Modeling reflection and refraction of freeform surfaces

J. E. Gómez-Correa[a], A. L. Padilla-Ortiz[b,*], and S. Chávez-Cerda[a]

[a]*Instituto Nacional de Astrofísica, Óptica y Electrónica, Coordinación de Óptica, Tonantzintla Puebla 72840, Mexico*

[b]*SECIHTI - Instituto de Ciencias Aplicadas y Tecnología, Universidad Nacional Autónoma de México, Ciudad de México 04510, México,*

[*]*e-mail: laura.padilla@icat.unam.mx*

Received 8 August 2024; accepted 17 September 2024

In this work, we present a detailed procedure for computer implementation of the laws of refraction and reflection on an arbitrary surface with rotational symmetry with respect to the propagation axis. The goal is to facilitate the understanding and application of these physical principles in a computational context. This enables students and instructors alike to develop simulations and interactive applications that faithfully replicate the behavior of light and sound propagating in a diversity of media separated by arbitrary surfaces. In particular, it can help to explore freeform optics. Additionally, we include a practical example demonstrating these implementations using either `Matlab` or open-source `Octave` programming language.

Keywords: Snell's law; reflection law; ray tracing; total internal reflection; freeform optics; non-imaging optics.

DOI: https://doi.org/10.31349/RevMexFisE.22.020211

1. Introduction

The rapid advances in LED technology have opened the necessity to investigate the focusing and reflecting properties of a variety of surfaces other than those commonly obtained from conics, giving birth to a new technology, freeform Optics [1]. This technology is based on Snell's law of refraction and the law of reflection that are fundamental in optics and acoustics, governing what happens to the propagation of light or sound in a medium when they encounter an interface with a different medium.

Refraction Snell's law describes how light and sound change direction when transitioning from one medium to another with different refractive indices, based on their respective angles of incidence and refractive indices [2-4]. This principle is essential for understanding image formation in lenses and the propagation of sound in various acoustic environments. Mathematically, this law can be expressed as:

$$n_i \sin\theta_i = n_t \sin\theta_t. \tag{1}$$

Here, a ray, representing the propagation of light or sound, initially propagates in a medium with a refractive index n_i and is incident on a medium with a refractive index n_t. The incident ray forms an angle θ_i with respect to the normal of the surface of the new medium, while the transmitted or refracted ray changes its direction of propagation and travels at an angle θ_t relative to the normal of the surface at the same point, see Fig. 1.

On the other hand, when light or sound strikes a surface the law of reflection states that the angle of incidence, θ_i, equals the angle of reflection, θ_r, with respect to the normal to the surface at the point of incidence, see Fig. 1. In mathematical terms [2,3], this law is represented as:

$$\theta_i = \theta_t. \tag{2}$$

This law is crucial for explaining how acoustic waves reflect in enclosed spaces and how images form in optical mirrors.

Ray tracing is a widely used technique in both optics and acoustics to model wave propagation. In this approach, waves are approximated as rays that propagate in straight lines through homogeneous media and refract or reflect when they encounter interfaces between media with different optical or acoustic properties. In optics, ray tracing is essential in the design of optical systems such as lenses, mirrors, and imaging instruments, where accurately predicting ray paths is crucial for optimizing image quality and reducing aberrations. In acoustics, this technique is useful for predicting sound propagation in enclosed spaces or urban environments, where reflections and refractions from surfaces are key to correctly modeling sound distribution. There are several ray tracing methods, with the most common being: exact ray tracing, paraxial ray tracing, matrix methods, and the y-nu method, which uses paraxial ray-trace equations to estimate

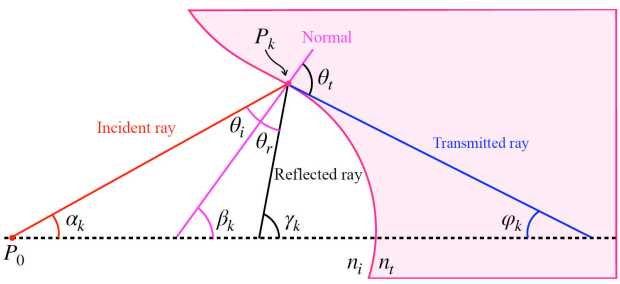


FIGURE 1. Geometric parameters of incident, reflected, and transmitted rays on a surface.

ray heights and slopes at each surface in an optical system [2,5-9].

In this work, we aim to provide readers with the necessary tools to implement Snell's Law of refraction and the law of reflection on an arbitrary surface with rotational symmetry about the propagation axis, using any programming language. In particular, we present a practical example providing `Matlab` code, compatible with `Octave` programming language (open access software), to demonstrate how these implementations can be executed [10-12]. The primary objective is to improve the understanding and extend the application of these physical principles using a computational tool. This will empower students and educators alike to explore and experiment with interactive simulations that accurately depict the behavior of light and sound interacting with diverse freeform structures.

