

Optics

Jim Emery

10/22/2009

Contents

1	Introduction	5
2	Gaussian Optics	5
3	Graphical Analysis	6
4	Gauss' Formula For Refraction At A Single Spherical Surface	6
5	Characterizing a Ray	11
6	The Simple Translation Matrix in a Uniform Medium	12
7	The Simple Refraction Matrix	13
8	The Matrix of a Thick Lens	14
9	The Matrix of the Thick Lens in Air	15
10	The Thin Lens Matrix	15
11	Calculating the Image of an Object.	17
12	Restating the Object-Image Matrix Condition	20
13	A Program to Construct A Composite Lens Matrix	20
14	Experimentally Determining the Lens Matrix Parameters	41

15	The Cardinal Points: Focal points, Principal Points, and Nodal Points	42
16	Principal Planes	43
17	Nodal Points	49
18	The Nodal Slide	51
19	Focal Points	51
20	The Focal Length of a Lens	53
21	The Lens Makers Equation	54
22	Measuring the Radius of a Spherical Cap	55
23	The Power of a Lens: the Diopter	56
24	Determining the Matrix Coefficients From the Cardinal Points	56
25	Determining The Six Cardinal Points From The Matrix Parameters	57
26	Computing The Lens Matrix Parameters From the Nodal and Focal Points	57
27	The Caustic Curve	57
28	Paraxial Thick Lenses	57
29	A Computer Program for Ray Tracing Through Spherical Surfaces	57
30	Tracing Rays Through a Single Surface	58
31	Optical Instruments	58
31.1	Glasses for Vision Correction	58
31.2	The Camera for a Diamond Gemstone Grading Machine.	58
31.3	Camera Orientation: Euler's Angles	58
31.4	Locating a Point From Two Translated Camera Views	61

31.5	Locating Surface Points From Two Camera Views of a Rotated Object	61
31.6	Locating Internal Points From Two Camera Views of a Rotated Object	61
31.7	The Simple Magnifier	62
31.8	An Analysis of the Angular Magnification of a Magnifying Glass.	62
31.9	The Single Lens Microscope.	65
31.10	The Compound Microscope	65
31.11	The Astronomical Refracting Telescope	66
31.12	Terrestrial Telescopes	68
31.13	Galileo's Telescope	68
31.14	Newton's Telescope	68
31.15	Cassegrain's Telescope	68
32	The Eye and Measuring Machines	68
33	The Aspherical Lens Model	69
34	Outline of Geometrical Ray Tracing	71
35	A Ray Tracing Algorithm	72
36	Laser Accuracy	73
37	Accuracy of Existing Machines	73
38	The Location of Deflection Mirrors	74
39	Pitfalls	75
40	Interference	76
41	Vision: Corrective Lenses	76
42	Diffraction	76
43	The Kirchoff Diffraction Formula	77
44	Resolving Power	77

45 Fourier Optics	77
46 Laser Optics	77
47 Acousto-Optics	79
48 Geometrical Ray Tracing	80
49 An Optical Ray Tracing Program	82
50 Locating The Point Of Refraction On A Plane Surface	94
51 The Intensities of Refracted and Reflected Rays	96
52 Geometrical Derivation of the Thin Lens Equation	96
53 The Apparent Position of a Source Point In Medium 1, Viewed From Medium 2	97
54 Snell's Law as an Extremum of Time	98
55 Indices of Refraction	100
56 The Focal length of a Pair of Thin Lenses	101
57 Computing With The Thin Lens Equation	101
58 The Single Lens Microscope	106
59 Polarized Light	107
59.1 Circular Polarization	107
60 Jones Matrices	107
61 Lasers and Einstein's Equation for Stimulated Emission	107
62 To Quickly Determine Whether a Lens is Positive or Nega- tive	107
63 Holography	108

1 Introduction

The first part of this document deals with Geometrical Optics. Geometrical Optics deals with the paths of light rays, which are taken to be straight lines in a homogeneous medium. The index of refraction n is the ratio of the speed of light v in the medium to the speed of light c in a vacuum.

$$n = \frac{c}{v}.$$

Thus n is never less than 1. At the interface between two media of different indices of refraction, light rays bend and are said to be refracted. This occurs at the surface separating two media. This behavior is characterized by Snell's law,

$$n_1 \sin(\theta_1) = n_2 \sin(\theta_2),$$

where n_1 is the index of refraction in the first medium, and n_2 is the index of refraction in the second. The angle θ_1 is the angle that the ray makes with the separating surface normal in the first medium, and θ_2 is the angle that the ray makes with the normal in the second medium. In analyzing spherical lenses and determining the cardinal points such as the focal points, it is sufficient to confine the analysis to the paraxial approximation where rays are nearly parallel to the optic axis. In this case, the sines of angles in Snell's law may be replaced by the angles themselves. The advantage of this is that refraction becomes a linear transformation and the properties of a system of lenses can then be determined by multiplying together lens matrices.

The other branch of Optics is known as Physical Optics, and deals with wave topics such as the interference of light, diffraction, polarized light, particle and photon properties, such as the Compton effect, the Photoelectric effect, relativistic effects and so. Some of these topics occur later in the document.

2 Gaussian Optics

Paraxial optics is also called Gaussian optics because Gauss originated this approximation for rays that are nearly parallel to the optic axis and for

angles that are small. Then a ray angle θ can be used in place of $\sin(\theta)$ and $\tan(\theta)$. This makes the ray transformations linear so that they can be represented by matrices, and the composition of transformations by matrix multiplication. Gauss developed the method in a classical memoir called **Dioptrische Untersuchungen** in 1840. The use of matrices in Gaussian or paraxial optics is a relatively recent development. And may still not be commonly introduced in courses on optics. Gauss is known as the prince of mathematicians and seems to have had his hand in everything.

3 Graphical Analysis

For a thin lens one can graphically locate images by taking a parallel ray from a point on the object and using the fact that the refracted ray passes through the second focus. One assumes that these rays are refracted at the center of the thin lens. Then one can take a ray from the object point that passes through the first focus so that the resulting refracted ray is parallel. Then the image point is located by the intersection of these two refracted rays.

For a thick lens one can use a similar technique, but using as the planes of refraction, two planes known as the principal planes. For the thin lens, these planes are located very near the lens center. For a composite lens system consisting of many lenses one can again locate a pair of principal planes and locate images in a similar way. The principal planes are determined by the composite lens transfer matrix for the system. The paraxial approximation technique starts with Gauss's Refraction Formula.

4 Gauss' Formula For Refraction At A Single Spherical Surface

We suppose that a spherical surface is located at the origin. The optical axis is taken to be the z direction. The z axis is pictured in figures as the horizontal axis. The y axis is vertical, and the x axis goes into the page. The left direction is the negative z direction, and the right is the positive z direction. We shall make the paraxial approximation where rays are nearly parallel to the optic axis, and angles are small. Then we can take sines and

tangents of angles to equal the angles themselves. The z axis will be called the optical axis.

We let R be the signed distance from the sphere surface to the sphere center. That is, if the center of the lens surface lies to the right of the surface, that is the z coordinate of the center is greater than the z coordinate of the intersection of the optic axis with the surface, then R is taken positive. This is called a positive surface. The center of the sphere has in this derivation a z coordinate R . The magnitude of the sphere radius is always $|R|$. For this derivation refer to the Gauss' Refraction Formula figure.

Let n be the index of refraction. Using Snell's law at the spherical surface, and by replacing sines and tangents by angles, we get Gauss' paraxial refraction formula

$$\frac{n_1}{z_1} + \frac{n_2 - n_1}{R} = \frac{n_2}{z_2}.$$

This says that a point on the $z = z_1$ is brought to a focus on the $z = z_2$ plane as determined by this formula. Let us proceed with the derivation. Suppose a ray originates at an object plane $z = z_1$, where z_1 is negative, and arrives at a focus on an image plane $z = z_2$. The spherical surface is located at $z = 0$. The index of refraction on the object side of the surface (negative side) is n_1 . The index of refraction on the other side of the surface (positive side) is n_2 .

We take it as known that the spherical surface refracts each ray from an object point so that each refracted ray passes through a common image point. To locate the image plane, we can look at the optic axis intercept of a refracted ray from a ray originating from an object point, which is located on the optic axis. To derive the formula we use the angles of incidence and the angles the rays make with the optical axis. We can use the law of sines and the paraxial approximation to derive the formula for the case of the object point not being on the optic axis.

We give here a special derivation for the case of an object point on the optic axis. Let the rays lie in the $x = 0$ plane. Let the object point be $(0, 0, z_1)$ and the image point be $(0, 0, z_2)$. We consider the case where $z_1 < 0$ and $z_2 > 0$, and $R > 0$. The derivation of the formula for other cases is similar to this one. Let α be the angle that the ray from the object makes with the optic axis. Let the point where the object ray intersects the spherical surface have coordinates $(0, y_0, z_0)$. Because of the paraxial assumption, z_0 will be nearly zero. Let β be the acute angle between the line from the sphere center to the intersection point. Let γ be the acute angle between the refracted ray and

the optic axis. Let θ_1 and θ_2 be the refraction angles. Snell's law is

$$n_1 \sin(\theta_1) = n_2 \sin(\theta_2).$$

This reduces to

$$n_1 \theta_1 = n_2 \theta_2$$

in the paraxial case. We do not assume anything about the initial ray angle α other than that it is small. Using the fact that for a triangle an external angle equals the sum of the opposite interior angles, we have the following two equations.

$$\theta_1 = \alpha + \beta.$$

$$\beta = \theta_2 + \gamma.$$

The approximate length of the arc from the origin to the intersection point $(0, y_0, z_0)$, which is y_0 is given by three different expressions obtained from approximate triangles in the figure.

$$y_0 = -\alpha z_1.$$

$$y_0 = \beta R.$$

$$y_0 = \gamma z_2.$$

Using these six equations we shall derive Gauss's formula. Substituting the first equation in the third, we get

$$\beta = \frac{n_1}{n_2} \theta_1 + \gamma.$$

Using the second equation and rearranging we get

$$\beta(n_2 - n_1) = n_1 \alpha + n_2 \gamma.$$

Using the last three equations to replace the angles α, β, γ , and dividing out the common factor y_0 , we get

$$\frac{n_2 - n_1}{R} = -\frac{n_1}{z_1} + \frac{n_2}{z_2}.$$

This is Gauss's formula

$$\frac{n_1}{z_1} + \frac{n_2 - n_1}{R} = \frac{n_2}{z_2}.$$

Note that this result is independent of the initial ray angle α , so it shows that all refracted rays pass through a common image point on the optic axis. This special result is sufficient to introduce the method of lens matrices, which will handle the case of object points being off of the optic axis. The method also shows that the points of a plane object are brought to a focus at an image plane $z = z_2$, which is given by Gauss's formula. Without the paraxial assumption, rays do not pass through an exact image point. This causes fuzzy images for lenses of large aperture. This fuzziness is called spherical aberration. A good way to study spherical aberration is with a computer program that does ray tracing, such as the program **lens.ftn**.

For the nonparaxial case, see Joseph Morgan **Introduction to Geometrical and Physical Optics** 1953, chapter 2.

Example 1 Let

$$n_1 = 1, n_2 = 1.4, R = 3, z_1 = -20$$

Then an image at the $z = z_1$ plane is brought to a focus at the plane $z = z_2$, where

$$\begin{aligned} z_2 &= n_2 / \left[\frac{n_1}{z_1} + \frac{n_2 - n_1}{R} \right] \\ &= 16.8 \end{aligned}$$

The first focal point occurs where z_2 goes to infinity. This is where the denominator in the expression for z_2 is zero. It is

$$z_1 = -\frac{n_1 R}{n_2 - n_1} = -\frac{3}{.4} = -7.5$$

The limit as $z_1 \rightarrow -\infty$ is

$$z_2 = R \frac{n_2}{n_2 - n_1}.$$

This is the second focal point. In this example the second focal point is

$$z_2 = R \frac{n_2}{n_2 - n_1} = 3.5R = 10.5.$$

Example 2 Let the object point be to the left of the first focal point:

$$n_1 = 1, n_2 = 1.4, R = 3, z_1 = -5$$

Then an image at the $z = z_1$ plane is brought to a focus at the plane $z = z_2$, where

$$z_2 = n_2 / \left[\frac{n_1}{z_1} + \frac{n_2 - n_1}{R} \right]$$

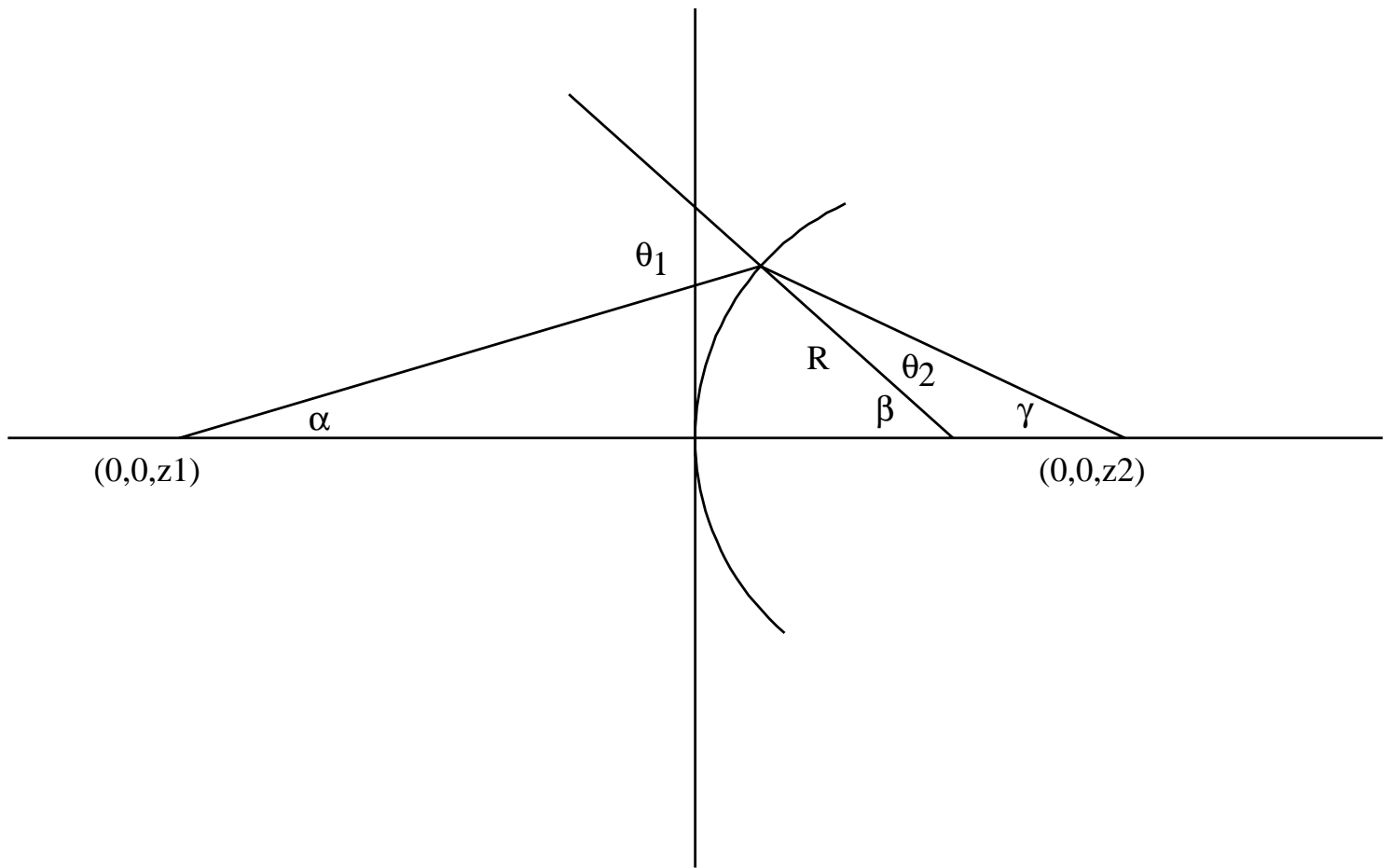


Figure 1: **Gauss's Refraction Formula Derivation.** *A ray originates at the $z = z_1$ plane, is refracted by the spherical lens, and is brought to a focus at the $z = z_2$ plane. We take these points to be on the optic axis, and the sphere radius positive. Because we are assuming paraxial optics the angle α would be much smaller than shown in the figure.*

$$= -21$$

So the image is virtual.

Example 3 Let the lens be concave

$$R = -3, n_1 = 1, n_2 = 1.4, z_1 = -20$$

Then an image at the $z = z_1$ plane is brought to a focus at the plane $z = z_2$, where

$$\begin{aligned} z_2 &= n_2 / \left[\frac{n_1}{z_1} + \frac{n_2 - n_1}{R} \right] \\ &= -7.63 \end{aligned}$$

So the image is virtual.

The first focal point occurs where z_2 goes to infinity and the denominator in the expression for z_2 is zero. Thus it is

$$z_1 = -\frac{n_1 R}{n_2 - n_1} = -\frac{3}{.4} = -7.5$$

The limit as $z_1 \rightarrow -\infty$ is

$$z_2 = R \frac{n_2}{n_2 - n_1} = 3.5R = 10.5,$$

which is the second focal point.

In this formula the object plane is usually thought of so that the plane lies to the left of the refracting surface and z_1 is taken to be negative. When R is negative and $n_2 > n_1$, we have refraction by a convex surface. When z_1 and z_2 differ in sign the image is real and the rays actually pass through the image. When z_1 and z_2 agree in sign the image is virtual. A virtual image can not appear as an image on a screen. A point in a virtual image can not be treated as an actual secondary light source because a bundle of rays does not physically pass through such a point. This formula suffers from a rather screwy sign convention. I have not thought of a good way to make it more mathematically elegant. Perhaps there is no way to do this with an arbitrary orientation of the lens system.

5 Characterizing a Ray

We will shortly introduce lens matrices. First we need to specify a ray. So a ray for our purposes will be a ray in the y-z plane. A ray is specified

by a point (y_1, z_1) in a reference plane $z = z_1$, an angle θ , and an index of refraction n . The ray makes an angle θ with the z axis, that is the optic axis.

We are interested in the transformed ray as it passes through the optic system. A transformation will be from one reference frame to another. A simple ray transformation may be due to either translation in identical media, or to refraction by a boundary surface. In general a transformation will be a composition of several simple transformations. Thus given a ray at some reference plane there will be a transformation taking the ray to a new ray at any other reference plane. The ray is specified by a reference plane $z = z_1$, and a vector

$$\begin{bmatrix} y_1 \\ n_1\theta_1 \end{bmatrix}.$$

These transformations, being linear, have matrix representations. The matrix for a general transformation will be the product of simple transformation matrices. A transformation matrix operates on the vector.

6 The Simple Translation Matrix in a Uniform Medium

Suppose that the ray is not refracted in moving from reference plane P_1 to reference plane P_2 . Suppose the signed distance between P_1 and P_2 is Δz . Then we have

$$z_2 = z_1 + \Delta z$$

where

$$n_1 = n_2.$$

and

$$\theta_1 = \theta_2.$$

Then since θ is small

$$y_2 = y_1 + \theta_1 \Delta z.$$

and

$$n_2\theta_2 = n_1\theta_1.$$

So the transformation matrix A is

$$\begin{bmatrix} 1 & \Delta z/n_1 \\ 0 & 1 \end{bmatrix}.$$

7 The Simple Refraction Matrix

Suppose that there is a refracting surface of radius R between P_1 and P_2 and that the distance between P_1 and P_2 is arbitrarily small.

Using our small angle approximation, and the small distance between P_1 and P_2 we have approximately $y_1 = y_2$, and $n_1\theta_1 = n_2\theta_2$. Recall that Gauss' Refraction Formula is

$$\frac{n_2 - n_1}{R} = -\frac{n_1}{z_1} + \frac{n_2}{z_2},$$

where $z = z_1$ is an arbitrary object plane, and $z = z_2$ is the focused image plane. Here the object plane and the image plane are not necessarily P_1 and P_2 . They are being used to get a form of Gauss' Formula involving angles.

Multiplying Gauss' formula by y_1 gives

$$\frac{n_1 y_1}{z_1} + \frac{(n_2 - n_1) y_1}{R} = \frac{n_2 y_1}{z_2}.$$

According to our lens sign convention the angle of the ray θ_1 will differ in sign from z_1 , as will θ_2 from z_2 . So

$$\theta_1 = -\tan(y_1/z_1) = -y_1/z_1,$$

and

$$\theta_2 = -\tan(y_1/z_2) = -y_1/z_2.$$

Then in terms of the ray angles, Gauss' lens formula becomes

$$-n_1\theta_1 + \frac{(n_2 - n_1)y_1}{R} = -n_2\theta_2.$$

So the refraction matrix A is

$$\begin{bmatrix} 1 & 0 \\ -(n_2 - n_1)/R & 1 \end{bmatrix}.$$

That is

$$\begin{bmatrix} y_2 \\ n_2\theta_2 \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ -(n_2 - n_1)/R & 1 \end{bmatrix} \begin{bmatrix} y_1 \\ n_1\theta_1 \end{bmatrix} =$$

$$\begin{bmatrix} y_1 \\ n_1\theta_1 - (n_2 - n_1)y_1/R \end{bmatrix}$$

Note that if the radius of curvature goes to zero, then we have refraction at a plane interface and

$$A = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}.$$

So that we get Snell's law

$$n_2 \sin(\theta_2) = n_2\theta_2 = n_1\theta_1 = n_1 \sin(\theta_1).$$

In this case of simple refraction we may take the first reference plane P_1 and the second reference plane P_2 to be the same plane.

