Refraction by Surfaces

Tim Poston

1 Rays, surfaces and normals

1.1 Surfaces

We look at piecewise straight rays, bending where they meet surfaces.

For simplicity, assume that every surfaces is of the form

$$\begin{aligned} 0 &= f(x,y,z)\,, \qquad \text{where} \\ f(x,y,z) &= a_0 + a_x x + a_y y + a_z z + a_{xx}x^2 + 2a_{xy}xy + a_{yy}y^2 + 2a_{yz}yz + a_{zz}z^2 + 2a_{zx}zx\,. \end{aligned} \tag{1}$$

At a point (X,Y,Z), this has the gradient

$$\nabla f(X,Y,Z) = \begin{bmatrix} \frac{\partial f}{\partial x}(X,Y,Z) \\ \frac{\partial f}{\partial y}(X,Y,Z) \\ \frac{\partial f}{\partial z}(X,Y,Z) \end{bmatrix} \tag{2}$$

$$= \begin{bmatrix} a_x + 2a_{xx}X + 2a_{xy}Y + 2a_{zx}Z \\ a_y + 2a_{xy}X + 2a_{yy}Y + 2a_{yx}Z \\ a_z + 2a_{zx}X + 2a_{yz}Y + 2a_{zz}Z \end{bmatrix} \tag{3}$$

$$\tag{4}$$

which gives us an easily computed normal

$$\mathbf{N}(X,Y,Z) = \frac{\nabla f(X,Y,Z)}{\|\nabla f(X,Y,Z)\|}\,. \tag{5}$$

In particular, look at a single lens, at right angles to the z-axis, with refractive index n_l, surrounded by air (refractive index $n_a = 1$), and centred on $\mathcal{O} = (0,0,0)$ For simple astigmatism, choose a first surface with

$$\begin{aligned} f_1(x,y,z) &= A^1_{xx}x^2 + A^1_{yy}y^2 + A^1_{zz}\,(z-\overline{z}_1)^2 - L_1 \quad \text{with } L_1 > 0 \\ &= \Big((\overline{z}_1)^2 - L_1\Big) + \left(-2\overline{z}^1\right)z + A^1_{xx}x^2 + A^1_{yy}y^2 + A^1_{zz}z^2 \end{aligned} \tag{6}$$

in the form (1). (A non-zero xy term would revolve the astigmatism axis: non-zero xz or yz would prevent the z-axis from being an undeflected ray, except if we are calculating at the origin.) In the case $A^1_{xx} = A^1_{yy} = A^1_{zz} = 1$, this is a sphere of radius $\sqrt{L_1}$ with centre $(0,0,\overline{z}_1)$. We will use $A^1_{xx} \neq A^1_{yy}$, but without loss of generality we can fix $A^1_{zz} = 1$. With $A^1_{xx} \approx A^1_{yy}$, we get an ellipsoidal but near-spherical surface.

Similarly, a second surface is given by

$$f_2(x,y,z)] = \Big((\overline{z}_2)^2 - L_2\Big) + (-2\overline{z}_2)\,z + A^2_{xx}x^2 + A^2_{yy}y^2 + z^2\,. \tag{7}$$

We *could* use one formula for a whole lens, like the sphere $x^2+y^2+z^2 = 1$ or the ellipsoid $x^2+4y^2+400z^2 = 1$, (both have x-radius 1, but the ellipsoid's y-radius is $\frac{1}{2}$ and z-thickness $\frac{1}{10}$), but this would be limiting. With two formulæ, we must choose where they apply: for instance, a biconvex 'spherical' lens uses only

$$\begin{aligned} \text{points with } z &< \overline{z}_1 \text{ for the first surface,} \\ \text{points with } z &> \overline{z}_2 \text{ for the second.} \end{aligned} \tag{8}$$

We could use a flat first surface by setting $(a_0, a_x, a_y, a_z, a_{xx}, a_{xy}, a_{yy}, a_{yz}, a_{zz}, a_{zx}) = (0,0,0,1,0,0,0,0,0,0)$, with a fixed normal $\mathbf{N} = (0,0,1)$, but the separate coding makes more effort, not less. For simplicity, fix $A^1_{xx} = A^2_{xx}$ and $A^1_{yy} = A^2_{yy}$, so that the lens is symmetrical front to back, But keep the first and second surface constants as separately named numbers, for flexibility if/when we want to be less symmetrical later.