The rest of the paper is organized as follows: Section 2 explains the generation of an arbitrary surface where the rays are incident. Section 3 presents the generation of the incident rays, followed by Sec. 4, which details the calculation of the normals to the surface at each point where the rays are incident. Section 5 then covers the calculation of the refracted and reflected rays by the surface. In Sec. 6, a series of classic examples from specialized literature are presented. Section 7 explains the phenomenon of Total Internal Reflection. Finally, the conclusions are provided in Sec. 8.

2. Surface of the medium

In this work, the interface between the two media is described by a curve, as ray tracing is conducted on an arbitrary surface with rotational symmetry around the propagation axis. This means that both the curve and the rays traced along it can be rotated around the propagation axis, generating a three-dimensional ray-tracing model. Therefore, we will explain how such a curve can be generated.

It is well known that a curve in a plane can be represented in three different forms, namely, using explicit functions, implicit functions or parametric functions. Some texts also refer to them indistinctly as equations instead of functions [13,14].

An explicit function involves a correspondence rule with one independent variable and one dependent variable, as shown in Eq. (3):

$$y = f(x), \tag{3}$$

where y is the dependent variable and x is the independent variable. An example of an explicit function is $y = \sqrt{1-x^2}$.

An implicit function, on the other hand, does not allow for a clear distinction between the independent and dependent variables; the dependent variable is not isolated. This is illustrated in Eq. (4):

$$f(x, y) = a. \tag{4}$$

The equation of the unit circle, $x^2 + y^2 = 1$, is an example of an implicit function. Notice that one can be tempted to solve for y but then one reaches the point where y is not uniquely determined for a given value of x as it occurs for explicit functions. Another more intricate example is the mathematician's love equation, $(x^2 + y^2 - 1)^3 - x^2y^3 = 0$, in which it is not possible to solve for any of the variables.

In a parametric function, the variables are written in terms of functions of a third independent variable called a parameter, commonly represented by t, and are thus independent of each other, namely,

$$\begin{aligned} x &= f(t), \\ y &= g(t). \end{aligned} \tag{5}$$

When the point coordinates (x, y) in the curve are described as functions of t, as above, it is said that the curve is parametrized in terms of the parameter t.

Generally, to construct a curve, we use the explicit equation or the parametric equation of the desired curve. The implicit equation is rarely used. Due to its simplicity, in this work, we will focus only on explicit and parametric functions.

The first step is to generate the curve computationally once the parametric functions have been established and together with the range of the parameter t. The latter consisting of a vector with m elements. We will consider t_i and t_f as the limits of the desired parameter t and they are such that $x_i = f(t_i)$, $y_i = g(t_i)$ and $x_f = f(t_f)$, $y_f = g(t_f)$ are the endpoints of the curve. It is clear that the value of m depends on the desired resolution to obtain smooth graphs, and the larger value for a better resolution. Then we proceed to evaluate the parametric functions.

To illustrate these steps, we will generate a curve using the `MATLAB` (`Octave`) programming language. The chosen curve is given by the following parametric equations:

$$\begin{aligned} x(t) &= a\cos(t)\sin(t)^2, \\ y(t) &= b\sin(t). \end{aligned} \tag{6}$$

The code snippet to generate and plot this curve looks like this:

```
% -- Input parametric curve functions --
%
m = 101;  %t-parameter sampling
n = 21;   % number of rays
ti = -pi/2.3;
tf = pi/2.3;
t = linspace(ti,tf,m);
a = 3;
b = 3;
x = a.*cos(t).*sin(t).^2;
y = b.*sin(t);
% ------------- Point P_0 --------------
x0 = -10;
```

```
y0 = 0;
% ---------- Plot the curve ----------
figure
plot(x,y,'m','linewidth',3)
axis equal
axis([-10 20 -6 6])
set(gca,'fontsize',18,'LineWidth',2)
set(gcf,'Color',[1,1,1])
% ------------------------------------
```

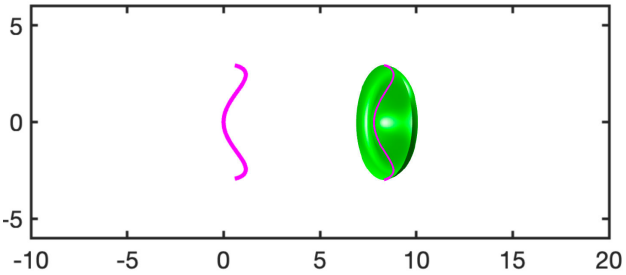


FIGURE 2. The curve was obtained from Eq. (6). For visualization purposes, we present the surface generated by rotating the curve around the propagation axis.