8 The Matrix of a Thick Lens

We obtain the matrix of a thick lens by composing three transformations, that is by multiplying three matrices together

$$A = A_3 A_2 A_1,$$

where the first reference plane of the product is the first reference plane of A_1 and the second reference plane is the second reference plane of A_3 . Let the lens consist of spherical surfaces of radii R_1 and R_2 with the surfaces separated by a distance t . Let the index of refraction of the medium to the left of the lens where the light enters be n_1 , Let the index of refraction of the lens be n_2 . Let the index of refraction of the medium to the right of the lens where the light exits be n_3 .

Then the system matrix with a reference plane located at each of the first and second surfaces is

$$A_3 A_2 A_1 = \begin{bmatrix} 1 & 0 \\ -(n_3 - n_2)/R_2 & 1 \end{bmatrix} \begin{bmatrix} 1 & t/n \\ 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 \\ -(n_2 - n_1)/R_1 & 1 \end{bmatrix}$$

We can write out this product, but perhaps it is easier to evaluate each matrix and then do the numerical multiplication, rather than substituting numbers in a complicated product matrix formula.

9 The Matrix of the Thick Lens in Air

Let the lens consist of spherical surfaces of radii R_1 and R_2 , with the surfaces separated by a distance t . Let the lens be in air. So for the thick lens matrix of the previous section we have $n_1 = n_3 = 1$. Let us write $n = n_2$. The system matrix for the thick lens becomes

$$\begin{aligned} & \begin{bmatrix} 1 & 0 \\ -(1-n)/R_2 & 1 \end{bmatrix} \begin{bmatrix} 1 & t/n \\ 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 \\ -(n-1)/R_1 & 1 \end{bmatrix} \\ &= \begin{bmatrix} 1 - \frac{t(n-1)}{R_1 n} & \frac{t}{n} \\ -\frac{(n-1)(n(R_2 - R_1) + t(n-1))}{nR_1 R_2} & 1 + \frac{t(n-1)}{nR_2} \end{bmatrix} \end{aligned}$$

The matrix coefficients are

$$\begin{aligned} A &= 1 - \frac{t(n-1)}{R_1 n} \\ B &= \frac{t}{n} \\ C &= -\frac{(n-1)(n(R_2 - R_1) + t(n-1))}{nR_1 R_2} \\ D &= 1 + \frac{t(n-1)}{nR_2} \end{aligned}$$

where the lens matrix is

$$\begin{bmatrix} A & B \\ C & D \end{bmatrix}$$

10 The Thin Lens Matrix

As the thickness t goes to zero, we get the matrix for the thin lens

$$\begin{bmatrix} 1 & 0 \\ -(n-1)(1/R_1 - 1/R_2) & 1 \end{bmatrix}$$

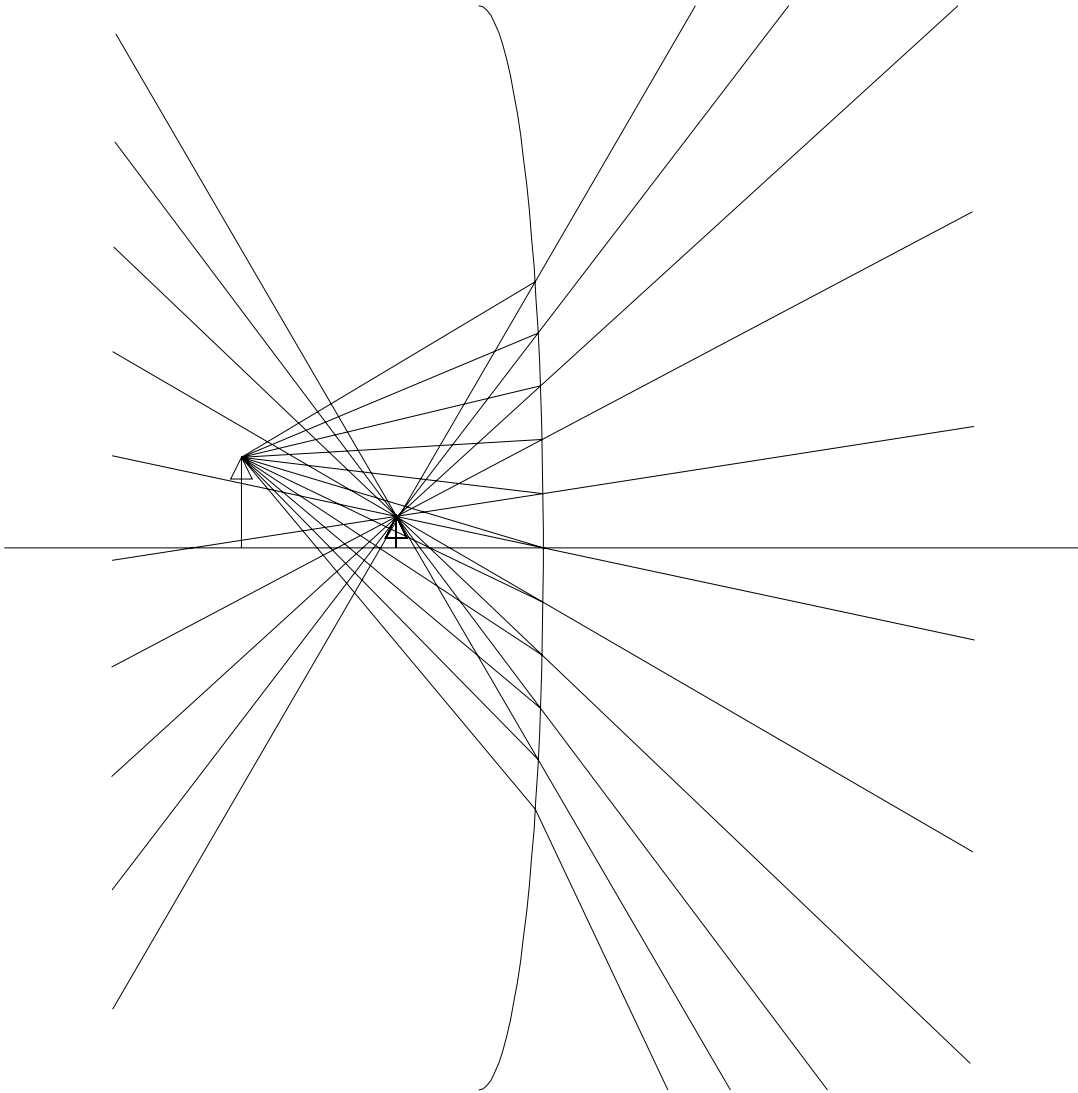


Figure 2: **Refraction By a Concave Spherical Surface:** *Virtual Image,*
 $R = -3, x_{object} = -14.$

11 Calculating the Image of an Object.

Let an optical system have matrix

$$\begin{bmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{bmatrix},$$

with a first reference plane $z = z_1$ and a second reference plane $z = z_2$. We also write the matrix as

$$\begin{bmatrix} A & B \\ C & D \end{bmatrix}.$$

Let an object plane be located at position z_O . Let

$$d_1 = \frac{z_1 - z_O}{n_O}$$

be the optical distance from the object to the lens first reference plane. Recall that this appears as the element of the first row and second column of the simple translation matrix. As an aside note that this is not the optical path length. The optical path length is defined to be the distance that light would travel in a vacuum, in the time that the light takes to travel along a specified nonvacuum path. Thus for a distance δs the optical path length is $n\delta s$. This is confusing notation. The optical path lengths between two wave fronts, (curves of equal phase) are independent of the media through which they pass. That is, any two paths connecting the same wave fronts (surfaces of equal phase) have equal optical path lengths. Consider a distance δs in a media of index of refraction n_1 and wavelength λ_1 . The number of wavelengths in the path is

$$m = \frac{\delta s}{\lambda_1} = \frac{\delta s f}{v}.$$

This corresponds to a distance of travel of light in a vacuum of

$$\delta s' = \lambda m = \frac{c}{f} m = \frac{c}{f} \frac{\delta s f}{v} = \delta s n.$$

We assume that the object is focused on some unknown image plane. Let an image plane be located at position z_I , and recall that the second reference plane of the lens system is located at z_2 . Let

$$d_2 = \frac{z_I - z_2}{n_I}$$

be the optical distance from the second reference plane to the image. Usually d_1 will be positive because the object will be located to the left of the lens system. Also usually n_O will have value 1. However, in dealing with the concepts of principle planes and nodal planes as objects, d_1 will usually be negative, because these planes are usually located inside of the lens system. These planes are somewhat fictional, for example a ray is assumed to travel from outside into the lens to the first principal plane as if it were travelling in air. Although physically this is not possible, it is possible mathematically. Again the object will usually be assumed located to the left of reference plane 1 and d_1 will be positive, being the distance from the object point to the first optical system reference plane, usually the first lens surface. d_2 may be either positive or negative because an image may be to the left or the right of the second lens surface. These optical distances are the actual distances divided by the index of refraction. The matrix of the system with respect to these new reference planes is

$$\begin{aligned} & \begin{bmatrix} 1 & d_2 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{bmatrix} \begin{bmatrix} 1 & d_1 \\ 0 & 1 \end{bmatrix} \\ &= \begin{bmatrix} A_{11} + d_2 A_{21} & (A_{11} d_1 + A_{12}) + d_2 (A_{21} d_1 + A_{22}) \\ A_{21} & A_{21} d_2 + A_{22} \end{bmatrix} = \begin{bmatrix} A'_{11} & A'_{12} \\ A'_{21} & A'_{22} \end{bmatrix} \end{aligned}$$

Let us consider how, given an object, the location and size of the image may be determined. Suppose there is a point of the object at height h . If a ray leaves this point at angle θ , then the image height is

$$h' = A'_{11} h + A'_{12} \theta n_1$$

This must be independent of the ray angle. So we must have

$$A'_{12} = 0 = (A_{11} d_1 + A_{12}) + d_2 (A_{21} d_1 + A_{22}).$$

Solving for d_2 we get the image distance in terms of the object distance

$$d_2 = -\frac{A_{11} d_1 + A_{12}}{A_{21} d_1 + A_{22}}$$

The first focal point is the limiting object point as the image point, d_2 , goes to infinity. It corresponds to a d_1 that makes the denominator zero:

$$d_1 = -\frac{A_{22}}{A_{21}}$$

The first focal point is located at

$$z_{f_1} = z_1 - n_1 d_1 = z_1 + n_1 \frac{A_{22}}{A_{21}}.$$

Normally the object point is to the left of the lens. So d_1 is positive. The second focal point is the image point as d_1 goes to infinity and corresponds to where

$$d_2 = -\frac{A_{11}}{A_{21}}.$$

The second focal point is located at

$$z_{f_2} = z_2 + n_1 d_1 = z_2 - n_2 \frac{A_{11}}{A_{21}}.$$

Hence the second focal point is always located to the right of the second lens surface at a positive distance.

We return to the general image-object case. Since $A'_{12} = 0$ the height of the image is

$$h' = A'_{11} h = (A_{11} + d_2 A_{21}) h.$$

The linear magnification is

$$m_\ell = \frac{h'}{h} = A_{11} + d_2 A_{21}$$

In the case of a thin lens, with $n_1 = 1$ and $n_2 = 1$, we have

$$A_{11} = 1, A_{12} = 0, A_{21} = -1/f, A_{22} = 1,$$

where f is the magnitude of the first and the second focal point. So

$$0 = d_1 + d_2(-d_1/f + 1)$$

which becomes the thin lens equation

$$1/f = 1/d_1 + 1/d_2$$

The magnification is

$$A_1 1 + d_2 A_{21} = 1 - d_2/f$$

12 Restating the Object-Image Matrix Condition

It is worth restating a result of the last section. Suppose we have a lens system with matrix

$$\begin{bmatrix} A & B \\ C & D \end{bmatrix}$$

Let d_1 be the directed distance from an object to the first reference surface. Let d_2 be the directed distance from the second reference surface to the image. An image point must be independent of the direction of any ray leaving the object point. Then for the composite matrix mapping the object plane to the image plane, the top right hand element, namely

$$(Ad_1 + B) + d_2(Cd_1 + D),$$

must be zero.

13 A Program to Construct A Composite Lens Matrix

The constructor program, which is named **lensmat.cpp**, asks for lens surfaces. From the surface definitions, the program creates a matrix for the composite lens system. This lens matrix takes a ray from the first lens surface on the left to the last lens surface on the right. That is, the first lens surface serves as the first reference plane for the matrix and the last lens surface serves as the second reference plane for the lens matrix. Then it asks for an object plane, computes the image plane, and the matrix taking a ray from the object plane to an image plane, and the magnification. Here is an example of running the program, a simple convex lens of radius 3 with zero lens thickness. A radius is positive if the surface lies to the left of the center of the spherical surface. Thus a convex lens might have a radius of 3 for the left surface and a radius of -3 for the right surface.

```
Enter the index of refraction at the left side of the first lens
1
Enter the radius of the next lens surface
3
Enter the index of refraction to the right of the lens surface.
```

```

1.5
Enter the location of the lens
0
refraction matrix =
    1      0
   -0.1667  1
lens system matrix =
    1      0
   -0.1667  1
det = 1
Enter another lens surface? [y n] .
y
Enter the radius of the next lens surface
-3
Enter the index of refraction to the right of the lens surface.
1
Enter the location of the lens
0
refraction matrix =
    1      0
   -0.1667  1
lens system matrix =
    1      0
   -0.3333  1
det = 1
Enter another lens surface? [y n] .
n
Location of left lens surface (reference plane 1), z= 0
Location of right lens surface (reference plane 2), z= 0
Final lens system matrix taking ray on plane 1 to ray on
plane 2:
    1      0
   -0.3333  1
Principal plane 1, z= 0
Principal plane 2, z= 0
Nodal plane 1, z= 0
Nodal plane 2, z= 0
First focal point: z= -3
Focal distance: 3
Second focal point: z= 3
Focal distance: 3
Enter location of an object plane (to left of system in air)
-2
Image plane, z= -6
Object translation matrix =
    1      2
    0      1
Image translation matrix =
    1     -6
    0      1
Object to image system matrix =
    3  7.405e-17
   -0.3333  0.3333
Magnification, z= 3
Enter another object? [y n] .
y
Enter location of an object plane (to left of system in air)

```

```

-4
Image plane, z= 12
Object translation matrix =
    1      4
    0      1
Image translation matrix =
    1      12
    0      1
Object to image system matrix =
    -3 -2.962e-16
    -0.3333 -0.3333
Magnification, z= -3
Enter another object? [y n] .
n

```

Here is a listing of the program called lensmat.cpp. It uses include file lensmat.h.

```

%use program includer
%:include lensmat.cpp
//lensmat.cpp construct a composite lens matrix 4/4/2003
//james d emery
#include <iostream.h>
#include <fstream.h>
#include <stdio.h>
#include <math.h>
#include <string.h>
#include <stdlib.h>
#include "lensmat.h"
#define PI 3.14159265358979
FILE* fh1;
FILE* fh2;
FILE* fh3;
int main(int argc,char** argv){
//int main(){
int i,j,ier;
int count;
char more;
double x,y;
double r;
double xmin,xmax,ymin,ymax;
int n;
double eta_s,eta_1,eta_2;
double z_s,z_1,z_2;
double z_o;
double z_i;
double d2;
double z_p1,z_p2,z_f1,z_f2;
double z_n1,z_n2;
double dz;
double nearpoint,s1,s2,angmag,mag;
dmatrix a(2,2);
dmatrix b(2,2);
dmatrix c(2,2);
dmatrix cc(2,2);
dmatrix d(3,3);

```

```

dmatrix e(3,3);
dmatrix f(10,10);
//dmatrix& mm=(new dmatrix(3,3)); (moved below)
dvector v1(3),v2(3),v3(3);
double v,ang,ain[20];
if(argc<2){
    printf(" lensmat.cpp 4/4/2003, James D Emery\n");
    printf(" This program constructs a lens matrix. The lens system is assumed\n");
    printf(" to have a horizontal optic axis, where light enters from the\n");
    printf(" left, and exits to the right. The user defines lens surfaces,\n");
    printf(" locations, and indices of refraction. The program computes the \n");
    printf(" principal points, nodal points, and focal points. Image planes \n");
    printf(" are computed from entered object planes. For more information\n");
    printf(" see the document: Optics, James Emery (optics.tex) \n");
    printf(" Also see lensdble.ftn, which is a related program. \n");
    printf(" The input data is echoed to the output file with the word \"input:\" prefaced. \n");
    printf(" One can extract this data from a first run and use it for a second.\n");
    printf(" Example:\n");
    printf(" grep input: outputfile > inputfile\n");
    printf(" Then one can edit this file, and use it as input.\n");
    printf(" Example:\n");
    printf(" lensmat outputfile2 < inputfile \n");
    printf(" Usage: lensmat outputfile\n");
    return(0);
}
fh1=fopen(argv[1],"w");
for(i=1;i <= a.gr(); i++){
    for(j=1; j <= a.gc(); j++){
        if(i==j){
            a.p(i,j,1.);
        }
        else{
            a.p(i,j,0.);
        }
    }
}
cout<<"Enter the eye nearpoint (for calculating magnification, 25 centimeters)"<<endl;
fprintf(fh1,"Enter the eye nearpoint, (25 centimeters) \n");
cin >> nearpoint;
fprintf(fh1,"input: %g\n",nearpoint);
fprintf(fh1,"Eye nearpoint , %g \n",nearpoint);
cout << "Enter the index of refraction at the left side of the first lens " << endl;
fprintf(fh1,"Enter the index of refraction at the left side of the first lens. \n");
cin >> eta_s;
fprintf(fh1,"input: %g\n",eta_s);
eta_1=eta_s;
fprintf(fh1, "index of refraction= %15.8g \n",eta_1);
more='y';
count=1;
while(more=='y'){
    cout << "Enter the radius of the lens surface" << endl;
    fprintf(fh1, "Enter the radius of the lens surface \n" );
    cin >> r;
    fprintf(fh1,"input: %g\n",r);
    cout << "Enter the location of the lens surface along the axis" << endl;
    fprintf(fh1, "Enter the location of the lens surface along the axis \n" );
    cin >> z_2;
}

```

```

fprintf(fh1,"input: %g\n",z_2);
cout << "Enter the index of refraction to the right of the lens surface." << endl;
fprintf(fh1, "Enter the index of refraction to the right of the lens surface. \n" );
cin >> eta_2;
fprintf(fh1,"input: %g\n",eta_2);
fprintf(fh1, "lens radius= %15.8g, lens location= %15.8g \n",r,z_2);
if(count==1){
    z_1=z_2;
    z_s=z_1;
}
if( z_2 > z_1){
    //create translation matrix
    dz=z_2-z_1;
    b.p(1,1,1.);
    b.p(1,2,dz/eta_1);
    b.p(2,1,0.);
    b.p(2,2,1.);
    matm(b,a,c);
    matcp(c,a);
    printf("translation matrix = \n");
    fprintf(fh1,"translation matrix = \n");
    printm(a);
    printm(fh1,a);
}
b.p(1,1,1.);
b.p(1,2,0.);
b.p(2,1,(eta_1-eta_2)/r);
b.p(2,2,1.);
printf("refraction matrix = \n");
fprintf(fh1,"refraction matrix = \n");
printm(b);
printm(fh1,b);
matm(b,a,c);
printf("lens system matrix = \n");
fprintf(fh1,"lens system matrix = \n");
printm(c);
printm(fh1,c);
matcp(c,a);
//printf("a = \n");
//printm(a);
printf("det = %g \n",a.g(1,1)*a.g(2,2)-a.g(2,1)*a.g(1,2));
z_1=z_2;
eta_1=eta_2;
cout << "Do you want another lens surface? [y n] .\n";
fprintf(fh1, "Do you want another lens surface? [y n] .\n");
count++;
cin >> more;
fprintf(fh1,"input: %c\n",more);
}
z_1=z_s;
eta_1=eta_s;
printf("Location of left lens surface (reference plane 1) = %g \n",z_1);
printf("Location of right lens surface (reference plane 2) = %g \n",z_2);
printf("Final lens system matrix : \n");
fprintf(fh1,"Location of left lens surface (reference plane 1), z= %g \n",z_1);
fprintf(fh1,"Location of right lens surface (reference plane 2), z= %g \n",z_2);
fprintf(fh1,"Final lens system matrix: \n");