Choose constants to make the focal length (in the (x,z)-plane paraxial approximation) three or four times the lens diameter.

1.2 Rays meeting surfaces

We can write any straight ray segment as

$$\Big(x(s), y(s), z(s)\Big) \quad = \quad (x_0, y_0, z_0) + s\mathbf{v} \;=\; (x_0 + sv_x,\, y_0 + sv_y,\, z_0 + sv_z)\,. \tag{9}$$

For convenience later, we use a unit direction vector, requiring $\|\mathbf{v}\|^2 \;=\; v_x^2 + v_y^2 + v_z^2 \;=\; 1$.

Substituting into the formula for surface i gives us a quadratic equation

$$f_i\Big(x(s), y(s), z(s)\Big) \quad = \quad 0\,, \tag{10}$$

with every number but s given by the constants in (1) and (9), so we solve for s by the quadratic formula. If the roots are complex we catch an error: if they are real, we use (8) to choose the relevant one. Put this value into (9), and we get a point (X, Y, Z) on the surface.

1.3 Refraction

At the point (X, Y, Z) just found, we have the normal $\mathbf{N}$ from (5) and the incident vector $\mathbf{v}_i = \big(v_x^i, v_y^i, v_z^i\big)$. If the incident ray is in a region with refractive index n_i, and the refracted ray is in a region with refractive index n_r, Snell's Law gives us the new direction vector $\mathbf{v}_r = \big(v_x^r, v_y^r, v_z^r\big)$. Specifically, $\mathbf{v}_i$ makes an angle

$$\theta_i = \arccos\left(\mathbf{v}_i \bullet \mathbf{N}\right) \tag{11}$$

with the normal, and the vector

$$\mathbf{d}_i \quad = \quad \mathbf{N} - \frac{1}{\cos\theta_1}\mathbf{v}_i \quad = \quad \mathbf{N} - \frac{1}{\mathbf{v}_i \bullet \mathbf{N}}\;\mathbf{v}_i \tag{12}$$

is tangent to the surface, with length $\tan\theta_i$. Correspondingly, the vector

$$\mathbf{d}_r \quad = \quad -\frac{n_i}{n_r}\mathbf{d}_i \tag{13}$$

is tangent in the same $(\mathbf{v}_i, \mathbf{N}\,)$ plane but in the opposite direction, with length

$$\frac{\sin\theta_r}{\sin\theta_i}\tan\theta_i \quad = \quad \frac{\sin\theta_r}{\cos\theta_i}\,. \tag{14}$$

Multiply it by $\cos\theta_i$ and we have a vector of length $\sin\theta_r$, at right angles to the unit vector $\mathbf{N}$. Thus

$$\widetilde{\mathbf{v}} \quad = \quad \mathbf{N} + \cos\theta_i \mathbf{d}_i \tag{15}$$

$$\quad = \quad \mathbf{N} + \frac{n_i\left(\mathbf{v}_i \bullet \mathbf{N}\right)}{n_r}\left(\mathbf{N} - \frac{1}{\mathbf{v}_i \bullet \mathbf{N}}\;\mathbf{v}_i\right) \tag{16}$$

points in the refracted direction, giving a unit vector

$$\mathbf{v}_r \quad = \quad \frac{1}{\|\widetilde{\mathbf{v}}\|}\;\widetilde{\mathbf{v}} \tag{17}$$

for the new ray segment. (Notice that $\|\;\|$ needs a $\sqrt{\;}$, but we do not evaluate any trig functions explicitly.) Setting (x_0, y_0, z_0) in (9) to the current (X, Y, Z), and using this vector, we have the constants for the new segment. Returning to §1.2, we find the next intersection, and so on through the ray's successive encounters.

The final segment of ray meets a **projection screen** $\mathcal{S}$, as a square in the plane $z = Z_{\mathcal{S}}$ for varying distances $Z_{\mathcal{S}}$. The value of s — using the latest $\mathbf{v}$ and segment starting point — is given by

$$s \quad = \quad \frac{Z_{\mathcal{S}} - z_0}{v_z}\,. \tag{18}$$

Substituting this s in (9) gives the other two coordinates.