This code defines the parameter t over the interval $[-\pi/2.3, \pi/2.3]$ with 100 points, calculates the corresponding x and y values using the parametric functions (6), and then plots the resulting curve. Figure 2 shows the plot of the curve described by the given parametric equations. Having generated the curve, we will proceed to create the incident rays.

3. Incident rays on the curve

Let us consider n rays originated from a point source located at $P_0 = (x_0, y_0)$ arriving at the curve on a point $P_k = (x_k, y_k)$, with $k = 1, 2, ...n$. Consider $n < m$ to avoid saturation of the plot. The parameter m refers to the number of samples for the parameter t, as defined in the code snippet for the input parametric curve functions. Code snippet defining incident points P_k:

```
% -- Rays from P_0 to Curve at P_{k} --
% Parameter t is redefined to n elements
% Coordinates (x(k),y(k)) are calculated
% ---------- Curve points P_k ----------
t = linspace(ti,tf,n);
x = a.*cos(t).*sin(t).^2;
y = b.*sin(t);
% ------------------------------------
```

Since the points P_k on the curve are given by the parametric functions (6), to plot each of the rays emanating from the point source P_0 we use the equation of the straight line in two-point form, namely

$$y = \frac{y_k - y_0}{x_k - x_0}(x - x_0) - y_0. \tag{7}$$

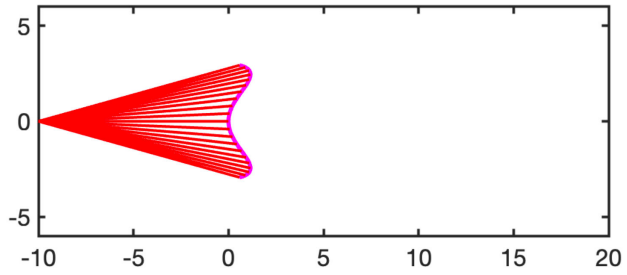


FIGURE 3. Incident rays on the surface.

Next is the code snippet to implement these equations:

```
% Incident Rays plotted from P_0 to P_k
hold on; %Keep curve to plot Rays
for k = 1:n
     xi = linspace(x0,x(k),m);
     mi = (y(k)-y0)/(x(k)-x0);
     yi = mi*(xi-x0)-y0;
     plot(xi,yi,'r','LineWidth',1);
end
hold off; %Release plot
axis equal
axis([-10 20 -6 6])
set(gca,'fontsize',18,'LineWidth',2)
set(gcf,'Color',[1,1,1])
clear xi yi mi
% ------------------------------------
```

The results are shown in Fig. 3.

Once having defined the points where the incident rays intersect the surface, the next step involves calculating the surface normal at each of these points.

4. Normal lines

A normal line to a curve at a specific point P_n is a line that is perpendicular to the tangent of the curve at that point. The normal line intersects the curve at the point of tangency and has a slope that is the negative reciprocal of the slope of the tangent at that point. The equation of the normal line can be expressed as:

$$y_N = -\frac{1}{M_k}(x_N - x_k) + y_k, \tag{8}$$

where (x_k, y_k) is the point of tangency, *i.e.*, the points P_k, while M_k is the slope of the tangent to the curve at that point.

For a surface defined by a parametric function, the slope of this tangent line is given by [13,14]

$$M_k = \left.\frac{dy}{dx}\right|_{P_k} = \frac{y'(t_k)}{x'(t_k)}, \tag{9}$$

where $y'(t_k) = \mathrm{d}y(t)/\mathrm{d}t|_{P_k}$, and $x'(t_k) = \mathrm{d}x(t)/\mathrm{d}t|_{P_k}$. This expression represents the derivative of y with respect to x evaluated at the point P_k in terms of the derivatives with respect to the parameter t.

Notice that if we substitute Eq. (9) into Eq. (8), we obtain the equation of the normal line at each of the points P_k on the surface.

