be carried out by using beam splitters which are often used in optical experiments. A beam splitter consists of a partially reflective surface so that two beams from different directions can emerge from the same surface as in Fig. 14.

Since $m\xi \bmod 1 = 0$ if and only if $m = 0$, we have:

Lemma 5.1. *This optical system can simulate a counter.*

Basic Boxes. First we describe basic boxes that are used in this system.

- Shift box.
  This box shifts rays by an irrational distance ξ. It uses a rectangular refractive slab as shown in Fig. 12.
- Modulo 1 box.
  The modulo operation uses a reflective surface and a partially reflective surface as in Fig. 13.

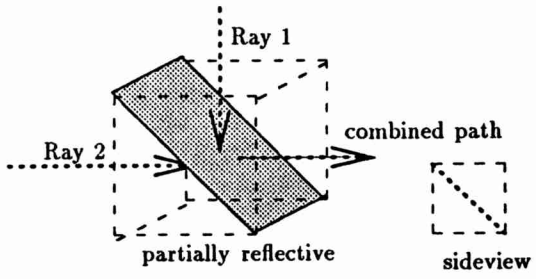

Fig. 14. Beam combiner box.

- Beam combiner box.
  This box merges two windows into one. It requires only one partially reflective surface as in Fig. 14.

Undecidability. Since a 2-counter machines can simulate universal Turing machines, we have:

Theorem 5.3. *The ray tracing of three-dimensional optical systems consisting of linear rectangular reflective and refractive objects is undecidable, even if all the rectangular objects are represented by a system of rational linear inequalities.*

5.4. *Three-Dimensional Linear Optical Systems with Reflective and Partially Reflective Surfaces*

Optical System. Next we consider optical systems that consist purely of reflective and partially reflective surfaces—without any "slabs" containing nonreflective surfaces. We use the same approach as the previous model, but instead of using rationally represented refractive objects, we use irrationally represented reflective surfaces to implement the shifting operations.

Basic Boxes. We use the same basic boxes as in the previous model, but they are implemented differently.

- Shift in modulo 1 box.
  This box uses six rectangular reflective surfaces as shown in Fig. 15. The surfaces are oriented at an angle whose tangent is rational, but the distance of the shift becomes irrational.
- Beam combiner.
  This box uses a reflective surface and a partially reflective surface as before.

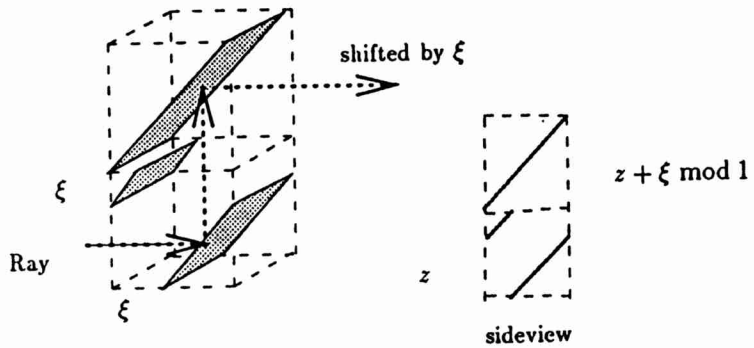

Fig. 15. Shift in modulo 1 box.

We immediately have:

Theorem 5.4. *The ray tracing of three-dimensional optical systems consisting of linear reflective and partially reflective surfaces is undecidable, if some surfaces are allowed to be represented by irrational linear inequalities.*

5.5. *Three-Dimensional Rational Linear Optical Systems with Reflective Surfaces*

Optical System. Next we restrict ourselves to rational optical systems with only reflective surfaces.

Augmented Bounded 2-Counter Machine. First we define a new machine type:

Definition 5.1. A 2^n augmented bounded 2-counter machine has two counters that count up to $2^n - 1$. Furthermore, it can add or subtract 2^{n-1} from a number and read the nth and the $(n-1)$th bits of the binary representation of a number.