2 What to draw

2.1 The lens

The lens has a first surface, a second surface, and probably a rim. We choose a mesh of $(x.y)$ points such as

$$(x_{i,j}, y_{i,j}) \quad = \quad \left(\frac{j}{10}\cos\frac{2\pi i}{30}, \frac{j}{10}\cos\frac{2\pi i}{30}\right) \tag{19}$$

for $0 \le j \le 10,\ \ 0 \le i \le 10$ and find the corresponding $z_{i,j}$ values for the first surface, and for the second surface. Add edges across the rim, joining front rim points to back rim points, and polygons between them.

Standard techniques draw this as a wire mesh, as an opaque shaded surface, or as a transparent surface.

For varying distances $Z_{\mathcal{S}}$, under user control by a slider, we draw a **projection screen** $\mathcal{S}$, as a square $-w \le x, y \le w$ in the plane $z = Z_{\mathcal{S}}$.

2.2 The light

In 3D, taking up space with closely-spaced rays (as we do with area, in 2D) would create a visual mess: we need to be more selective. Drawing the caustic surfaces will require some subtle calculations with eigenvalues: but first, let us see by slices of the ray pattern.

Consider a parallel beam, starting at a distant point $P_0 = (X_0, Y_0, Z_0)$ with Z_0 fairly large negative, and pointed at $\mathcal{O}$. The initial direction for all the rays is

$$\mathbf{v}_0 \quad = \quad \frac{1}{\sqrt{X_0^2 + Y_0^2 + Z_0^2}}\left(X_0, Y_0, Z_0\right), \tag{20}$$

but they need different starting points. (Alternatively, we can handle a point source where they all start, but in different directions.) If we want these points in a 'wavefront' plane $\mathcal{P}$ orthogonal to the beam, we must parametrise it nicely.

The plane $\mathcal{P}$ need not contain lines in the x or y directions, but it must contain the cross-product vector

$$\mathbf{V} \quad = \quad \mathbf{v}_0 \times (0, 1, 0), \tag{21}$$

which is automatically horizontal. (It is non-zero, because $Z_0 \neq 0$.) It also contains

$$\mathbf{U} \quad = \quad \mathbf{v}_0 \times \mathbf{V}. \tag{22}$$

A point

$$P_0^{u,v} \quad = \quad P_0 + u\mathbf{U} + v\mathbf{V} \tag{23}$$

thus lies on $\mathcal{P}$ for any (u, v), and every point on $\mathcal{P}$ can be uniquely written that way. Defining a curve like

$$\Big(u(t), v(t)\Big) \quad = \quad \Big(c\cos(t), c\sin(t)\Big) \tag{24}$$

$$\text{or} \quad = \quad (t, c) \tag{25}$$

gives a circle or straight line $P^{\,u(t),v(t)}$ in $\mathcal{P}$, at a distance c from P_0. We will use a set of such curves, equally spaced in c. Graphically, each becomes a sequence of 3D points for t values $t_0^0, t_0^1, \ldots, t_0^k, \ldots, t_0^M$, to be joined by short straight segments: but also, each point $P_0^{u(t_k),v(t_k)}$ is the 3D start point p_0^k of a ray. Find the corresponding points p_1^k where the rays meet the first surface, p_2^k on the second surface, and $p_{\mathcal{S}}^k$on the screen.

We can then draw (in a 3D view) the curves for different c on the initial plane $\mathcal{P}$, and the corresponding curves on the lens surfaces and the screen $\mathcal{S}$. Interesting things happen as the user moves $\mathcal{S}$ through the focussing region!

For a particular c, we can draw polygons like

$$\begin{array}{ccc} p_2^k & \text{———} & p_{\mathcal{S}}^k \\ | & & | \\ p_2^{k+1} & \text{———} & p_{\mathcal{S}}^{k+1} \end{array}, \tag{26}$$

preferably split into triangles. They combine to give a surface representing a sheet of light, creased where it meets the caustic. Drawn for multiple c values together, these surfaces would hide each other, but animating a changing c would be dramatic.

3 User controls

For the parallel-beam version, we fix Z_0 but let the user choose (X_0, Y_0) with a cursor in a 2D panel.

We also need to control the x-strength and y-strength parameters A^i_{xx} and A^i_{yy} (keeping the first and second ones equal, for simplicity), and the distance $Z_{\mathcal{S}}$ from the lens to the screen.

The usual cursor-drag controls to 3D-rotate the whole picture would be good to include.