For our example, the derivatives of the parametric equations $x(t)$ and $y(t)$ are:

For $x(t) = a\cos(t)\sin^2(t)$:

$$\left.\frac{dx}{dt}\right|_{P_k} = a\left[2\cos^2(t_k)\sin(t_k) - \sin^3(t_k)\right]. \tag{10}$$

For $y(t) = b\sin(t)$:

$$\left.\frac{dy}{dt}\right|_{P_k} = b\cos(t_k). \tag{11}$$

Then, the slope of the tangent line at the point P_k in the curve is:

$$M_k = \frac{b\cos(t_k)}{a\left[2\cos^2(t_k)\sin(t_k) - \sin^3(t_k)\right]}, \tag{12}$$

with this equation, we can determine the equation of the normal line at each of the points P_k on the curve, which is given by:

$$y_N = \frac{a\left[\sin^3(t_k) - 2\cos^2(t_k)\sin(t_k)\right]}{b\cos(t_k)} \times (x_N - x_k) + y_k. \tag{13}$$

To implement this equation, we define a domain for x_N large enough to contain the x_k of the curve; the code snippet in MATLAB is as follows:

```
% ----------- Normal lines -------------
% ---- x-interval for normals [x1,x2]---
x1 = -2;
x2 = 2;
hold on; %Keep curve to plot Normals
for k = 1:n
    xN = linspace(x1,x2,m);
    Mn = (b*cos(t(k)))/(a*(2*cos(t(k))
        ^2*sin(t(k))-sin(t(k))^3));
    yN = -(1/Mn)*(xN-x(k))+y(k);
    plot(xN,yN,'LineWidth',1);
end
hold off; % Release plot
axis equal
axis([-10 20 -6 6])
set(gca,'fontsize',18,'LineWidth',2)
set(gcf,'Color',[1,1,1])
% ---------------------------------------
```

Notice that the code finds the normal lines to the curve at each point P_k as shown in Fig. 4.

Once the lines normal to the curve have been calculated, we will proceed to calculate the reflected and transmitted rays.

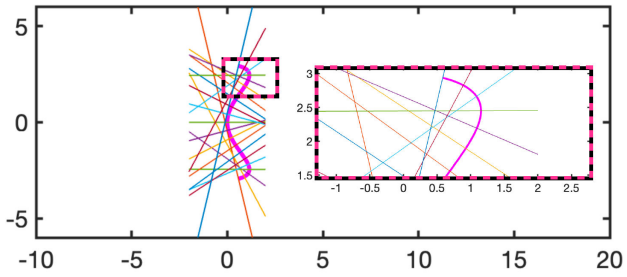


FIGURE 4. Normals to the surface (curve).

5. Reflected or transmitted rays by the surface

With all of the above, we have the necessary to obtain the refracted and reflected trajectories of the incident light or sound at the surface according to the Snell's and reflection laws described in the introduction. For this purpose, we endeavor to determine the slopes of the reflected and transmitted rays. We will make use of auxiliary angles α_k, β_k, γ_k and φ_k defined with respect to the Cartesian reference frame and determined by the point P_k at the curve as shown in Fig. 1.

A simple trigonometric calculation shows that the auxiliary reflected angle is given by $\gamma_k = 2\beta_k - \alpha_k$ with $\beta_k = \tan^{-1}(-1/M_k)$, $\alpha_k = \tan^{-1}(m_k)$; m_k and M_k the slopes of the incident ray and of the tangent to the curve at point P_k, respectively. Then, the slope of the reflected ray at any point P_k at the curve is given by $m_{rk} = \tan(\gamma_k)$ and the reflected rays are determined by the line equation given by

$$y_{rk} = \tan(\gamma_k)(x_{rk} - x_k) + y_k. \tag{14}$$

Below is the corresponding Matlab (Octave) code snippet to plot the reflected rays shown in Fig. 5.

```
% ----------- Reflected rays -----------
hold on; %Keep curve to plot Normals
for k = 1:n
    Mk = (b*cos(t(k)))/(a*(2*cos(t(k))
        ^2*sin(t(k))-sin(t(k))^3));
    betak = atan(-1/Mk);
    mk = (y(k)-y0)/(x(k)-x0);
    alphak = atan(mk);
    gammak = 2*betak-alphak;
    gammakGrad = gammak*180/pi;
    if (abs(gammakGrad)>90)
        xEnd = 6;
    else
        xEnd = -6;
end
xrk = linspace(x(k),xEnd,m);
    yrk = tan(gammak)*(xrk-x(k))+y(k);
    plot(xrk,yrk,'k','LineWidth',1);
 end
hold off; % Release plot
axis equal
axis([-10 20 -6 6])
set(gca,'fontsize',18,'LineWidth',2)
set(gcf,'Color',[1,1,1])
% ---------------------------------------
```

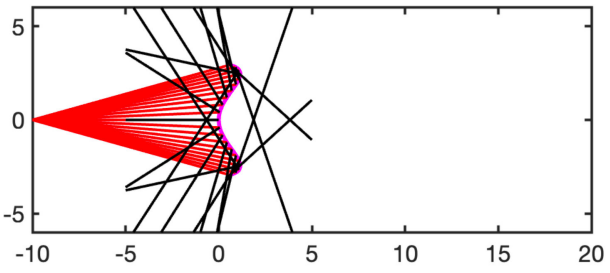


FIGURE 5. Rays reflected by the surface.