```

```

printm(c);
printm(fh1,c);
y=(1.-a.g(1,1))/a.g(2,1);
x=-(a.g(1,2)+y*a.g(2,2))/(a.g(1,1)+y*a.g(2,1));
z_p1=z_1-x*eta_1;
printf("Principal plane 1, z= %g \n",z_p1);
fprintf(fh1,"Principal plane 1, z= %g \n",z_p1);
z_p2=z_2+y*eta_2;
printf("Principal plane 2, z= %g \n",z_p2);
fprintf(fh1,"Principal plane 2, z= %g \n",z_p2);
x=(1.-a.g(2,2))/a.g(2,1);
y=-(a.g(1,1)*x+a.g(1,2))/(a.g(2,1)*x+a.g(2,2));
z_n1=z_1-x*eta_1;
z_n2=z_2+y*eta_2;
printf("Nodal plane 1, z= %g \n",z_n1);
printf("Nodal plane 2, z= %g \n",z_n2);
fprintf(fh1,"Nodal plane 1, z= %g \n",z_n1);
fprintf(fh1,"Nodal plane 2, z= %g \n",z_n2);
z_f1=z_1+(a.g(2,2)/a.g(2,1))*eta_1;
printf("First focal point: z= %g \n",z_f1);
printf("Focal distance: %g \n",fabs(z_p1-z_f1));

fprintf(fh1,"First focal point: z= %g \n",z_f1);
fprintf(fh1,"First focal distance: %g \n",fabs(z_p1-z_f1));
z_f2=z_2-(a.g(1,1)/a.g(2,1))*eta_2;
printf("Second focal point: z= %g \n",z_f2);
printf("Focal distance: %g \n",fabs(z_p2-z_f2));
fprintf(fh1,"Second focal point: z= %g \n",z_f2);
fprintf(fh1,"Second focal distance: %g \n",fabs(z_p2-z_f2));
more='y';
while(more=='y'){
    cout<<"Enter the location of the object plane "<<endl;
    fprintf(fh1,"Enter the location of the object plane \n");
    cin >> z_o;
    fprintf(fh1,"input: %g\n",z_o);
    printf("Object plane, z= %g \n",z_o);
    fprintf(fh1,"Object plane, z= %g \n",z_o);
    //locate image plane
    dz=(z_s-z_o)/eta_s;
    if(dz < 0.){
        printf(" Error, the object plane must be left of the first lens surface.\n");
    }
    d2=-(a.g(1,1)*dz+a.g(1,2))/(a.g(2,1)*dz+a.g(2,2));
    z_i=z_2+d2*eta_2;
    printf("Image plane, z= %g \n",z_i);
    fprintf(fh1,"Image plane, z= %g \n",z_i);
    //create object translation matrix
    b.p(1,1,1.);
    b.p(1,2,dz);
    b.p(2,1,0.);
    b.p(2,2,1.);
    matm(a,b,c);
    printf("Object translation matrix = \n");
    printm(b);
    fprintf(fh1,"Object translation matrix = \n");
    printm(fh1,b);
    //create translation matrix

```

```

dz=d2;
b.p(1,1,1.);
b.p(1,2,dz);
b.p(2,1,0.);
b.p(2,2,1.);
matm(b,c,cc);
printf("Image translation matrix = \n");
printm(b);
fprintf(fh1,"Image translation matrix = \n");
printm(fh1,b);
printf("Object to image system matrix = \n");
printm(cc);
fprintf(fh1,"Object to image system matrix = \n");
printm(fh1,cc);
mag=cc.g(1,1);
printf("Magnification, z= %g \n",mag);
fprintf(fh1,"Magnification, z= %g \n",mag);
if(z_i < z_2){
    s1=fabs(z_o-z_1);
    if(s1 < nearpoint)s1=nearpoint;
    s2=fabs(z_i-z_2);
    if(s2 < nearpoint) s2=nearpoint;
    cout << "Distance from first lens to object= " << s1 << endl;
    cout << "Distance from eye to image " << s2 << endl;
    fprintf(fh1,"Distance from first lens to object %g \n",s1);
    fprintf(fh1,"Distance from eye to image %g \n",s2);
    angmag=s1*mag/s2;
    printf("Angular Magnification, z= %g \n",angmag);
    fprintf(fh1,"Angular Magnification, z= %g \n",angmag);
}
cout << "Do you want another object plane? [y n] .\n";
fprintf(fh1, "Do you want another object plane? [y n] .\n");
cin >> more;
fprintf(fh1,"input: %c\n",more);
}
printf("Output was written to file %s \n",argv[1]);
fclose(fh1);
return(0);
}
//c+ crsspr vector cross product
void crsspr(dvector &a,dvector &b,dvector &c){
    double v;
    v= a.g(2)*b.g(3)-a.g(3)*b.g(2);
    c.p(1,v);
    v=a.g(3)*b.g(1)-a.g(1)*b.g(3);
    c.p(2,v);
    v=a.g(1)*b.g(2)-a.g(2)*b.g(1);
    c.p(3,v);
}
//c+ dotpr vector dot product
double dotpr(dvector& a,dvector& b){
    double v;
    v=a.g(1)*b.g(1) + a.g(2)*b.g(2) + a.g(3)*b.g(3);
    return(v);
}
//c+ length length of 3d vector

```

```

double length(dvector& a){
    double v;
    v=a.g(1)*a.g(1) + a.g(2)*a.g(2) + a.g(3)*a.g(3);
    return(sqrt(v));
}
//c+ matrot generate rotation matrix from axis and angle
int matrot(dvector &x,double &t,dmatrix &a){
    // Input:
    // x-vector in the direction of the rotation axis
    // t-rotation angle
    // Output:
    // a- 3 by 3 rotation matrix
    // See: Jay Fillmore, A Note On Rotation Matrices,
    // IEEE Computer Graphics and Applications, Feb. 1984.
    dmatrix l(3,3),l2(3,3);
    double lambda,c1,c2;
    int i,j;
    lambda=sqrt(x.g(1)*x.g(1)+x.g(2)*x.g(2)+x.g(3)*x.g(3));
    for(i=1;i<4;i++){
        l.p(i,i, 0.);
    }
    l.p(1,2 , -x.g(3));
    l.p(1,3 , x.g(2));
    l.p(2,3 , -x.g(1));
    l.p(2,1 , -l.g(1,2));
    l.p(3,1 , -l.g(1,3));
    l.p(3,2 , -l.g(2,3));
    matm(l,l,l2);
    c1=sin(t)/lambda;
    c2=(1.-cos(t))/(lambda*lambda);
    for(i=1;i <= 3;i++){
        for(j=1;j<=3;j++){
            if(i == j){
                a.p(i,j, 1.0+c1*1.g(i,j)+c2*12.g(i,j));
            }
            else{
                a.p(i,j, c1*1.g(i,j)+c2*12.g(i,j));
            }
        }
    }
    return(0);
}
//c+ printm print a dmatrix
int printm(dmatrix& a){
    int i,j;
    for(i=1;i <= a.gr(); i++){
        for(j=1; j <= a.gc(); j++){
            printf("%10.4g ",a.g(i,j));
        }
        printf("\n");
    }
    return(0);
}
//c+ printm print a dmatrix
int printm(FILE* f,dmatrix& a){
    int i,j;
    for(i=1;i <= a.gr(); i++){

```

```

    for(j=1; j <= a.gc(); j++){
        fprintf(f,"%10.4g ",a.g(i,j));
    }
    fprintf(f,"\n");
}
return(0);
}
//c+ printv print a dvector
int printv(dvector& v){
    int i,m;
    m=v.gr();
    for(i=1;i <= m; i++){
        printf("%10.4g\n",v.g(i));
    }
    return(0);
}
//c+ angle between two vectors
double angle(dvector &a,dvector &b){
    void crsspr(dvector&,dvector&,dvector&);
    double dotpr(dvector&,dvector&);
    double length(dvector& a);
    dvector c(3);
    double x,y,v;
    crsspr(a,b,c);
    y=length(c);
    x=dotpr(a,b);
    v=atan2(y,x);
    return(v);
}
int readdvector(FILE *f,dvector &a){
    double vin[512];
    int m,i,v,size;
    int readr(FILE *f,double *vin);
    //v=0;
    size=a.gsize();
    m=readr(f,vin);
    if(m > 0){
        if(m <= size){
            v=0;
        }
        else{
            m=size;
            v=1;
        }
        a.pr(m);
        for(i=1;i <= m;i++){
            a.p(i,vin[i-1]);
        }
        return(v);
    }
    else{
        return(1);
    }
}
//c+ matmv matrix multiplication of vector
int matmv(dmatrix &a,dvector &b,dvector &c){
    // c=a*b

```

```

int i,k,ma,na;
ma=a.gr();
na=a.gc();
if((na != b.gr()) || (na != c.gr())){
    return(1);
}
for(i=1;i<=ma;i++){
    c.p(i,0.);
    for(k=1;k<=na;k++){
        c.p(i,c.g(i)+a.g(i,k)*b.g(k));
    }
}
return(0);
}
//c+ matzero fill a matrix with zeros
int matzero(dmatrix &a){
    int i,j,m,n;
    m=a.gr();
    n=a.gc();
    for(i=1;i <= m; i++){
        for(j=1;j <= n; j++){
            a.p(i,j,0.);
        }
    }
    return(0);
}
//c+ matone fill a matrix with ones
int matone(dmatrix &a){
    int i,j,m,n;
    m=a.gr();
    n=a.gc();
    for(i=1;i <= m; i++){
        for(j=1;j <= n; j++){
            a.p(i,j,1.);
        }
    }
    return(0);
}
//c+ matident create an identity matrix
int matident(dmatrix &a,int n){
    int i,j,size;
    size=a.gsize();
    if(n*n <= size){
        a.pr(n);
        a.pc(n);
        for(i=1;i <= n; i++){
            for(j=1;j <= n; j++){
                if(i == j){
                    a.p(i,j,1.);
                }
                else{
                    a.p(i,j,0.);
                }
            }
        }
    }
    return(0);
}

```

```

}
//c+ matrotv2v rotation matrix taking direction vector1 to vector2
int matrotv2v(dvector &v1,dvector &v2,dmatrix &a){
// Input:
// v1-vector one
// v2-vector one
// Output:
// a- 3 by 3 rotation matrix
// calls matrot
dvector axis(3);
double t;
crsspr(v1,v2,axis);
t=angle(v1,v2);
matrot(axis,t,a);
return(0);
}
//c+ matputsub copy matrix a to a submatrix of b
int matputsub(dmatrix &a,int i,int j,dmatrix &b){
// Input:
// a-submatrix
// i,j-a is copied into b starting at row i column j of b
// Output:
// b- matrix to which submatrix is copied
// copy will be done only if the submatrix fits in the matrix
// in that case 0 is returned, otherwise 1 is returned.
int am,an,bm,bn,p,q;
am=a.gr();an=a.gc();bm=b.gr();bn=b.gc();
if((i-1+am > bm) || (j-1+an > bn)){
return(1);
}
else{
for(p=1;p<=am;p++){
for(q=1;q<=an;q++){
b.p(i-1+p,j-1+q,a.g(p,q));
}
}
return(0);
}
}
//c+ matgetsub copy submatrix of a to matrix b
int matgetsub(dmatrix &a,int i,int j,dmatrix &b){
// Input:
// a-matrix containing submatrix
// i,j-a is copied starting at row i column j of a to b
// Output:
// b- matrix to which submatrix of a is copied
// the shape of the submatrix is assumed equal to the shape b,
// in that case 0 is returned, otherwise 1 is returned.
int am,an,bm,bn,p,q;
am=a.gr();an=a.gc();bm=b.gr();bn=b.gc();
if((i-1+ bm > am) || (j-1+ bn > an)){
return(1);
}
else{
for(p=1;p<=am;p++){
for(q=1;q<=an;q++){
b.p(p,q,a.g(i-1+p,j-1+q));
}
}
}
}

```

```

    }
  }
  return(0);
}
}
//c+ matputvec copy vector a to a submatrix of b
int matputvec(dvector &a,int i,int j,dmatrix &b){
  // Input:
  // a -vector
  // i,j-a is copied into b starting at row i column j of b
  // Output:
  // b- matrix to which vector is copied
  // copy will be done only if the vector fits in the matrix
  // in that case 0 is returned, otherwise 1 is returned.
  int am,bm,bn,p;
  am=a.gr();bm=b.gr();bn=b.gc();
  if((i-1+am > bm) || (j > bn)){
    return(1);
  }
  else{
    for(p=1;p<=am;p++){
      b.p(i-1+p,j,a.g(p));
    }
    return(0);
  }
}
//c+ matgetvec copy submatrix of a to vector b
int matgetvec(dmatrix &a,int i,int j,dvector &b){
  // Input:
  // a-matrix containing submatrix
  // i,j-a is copied starting at row i column j of a to b
  // Output:
  // b- vector to which submatrix of a is copied
  // the shape of the submatrix is assumed equal to the shape b,
  // in that case 0 is returned, otherwise 1 is returned.
  int am,an,bm,p;
  am=a.gr();an=a.gc();bm=b.gr();
  if((i-1+ bm > am) || (j > an)){
    return(1);
  }
  else{
    for(p=1;p<=am;p++){
      b.p(p,a.g(i-1+p,j));
    }
    return(0);
  }
}
}
//c+ gaussr solution real linear eqs., matrix inv., determinant (real*8)
int gaussr(dmatrix& a,dmatrix& b,int inv,double eps,int idet,double& det){
  // solves the equation a*c=b for c, where a is an n by n matrix
  // c and b are n row by m column matrices. c is returned as b.
  // algorithm -gaussian elimination with partial pivoting.
  // input:
  // a n by n matrix containing the coefficients of
  // the linear system.
  // b On input, n by m matrix containing the m right sides
  // of the equations. Output:

```

```

//      b contains the solutions. The inverse of a is
//      returned in b when inv=1
// inv  the inverse of a is calculated and returned in b when inv=1.
//      the shape of b (n by n) is computed.
// eps  each equation is normalized so that the
//      coefficients are <= 1 in magnitude.
//      when a pivot is less than eps the matrix is
//      considered singular, and ier is set to 2
//      one may set eps=1.e-12 for double.
//      eps does not effect any calculation.
//      normalization may also prevent exponent overflow.
// idet  determinant computed if idet = 1
//      determinants are products of n numbers.
//      overflow can occur if the elements of the
//      matrix have large exponents.
//      set idet=0 if the determinant is not needed.
// output:
// b     b is both input and output
// det   determinant of a.
// value returned:
// 0     normal return
// 1     error: a matrix is not a square matrix
// 2     error: matrix is singular
//
// warning!! the procedure changes a and b. if they need to be
// saved, copies must be made before calling the subroutine.
// can use copy procedure matcp.
// see also corresponding fortran subroutine gaussr.
// language libraries: mathlib.cpp, mathlib.c, mathlibd.ftn, mathlib.pas
// last revision 10/31/96
int matshape(dmatrix&,int,int);
int cdet;
int m,i,n,j,k,kk,nn;
int jj,l,ni,nj,ki;
double zero,ab,c,am,biggest;
cdet = idet == 1;
zero=0.;
det=1.;
n=a.gr();
if(a.gc() != n){
    return 1;
}
if(inv != 1){
    m=b.gc();
    if(m <= 0){
        return 1;
    }
    if(b.gr() != n){
        return 1;
    }
}
else{
    //if computing the inverse, set b equal to the identity.
    matshape(b,n,n);
    for(i=1;i<=n ;i++){
        for(j=1;j<=n ;j++){
            b.p(i,j, 0. );

```

```

        if(i == j){
            b.p(i,j, 1. );
        }
    }
}
m=n;
}
//c normalize rows by dividing row by largest magnitude element.
for(i=1;i<=n ;i++){
    biggest=a.g(i,1);
    for(j=2;j<=n ;j++){
        ab=a.g(i,j);
        if(fabs(ab) > fabs(biggest)){
            biggest=ab;
        }
    }
    if(biggest == zero){
        det=0.;
        return 2;
    }
    if(cdet){
        det=det*biggest ;
    }
    for(j=1;j<=n ;j++){
        a.p(i,j, a.g(i,j)/biggest );
    }
    for(j=1;j<=m ;j++){
        b.p(i,j, b.g(i,j)/biggest );
    }
}
//start the elimination
j=1;
while( j < n){
    kk=j+1;
    l=j;
    //c find row l with largest pivot
    for(i=kk;i<=n ;i++){
        if(fabs(a.g(i,j)) > fabs(a.g(l,j))){
            l=i ;
        }
    }
    if(fabs(a.g(l,j)) == zero){
        det=0.;
        return 2;
    }
    if(fabs(a.g(l,j)) <= fabs(eps)){
        det=0.;
        return 2;
    }
    if(l != j){
        //c interchange rows l and j
        for(k=1;k<=n ;k++){
            c=a.g(l,k);
            a.p(l,k, a.g(j,k) );
            a.p(j,k, c );
        }
        for(k=1;k<=m ;k++){

```

```

    c=b.g(1,k);
    b.p(1,k, b.g(j,k) );
    b.p(j,k, c );
}
if(cdet){
    det=det*(-1.) ;
}
}
if(cdet){
    det=det*a.g(j,j) ;
}
//c divide row by pivot
c=a.g(j,j);
for(k=j;k<=n ;k++){
    a.p(j,k, a.g(j,k)/c );
}
for(k=1;k<=m ;k++){
    b.p(j,k, b.g(j,k)/c );
}
//add multiple of row j to lower rows
//to eliminate jth coefficients
jj=j+1;
for(i=jj;i<=n ;i++){
    am=a.g(i,j);
    for(k=1;k<=n ;k++){
        a.p(i,k, a.g(i,k)-am*a.g(j,k) );
    }
    for(k=1;k<=m ;k++){
        b.p(i,k, b.g(i,k)-am*b.g(j,k) );
    }
}
j=j+1;
}
am=a.g(n,n);
if(fabs(am) == zero){
    det=0.;
    return 2;
}
if(fabs(am) <= fabs(eps)){
    det=0.;
    return 2;
}
}
if(cdet){
    det=det*am ;
}
//c a is now in triangular form
//c compute nth component of solution
for(k=1;k<=m ;k++){
    b.p(n,k, b.g(n,k)/am );
}
//back substitute to compute n-i component
//i=1,2,3,...
nn=n-1;
for(i=1;i<=nn ;i++){
    ni=n-i;
    for(j=1;j<=m ;j++){
        nj=ni+1;

```

```

    for(ki=nj;ki<=n ;ki++){
        b.p(ni,j, b.g(ni,j)-a.g(ni,ki)*b.g(ki,j) );
    }
}
return 0;
}
//c+ matshape set the rows and columns of a matrix
int matshape(dmatrix &a,int m,int n){
    int size;
    size=a.gsize();
    if(m*n > size){
        return(1);
    }
    else{
        a.pr(m);
        a.pc(n);
        return(0);
    }
}
//c+ readdmatrix read a matrix
int readdmatrix(FILE *f,dmatrix &a){
// reads rows of numbers and attempts
// to redefine the matrix with these numbers
// as rows of the matrix.
// the column size will be equal to
// the length of the first row read provided
// it is less than the total size of the matrix
// if succeeding rows have fewer numbers
// the row will be filled with zeroes
// a row will be put into the matrix only if
// the size of the matrix allows it
// if the row lengths vary, or the number
// of rows exceeds the allocated size of the matrix,
// then an error value of 1 is returned,
// otherwise a value of zero is returned
// that is if there are no input errors a value 0 will be returned
double vin[100];
int m,n,nr,j,size,v;
int readr(FILE *f,double *vin);
size=a.gsize();
m=0;
v=0;
while((nr=readr(f,vin)) > 0){
    if(m == 0){
        n=nr;
        if(n <= size){
            a.pc(n);
        }
        else{
            n=size;
            a.pc(n);
        }
    }
    if(nr != n)v=1;
    if((m+1)*n <= size){
        m++;
    }
}
}

```

```

    if(nr >= n){
        for(j=1;j <= n;j++){
            a.p(m,j,vin[j-1]);
        }
    }
    if(nr < n){
        for(j=1;j <= nr;j++){
            a.p(m,j,vin[j-1]);
        }
        for(j=nr+1;j <= n;j++){
            a.p(m,j,0.0);
        }
    }
}
else{
    v=1;
}
}
if(m > 0){
    a.pr(m);
}
else{
    v=1;
}
return(v);
}
//c+ readr read row of numbers
int readr(FILE *fn,double *a){
    // input:
    // fn file pointer
    // output:
    // a-array of numbers read
    // returned value is:
    // -1, end of file
    // 0, empty line
    // n, n is number of values read
    // Remarks:
    // separate numbers by blanks.
    // to read from keyboard use: n=readr(stdin,a).
    // modified for c++, 10/29/96
    // extern double atof(const char *s);
    char b[200],c[25],d[2];
    int i,l,nr;
    strcpy(c,"");
    if(fgets(b,200,fn)==NULL){
        nr=-1;
    }
    else{
        nr=0;
        // fgets puts newline character into string, so subtract 1
        l=strlen(b)-1;
        d[0]=' ';
        for(i=0;i<l;i++){
            d[0]=b[i];
            if(d[0] != ' ')strncat(c,d,1);
            if ((d[0]==' ') || (i==l-1)){
                if(strlen(c) != 0){