We simulate a 2^n augmented bounded 2-counter machine as a rational optical system. The augmented 2-counter machine is unusual in that it has the ability to add or subract 2^{n-1} from a number and to read the nth and $(n-1)$th bits of the binary representation of a number. The relevance of this ability for our application is explained later.

Representation of Operations. We describe the operations which we need to simulate a 2^n augmented bounded 2-counter machine. Incrementing or decrementing a number are handled as before, except ξ is set to the rational value 2^{-n}. We use U and V to denote two numbers. We consider n to be the input size, since the optical system can be described in a polynomial in n number of bits.

Once again, we represent U and V as a position (U mod 1, V mod 1).

Lemma 5.2. *Each optical counter can count up to* $2^n - 1$.

Proof. Since ξ is a rational number 2^{-n}, for any integer m we have $m\xi \bmod 1 = 0$ if and only if $m = k2^n$, where k is an integer. Hence the counter can count up to $2^n - 1$. □

Next we consider the additional operations of the augmented 2-counter machine:

- Add or subtract 2^{n-1} from a number.
  This can be implemented by shifting the ray along the X or Y axis by 0.5.
- Test if a number is less than 2^{n-1}.
  We use the "readout box" introduced in Section 5.1.

Using these boxes, we can build an optical system that simulates a 2^n augmented bounded 2-counter machine.

Complexity.

Lemma 5.3. *Augmented bounded 2-counter machines can decide any PSPACE problem.*

Proof. Karp [16] has shown that the problem of determining whether or not a deterministic n-space bounded Turing machine accepts its input is PSPACE-complete. As for the reversibility of space-bounded Turing machines, Bennett [2] has shown that any ordinary Turing machine running in space S can be simulated by a reversible one running in space $O(S^2)$. Thus, suffice it to show that the 2^n augmented bounded 2-counter machine can simulate a deterministic reversible n-space Turing machine. Any linear space Turing machine has a storage tape of length n, where n is the input size. We show that the 2^n augmented bounded 2-counter machine simulates a Turing machine. We consider the tape of the Turing machine to be cyclic as in Fig. 16. We can view the storage tape as a binary number, where the second highest bit of the number lies under the tape head. This arrangement partitions the unit square into eight zones using the three bits near the tape head. Assuming the Turing machine is reversible, these zones must be interleaved together without overlapping when two exit windows are routed to the entrance window of a particular state.

Let C_{TM} be the binary representation of the tape. Then C_{TM} is an integer that satisfies $0 \leq C_{\mathrm{TM}} < 2^n$. One of the counters of the augmented bounded 2-counter machine stores this number, C_{TM}. Each time when the tape head moves left or right, we must compute the new C_{TM} which represents the tape at the subsequent step. Reading symbols at the tape head and its right can be done by using two half-size readout boxes placed in a unit cube. In order to compute the new C_{TM} for a left move, we first remove the nth bit, u_{n-1}, and replace u_{n-2} with c'. Then we multiply the number by two (using the second counter), and finally we add one if the original nth bit was 1. Similarly, the new C_{TM} for a right move can be computed. The simulation is straightforward, completing the proof. □

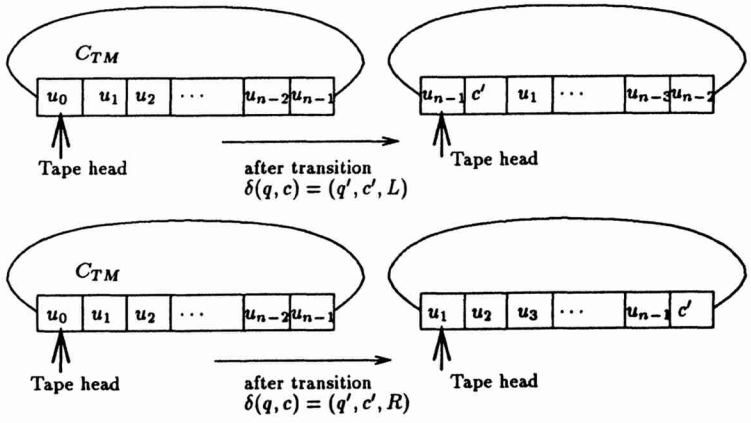

Fig. 16. Cyclic Turing machine and its tape.