It is important to note that the reflected rays are plotted from the points P_k, where the rays strike the curve, to a certain plane located either in front of or behind the curve. The latter case occurs for some curves or physical situations in which the reflected rays propagate towards another point at the curve, as can be observed in Fig. 5 near the endpoints of the curve. At this point, for simplicity these secondary reflections will be neglected, but if required their trajectories can be obtained following the procedure just described. The choice of plane depends on the value of the slope: if $\gamma_k < 90°$, a plane located before the curve is chosen; if $\gamma_k > 90°$, a plane located after the curve is chosen.

For the transmitted rays, the slope is given by (see Fig. 1):

$$m_{tk} = \tan(\varphi_k), \tag{15}$$

where

$$\varphi_k = \beta_k - \theta_t. \tag{16}$$

The value of θ_t is easily found using Snell's law, that is,

$$\theta_t = \sin^{-1}\left(\frac{n_1}{n_2}\sin\theta_i\right). \tag{17}$$

Then, the equation of the line with which we will plot the refracted rays is given by:

$$y_{tk} = \tan(\varphi_k)(x_{tk} - x_k) + y_k. \tag{18}$$

The transmitted rays will be plotted from the curve to a plane located after the curve, as shown in the following Matlab code snippet and in Fig. 6.

Observe that in this example, the rays are incident from a medium with a refractive index of $n_1 = 1$ into a medium with a refractive index of $n_2 = 1.2$.

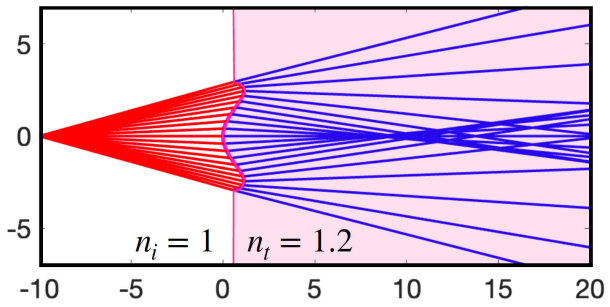


FIGURE 6. Transmitted rays through the surface.

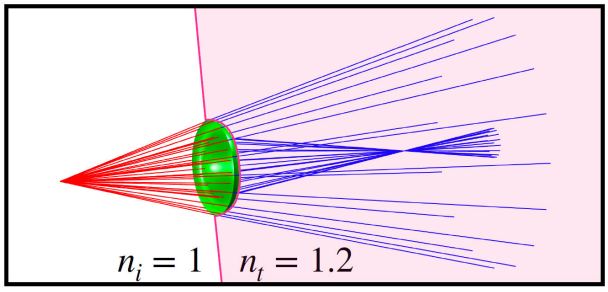


FIGURE 7. Three-dimensional ray tracing through the surface.

```
% ---- Transmitted or refracted rays ---
% ------ Define refractive indexes -----
n1 = 1;
n2 = 1.2;
hold on;  %Keep curve to plot Normals
for k = 1:n
    Mk = (b*cos(t(k)))/(a*(2*cos(t(k))
        ^2*sin(t(k))-sin(t(k))^3));
    betak = atan(-1/Mk);
    mk = (y(k)-y0)/(x(k)-x0);
    alphak = atan(mk);
    gammak = betak-alphak;
    Thetat = asin((n1/n2).*sin(gammak));
    xtk = linspace(x(k),20,m);
    ytk = tan(gammak-Thetat)*(xtk-x(k))+
       y(k);
    plot(xtk,ytk,'b','LineWidth',1);
end
hold off;  % Release plot
axis equal
axis([-10 20 -6 6])
set(gca,'fontsize',18,'LineWidth',2)
set(gcf,'Color',[1,1,1])
% ------------------------------------
```

Only for visualization purposes, in Fig. 7 we show the ray tracing on the three-dimensional surface generated using Eq. (6). This ray tracing was performed by applying a three-dimensional rotational matrix to rotate the entire system from Fig. 6.

6. Other examples

We have developed above a simple code capable of calculating the reflected and refracted rays from an arbitrary curve given by the parametric functions Eq. (6). However, the code is general and it can work for any other curve expressed in its parametric form. The only required modification is defining the parametric functions for the coordinates of the curve in question. For example, we can calculate the reflected rays by a circular, elliptical, parabolic, or hyperbolic mirror, as shown in Figs. 8-12, respectively.

We can also calculate the transmitted rays through a circular, elliptical, parabolic, or hyperbolic surface, as shown in Figs. 13-17, respectively.