```

```

        nr=nr+1;
        a[nr-1]=atof(c);
        strcpy(c,"");
    }
}
}
return(nr);
}
//c+ matm matrix multiplication
int matm(dmatrix &a,dmatrix &b,dmatrix &c){
    // c=a*b
    // shape of c computed from a and b
    // revision 10/31/96
    int i,j,k,ma,na,nb;
    ma=a.gr();
    na=a.gc();
    if(na != b.gr()){
        return(1);
    }
    nb=b.gc();
    matshape(c,ma,nb);
    for(i=1;i<=ma;i++){
        for(j=1;j<=nb;j++){
            c.p(i,j,0.);
            for(k=1;k<=na;k++){
                c.p(i,j,c.g(i,j)+a.g(i,k)*b.g(k,j));
            }
        }
    }
    return(0);
}
//c+ matsc scalar multiplication of matrix
int matsc(double& s,dmatrix& a){
    int ma,na,i,j;
    double v;
    ma=a.gr();
    na=a.gc();
    for(i=1; i <= ma;i++){
        for(j=1; j <= na; j++){
            v= a.g(i,j);
            a.p(i,j,s*v);
        }
    }
    return(0);
}
//c+ mata matrix addition
int mata(dmatrix &a,dmatrix &b,dmatrix &c){
    //c=a+b
    //computes shape of c
    int ma,na,mb,nb,i,j;
    double v;
    ma=a.gr();
    na=a.gc();
    mb=b.gr();
    nb=b.gc();
    if((ma != mb) || (na != nb))return(1);
}

```

```

matshape(c,ma,na);
for(i=1; i <= ma;i++){
  for(j=1;j <= na; j++){
    v= a.g(i,j) + b.g(i,j);
    c.p(i,j,v);
  }
}
return(0);
}
//c+ matcp matrix copy
int matcp(dmatrix &a,dmatrix &b){
  //b=a
  //computes shape of b
  int ma,na,i,j;
  double v;
  ma=a.gr();
  na=a.gc();
  matshape(b,ma,na);
  for(i=1; i <= ma;i++){
    for(j=1;j <= na; j++){
      v= a.g(i,j);
      b.p(i,j,v);
    }
  }
  return(0);
}
}

```

The needed include file **lensmat.h** is

```

//c+ matrix classes
//double matrix class
class dmatrix{
  int m;
  int n;
  int size;
  double *data;
public:
  dmatrix();
  dmatrix(int rows, int columns);
  ~dmatrix(void);
  double g(int i,int j);
  void p(int i,int j,double v);
  int gr(void){return m;};
  int gc(void){return n;};
  int gsize(void){return size;};
  void pr(int mm){m=mm;};
  void pc(int nn){n=nn;};
};
dmatrix::dmatrix(int rows,int columns){
  size=rows*columns;
  data= new double[size];
  m=rows;
  n=columns;
  // printf(" constructor is creating matrix\n");
}
dmatrix::dmatrix(){

```

```

size=9;
data= new double[9];
m=3;
n=3;
// printf(" constructor is creating matrix\n");
}
dmatrix::~dmatrix(void){
delete data;
// printf("distructor is deleting matrix\n");
}
void dmatrix::p(int i,int j,double v){
//printf(" store: %d n= %d, i= %d, j= %d, v=%g\n",data,n,i,j,v);
data[(i-1)*n + (j-1)]=v;
}
double dmatrix::g(int i,int j){
double v;
v=data[(i-1)*n + (j-1)];
//printf(" get: %d n= %d, i= %d, j= %d, v=%g\n",data,n,i,j,v);
return v;
}
//double vector class
class dvector{
int m;
int size;
double *data;
public:
dvector(){
data= new double[3];
m=3;
size=3;
// printf(" constructor is creating vector\n");
}
dvector(int rows);
~dvector(void);
double g(int i);
void p(int i,double v);
int gr(void){return m;};
int gsize(void){return size;};
void pr(int mm){m=mm;};
};
dvector::dvector(int rows){
data= new double[rows];
m=rows;
size=rows;
// printf(" constructor is creating vector\n");
}
dvector::~dvector(void){
delete data;
// printf("distructor is deleting vector\n");
}
void dvector::p(int i,double v){
//printf(" store: %d n= %d, i= %d, j= %d, v=%g\n",data,n,i,j,v);
data[(i-1)]=v;
}
double dvector::g(int i){
double v;
v=data[(i-1)];
}

```

```

//printf(" get: %d n= %d, i= %d, j= %d, v=%g\n",data,n,i,j,v);
return v;
}
//integer vector class
class ivector{
int m;
int size;
int *data;
public:
ivector(){
data= new int[3];
m=3;
size=3;
// printf(" constructor is creating vector\n");
}
ivector(int rows);
~ivector(void);
int g(int i);
void p(int i,int v);
int gr(void){return m;};
int gsize(void){return size;};
void pr(int mm){m=mm;};
};
ivector::ivector(int rows){
data= new int[rows];
m=rows;
size=rows;
// printf(" constructor is creating vector\n");
}
ivector::~ivector(void){
delete data;
// printf("distructor is deleting vector\n");
}
void ivector::p(int i,int v){
//printf(" store: %d n= %d, i= %d, j= %d, v=%g\n",data,n,i,j,v);
data[(i-1)]=v;
}
int ivector::g(int i){
int v;
v=data[(i-1)];
//printf(" get: %d n= %d, i= %d, j= %d, v=%g\n",data,n,i,j,v);
return v;
}
//c+ function declarations
int readdmatrix(FILE* ,dmatrix&);
int printm(dmatrix&);
int printm(FILE* ,dmatrix&);
int printv(dvector&);
int printv(FILE* ,dvector&);
int mata(dmatrix&,dmatrix&,dmatrix&);
int matm(dmatrix&,dmatrix&,dmatrix&);
int matmv(dmatrix&,dvector&,dvector&);
int matrot(dvector&,double&,dmatrix&);
int matrotv2v(dvector&,dvector&,dmatrix&);
int matident(dmatrix &a,int n);
int matone(dmatrix &a);
int matzero(dmatrix &a);

```

```

int matshape(dmatrix &a,int m,int n);
int matputsub(dmatrix&,int,int,dmatrix&);
int gaussr(dmatrix&,dmatrix&,int,double,int,double&);
int matsc(double&,dmatrix&);
int matcp(dmatrix&,dmatrix&);
void crsspr(dvector&,dvector&,dvector&);
double dotpr(dvector&,dvector&);
double angle(dvector &a,dvector &b);
double length(dvector& a);
int readdvector(FILE* ,dvector&);
int readr(FILE*,double*);

```

14 Experimentally Determining the Lens Matrix Parameters

We can determine the parameters by examining properties of various object image pairs. Let an object be positioned so that the optical distance from it to the left reference plane is x . Let the image be positioned at optical distance y from the right reference plane. Then the matrix of the system with respect to these new reference planes is

$$\begin{aligned}
& \begin{bmatrix} 1 & y \\ 0 & 1 \end{bmatrix} \begin{bmatrix} A & B \\ C & D \end{bmatrix} \begin{bmatrix} 1 & x \\ 0 & 1 \end{bmatrix} \\
& = \begin{bmatrix} A + yC & (Ax + B) + y(Cx + D) \\ C & Cx + D \end{bmatrix}
\end{aligned}$$

For an object distance x_i we measure the image distance y_i . We make n measurements. Let the magnification be the image height divided by the object height. We will use the set of reciprocal magnification values, the i th element of which we shall call α_i . The magnification is

$$\frac{1}{\alpha_i} = y_i C + A.$$

We shall show that by Doing a least squares fit, we can obtain the parameters C and D . Because the object and image planes are the reference planes, the matrix element in the upper left corner of the matrix vanishes, which means that the determinate value, which equals 1, equals the product of the diagonal elements. Hence the magnification equals

$$\frac{1}{\alpha_i} = y_i C + A = \frac{1}{Cx_i + D},$$

so that we have a set of n linear equations in the unknowns C and D ,

$$Cx_i + D = \alpha_i.$$

Using the least squares method, we compute the values C and D .

According to the object-image condition, the upper right matrix value vanishes:

$$Ax_i + B + y_i(Cx_i + D) = 0.$$

That is

$$Ax_i + B = -y_i(Cx_i + D).$$

Let $\beta_i = -y_i(Cx_i + D)$. Then we have a set of n linear equations

$$Ax_i + B = \beta_i,$$

in the unknowns A and B , from which A and B can be determined by least squares.

15 The Cardinal Points: Focal points, Principal Points, and Nodal Points

A lens system is characterized by the six cardinal points on the optic axis, two focal points, two principal points, and two nodal points. The focal points are the images of objects at infinity. Through the principle points are principal planes. The second principal point called h_2 is determined by the intersection of a parallel ray and its refracted ray that passes through the second focal point. The first principal point called h_1 is the intersection of a ray through the first focus f_1 and its refracted ray, which is parallel. The principle points define principle planes. principle points are characterized by unit magnification where h_1 is considered the object plane, and h_2 the image plane. Using the principle planes one can graphically locate an image of an object. At a point on the object we take a parallel ray to the second principal plane and from there take a ray through the second focal point. Then take a ray from the point on the object trough the first focal point to the first principal plane, and then from there a parallel ray. Then the intersection of these two rays gives the image point. The nodal points n_1 and n_2 are such that a ray directed at point n_1 has a refracted ray that is directed from n_2

parallel to the first ray. So the nodal points are characterized by unit angular magnification. These points may also be used to locate images.

For a thin lens, the principle points are taken to coincide. For a lens in air the principle points and the nodal points coincide. Below we shall see how to calculate the cardinal points from the matrix coefficients.

Conversely the cardinal points determine the matrix coefficients.

16 Principal Planes

Suppose a ray parallel to the optics axis enters a lens system. The refracted ray will pass through the second focus. The intersection point of the entering ray and the exiting ray locates a plane that is called the second principal plane, which we shall call H_2 . In the same way an entering ray that passes through the first focus will exit as a parallel ray, and the intersection of these two rays define the first principal plane, which we shall call H_1 . These principal planes may be used to graphically locate images of objects.

Refer to the **Principal Plane Definition** figure, or to figure 2.11 in Warren Smith, 1966.

Unit magnification can be used to locate the principal planes. And the principal planes may be defined by this unit magnification property.

We will now show that principal planes are characterized by unit magnification. So consider an object plane and an image plane located so that there is unit magnification. Then we claim that the object plane is the first principal plane and the image plane is the second principal plane. Let us first locate this pair of planes.

Let an object plane called O and an image plane called I be located so that there is unit magnification. Even though the location of these planes for unit magnification may be inside the lens system, they are treated as though they are outside so that the indices of refraction are those of the medium external to the lens system. We are treating this in an abstract mathematical way. For a ray at height y , and angle 0, at the object plane, the height at the image plane is

$$(d_2C + A)y.$$

So a necessary condition for unit magnification is that

$$d_2C + A = 1.$$

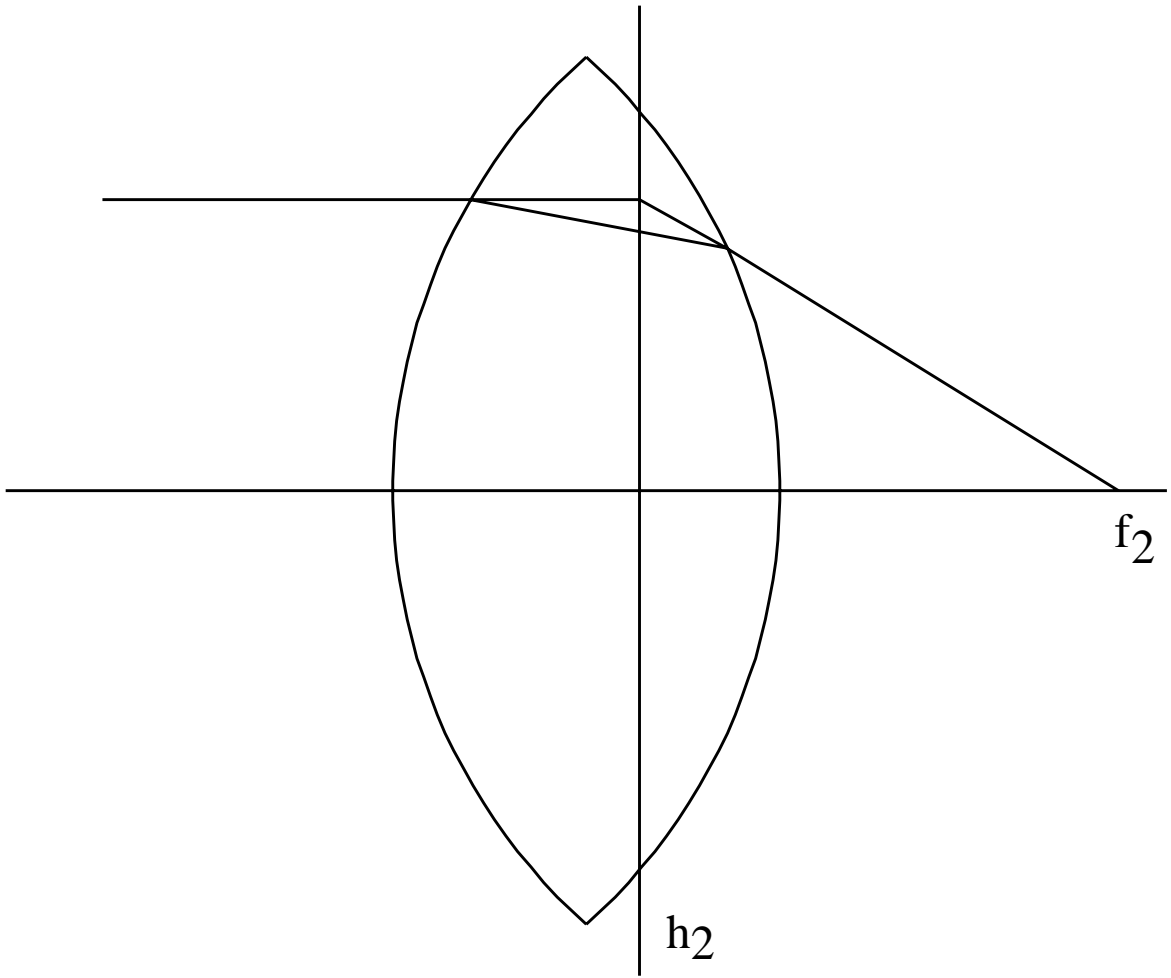


Figure 3: **Principle Plane Definition.** *The second principal plane h_2 is defined by the point of intersection of the parallel entrance ray and the refracted exit ray, which passes through the second focus f_2 . The principle planes located at h_1 and h_2 , when taken as object and image planes, give unit magnification which serves to locate them from the lens matrix parameters.*

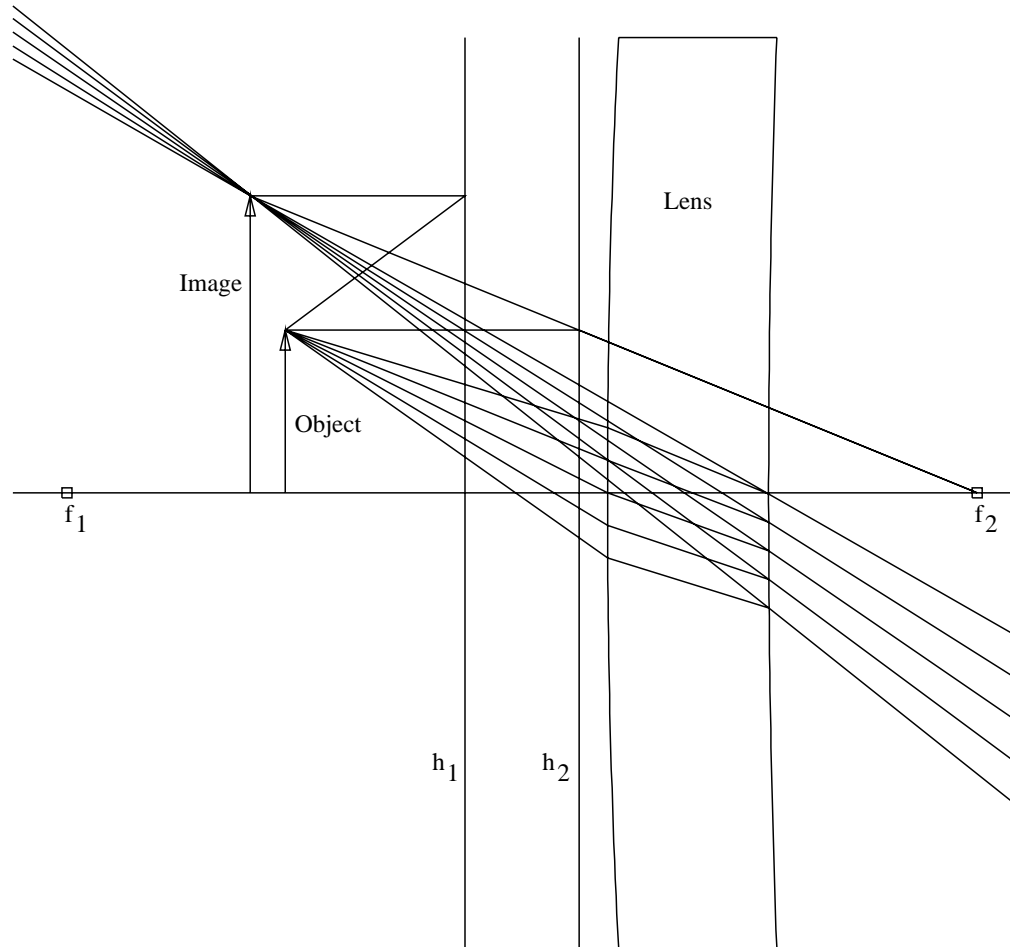


Figure 4: **An Example of Principal Points Outside of the Lens System.** The parameters for this example are as follows: The surface radii of this lens are $r_1 = 3$ and $r_2 = 4$, the thickness is 5. The first surface of the lens is located at $z = 0$. The index of refraction of the lens is $n = 1.4$. The focal point f_1 is located at $z = -16.76$, and the focal point f_2 is located at $z = 11.47$. The principal plane h_1 is located at $z = -4.41$, and the principal plane h_2 is located at $z = -.88$. The object is located at $z = -10$ and has height .5. The image is located at -11.09 and has height 1.83. The focal length of the lens is 12.35. The figure was generated with programs `lensdble.ftn`, `eg2ps.c`, and `addpstxt.bat`.

Then

$$d_2 = (1 - A)/C.$$

The object plane O is located at this distance d_2 from the second reference plane of the lens system. That is, it is located at

$$z = z_2 + d_2 n_I$$

The upper right element of the object-image vanishes, so that

$$Ad_1 + B + d_2(Cd_1 + D) = 0.$$

Rearranging we get

$$(A + d_2C)d_1 = -(B + d_2D).$$

Then

$$\begin{aligned} d_1 &= -\frac{B + d_2D}{A + d_2C} \\ &= -(B + d_2D) \\ &= -\frac{BC + (1 - A)D}{C}. \\ &= -\frac{BC - AD + D}{C}. \\ &= \frac{1 - D}{C}. \end{aligned}$$

We have used the fact that the determinant of the lens matrix $AD - BC$ equals 1.

The first reference plane is at directed distance $d_1 n_O$ from the object plane. Therefore the object plane O is located at a distance $-d_1 n_O$ from the first reference plane. That is, O is located at

$$z = z_1 - d_1 n_O.$$

Let us now establish that the image plane, I , calculated above, is the second principal plane. Let a point of an object be on the object plane O above, where we have unit magnification. Let the point have height y and let an initial ray be defined at this point with angle $\theta = 0$. It does not matter mathematically whether O is to the left of the first reference plane

of the lens system or to the right. This is a transformation of rays and does not mean necessarily that a light ray originates at the a point on O , travels to the first lens surface, then to the last lens surface, then to a point on the second principle plane. Rather it is just a mathematical mapping of rays according to the lens matrix. So this ray on O is mapped to a ray on the image plane I that passes through a point at height y because of unit magnification. Say this this transfer from O to the first reference surface of the lens system is done by the transfer matrix A_1 . Suppose this initial ray has vector v_1 . Then A_1v_1 is now a parallel ray at the first reference plane of the lens system. Then $A_2A_1v_1$ is a ray passing through the second focus, because it is the refracted ray of an incident parallel ray. Suppose A_2 is the lens system transfer matrix, and A_3 is the transfer matrix taking the ray at the second lens reference plane to the image plane I . This is a simple intersection of the exit ray with the image plane I , $A_3A_2A_1v_1$, but because of unit magnification this intersection point is at height y . Therefore the point on the object plane is the intersection of a parallel line at height y and the exiting refracted ray through the second focus. Therefore by definition I is the second principal plane.