Thus, any linear-space deterministic reversible Turing machine can be transformed in polynomial time to a 2^{n-1} augmented bounded 2-counter machine that is described as an optical system. There are a finite number of boxes in the optical system, where each box can be represented by a polynomial of n bits. Thus, the description of the optical system is of size polynomial of n, otherwise the transformation would be nonpolynomial time. We immediately have:

Theorem 5.5. *The ray tracing of three-dimensional optical systems consisting of linear reflective surfaces is PSPACE-hard, if all the surfaces are represented by a system of rational linear inequalities.*

5.6. *d-Dimensional Rational Linear Optical Systems with Orthonormal Reflective Surfaces*

Optical System. In this model we consider rational optical systems where the mirrors lie parallel or perpendicular to each another in d dimensions.

Complexity. We show that ray tracing in the system is in PSPACE. First we show the problem is in PSPACE if the angle that the initial ray makes with the first mirror it strikes is $\pi/4$.

Lemma 5.4. *The ray tracing in this model is in PSPACE, if the angle that the initial ray makes with the system is* $\pi/4$.

Proof. Since the input to the optical system is encoded in n bits, we can construct grid lines at every $2^{-p(n)}$ interval such that the ray always intersects grid lines at the grid points, where $p(n)$ is a polynomial of n. Figure 17 illustrates an example.

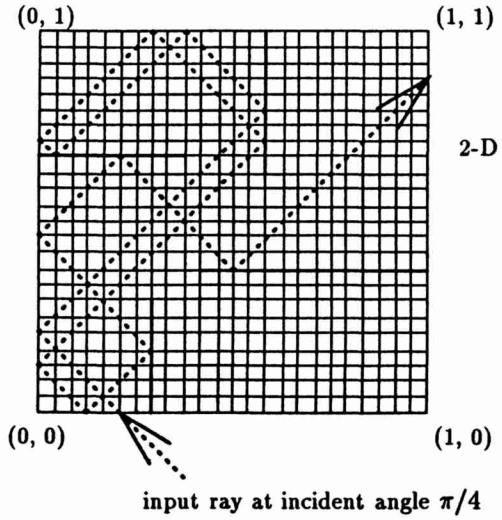

Fig. 17. Grid lines.

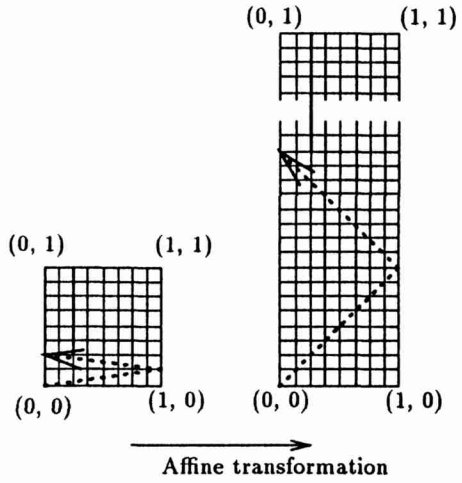

Fig. 18. Expanding grid lines.