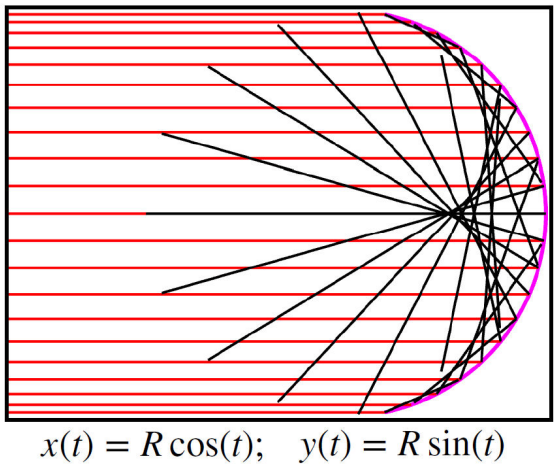


FIGURE 8. Rays reflected by a circular mirror.

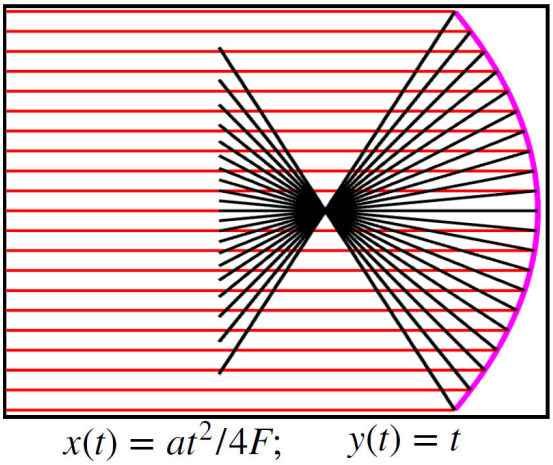


FIGURE 11. Rays reflected by a parabolic mirror. F is the focus of the parabola.

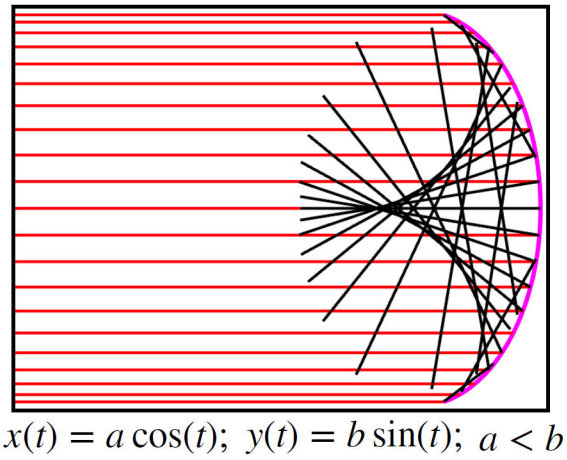


FIGURE 9. Rays reflected by an elliptical mirror with $a < b$ (prolate surface).

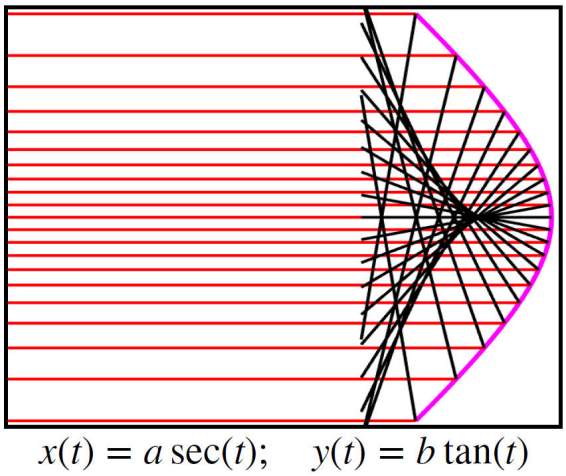


FIGURE 12. Rays reflected by a parabolic mirror. F is the focus of the parabola.

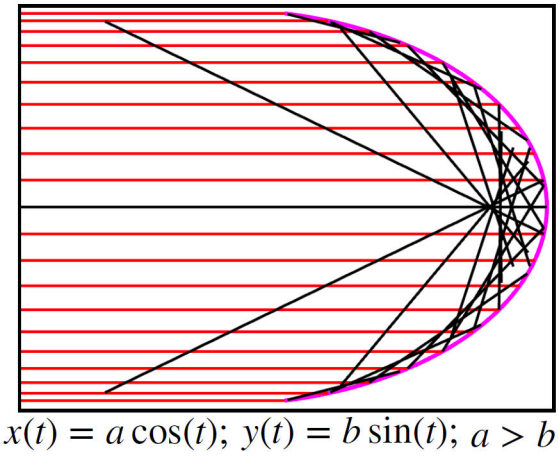


FIGURE 10. Rays reflected by an elliptical mirror with $a > b$ (oblate surface).