By a similar argument we can establish that the object plane above called O is the first principal plane. So suppose we have a ray at object plane O with a vector v . Suppose the ray passes through a point at distance y from the optic axis, and suppose this ray also passes through the first focal point. Let A_1 be the transfer matrix from O to the first reference point of the lens system. Then A_1v is a ray entering the lens system at the first reference plane and passes through the first focal point. Thus if A_2 is the lens system matrix, A_2A_1v is the refracted exit ray from the system and so is parallel to the optic axis. Let the matrix A_3 take the ray at the second lens reference plane to the image plane I . Thus $A_3A_2A_1v$ is a ray at the image plane I . It has the same angle direction as A_2A_1v because A_3 is a simple ray translation. On the other hand $A_3A_2A_1v$ is a unit magnification matrix, so the point on the image plane I has point y . Thus the parallel exit ray meets object plane O at distance y from the object axis and is the point where our original ray originated. Therefore it is the intersection of a ray through the first focus with the parallel refracted ray. Hence by definition the object plane O is the first principal plane.

The principal planes may be used to locate an image graphically. Let two rays emerge from an object point. Let the first ray be parallel to the optic axis and the second ray pass through the first focus. From the point where

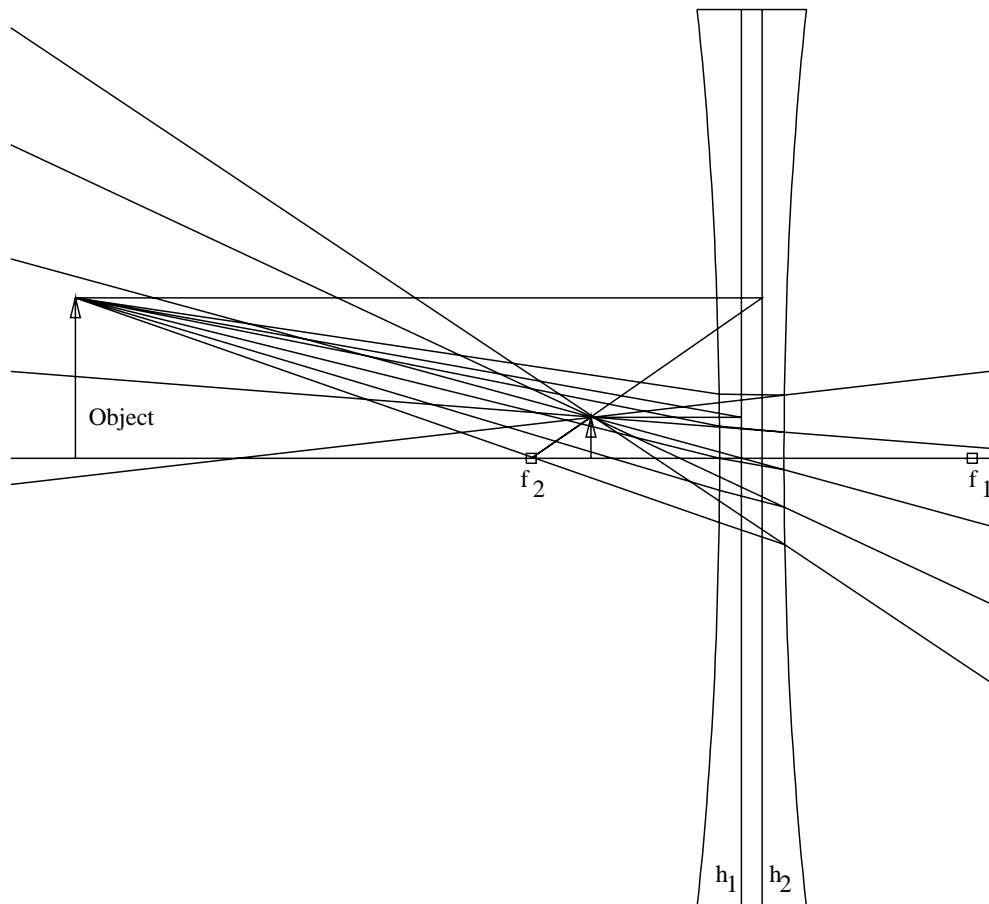


Figure 5: **A diverging lens with a virtual image.** *The surface radii of this lens are $r_1 = -3$ and $r_2 = 3$, the thickness is 1. The index of refraction is 1.4. The principal planes, h_1 and h_2 , are inside the lens. The first focal point f_1 is to the right, and the second focal point f_2 is to the left. The object is at $z = -10$, and has height $1/2$. The image is at $z = -2$ and has height $.129$. The focal length is -3.58 . Five rays are traced through the lens giving a virtual image.*

the first parallel ray meets the second principal plane draw a line through the second focus. From a point where the second object ray passes through the first principal plane draw a parallel line. The intersection of these two drawn lines is the image point (see the figure). The principal planes usually fall inside of the lens system. See figure 2.12 of Warren Smith 1966 for the location of the principle planes for common single lenses. When locating images in this way for a thin lens, we commonly use the center plane of the lens for both principal planes.

17 Nodal Points

As in the case of principal planes, we have a unit angular magnification between a specific object plane and an image plane. These planes are called the nodal planes. The nodal points are the intersections of the nodal planes with the optical axis. Let us find these planes. Using the object-image matrix given above, if the coordinates of a ray leaving the first nodal point are $(0, \theta)$, then the coordinates at the image plane are $(0, (Cd_1 + D)\theta \frac{n_o}{n_i})$. So for unit angular magnification, we must have

$$(Cd_1 + D)\frac{n_o}{n_i} = 1$$

That is the optical distance d_1 is

$$d_1 = \frac{n_i}{n_o}(1 - D)/C.$$

This locates the first nodal point relative to the first reference plane.

Using the object-image condition

$$Ad_1 + B + d_2(Cd_1 + D) = 0,$$

we have

$$d_2 = -\frac{Ad_1 + B}{Cd_1 + D}.$$

If the media on both sides of the lens are the same then,

$$\frac{n_i}{n_o} = 1,$$

so that

$$d_1 = (1 - D)/C.$$

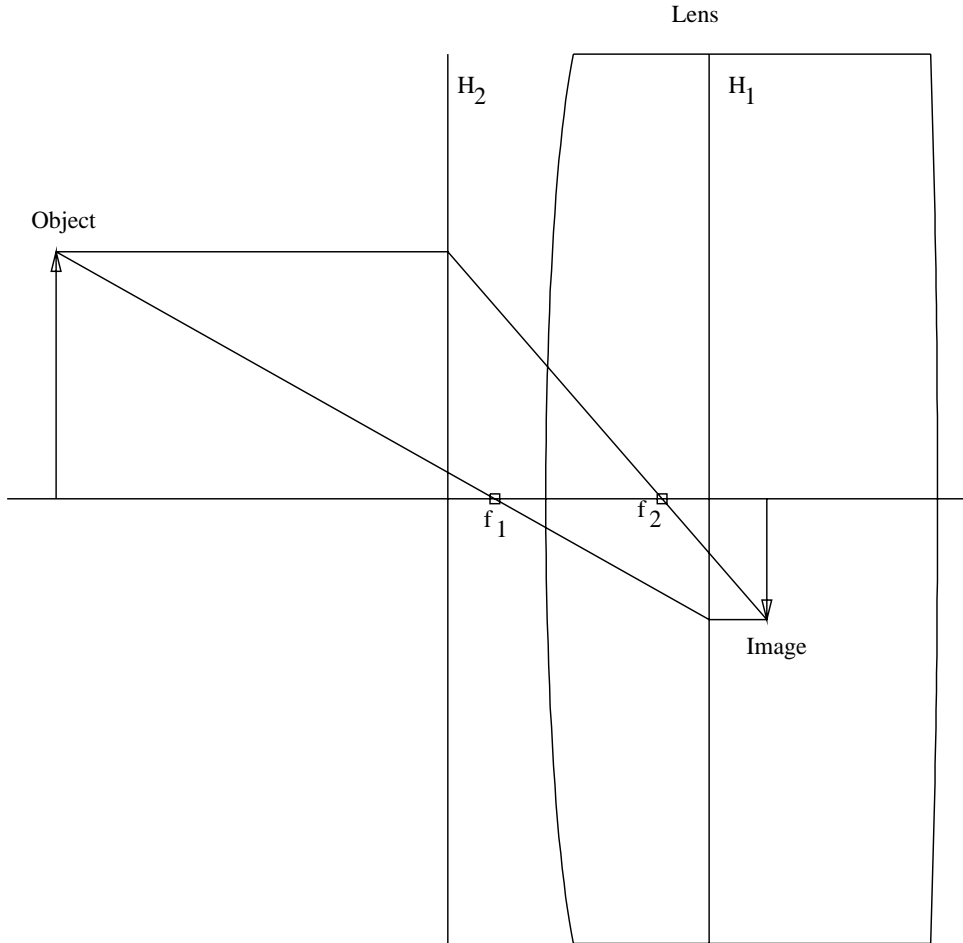


Figure 6: **Reversed Position of Principal Planes** Usually the first principal plane lies to the left of the second principal plane, but as shown here the opposite can occur. The surface radii of this lens are $r_1 = 1$ and $r_2 = -3$, the thickness is 8. The first surface of the lens is located at $z = 0$. The index of refraction of the lens is $n = 1.4$. The focal point f_1 is located at $z = -1.04$, and the focal point f_2 is located at $z = 2.375$. The principal plane h_1 is located at $z = 3.33$, and the principal plane h_2 is located at $z = -2.0$. The object is located at $z = -10$. and has height .5. The image is located at 4.51 and has height -2.44 . The image is virtual. The focal length of the lens is 4.375. The figure was generated with `lensdble.ftn`.

and

$$\begin{aligned}d_2 &= -\frac{A(1-D) + BC}{C} \\ &= \frac{AD - BC - A}{C} \\ &= \frac{1-A}{C}.\end{aligned}$$

Again we have used the fact that the determinant of the lens matrix is 1. This locates the second nodal plane.

Explicitly, the location of the first nodal plane is

$$z = z_1 - d_1 n_O$$

The location of the second nodal plane is

$$z = z_2 + d_2 n_I$$

Usually the planes are in air so that $n_O = n_I = 1$. Notice that when the object space and the image space are the same media, then the principle planes and the nodal planes coincide.

18 The Nodal Slide

The nodal points of a lens system may be found experimentally with a device known as a nodal slide. It consists of a source of collimated light, and a screen. Between the light source and the screen is a mounting that may slide transversally along the axis and may also rotate about a point. The lens system is attached to the mounting. The slide is moved so that the lens focuses the parallel light on the screen. Now the lens system is moved on the slide, and the slide itself translated so that the image remains focused. This is done until rotating the slide does not move the focused image. Then the slide rotation point is at the second nodal point of the lens system.

19 Focal Points

The location of the focal points, which were determined in the section on the object-image section, is given again here. The lens matrix is

$$\begin{bmatrix} A & B \\ C & D \end{bmatrix} = \begin{bmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{bmatrix}$$

The image distance is computed from the object distance d_1 as

$$d_2 = -\frac{A_{11}d_1 + A_{12}}{A_{21}d_1 + A_{22}}$$

The first focal point is the limiting object point as the image point, d_2 , goes to infinity. It corresponds to a value d_1 that makes the denominator zero. Thus the first focal distance is

$$f_1 = -\frac{A_{22}}{A_{21}} = -D/C.$$

This is called the front focal length. It is the distance from the first focal point, the point at which incident parallel rays are focused, to the first lens surface. It will be positive for a converging lens and negative for a diverging lens. Let z_1 and z_2 be the z coordinates of the first and second reference planes for the lens system. The first focal point is located at

$$z_{f_1} = z_1 - n_1 f_1 = z_1 + n_1 \frac{A_{22}}{A_{21}}.$$

The second focal distance is the limiting distance d_2 as d_1 goes to infinity

$$f_2 = -\frac{A_{11}}{A_{21}} = -A/C.$$

This is called the back focal length. It is the distance from the second lens surface to the second focal point. It can be positive or negative. The second focal point is located at

$$z_{f_2} = z_2 + n_1 f_2 = z_2 - n_2 \frac{A_{11}}{A_{21}}.$$

If the object and image are located in air, then the first focal point is located at

$$z_{f_1} = z_1 + \frac{D}{C}.$$

The distance from the first focal point to the first lens surface is

$$-\frac{D}{C}.$$

20 The Focal Length of a Lens

The focal lengths of a lens are considered to be distances between principal planes and focal points. We saw above that the directed optical distance from first focal point to the first reference plane is

$$f_1 = -\frac{D}{C}.$$

And that the directed optical distance from the first principal plane to the first reference plane is

$$d_1 = \frac{1 - D}{C}.$$

Therefore the directed optical distance from the first reference surface to the first principal plane is $-d_1$. It follows that the directed optical distance from the first focal point to the first principal plane is

$$f_1 - d_1 = -\frac{1}{C}.$$

This is called the first optical focal length. Similarly, because

$$f_2 = -\frac{A}{C}$$

and

$$d_2 = \frac{1 - A}{C},$$

the directed optical distance from the second principal plane, to the second focal point is

$$f_2 - d_2 = -\frac{1}{C}.$$

This is the second optical focal length. Notice that the first length is from the focal point to the principal plane, and the second length is from the principal plane to the focal point. The distance

$$f = -\frac{1}{C}$$

is called the paraxial focal length of the lens. To get the focal lengths themselves from the optical focal lengths, we must multiply each optical focal length by the proper index of refraction. Thus the focal lengths are

$$F_1 = -\frac{n_1}{C},$$

and

$$F_2 = -\frac{n_3}{C}.$$

When the index of refraction, n , is the same on both sides of the lens, we have a common focal length

$$F = -\frac{n}{C}.$$

For air n is approximately 1, so for air

$$F = -\frac{1}{C}.$$

For a lens where both lens surfaces are convex, F is positive. In this case, the first focal point is to the left of the lens, and the second focal point is to the right. If both lens surfaces are concave, then F is negative, and the first focal point is to the right of the lens, and the second focal point is to the left. If F is positive, the lens is known as a positive, or converging lens. If F is negative, the lens is called a negative or diverging lens. The focal length is commonly given as the reciprocal

$$1/F.$$

This is convenient for optical calculations. When F is in meters, then the unit of $1/F$ is meter⁻¹. The unit meter⁻¹ is called a diopter. So for example, a -5 diopter lens is a diverging lens of focal length .2 meters, or 20 centimeters.

21 The Lens Makers Equation

For a lens of refractive index n in air the focal length is

$$F = -\frac{1}{C}$$

Because

$$C = -\frac{(n-1)(n(R_2 - R_1) + t(n-1))}{nR_1R_2},$$

we can give the focal length of a lens in terms of the geometric properties of the lens. We get the lens makers equation

$$\frac{1}{F} = (n-1)\left(\frac{1}{R_1} - \frac{1}{R_2} + \frac{(n-1)t}{nR_1R_2}\right).$$

For a thin lens this becomes

$$\frac{1}{F} = (n - 1)\left(\frac{1}{R_1} - \frac{1}{R_2}\right).$$

For example, if we have a thin convex lens where say $R_1 = 3$ and $R_2 = -3$, and $n = 3/2$, then

$$\frac{1}{F} = \frac{1}{2} \frac{2}{R_1}.$$

So the focal length is equal to the radius

$$F = R_1 = 3.$$

Here is a MatLab script for the lens makers equation calculation.

```
% lensmaker.m, The lensmaker equation.
n=1.5
% by convention a lens surface radius is positive if the center lies to the right (lens matrix convention)
r1=12.4*2.54
r2=-12.4*2.54
t=0
t=.6
g=(n-1)*(1/r1 - 1/r2 + (n-1)*t/(r1*r2))
h=(n-1)*(1/r1 - 1/r2 )
% f is the focal length
focallength=1/g
```

22 Measuring the Radius of a Spherical Cap

Suppose we want to physically measure the radius of a lens surface. Let b be half the cord length across the cap. Let a be the depth of the cap, that is the distance from the middle of the cord to the sphere surface. Let the unknown sphere radius be r . Let

$$c = r - a.$$

We have

$$c^2 = r^2 - b^2.$$

So

$$(r - a)^2 = r^2 - b^2.$$

Expanding, we find

$$2ra = a^2 + b^2,$$

or

$$r = \frac{a^2 + b^2}{2a}.$$

In the case of a convex lens with $r_1 = r_2 = r$, diameter d , edge thickness h , and center lens thickness t , we find that

$$b = \frac{d}{2}$$

and

$$a = \frac{t - h}{2}.$$

```
% sphericalradius.m measuring the radius of a sphere.  
d=1+7/16  
a= 1/48  
b=d/2  
sphererad=(a^2+b^2)/(2*a)
```

23 The Power of a Lens: the Diopter

The reciprocal of the focal length is called the power of the lens. It is measured in diopters, which are reciprocal meters (the diopter originated in France and is spelled diop^{tr}e). The power of two thin lenses is approximately the sum of their individual powers.

24 Determining the Matrix Coefficients From the Cardinal Points

Suppose for the case of a lens located in air, we know the principle points and the focal points and positions of lens surfaces. Then clearly from the previous section we know the coefficient C determined by the distance from a principal plane to a focal point. Then measuring distances from the focal points to the lens surfaces we can find A/C and D/C , and so A and D . Finally because the determinant is 1, we can calculate B .

25 Determining The Six Cardinal Points From The Matrix Parameters

The cardinal points are the two principal points, the two nodal points, and the two focal points, which we have just determined. Because they determine the matrix parameters they completely characterize the lens system for paraxial rays.

26 Computing The Lens Matrix Parameters From the Nodal and Focal Points

Suppose the two nodal points and the front and back focal length are given. Then we may compute A, B, C and D . The computation is: Once we have these then the principle points are:

27 The Caustic Curve

See Morgan, **Geometrical and Physical Optics**. The dicaustic curve, pp5-6 and Appendix p417. The curve is the envelope of plane refracted rays. The caustic is the evolute of an ellipse.

28 Paraxial Thick Lenses

The paraxial assumption is that the angles in the lens system are small so that \sin and \tan may be replaced by the angle arguments. Thus spherical aberration is neglected. For thick lenses we do not make the thin lens assumption that the distance between lens surfaces is zero.

29 A Computer Program for Ray Tracing Through Spherical Surfaces

Previously we discussed the programs **lens.ftn** and **lensdble.ftn** which trace rays through a single spherical and through a thick lens made of two spherical surfaces.

30 Tracing Rays Through a Single Surface

Program **lens.ftn** traces a ray through a single spherical surface and makes a plot file. From the figures, which were produced by **lens.ftn**, we see that spherical aberration is very large for moderate ray angles, which makes one wonder that a spherical lens creates a focused image at all. The secret is that although rays from a point come to a focus over a large interval they do so in such a way that only in a small neighborhood is the summed intensity large. Thus the location of the image can be computed from paraxial rays. The program **lensdble.ftn** traces rays through a thick lens defined by two spherical surfaces.

31 Optical Instruments

31.1 Glasses for Vision Correction

Correction for near sightedness.

Correction for far sightedness.

31.2 The Camera for a Diamond Gemstone Grading Machine.

The camera is a wide angle optical instrument that focuses the image onto a plane. A traditional camera uses photographic film located at the image plane. A digital camera uses a CCD (Charge Coupled Device) to measure the intensity of the image.

31.3 Camera Orientation: Euler's Angles

Suppose that a camera is located so that one of the nodal points of the lens system serves as its center. We may introduce a local camera coordinate system which is defined from the inertial coordinate system by the three Euler angles. In this coordinate system we take the origin as one of the lens nodal points, and such that the Z axis is directed along the lens axis from the front of the camera to the back. Reference for Euler angles: Thomson, **Introduction to Space Dynamics**, John-Wiley 1961, p. 33.