The number of the grid points in the d-dimensional system is $2^{dp(n)}$. We give a nondeterministic polynomial-space algorithm for the ray-tracing problem. Let M_n be a nondeterministic polynomial-space Turing machine which solves this problem. M_n has two vectors, P_c and P_n. P_c represents the current position and direction of the ray. P_n represents the position and direction of the next reflected ray after leaving the current position. M_n nondeterministically guesses P_n from P_c, and determines the correct P_n in a polynomial space. Initially, P_c stores the position and direction of the ray. We assume inductively that P_c stores the current position and direction of the ray. M_n halts if it reaches the specified final position. Since these vectors can be encoded as a polynomial of n, we can construct M_n to solve the problem. By Savitch's theorem [24], the problem is in PSPACE.

Next, we consider the case in which the incident angle α is not $\pi/4$. We only give a sketch of the proof. We simply note that the system can be reduced to the $\pi/4$ case by transforming the system via the multiplication of one of the axes by the tangent of α as illustrated in Fig. 18. The proof follows from the argument above. □

We hence have:

Theorem 5.6. *The ray tracing of d-dimensional rational optical systems consisting of a finite set of orthonormal reflective surfaces is in PSPACE.*

5.7. Other Optical Ray-Tracing Phenomena

By using linear iterative maps which generate ∞-distributed sequences [18], we show that an optical system exists such that if a ray enters from a single fixed

point, it visits an edge of unit length at every point densely and uniformly in the range [0, 1).

First we describe an ∞-distributed sequence which is dense and uniform in the range [0, 1).

Lemma 5.5. *$\{\xi^i \bmod 1 | i$ integer $\geq 0\}$ is dense and uniform in the range [0, 1) (∞-distributed sequence in the range [0, 1)) for almost all real numbers $\xi > 1$.*

Proof. The proof of this lemma is found in [9] and [18]. □

This leads to:

Theorem 5.7. *An optical system exists such that if a ray enters from a single fixed point, the set of points it visits on an edge of unit length is dense and uniform over the range [0, 1).*

Proof. The idea is to construct an optical box which performs the multiplication by an irrational number ξ which satisfies Lemma 2.5. The implementation of modulo operations has already been introduced in Sections 5.3 and 5.4. To perform the multiplication by an irrational number ξ, we use two lenses or mirrors, where one of the ratios of their focal lengths is ξ. Then, by placing them together as in the multiply2 box in Section 5.1, we can construct an optical box which implements the multiplication by ξ. □

We can extend this type of problem by considering polygons whose sides form mirrors. We can then pose the *illumination problem*: To find an initial position and orientation which would cause a light beam to visit all the interior edges of the polygon densely. These problems are often called *billiard-ball* problems, and the literature contains a number of "art-gallery theorems" about these models. One particular result [3], [17], [22] states that for every polygon whose angles are rational multiples of π, the illumination problem has a solution. We ask the following extension of the illumination problem: Can the points on the polygon be visited densely and uniformly?

6. Conclusion and Further Problems

This paper has classified a wide degree of ray-tracing scenarios and given lower bounds for many of the problems. This work considers only geometric constructs of reflective and refractive objects. These models give interesting results in terms of pure computational geometry.

There are some interesting ray-tracing problems that remain open. Here are a few:

- Find a lower bound for computations of ray tracing in *two*-dimensional rational optical systems with reflective and/or refractive objects.

- Find a lower bound for computations of ray tracing in *two*-dimensional rational linear optical systems with only reflective and partially reflective surfaces. Note that the *three*-dimensional case is PSPACE-hard, but we have no lower bound in the *two*-dimensional case.
- Determine if ray tracing in rational optical systems with only reflective linear surfaces is decidable.