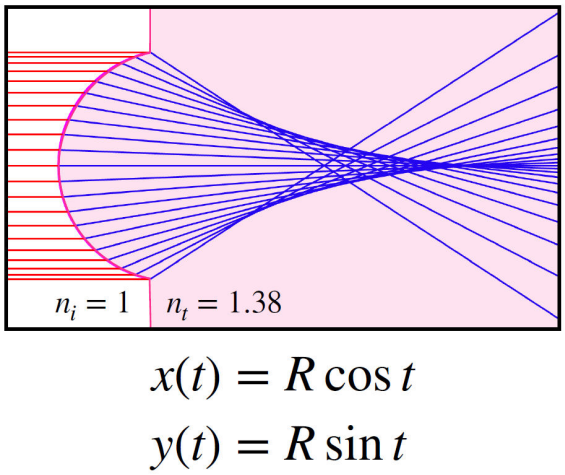
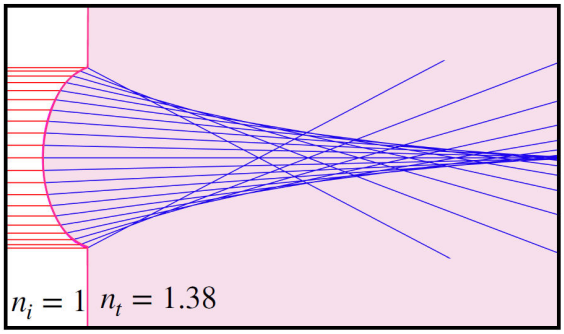
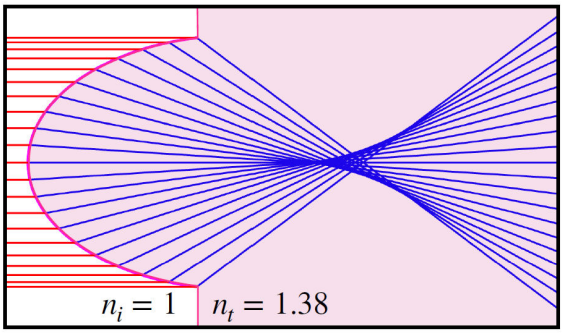


FIGURE 13. Rays reflected by a hyperbolic mirror.



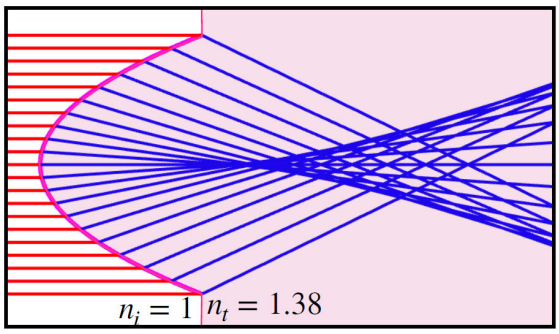
$$x(t) = a\cos t \qquad y(t) = b\sin t \qquad a < b$$

FIGURE 14. Rays refracted by an elliptical surface with $a < b$ (prolate surface).



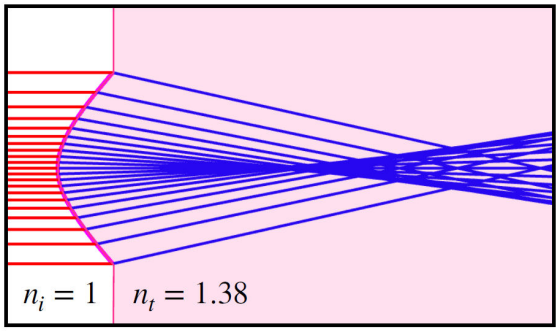
$$x(t) = a\cos t \qquad y(t) = b\sin t \qquad a > b$$

FIGURE 15. Rays refracted by an elliptical surface with $a > b$ (oblate surface).



$$x(t) = \frac{at^2}{4F} \qquad y(t) = t$$

FIGURE 16. Rays refracted by a parabolic surface. F is the focus of the parabola.



$$x(t) = a\sec t \qquad y(t) = b\tan t$$

FIGURE 17. Rays refracted by a hyperbolic surface.