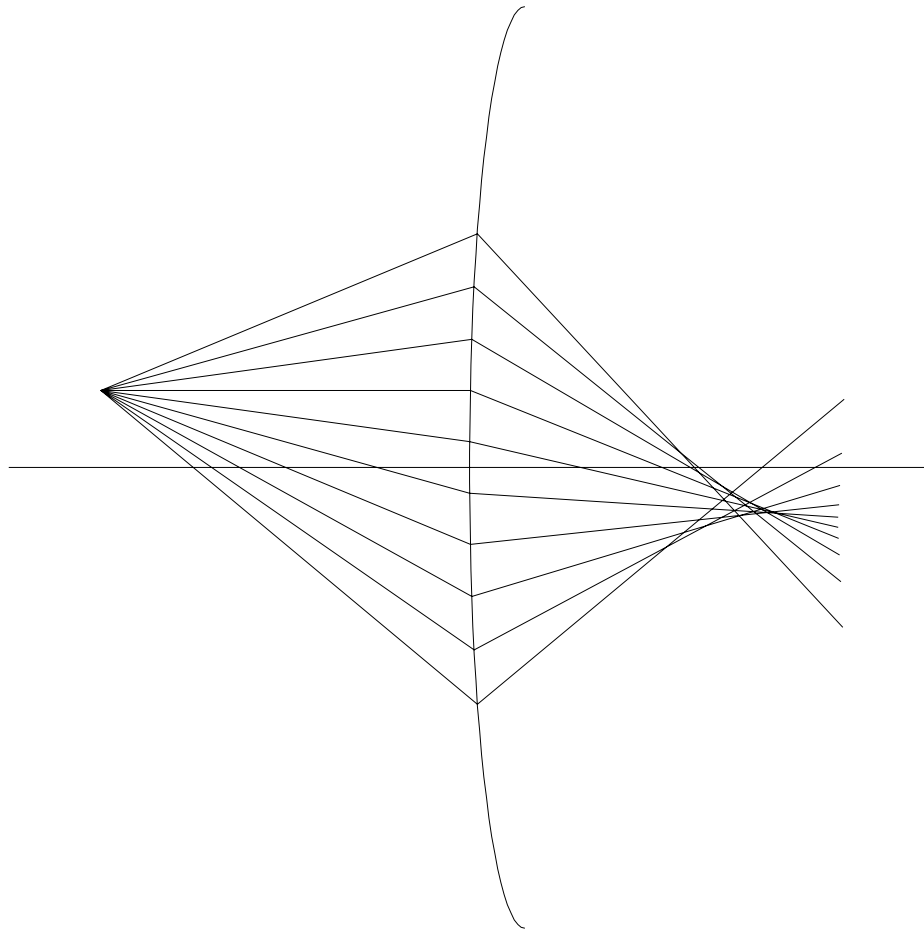


Figure 7: **Refraction By a Spherical Surface.** *This Figure is stretched in the vertical direction.*

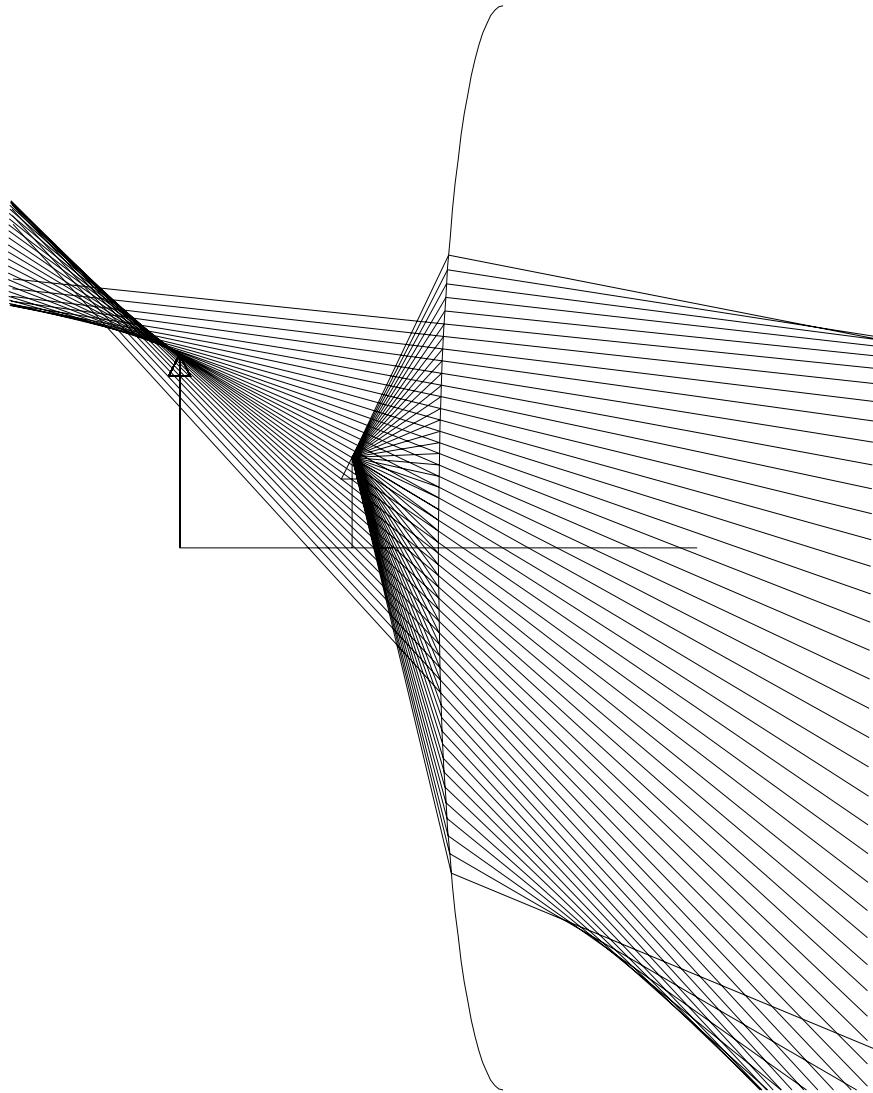


Figure 8: **Refraction By a Spherical Surface:** *Virtual Image*, $R = 3$, $x_{object} = -4$.

31.4 Locating a Point From Two Translated Camera Views

From the second nodal point of the camera consider a ray that meets the focused plane (i.e. the film) at point $(x, y, -d)$. From the distance to the film from the second nodal point, d , we can compute the line in the local camera coordinate system. From this line we can generate the parallel line that originates at the first nodal point and thus is the line containing a given point P . We can transform this line to the global coordinate system, getting say line ℓ_1 . From another camera position, in the same way, we get a second line ℓ_2 , which also contains the point P . Then P is given by

$$P = \ell_1 \cap \ell_2.$$

Because of inevitable error the lines will not meet exactly. We may compute a shortest line segment with endpoints Q_1, Q_2 joining the two lines. We can take P to be the midpoint of this segment

$$P = \frac{Q_1 + Q_2}{2}.$$

31.5 Locating Surface Points From Two Camera Views of a Rotated Object

Let P_1 be a point on an object, which may be rotated about a center C . Let P'_1 be the corresponding point on the image plane. Let n_1 and n_2 be the first and second nodal points of the lens. Let ray Q_1 go from n_1 to P_1 parallel to the ray from P'_1 to nodal point n_2 . Rotate the object by angle θ about the x axis (the vertical axis). The point moves to P_2 giving an image P'_2 . From the image point P'_2 , we compute ray Q_2 from n_1 to P_2 . Then we rotate Q_2 by angle $-\theta$ giving ray Q_3 . Then the original point P_1 is the intersection of the rays Q_1 and Q_3 , which in practice is computed approximately as the center of the shortest line segment joining the rays.

31.6 Locating Internal Points From Two Camera Views of a Rotated Object

Suppose an object point is located inside a transparent polyhedral body, then the previous technique is modified as follows. Each rays Q_1 and Q_2 is traced

to where it enters a facet of the body. The refracted ray is computed and is substituted for the original ray. Then the calculation proceeds as in the previous section.

31.7 The Simple Magnifier

The single lens magnifies when an object is placed between the lens and its first focal point. A virtual image appears to the left of the first focal point. As the object is brought to the focal point, the image goes to infinity. If y is the height of the object and d_0 is the near point of the eye, then the maximum angle observed by the unaided eye is $\arctan(y/d_0)$ or approximately y/d_0 . A parallel ray from the top of the object is refracted to pass through the second focal point. So the angle that the image makes with the second focal point is y/f , where f is the focal length of the lens. The angle the image makes with the eye depends upon the location of the eye. If the eye can be brought closer to the lens than the focal point, then the magnification will be somewhat larger. As the image is made to go to infinity by letting the object approach the first focal point, the difference between the angle the image makes with the eye and the angle the image makes with the focal point goes to zero. Hence the approximate angular magnification of the simple lens is d_0/f . In theory, the magnification can be made arbitrarily large by choosing a lens of very small focal length. But diffraction and aberration limits the size of the lens that can be used to resolve an image. The original Leeuwenhock microscope at the Royal Zoological Society of Antwerp is a single lens consisting of, a glass bead, a few millimeters in diameter, and two clamping brass plates. It has a magnification of about 266. For more about this, see **Single Lens** by Brian Ford, Harper and Row, 1981.

31.8 An Analysis of the Angular Magnification of a Magnifying Glass.

When we view an object with the eye, the focused size of the image at the retina of the eye, is proportional to the angle that the object makes with the center point of the lens of the eye. Hence the magnification of the image we observe is proportional to the angular magnification.

Suppose a thin lens magnifier has focal length f . Suppose an object of height h is located between the lens and the focal point at $d_O = \alpha f$, where

$0 < \alpha < 1$. Suppose the near point of the observers eye is β . Assuming that h is very small the angle of the largest image that can be viewed without the magnifier is

$$\theta_1 = \frac{h}{\beta}.$$

The image distance is given using the thin lens equation

$$\frac{1}{f} = \frac{1}{d_O} + \frac{1}{d_I} = \frac{1}{\alpha f} + \frac{1}{d_I}.$$

Or

$$\begin{aligned} \frac{1}{d_I} &= \frac{1}{f} - \frac{1}{\alpha f} = \frac{\alpha - 1}{\alpha f}. \\ d_I &= \frac{\alpha f}{\alpha - 1}, \end{aligned}$$

which is negative, the image being virtual. Notice that as α goes to zero and the object approaches the lens, the image also approaches the lens.

Now the linear magnification is

$$\frac{-d_I}{d_O}$$

So the angle that an observing eye placed at the lens sees is

$$\frac{h(d_I/d_O)}{d_I} = \frac{h}{d_O}.$$

Therefore the angular magnification is

$$\frac{h/d_O}{h/\beta} = \frac{\beta}{d_O} = \frac{\beta}{\alpha f}.$$

So the angular magnification is larger as the object is brought nearer the lens. However, as the object approaches the lens, the image also approaches the lens. As the object moves closer to the lens, the image can move closer to the eye than the near point of the eye. So it would eventually not be in focus.

As α goes to 1, the angular magnification goes to

$$\frac{\beta}{f}.$$

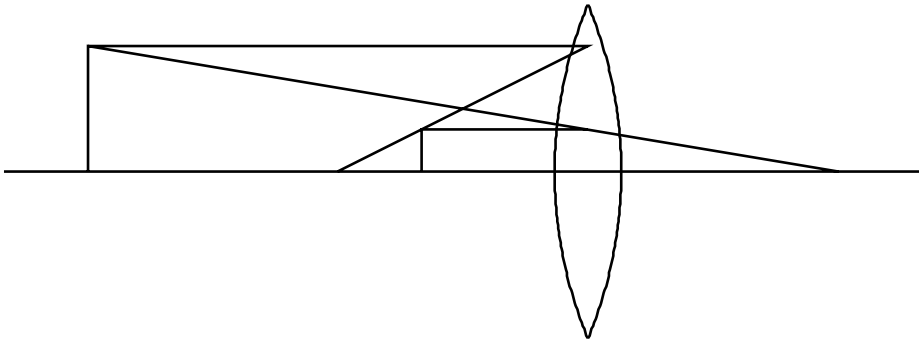


Figure 9: **Angular Magnification.** The object distance is $d_O = 12$ cm, the image distance is $d_I = -36$ cm (the image is virtual). Let the height of the object point be h . The computed focal length is $f = 18$ cm, the magnification is 3, and the height of the image is $3h$. Assuming a small h value, the angle the image makes at the eye located at the lens is $3h/36$. If the near point of the eye is $\alpha = 25$ cm, the largest angle that the eye can focus at is $h/25$. Then the angular magnification is $3(25)/36 = 75/36 = 2.0833$. The expression for angular magnification as the image goes to infinity is $25/f = 25/18 = 1.39$. This results from assuming that the observing eye is at the back focal point. The object is located at $\alpha f = 12$. So $\alpha = 2/3$, and the angular magnification from the expression we derived in the text is $\beta/(\alpha f) = 2.0833$.

The greatest in-focus magnification occurs where the image distance d_I equals the near point of the eye β . So

$$\beta = |d_I| = \frac{\alpha f}{1 - \alpha}.$$

Or

$$\beta(1 - \alpha) = \alpha f$$

$$\beta = \alpha(\beta + f)$$

$$\alpha = \frac{\beta}{\beta + f}$$

In our example given in the **Angular Magnification** figure, $\beta = 25$ and $f = 18$, therefore the best value for α is $\alpha = .5814$.

A very near sighted person has a small near point β and so has a great ability to view tiny objects up close. Notice also that the angular magnification, being proportional to β decreases for a near-sighted person, and therefore such a person derives less benefit from a magnifying glass, then does a far-sighted person.

31.9 The Single Lens Microscope.

Early discoveries in microscopy were done with a single lens microscope. It consisted of a single lens of small diameter, a millimeter or two. Objects to be examined were mounted on a point with wax. The object was brought into focus with screws and other ingenious mechanisms. Van Leeuwenhoek's microscope was made of brass and was brought very close to the eye while looking through the lens toward a light source. Such microscopes were capable of a magnification of up to 200 times. See Brian Ford, "The Single Lens," and the Giordana Collection of single lens microscopes which was exhibited at Linda Hall Library in Kansas City in the summer of 2009. This collection will reside permanently at Musee des Confluences in Lyon France.

31.10 The Compound Microscope

The compound microscope, in its simple form, consists of two lens systems. The objective is a positive lens of short focal length and creates a magnified image of an object. The object is placed just left of the first focal point of the objective. The image is located just to the right of the first focal point of the

eyepiece. We have placed the objective lens of the microscope to the left and the eyepiece to the right. This real image is further magnified by the eyepiece, which operates as a simple magnifier and so produces a virtual image. The matrix for a simple compound microscope is the composition of two thick lens matrices. The magnification is inversely proportional to the product of the two focal lengths. To show this suppose the objective has focal length f_o and the eyepiece has focal length f_e . Let T be the optical tube length. This is the distance from the second focal point of the objective to the position in the tube where the object is brought to a focus. This would be a place just to the right of the first focal point of the eyepiece. Then by similar triangles, the lateral magnification produced by the objective is T/f_o . The angular magnification produced by the eyepiece is d_0/f_e , where d_0 is the near point of the eye, which is usually taken to be 25 cm. So the total magnification of the compound microscope is the product of these magnifications

$$\frac{Td_0}{f_o f_e}.$$

So if the tube length were 16 cm, the eye near point 25 cm, the objective focal length .2 cm and the eyepiece focal length 2 cm, then the magnification would be 1000. See Meyer-Arendt for more details.

31.11 The Astronomical Refracting Telescope

This instrument is optically similar to the compound microscope, but the object is placed at a great distance from the objective. The focal length of the objective is long and that of the eyepiece is short. The second focal point of the objective is located just beyond the first focal point of the eyepiece. For a distant object the image of the objective is located just beyond the second focal point. The eyepiece produces a virtual image of the real objective image. The virtual image is inverted and the magnification is approximately

$$M = \frac{f_o}{f_e},$$

where f_o is the focal point of the objective and f_e is the focal point of the eyepiece. The purpose of an astronomical telescope is to collect parallel rays of light over as wide an area as possible and to bring these rays to a focus. Magnification is not relevant for this purpose.

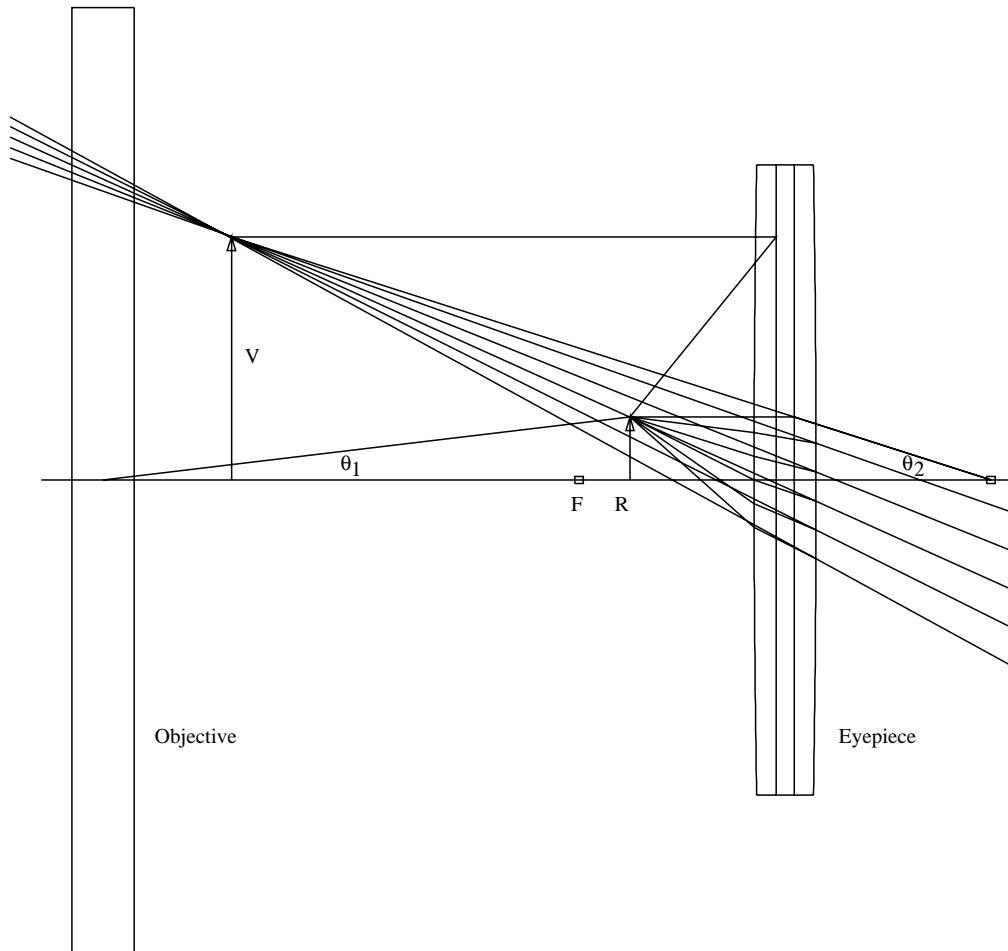


Figure 10: **The Astronomical Telescope.** F is a common focal point of the Objective and the Eyepiece. The objective produces a real image labelled R . The eyepiece produces a virtual image of R at V . A ray from R passes through the center of the objective to the distant object. The angle made by this ray is θ_1 . The angle made by a ray through the center of the eyepiece to the virtual image is approximately θ_2 . This becomes more exact as the object recedes and the real image approaches the focal point. The angular magnification is θ_2/θ_1 . If h is the real image height, then θ_1 is approximately h/f_o and θ_2 is approximately h/f_e , where f_o is the focal length of the objective, and f_e is the focal length of the eyepiece. Therefore the angular magnification is approximately f_o/f_e .

31.12 Terrestrial Telescopes

A terrestrial telescope is similar to the astronomical telescope, except that some means is used to prevent an inverted image. One method is to place an erector lens in the middle of the tube. This can be done by lengthening the tube so that the focal points of the objective and the eyepiece are separated by a distance d . Then one places a convex lens between the focal points. If the lens has focal length close to $d/4$ then the image from the objective will be transformed into an inverted image of equal size located a distance d from the focal point of the objective. Also erection can be effected by using an inverting prism. This is the method used in binoculars. The magnification will again be approximately f_o/f_e . The actual magnification experienced by the viewer depends a bit on the near point of the eye, and on how the viewer focuses the telescope.

31.13 Galileo's Telescope

This telescope has an eyepiece that is a concave lens. So it has an upright image already.

31.14 Newton's Telescope

This is the basic reflecting telescope using a spherical mirror with an eyepiece located perpendicular to the telescope axis.

31.15 Cassegrain's Telescope

This is a reflecting telescope like Newton's, but uses an eyepiece along the telescope axis.

32 The Eye and Measuring Machines

The next few sections relate to models of the eye and to techniques and possible machines for measuring the cornea.

33 The Aspherical Lens Model

Reference: Warren Smith, **Modern Optical Engineering**, 1966, p257. Suppose a surface of revolution is located tangent to the yz -plane where the x -axis is the axis of symmetry. Let

$$s^2 = y^2 + z^2.$$

Let the surface be nearly spherical and lying on the positive side of the yz -plane. Then if $p = (x, y, z)$ is a point on the surface, x is a function of s , which is the distance from p to the x -axis. Let

$$x = f(s).$$

If the surface were exactly spherical with radius R , then its equation would be

$$(R - x)^2 + y^2 + z^2 = R^2,$$

which becomes

$$x^2 - 2Rx + s^2 = 0.$$

Solving for x , which must be less than R , we have

$$\begin{aligned} x &= R(1 - \sqrt{1 - s^2/R^2}) \\ &= R \frac{s^2/R}{1 + \sqrt{1 - s^2/R^2}}. \end{aligned}$$

In order to model a nearly spherical surface we add a power series in s^2 . Let

$$x = f(s) = R \frac{s^2/R}{1 + \sqrt{1 - s^2/R^2}} + A_1 s^2 + A_2 s^4 + \dots$$

The A_2, A_3, \dots are called the aspherical coefficients. If we know points on the surface we can determine the parameters R, A_2, \dots, A_k using least squares (perhaps nonlinear least squares). The implicit equation of the surface is

$$0 = g(x, y, z) = x - f(s).$$

Let a ray be represented by

$$p(t) = p_0 + v_0 t.$$

where t is a parameter, and p_0 is the starting point of the ray directed along the unit vector v . The parameter value t_1 , where the ray meets the aspherical surface is a zero of the function

$$h(t) = g(p(t)).$$

This zero may be found in various ways, perhaps analytically, but certainly numerically. The gradient of g , ∇g is normal to the surface. From the incident ray direction v_i and the surface normal, we can compute the reflected ray v_r . We wish to study the angle sensitivity of the reflected ray. Let a ray meet the surface at surface coordinates (y, z) . The direction of the incident ray is determined by two other parameters. Thus the reflected ray is a function of four parameters. Let q be a vector representing the point where the reflected ray meets a detector. Suppose q is a two dimensional vector, say $q = (q_1, q_2)$. q is a function of four variables. Suppose the variables are x_1, x_2, x_3, x_4 . To study the sensitivity, we compute the linear approximation, namely the derivative (essentially the Jacobian). Thus we compute the partial derivatives

$$\frac{\partial q_i}{\partial x_j}.$$

Using the linear approximation, we may study the sensitivities.