References

[1] C. Bennett, Logical Reversibility of Computation, *IBM J. Res. Develop.* **17** (1973), 525–532.
[2] C. Bennett, Time/Space Trade-Offs for Reversible Computation, *SIAM J. Comput.* **18** (1988), 766–775.
[3] C. Boldrighini, M. Keane, and F. Marchetti, Billiards in Polygons, *Ann. Probab.* **6** (1978), 532–540.
[4] M. Born and E. Wolf, *Principles of Optics*, 6th edn., Pergamon Press, Oxford, 1980.
[5] P. Bui-Tuong, Illumination for Computer-Generated Pictures, *Comm. ACM* **18**(6) (1975), 311–317.
[6] J. P. Eckmann and D. Ruelle, Periodic Theory of Chaos and Strange Attractors, *Rev. Modern Phys.* **57**(3) (1985), Part 1.
[7] R. Feynman, *QED: The Strange Theory of Light and Matter*, Princeton University Press, Princeton, NJ, 1985.
[8] E. Fiume, *The Mathematical Structure of Raster Graphics*, Academic Press, New York, 1989.
[9] J. N. Franklin, Deterministic Simulation of Random Processes, *Math. Comp.* **XVII**, No. 81–84 (1963), 28–59.
[10] A. Glassner, ed., *Introduction to Ray Tracing*, Academic Press, New York, 1989.
[11] D. Greenberg, Light Reflection Models for Computer Graphics, *Science* **244** (1989), 166–173.
[12] R. Guenther, *Modern Optics*, Wiley, New York, 1990.
[13] P. Hanrahan, Ray Tracing Algebraic Surfaces, *Comput. Graphics* **17** (1983), 83–90.
[14] J. Hopcroft and J. Ullman, *Introduction to Automata Theory, Languages, and Computation*, Addison-Wesley, Reading, MA, 1979.
[15] K. Iizuka, *Engineering Optics*, 2nd edn., Springer-Verlag, New York, 1987.
[16] R. Karp, Reducibility Among Combinatorial Problems, in *Complexity of Computer Computations*, R. Miller and J. W. Thatcher, eds., pp. 85–103, Plenum, New York, 1972.
[17] S. Kerckhoff, H. Masur, and J. Smillie, A Rational Billiard Flow Is Uniquely Ergodic in Almost Every Direction, *Bull. Amer. Math. Soc.* **13** (1985), 141–142.
[18] D. Knuth, *The Art of Computer Programming*, Vol. 2, 2nd edn., pp. 142–168, Addison-Wesley, Reading, MA, 1981.
[19] T. Y. Li and J. Yorke, Period Three Implies Chaos, *Amer. Math. Monthly* **82** (1975), 985–992.
[20] C. Moore, Unpredictability and Undecidability in Dynamical Systems, *Phys. Rev. Lett.* **64** (1990), 2354–2357.
[21] I. Newton, *Opticks, or A Treatise of The Reflections, Refractions, Inflections & Colours of Light*, 4th edn., London, 1730.
[22] J. O'Rourke, *Art Gallery Theorems and Algorithms*, Oxford University Press, New York, 1987.
[23] J. Reif, D. Tygar, and A. Yoshida, The Computability and Complexity of Optical Beam Tracing, *Proc. 31st Annual Symposium on Foundations of Computer Science*, Vol. I, 1990, pp. 106–114.
[24] W. J. Savitch, Relationships Between Nondeterministic and Deterministic Tape Complexities, *J. Comput. System Sci.* **4**(2) (1970), 151–156.
[25] A. Watt, *Fundamentals of Three-Dimensional Computer Graphics*, Addison-Wesley, Reading, MA, 1989.
[26] T. Whitted, An Improved Illumination Model for Shaded Display, *Comm. ACM* **23**(6) (1980), 343–349.

Received February 19, 1991, and in revised form June 14, 1993, and August 8, 1993.

[24] W. J. Savitch, Relationships Between Nondeterministic and Deterministic Tape Complexities, *J. Comput. System Sci.* **4**(2) (1970), 151–156.
[25] A. Watt, *Fundamentals of Three-Dimensional Computer Graphics*, Addison-Wesley, Reading, MA, 1989.
[26] T. Whitted, An Improved Illumination Model for Shaded Display, *Comm. ACM* **23**(6) (1980), 343–349.

Received February 19, 1991, and in revised form June 14, 1993, and August 8, 1993.