7. Total Internal Reflection

So far, all the examples we have performed meet the condition $n_i > n_t$. This is because the transmitted angle, θ_t, given by Eq. (17), always results in a real value under these conditions. However, if we consider the case where $n_t > n_i$, there will be an angle of incidence, θ_i, beyond which θ_t becomes imaginary [15]. This angle is known as the critical angle and is calculated using the following formula:

$$\theta_c = \sin^{-1}\left(\frac{n_t}{n_i}\right). \tag{19}$$

Physically, this equation tells us that when a ray passes from a medium with a higher refractive index (denser medium) to a medium with a lower refractive index (less dense medium), the following phenomena occur:

1. If the angle of incidence is less than the critical angle, the ray refracts out of the denser medium.

2. If the angle of incidence is equal to the critical angle, the refracted ray travels along the boundary.

3. If the angle of incidence is greater than the critical angle, the light ray is completely reflected back into the denser medium. This phenomenon is known as total internal reflection.

For example, the critical angle, θ_c, for the water-air interface (where $n_{\text{water}} = 1.33$ and $n_{\text{air}} = 1$) can be calculated as follows:

$$\theta_c = \sin^{-1}\left(\frac{1}{1.33}\right) \approx 48.6^\circ. \tag{20}$$

Thus, any ray hitting the water-air boundary at an angle greater than 48.6° will undergo total internal reflection, as shown in Fig. 18.

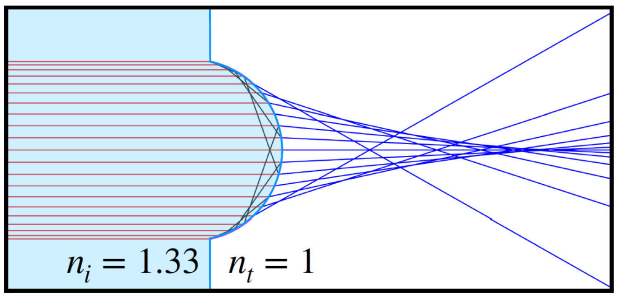


FIGURE 18. Representation of total internal reflection through a circular surface with $n_i > n_t$.

8. Conclusions

In this paper, we have presented a general method for implementing Snell's law and the law of reflection on a chosen curve, using only geometry and basic mathematics, such as differential calculus. This approach allows for the analysis and prediction of the behavior of light and sound as they encounter curved surfaces, facilitating the understanding of complex optical and acoustic phenomena.

Furthermore, we have shown a series of practical examples that illustrate how to apply these laws to different types of curves. These examples demonstrate the versatility and usefulness of our method in calculating Snell's law and the law of reflection and its applicability to Freeform Optics with rotational symmetry.

We conclude that this approach is a powerful tool for education and research in both optics and acoustics. With the basic ideas and tools used here an interesting challenge would be to extend the problem to a three-dimensional surface using vector calculus.

1. K. Falaggis, et al., Freeform optics: introduction, Opt. Express **30** (2022) 6450, https://doi.org/10.1364/OE.454788
2. E. Hecht, Optics, 4th ed. (AddisonWesley, San Francisco, CA, 2002).
3. D. T. Blackstock, Fundamentals of Physical Acoustics, 4th ed. (John wiley and Sons, New York, NY, 2000).
4. E. Kang and J. Park, Development of sound reflection and refraction experiment equipment, *Phys. Educ.* **57** (2022) 065015, https://doi.org/10.1088/1361-6552/ac8a84
5. M. Born and E. Wolf, Principles of optics: electromagnetic theory of propagation, interference and diffraction of light (Elsevier, 2013).
6. D. Malacara Hernández, Óptica básica, 3rd ed. (Fondo de cultura económica, 2015).
7. A. Cornejo Rodríguez and G. Urcid Serrano, Reporte Técnico: Óptica geométrica resumen de conceptos y fórmulas Parte I, INAOE (2005)
8. H. Kuttruff, Room acoustics, 5th ed. (CRC Press, UK, 2009).
9. A. Krokstad, S. Strom, and S. Sørsdal, Calculating the acoustical room response by the use of a ray tracing technique, *J. Sound Vib.* **8** (1968) 118, https://doi.org/10.1016/0022-460X(68)90198-3.
10. MathWorks, MATLAB 24.2.0.2637905 (R2024b) (The MathWorks Inc., Natick, Massachusetts, United States, 2024), URL https://www.mathworks.com.
11. J. W. Eaton *et al.*, GNU Octave version 9.2.0 manual: a high-level interactive language for numerical computations (2024), https://www.gnu.org/software/octave/doc/v9.2.0/.
12. J. Rogel-Salazar, Essential MATLAB and Octave (CRC Pres, Boca Raton, 2014).
13. J. Stewart *et al.*, Precalculus: Mathematics for calculus, 8th ed. (Cengage Learning, Boston, MA, 2024).
14. S. Lang, A first course in calculus, 5th ed. (Springer-Verlag, New York, 1986).
15. D. L. Lee, Electromagnetic principles of integrated optics (John wiley and Sons, USA, 1986).