To generalize, suppose there is a set of rays, mirrors, and lenses. The independent variables are the coordinates of the rays, the positions of the mirrors, and lenses, and the parameters of the cornea surface. Suppose there are n independent variables, x_1, x_2, \dots, x_n . There will be two coordinates for each ray, which eventually meets the detector. Say there are m dependent variables q_1, q_2, \dots, q_m . By holding some of the independent variables constant we can match the dimension of the domain space and the range space. Then given a function F , we can compute its inverse F^{-1} , and so determine for example the parameters of the surface from detector values. We can compute the derivative DF of F , to study sensitivities. Let the inverse of the linear transformation DF be M . Then given the known sensitivity of the detector, say Δq , we can compute the corresponding sensitivities of the x variables, that is, $\Delta x = M\Delta q$. The sensitivities will vary as x varies, i.e. the sensitivities will vary at different points of incidence on the aspherical surface. This calculation will determine how accurately we can measure the surface. We can make the model more complicated by using a model that

is not a surface of revolution. The technique of studying the sensitivity will remain essentially the same. If the technique is sensitive enough, using a given arrangement of mirrors, lenses, and a given detector, then calculations can be done to locate points on the surface and thus the parameters of the surface model may be found. Clearly algorithms can be found to do these calculations, but what we need are sufficiently fast and accurate algorithms. So given an arrangement and algorithms, we shall do a computer simulation to establish calculation time, sensitivity, and accuracy. We can trace rays recursively by triangulating all surfaces and as each ray is treated we find the closest triangle that it meets. If the triangle lies on a plane surface such as a planar mirror, we compute the reflected ray (and possibly a refracted ray) using the triangle itself. If the triangle belongs to a curved surface we compute the reflected and refracted ray using the actual curved surface, which the triangle approximates.

As an example, consider the simple situation of the aspherical model given above. Blah blah blah, here is the program, here is a picture, and here are the partial derivatives.

In a physical machine the reflected ray must be controlled, perhaps by a system of mirrors so that it meets the detector. This could be done by some sort of scanning of the detector, or by continuously controlling the incident ray and continuously deflecting the reflected ray to the detector.

34 Outline of Geometrical Ray Tracing

(see section geometrical ray tracing in optics.tex) Given a plane (surface tangent plane)

$$n \cdot p = d$$

separating medium 1 and medium 2, and a ray with start point p_1 and unit direction vector v_1 . We shall compute an intersection point p_2 , a refracted ray with unit vector v_2 , and a reflected ray with unit vector v_3 . Compute the intersection p_2 of the line and plane. If

$$n \cdot v_1 \geq 0,$$

then let

$$u_1 = n,$$

otherwise let

$$u_1 = -n.$$

Then u_1 has the same general direction as v_1 . If u_1 and v_1 are parallel, let

$$v_2 = v_1$$

and

$$v_3 = -v_1.$$

Otherwise, let

$$u_3 = u_1 \times v_1.$$

Let

$$u_2 = u_3 \times u_1.$$

This gives a right handed system of unit vectors. We define the angle of incidence θ_1 by

$$v_1 = \cos(\theta_1)u_1 + \sin(\theta_1)u_2.$$

Thus θ_1 is the angle between v_1 and u_1 . Let η_1 and η_2 be the indices of refraction of medium 1 and medium 2 respectively. Let θ_2 be the angle of refraction, so that

$$v_2 = \cos(\theta_2)u_1 + \sin(\theta_2)u_2.$$

By Snell's law

$$\sin(\theta_2) = \frac{\eta_1 \sin(\theta_1)}{\eta_2}.$$

The reflected ray is

$$v_3 = -\cos(\theta_1)u_1 + \sin(\theta_1)u_2.$$

35 A Ray Tracing Algorithm

We are given a collection of surfaces that divide space into two parts called halfspaces. Each halfspace is assigned optical properties. The surfaces will often be planes. A finite region of a plane may be represented by a set of triangles lying on the plane. Other suitable surfaces include spheres and cylinders, and some other quadric surfaces. For each surface we shall have a set of triangles defining the surface or approximating the surface. We will trace rays through an optical system. From an initial point outside of the

system, we will follow all of the reflected and refracted rays, until all of the rays leave the system. First we read the collection of triangles. We trace a ray to the nearest triangle. We write the ray from the starting point to the closest triangle intersection point. We compute the refracted and reflected ray. In the case of a triangle associated with a curved surface, we will replace the triangle with the tangent plane to the surface. We recursively trace the refracted ray, and the reflected ray. The recursion ends when a ray does not meet any triangle. We shall use the barycentric coordinates of each triangle to determine when a ray meets a triangle. For a perfect mirror we will delete the refracted ray. We must take into account polarization, dispersion, and absorption. We will use Fresnel's formulas and record intensities of the polarized components. We may also have to treat total reflection specially. (see program optict.ftn)

36 Laser Accuracy

The accuracy of the proposed machine will be limited by the properties of the laser and the detector. The light intensity reaching the detector must be above a threshold value, and the beam cross section must not be too large. We can study the properties of a laser beam, which has a Gaussian cross section, as it passes through the optical system and loses intensity, by using our ray tracing program. The simulation program must be validated by comparing with the results of physical experiments.

37 Accuracy of Existing Machines

According to a reference, current cornea measuring machines have an accuracy on the order of 1/10 of a diopter. Using the lens makers equation and an index of refraction of the cornea of $n_c = 1.376$, we can compute the change in corneal radius R . We have

$$s' = \frac{n'}{n' - n} R = \frac{1.376}{.376} R = 3.66R$$

So the optical power is

$$p = \frac{1}{s'} = \frac{1}{3.66R}$$

Then

$$R = \frac{1}{3.66p}.$$

Taking the derivative,

$$\frac{dR}{dp} = -\frac{1}{3.66p^2}.$$

So if $p = 49$ diopters, and $\Delta p = .1$, then

$$\Delta R = 1.14 \times 10^{-5} \text{meter}.$$

So the accuracy is about one 100th of a millimeter, or about 10 microns.

38 The Location of Deflection Mirrors

Suppose we are given a fixed beam with a horizontal direction along a coordinate axis. Suppose the beam is to be reflected from a mirror called m_1 to a mirror called m_2 to the cornea and then to a fixed point on a detector.

Given a point on the cornea, say with spherical coordinates θ, ϕ . Let us work backward to compute the required positions of the two mirrors. We compute the tangent plane and the ray from the detector to the cornea point. Then we get a backwards ray from the cornea to the mirror m_2 . Let us say that the center of m_2 is constrained to lie on a plane p_2 . So we move the center of the mirror m_2 to the intersection point of the backwards ray and the plane p_2 . Let us suppose that mirror m_1 has a fixed center and lies on the line in the direction of the fixed laser, which is in the plane p_1 , above plane p_2 . Let line l_1 pass through the centers of the two mirrors. This fixes the angular position of mirror m_2 , that is, its normal is the vector that bisects the ray from the cornea, and the ray to the first mirror. Then the angle of the second mirror is fixed in the same way by the backwards ray from the second mirror, and the backwards ray from the first mirror to the laser. All of this could be modified to assign an affine transformation to the eye itself. That is, the eye in general is not in its standard position, so before we do the above calculation, we move the eye mathematically (rotation and translation) from its standard position using the affine transformation. Note that mirror m_2 has four degrees of freedom (center (x, y) , rotation (θ_2, ϕ_2)), and mirror m_1 has two degrees of freedom (θ_1, ϕ_1) .

39 Pitfalls

Although it is conceivable that the mirrors could be designed to rotate quickly and accurately, it is not so clear that the mirror translation could be designed to be quick and accurate. At certain angles the mirror accuracy will be low. The time required to locate the mirror could be quite large. Mirror m_2 might interfere with the ray going to the detector.

40 Interference

The classic experiment that first showed that light is a wave was conducted by Thomas Young in 1801. He allowed sunlight to fall on a pinhole, producing a point source of light. The light from the point source then travelled to two closely located pinholes, and then from the two pinholes to a screen. This produced an interference pattern. Other classic interference effects result from thin oil films and from Newton's rings. Newton's rings appear when a lens is placed on a flat surface. In these two cases the interference is produced by light interfering after being reflected from two separated surfaces. Good interference effects require a coherent monochrome light source. The Young paper is reprinted in: **Great Experiments In Physics**, Morris Shamos, Dover, 1959.

41 Vision: Corrective Lenses

Given the near point or far point of the eye, using the thin lenses equation one can compute the properties of a thin lens to correct vision and get the respective near or far point to be moved to the normal 25 cm near point position for a normal eye. A near sighted person requires a corrective diverging lens, and a far sighted person requires a converging lens. Corrective lenses are specified in diopters. By noting how a near point is altered by a corrective lens, one can determine its power in diopters.

Example 1

42 Diffraction

Diffraction is the change of direction of light rays caused by such as: physical obstructions (edges, pinholes, and slits), surface scattering by a grating, scattering by the electron density of atoms and molecules, as in X-Ray diffraction by a crystal lattice. From Green's formula one can derive the Kirchhoff Integral Theorem

$$U_p = -\frac{1}{4\pi} \int_{\partial V} (U \nabla e^{ikr}/r - e^{ikr}/r \nabla U) \cdot ds$$

which gives the value of a scalar spherical wave function centered at p as a surface integral on the surface ∂V of volume V containing the center p . In

the scalar approximation the square of U gives the light intensity.

43 The Kirchoff Diffraction Formula

This formula, which is a consequence of the Kirchoff Integral Theorem, is equivalent to Huygens' principle.

$$U_p = -\frac{ikU_0e^{-i\omega t}}{4\pi} \int_{\partial V} \frac{e^{ik(r+r')}}{rr'} \left(\frac{n \cdot r}{\|r\|} - \frac{n \cdot r'}{\|r'\|} \right) ds,$$

where the volume encloses the image point p and integration is over the aperture surface. The distance from a point on the aperture surface to p is r , and the distance from a point on the aperture surface to the source point is r' . See Fowles, or Born and Wolf. For an elementary treatment of diffraction by various slits and holes, see Morgan. He uses Huygens' principle. Sommerfeld has a mathematical treatment of diffraction by an edge.

44 Resolving Power

See Morgan.

45 Fourier Optics

See the bibliography.

46 Laser Optics

The light intensity distribution function in the cross section of a laser beam ideally is Gaussian, but real beams may vary considerably from the ideal.

The laser has the capability of producing an intense narrow coherent collimated light beam. Such a beam can be used to identify small features of a diamond. The word LASER is an acronym for Light Amplification by Stimulated Emission. Lasers are devices that amplify light. They produce coherent light beams, which means that all of the light packets are in phase. They are also collimated, which means that the photons are moving in nearly

parallel directions. Lasers exist for producing light in the frequency range from infrared to ultraviolet.

Electrons in the atoms of a laser medium are pumped to an excited state by an energy source. Then a burst of radiation is produced by stimulated emission.

Albert Einstein in 1917 proposed stimulated emission, which is the underlying process for laser action (A. Einstein: Phys. Z., 18: 121 (1917)). Einstein's fundamental equation (Smith and Sorokin, page 5) is

$$p(\nu_{ij}) = h\nu_{ij}(n_i A_{ij} + (n_i - n_j) B_{ij} u(\nu_{ij})),$$

where p is the power per unit volume, n_i and n_j are atom energy level populations, u is the energy density of radiation, ν_{ij} is the frequency corresponding to the difference in energy levels, and A_{ij} and B_{ij} are called the Einstein coefficients of spontaneous and induced emissions respectively.

Arthur Schawlow and Charles Hard Townes outlined the construction of lasers in their 1958 patent application. The first Helium-Neon gas laser was built by Ali Javan. In 1966 a liquid laser was constructed by Peter Sorokin.

Lasers are classified according to the medium used. Lasers are classified as solid state, gas, semiconductor, or liquid. Types of lasers include the Helium Neon, Carbon Dioxide, Neodymium, Dye, Excimer, Gallium Arsenide Diode, Argon ion, Helium Cadmium, Copper Vapor, Ruby, Vibronic, and Nitrogen.

The HeNe laser is inexpensive and suitable for the diamond grading project. It is a continuous laser. The most common operating wavelength of the HeNe laser is 632.8 nanometers which is a red light. The frequency is

$$f = \frac{c}{\lambda} = \frac{3 \times 10^8}{632.8 \times 10^{-9}} = 4.74 \times 10^{14} \text{Hertz.}$$

Power ranges from a fraction of a milliwatt to 50 milliwatts. They are often used for aligning optics systems, and for reading and scanning. They are used in supermarket checkout counters, laser printers, in holography, and they are used in interferometers to measure surfaces.

The semiconductor laser is also cheap. The semiconductor laser consists of a junction between layers of semiconductors. Reflective boundaries confine the laser cavity to the semiconductor region. Gallium arsenide is the most common semiconductor. Semiconductor lasers are pumped by the direct application of electrical current across the junction.

Another very common laser is the Excimer laser (excited dimer). The Excimer laser involves the reaction of an inert gas with reactive halide. The

molecules involved are such as argon fluoride, krypton fluoride, and xenon chloride. These molecules only form for a few billionths of a second when they are excited by an energy source. They emit pulses of light when the molecule collapses back to the separate atoms. This type of laser is not suitable for the project.

The critical aspect of the laser for the project is the beam width, the intensity of the beam, and the divergence of the beam. The beam will travel a relatively long distance before it strikes the detector. The position error depends on the size of the detected beam spot and on the resolution of the detector. Unfortunately a very high resolution detector is very expensive. The beam width is controlled to a certain extent by the size of the mirror. The beam divergence $\Delta\theta$ is approximately

$$\Delta\theta = \frac{\lambda}{D},$$

where D is the mirror diameter. For $D = .01$ m , and $\lambda = 632.8 \times 10^{-9}$ m, we have

$$\Delta\theta = \frac{\lambda}{D} = 6.328 \times 10^{-5} \text{rad} = .004 \text{degrees}.$$

A collimator (telescope) may be used to decrease the beam size. A beam can be collimated with a telescope to have only 10^{-5} rad divergence, so that it would travel to the moon with a target image only 2 miles in diameter. However we need to both, decrease the beam size, and make the beam parallel. The two properties are in competition. The beam must be small when it strikes the diamond, and also small when it strikes the detector. We can use lenses to make the beam small at either one of these points, but perhaps not at both.

47 Acousto-Optics

- **Heterodyne Interferometer** Researcher: Ignatio Perez of the Naval Air Warfare Center. The Bragg defractor gives a second beam at 40mHz from the reference beam, which is modulated by the motion of the surface. This is converted to an electric signal with a solar cell. It is said to have 5 angstrom sensitivity, 5 volts. There are two commercial companies: Zygo and Wyko, located in Arizona and Connecticut.
- **NAWC**The Naval Air Warfare Center is located near Philadelphia.

- **The Bragg Cell** is a bulk-wave acousto-optic modulator (McGraw-Hill Encyclopedia of Science). The Bragg modulator consists of an acoustic cell and a laser source. Acoustic waves are generated by applying a radio-frequency signal to a planar piezoelectric transducer, producing a moving optical grating. At the Bragg angle, the diffracted beam makes an angle with the undiffracted beam, which is twice the Bragg angle. The diffracted beam has frequency

$$f_d = f_i \pm f_a,$$

where f_a is the acoustic frequency.

- **Laser Doppler Vibrometer**

48 Geometrical Ray Tracing

We present a method of tracing rays through surfaces bounding regions having different optical parameters. We assume that the surfaces are triangulated. We trace a ray from a starting point in a direction defined by a unit vector. We locate the first triangle through which the ray passes. Associated with the surface, which is approximated by the triangle, is an outward pointing unit surface normal. We take the index of refraction of the material region to which this outward normal points to be n_1 and the index of refraction of the material region on the opposite side of the surface to be n_2 .

Thus we have a plane (surface tangent plane)

$$n \cdot p = d$$

separating medium 1 and medium 2, and a ray with start point p_1 and unit direction vector v_1 . We shall compute an intersection point p_2 , a refracted ray with unit vector v_2 , and a reflected ray with unit vector v_3 . First we compute the intersection point p_2 of the line and the plane.

We shall establish a triad of orthogonal unit vectors centered at the intersection point. If

$$n \cdot v_1 \geq 0,$$

then we let

$$u_1 = n,$$

otherwise we let

$$u_1 = -n.$$

Then u_1 has the same general direction as v_1 .

If u_1 and v_1 are parallel then we can immediately determine the refracted and reflected directions and we are done with this step of the ray trace. So in this case the refracted direction is

$$v_2 = v_1$$

and the reflected direction is

$$v_3 = -v_1.$$

Suppose u_1 and v_1 are not parallel. Then we proceed with constructing the orthogonal frame of unit vectors. Let

$$u_3 = u_1 \times v_1,$$

and let

$$u_2 = u_3 \times u_1.$$

This gives a right handed system of unit vectors. Let θ_1 be the positive angle between vectors u_1 and v_1 . By the way we have constructed u_1 , this angle is between 0 and $\pi/2$, and is the angle of incidence between the ray with direction v_1 and the plane normal. For the unit vector v_1 we have

$$v_1 = \cos(\theta_1)u_1 + \sin(\theta_1)u_2.$$

Hence

$$\cos(\theta_1) = v_1 \cdot u_1,$$

and

$$\sin(\theta_1) = \sqrt{1 - \cos^2(\theta_1)}.$$

Let n_1 and n_2 be the indices of refraction of medium 1 and medium 2 respectively, where medium 1 is pointed to by the unit surface normal. If the incident ray meets the surface from inside of the bounding surface, that is

$$n \cdot v_1 \geq 0,$$

then n_1 and n_2 must be interchanged in the following application of Snell's law. Let θ_2 be the angle of refraction. By Snell's law

$$\sin(\theta_2) = \frac{n_1 \sin(\theta_1)}{n_2}.$$

The refracted ray is

$$v_2 = \cos(\theta_2)u_1 + \sin(\theta_2)u_2.$$

The reflected ray is

$$v_3 = -\cos(\theta_1)u_1 + \sin(\theta_1)u_2.$$

In general we shall repeat the ray trace using the refracted ray and the reflected ray. This ray tracing technique is implemented in program **opti-crt.ftn**, which is presented in a section of this document.

49 An Optical Ray Tracing Program

This program traces rays through surfaces defined as collections of triangles. As each surface is encountered, a reflected ray, and a refracted ray are generated. These rays become new ray sources and the algorithm is called recursively. The recursion ends when a ray does not meet any triangle. Future modifications could be: (1) To use the Fresnel equations to compute intensities with polarization, and (2) The association of a triangle with an underlying geometric surface, which would be used to compute the surface normals.

```
c program name opti-crt.ftn 12/6/95 recursive optical ray trace
c Author: jim emery
c Read collection of triangles, trace ray to nearest triangle,
c write ray from start point
c to triangle intersection point, compute refracted and reflected ray,
c Recursively trace refracted and reflected ray, recursion ends
c when ray does not meet any triangle.
  parameter(isize=1000)
  implicit real*8(a-h,o-z)
  dimension pn1(3),pn2(3),pn4(3)
  dimension x(3),y(3),z(3),vn(3)
  dimension p1(3),p2(3),p3(3),p4(3)
  dimension v1(3),v2(3),v3(3)
  dimension pe(3),ps(3)
  dimension xt(3, isize),yt(3, isize),zt(3, isize)
  dimension tn(3, isize),eta1(isize),eta2(isize)
  dimension ain(10)
```

```

c      open(1,file='icosaup.t',status='unknown')
      open(1,file='sphercap.t',status='unknown')
      open(2,file='p.n',status='unknown')
      open(3,file='p.l',status='unknown')
c      read triangles
      n=0
c      read triangles
40     continue
      call readr(1,ain,nr)
      if(nr .ne. 3)go to 50
      xt(1,n+1)=ain(1)
      yt(1,n+1)=ain(2)
      zt(1,n+1)=ain(3)

      call readr(1,ain,nr)
      if(nr .ne. 3)go to 50
      xt(2,n+1)=ain(1)
      yt(2,n+1)=ain(2)
      zt(2,n+1)=ain(3)

      call readr(1,ain,nr)
      if(nr .ne. 3)go to 50
      xt(3,n+1)=ain(1)
      yt(3,n+1)=ain(2)
      zt(3,n+1)=ain(3)
      n=n+1
c      write(*,*)'triangle ',n
c      do i=1,3
c        write(*,'(a,3(g12.5,1x))')xt(i,n),yt(i,n),zt(i,n)
c      enddo
      go to 40
50     continue
c      add a triangle
      n=n+1
      xt(1,n)=2.
      yt(1,n)=2.
      zt(1,n)=0.
      xt(2,n)=2.
      yt(2,n)=-2.
      zt(2,n)=0.
      xt(3,n)=3.
      yt(3,n)=0.
      zt(3,n)=2.
c 2 2 0
c 2 -2 0
c 3 0 2
      write(*,*)' number of triangles = ',n
c      read indices of refraction
      do i=1,n
      call readr(2,ain,nr)
      eta1(i)=ain(1)
      eta2(i)=ain(2)
      enddo
c      compute triangle normals
      do i=1,n
      do m=1,3
      x(m)=xt(m,i)

```

```

        y(m)=yt(m,i)
        z(m)=zt(m,i)
    enddo
c    write(*,*)'triangle ',i
c    do j=1,3
c        write(*,'(a,3(g12.5,1x))')x(j),y(j),z(j)
c    enddo
    call napoly(x,y,z,3,a,vn)
    do m=1,3
        tn(m,i)=vn(m)
    enddo
c    write(*,'(a,3(g12.5,1x))')'normal = ',(vn(ii),ii=1,3)
    enddo
c    define ray start point
    p1(1)=-2
    p1(2)=0
    p1(3)=1
    ps(1)=.1
    ps(2)=.1
    ps(3)=.0
    pe(1)=-.3
    pe(2)=-.1
    pe(3)=.0
    m=5
    one=1.
c    trace rays
    do j=1,m
        t=(j-1)*one/(m-1)
        do i=1,3
            p2(i) = (1.-t)*ps(i) + t*pe(i)
        enddo
c    define ray direction
    call unitv(p1,p2,v1)
    lev=1
    itri=0
    call trace(itri,p1,v1,xt,yt,zt,eta1,eta2,tn,n,lev)
    enddo
end
c+ unitv construct unit vector directed from point 1 to point 2
subroutine unitv(p1,p2,v)
implicit real*8(a-h,o-z)
dimension p1(*),p2(*),v(*)
vm=0.
do i=1,3
    v(i) = p2(i) - p1(i)
    vm = vm + v(i)**2
enddo
vm=sqrt(vm)
do i=1,3
    v(i) = v(i)/vm
enddo
return
end
c+ trace optical ray trace
subroutine trace(itri,p1,v1,xt,yt,zt,eta1,eta2,tn,n,lev)
c input:
c itri set to k, if p1 is on triangle k, otherwise 0

```

```

c p1 point on ray
c v1 unit vector in direction of ray
c xt,yt,zt triangle vertices
c pn plane normal
c p2 point on plane
c eta1 index of refraction in medium 1
c eta2 index of refraction in medium 2
c output:
c p3 point of ray incidence
c v2 unit vector direction of refracted ray
c v3 unit vector direction of reflected ray
c see optics.tex
implicit real*8(a-h,o-z)
dimension p1(*),v1(*),xt(3,*),yt(3,*),zt(3,*)
dimension tn(3,*),eta1(*),eta2(*)
dimension pn1(3),pn2(3),pn4(3)
dimension p2(3),p3(3),p4(3)
dimension v2(3),v3(3)
dimension pn(3),pp3(3),vv2(3),vv3(3)
dimension bc(3)
dimension q1(3),q2(3),q3(3)
dimension p10(3)
write(*,*)' into trace, recursion level ',lev
if(lev .gt. 3)return
zero=0.
dmn=zero
s=1.
met=0
do k=1,n
  if(k .ne. itri)then
c   trace ray through plane k of triangle k
c   define point on plane k
    p2(1)=xt(1,k)
    p2(2)=yt(1,k)
    p2(3)=zt(1,k)
    q1(1)=xt(1,k)
    q1(2)=yt(1,k)
    q1(3)=zt(1,k)
    q2(1)=xt(2,k)
    q2(2)=yt(2,k)
    q2(3)=zt(2,k)
    q3(1)=xt(3,k)
    q3(2)=yt(3,k)
    q3(3)=zt(3,k)
    do i=1,3
      pn(i)=tn(i,k)
    enddo
c   write(*,*)' k= ',k
c   write(*,*)' into refrac '
c   write(*,'(a,3(g12.5,1x))')'p1 = ',(p1(ii),ii=1,3)
c   write(*,'(a,3(g12.5,1x))')'v1 = ',(v1(ii),ii=1,3)
c   write(*,'(a,3(g12.5,1x))')'pn = ',(pn(ii),ii=1,3)
c   write(*,'(a,3(g12.5,1x))')'p2 = ',(p2(ii),ii=1,3)
c   call refrac(p1,v1,pn,p2,eta1(k),eta2(k),p3,v2,v3)
c   write(*,*)' out of refrac '
c   write(*,'(a,3(g12.5,1x))')'p3 = ',(p3(ii),ii=1,3)
c   write(*,'(a,3(g12.5,1x))')'v2 = ',(v2(ii),ii=1,3)

```

```

c      write(*,'(a,3(g12.5,1x))')v3 = ',(v3(ii),ii=1,3)
c      compute the barycentric coordinates of p3 relative to the triangle
c      call baryt(p3,q1,q2,q3,bc)
c      if((bc(1).lt.zero).or.(bc(2).lt.zero).or.(bc(3).lt.zero))then
c          meet=0
c      else
c          meet=1
c      compute directed distance
c          do i=1,3
c              p1(i)=p3(i)-p1(i)
c          enddo
c          d=dotpr(p10,v1)
c          d=(p3(1)-p1(1))**2+(p3(2)-p1(2))**2+(p3(3)-p1(3))**2
c          check that point is in the direction of ray, i.e. d is positive
c          if(d .lt. zero)then
c              meet=0
c          endif
c      endif
c      if(meet .eq. 1)then
c          write(*,*)' triangle was intersected '
c          met=1
c          if((d .lt. dmn) .or. (dmn .eq. zero))then
c              dmn=d
c              do j=1,3
c                  pp3(j)=p3(j)
c                  vv2(j)=v2(j)
c                  vv3(j)=v3(j)
c              enddo
c              ktri=k
c          endif
c      endif
c      end do
c      if(met .eq. 1)then
c          plot line segment from initial point to incident point
c          write(*,*)' plot of segment '
c          write(*,'(3(g12.5,1x))')p1(1),p1(2),p1(3)
c          write(*,'(3(g12.5,1x))')pp3(1),pp3(2),pp3(3)
c          write(3,'(3(g12.5,1x))')p1(1),p1(2),p1(3)
c          write(3,'(3(g12.5,1x))')pp3(1),pp3(2),pp3(3)
c          lev2=lev+1
c          itri=ktri
c          call trace(itri,pp3,vv2,xt,yt,zt,eta1,eta2,tn,n,lev2)
c          call trace(itri,pp3,vv3,xt,yt,zt,eta1,eta2,tn,n,lev2)
c      else
c          plot ray
c          write(*,*)' plot of ray '
c          write(*,'(3(g12.5,1x))')p1(1),p1(2),p1(3)
c          write(*,'(3(g12.5,1x))')p1(1)+s*v1(1),p1(2)+s*v1(2),p1(3)+s*v1(3)
c          write(3,'(3(g12.5,1x))')p1(1),p1(2),p1(3)
c          write(3,'(3(g12.5,1x))')p1(1)+s*v1(1),p1(2)+s*v1(2),p1(3)+s*v1(3)
c      endif
c      write(*,*)' out of trace '
c      return
c      end
c+ intlnp intersection of plane and line
c      subroutine intlnp(an,p,a,b,x,t,icase)

```

```

        implicit real*8(a-h,o-z)
c parameters:
c
c   an-normal to plane
c   p-point on plane
c   a-point on line
c   b-point on line
c   x-returned intersection point
c   t-parameter of intersection point
c   x=t*a+(1-t)*b
c icase-cases of intersection
c   icase=1: single intersection point
c   icase=2: line lies in plane
c   icase=3: line parallel to plane, intersection empty.
c
c   dimension an(3),p(3),a(3),b(3),x(3)
c   dimension an(*),p(*),a(*),b(*),x(*)
c calculate inner products
    zero=0.
    andota=0.
    andotb=0.
    andotp=0.
    do 10 i=1,3
    c=an(i)
    andota=andota+a(i)*c
    andotb=andotb+b(i)*c
10  andotp=andotp+p(i)*c
c check for line parallel to plane
    c=andota-andotb
    if(c .eq. zero)go to 100
c not parallel
c case 1: single intersection point
    icase=1
    t=(andotp-andotb)/c
    s=1.-t
    do 20 i=1,3
20  x(i)=t*a(i)+s*b(i)
    return
100 continue
    u=andota-andotp
    if(u .ne. zero)go to 200
c case 2: line in plane
    icase=2
    return
200 continue
c case 3 line parallel, intersection empty.
    icase=3
    return
end
c+ refrac ray refracted and reflected by plane
    subroutine refrac(p1,v1,pn,p2,eta1,eta2,p3,v2,v3)
c input:
c p1 point on ray
c v1 unit vector in direction of ray
c pn plane normal
c p2 point on plane
c eta1 index of refraction in medium 1

```

```

c eta2 index of refraction in medium 2
c output:
c p3 point of ray incidence
c v2 unit vector direction of refracted ray
c v3 unit vector direction of reflected ray
c see optics.tex
implicit real*8(a-h,o-z)
dimension p1(*),v1(*),pn(*),p2(*),p3(*),v2(*),v3(*)
dimension u1(3),u2(3),u3(3),b(3)
zero=0.
b(1)=p1(1)+v1(1)
b(2)=p1(2)+v1(2)
b(3)=p1(3)+v1(3)
call intlmp(pn,p2,p1,b,p3,t,icase)
cs1=dotpr(pn,v1)
if(cs1 .gt. zero)then
  u1(1)=pn(1)
  u1(2)=pn(2)
  u1(3)=pn(3)
else
  u1(1)=-pn(1)
  u1(2)=-pn(2)
  u1(3)=-pn(3)
  cs1=-cs1
endif
sn1=1.-cs1**2
call crsspr(u1,v1,u3)
call crsspr(u3,u1,u2)
sn2=eta1*sn1/eta2
cs2=1.-sn2**2
v2(1)=cs2*u1(1)+sn2*u2(1)
v2(2)=cs2*u1(2)+sn2*u2(2)
v2(3)=cs2*u1(3)+sn2*u2(3)
v3(1)=-cs1*u1(1)+sn1*u2(1)
v3(2)=-cs1*u1(2)+sn1*u2(2)
v3(3)=-cs1*u1(3)+sn1*u2(3)
return
end

c+ dotpr scalar product.
function dotpr(a,b)
implicit real*8(a-h,o-z)
dimension a(3),b(3)
s=0.
do 10 i=1,3
10 s=s+a(i)*b(i)
dotpr=s
return
end

c
c+ crsspr vector cross product.
subroutine crsspr(a,b,c)
implicit real*8(a-h,o-z)
c=product of a and b
dimension a(3),b(3),c(3)
c(1)=a(2)*b(3)-a(3)*b(2)
c(2)=a(3)*b(1)-a(1)*b(3)
c(3)=a(1)*b(2)-a(2)*b(1)

```

```

        return
    end
c+ readr read a row of floating point numbers
    subroutine readr(nf, a, nr)
        implicit real*8(a-h,o-z)
    c  numbers are separated by spaces
    c  examples of valid numbers are:
    c  12.13 34 45e4 4.78e-6 4e2,5.6D-23,10000.d015
    c  nf=file number, 0 for standard input file
    c  a=array of returned numbers
    c  nr=number of values in returned array,
    c      or 0 for empty or blank line,
    c      or -1 for end of file on unit nf.
    c requires functions val and length
        dimension a(*)
        character*200 b
        character*200 c
        character*1 d
        c=' '
        if(nf.eq.0)then
            read(*,'(a)',end=99)b
        else
            read(nf,'(a)',end=99)b
        endif
        nr=0
        l=lenstr(b)
        if(l.ge.200)then
            write(*,*)' error in readr subroutine '
            write(*,*)' record is too long '
        endif
        do 1 i=1,l
            d=b(i:i)
            if (d.ne.' ') then
                k=lenstr(c)
                if (k.gt.0)then
                    c=c(1:k)//d
                else
                    c=d
                endif
            endif
            if( (d.eq.' ').or.(i.eq.l)) then
                if (c.ne.' ') then
                    nr=nr+1
                    call valsub(c,a(nr),ier)
                    c=' '
                endif
            endif
        1 continue
        return
    99 nr=-1
        return
    end
c
c+ lenstr nonblank length of string
    function lenstr(s)
    c length of the substring of s obtained by deleting all
    c trailing blanks from s. thus the length of a string

```

```

c containing only blanks will be 0.
  character s*(*)
  lenstr=0
  n=len(s)
  do 10 i=n,1,-1
  if(s(i:i) .ne. ' ')then
    lenstr=i
    return
  endif
10 continue
  return
end

c+ valsub converts string to floating point number (r*8)
  subroutine valsub(s,v,ier)
  implicit real*8(a-h,o-z)
c examples of valid strings are: 12.13 34 45e4 4.78e-6 4E2
c the string is checked for valid characters,
c but the string can still be invalid.
c s-string
c v-returned value
c ier- 0 normal
c 1 if invalid character found, v returned 0
c

  logical p
  character s*(*),c*50,t*50,ch*15
  character z*1
  data ch/'1234567890+-.eE'/
  v=0.
  ier=1
  l=lenstr(s)
  if(l.eq.0)return
  p=.true.
  do 10 i=1,l
  z=s(i:i)
  if((z.eq.'D').or.(z.eq.'d'))then
    s(i:i)='e'
  endif
  p=p.and.(index(ch,s(i:i)).ne.0)
10 continue
  if(.not.p)return
  n=index(s,'.')
  if(n.eq.0)then
    n=index(s,'e')
    if(n.eq.0)n=index(s,'E')
    if(n.eq.0)n=index(s,'d')
    if(n.eq.0)n=index(s,'D')
    if(n.eq.0)then
      s=s(1:l)//'.'
    else
      t=s(n:l)
      s=s(1:(n-1))//'. '//t
    endif
    l=l+1
  endif
  write(c,'(a30)')s(1:l)
  read(c,'(g30.23)')v
  ier=0

```

```

        return
    end
c+ str floating point number to string
subroutine str(x,s)
implicit real*8(a-h,o-z)
character s*25,c*25,b*25,e*25
zero=0.
if(x.eq.zero)then
    s='0'
    return
endif
write(c,'(g11.4)')x
read(c,'(a25)')b
l=lenstr(b)
do 10 i=1,l
n1=i
if(b(i:i).ne.' ')go to 20
10 continue
20 continue
    if(b(n1:n1).eq.'0')n1=n1+1
    b=b(n1:l)
    l=l+1-n1
    k=index(b,'E')
    if(k.gt.0)e=b(k:l)
    if(k.gt.0)then
        s=b(1:(k-1))
        k1=index(b,'E+0')
        if(k1.gt.0)then
            e='E'//b((k1+3):l)
        else
            k1=index(b,'E+')
            if(k1.gt.0)e='E'//b((k1+2):l)
        endif
        k1=index(b,'E-0')
        if(k1.gt.0)e='E-'//b((k1+3):l)
        l=k-1
    else
        s=b
    endif
    j=index(s,'.')
    n2=1
    if(j.ne.0)then
        do 30 i=1,l
            n2=l+1-i
            if(s(n2:n2).ne.'0')go to 40
30         continue
    endif
40     continue
    s=s(1:n2)
    if(s(n2:n2).eq.'.')then
        s=s(1:(n2-1))
        n2=n2-1
    endif
    if(k.gt.0)s=s(1:n2)//e
    return
end
c+ napoly unit normal and area of polyhedral face

```

```

        subroutine napoly(x,y,z,n,a,vn)
        implicit real*8(a-h,o-z)
c input:
c  x,y,z-polyhedron vertices
c  n-number of vertices
c output:
c  a-face area
c  vn-normal vector
c method: compute magnetic moment integral of face boundary
c the contribution to the integral for edge p1,p2 is p1 cross p2.
c vn is determined by a global right hand rule, vn is
c in the direction of the magnetic moment of the circuit
        dimension x(*),y(*),z(*)
        dimension vn(3),r1(3),r2(3),dv(3)
        zero=0.
        do 5 i=1,3
            vn(i)=0.
5        continue
        do 20 k=1,n
            r1(1)=x(k)
            r1(2)=y(k)
            r1(3)=z(k)
            j=k+1
            if(j .gt. n)j=1
            r2(1)=x(j)
            r2(2)=y(j)
            r2(3)=z(j)
            call crsspr(r1,r2,dv)
            do 10 i=1,3
                vn(i)=vn(i)+dv(i)
10        continue
20    continue
        a=0.
        do 30 i=1,3
            a=a+vn(i)**2
30    continue
        if(a .gt. zero)then
            a=sqrt(a)
            do 40 i=1,3
                vn(i)=vn(i)/a
40    continue
        endif
        a=a/2.
        return
        end
c+ baryt barycentric coordinates of a point relative to a triangle in space
        subroutine baryt(p,p1,p2,p3,b)
c input:
c  p      3d point
c  p1,p2,p3 3d points of triangle
c output:
c  b      barycentric coordinates of the projection
c         of p to the plane of the triangle.
c         projection(p)=b(1)*p1+b(2)*p2+b(3)*p3
c         b(1)+b(2)+b(3) = 1
c         the projection is outside the triangle if
c         and only if some coordinate is negative

```

```

c Reference: Computer Graphics and Geometry, Jim Emery
implicit real*8(a-h,o-z)
dimension p(*),p1(*),p2(*),p3(*),b(*)
dimension a(3),a1(3),a2(3),b1(3),b2(3),u1(3),u2(3)
do i=1,3
  a1(i)=p1(i)-p3(i)
  a2(i)=p2(i)-p3(i)
  a(i)=p(i)-p3(i)
enddo
c1=-dotpr(a1,a2)
c2=dotpr(a1,a1)
do i=1,3
  b2(i)=c1*a1(i)+c2*a2(i)
enddo
c3=dotpr(b2,b2)
do i=1,3
  u1(i)=a1(i)/sqrt(c2)
  u2(i)=b2(i)/sqrt(c3)
enddo
x=dotpr(u1,a)
y=dotpr(u2,a)
x1=dotpr(u1,a1)
y1=dotpr(u2,a1)
x2=dotpr(u1,a2)
y2=dotpr(u2,a2)
d=x1*y2-y1*x2
b(1)=(x*y2-y*x2)/d
b(2)=(y*x1-x*y1)/d
b(3)=1.-b(1)-b(2)
return
end

c+ trintr linear interpolation in a triangle.
  subroutine trintr(px,py,pz,x,y,z,int,bary)
c function      -linear interpolation in a triangle.
c               calculates the barycentric coordinates of the point
c               (x,y) in the simplex (triangle) defined by px and
c               py. then z is calculated as a linear combination
c               of pz and the coordinates bary.
c parameters    px-array of dimension 3 containing the x coordinates
c               of the 3 points forming the triangle.
c               py-array of dimension 3 containing y coordinates of the
c               triangle.
c               pz-array of dimension 3 containing the functional z
c               values at each of the points of the triangle.
c               x,y-point for which the barycentric coordinates are to
c               be calculated.
c               z-interpolated value at (x,y) to be returned.
c               int-value is 1 if (x,y) is inside the triangle,
c               0 otherwise.
c               bary-array of dimension 3 containing the barycentric
c               coordinates.
c references     -see any book on algebraic topology for a discussion
c               of barycentric coordinates and simplicial
c               complexes.
dimension px(3),py(3),pz(3),bary(3)
int=1
p1=x-px(1)

```

```

p2=y-py(1)
q1=px(2)-px(1)
q2=py(2)-py(1)
r1=px(3)-px(1)
r2=py(3)-py(1)
epsilon=(1.e-10)*(abs(q1)+abs(q2))
c the barycentric coordinates satisfy the following system.
c p1=bary(2)*q1+bary(3)*r1
c p2=bary(2)*q2+bary(3)*r2
det=q1*r2-q2*r1
bary(2)=(p1*r2-p2*r1)/det
bary(3)=(p2*q1-p1*q2)/det
bary(1)=1.-(bary(2)+bary(3))
c the following vector equation holds
c (x,y)=bary(1)*(px(1),py(1))+bary(2)*(px(2),py(2))
c +bary(3)*(px(3),py(3))
c
c determine if (x,y) is inside or on the triangle, with tolerance
c epsilon.
do 10 i=1,3
if(bary(i).le.-epsilon)int=0
10 continue
z=bary(1)*pz(1)+bary(2)*pz(2)+bary(3)*pz(3)
return
end

```

50 Locating The Point Of Refraction On A Plane Surface

Suppose we have two optical media separated by a plane. Let point P be in medium 1, where the index of refraction is n_1 . Let Q be a point in medium 2, where the index of refraction is n_2 . Suppose a ray of light leaves point P and meets the plane at point R , so that the refracted ray passes through point Q . We call R the refraction point (see figure in notebook 2/4/97).

We wish to find the unique point R . Let P' be the projection of point P onto the plane, and Q' similarly the projection of point Q . (See geometry.tex for the projection calculation). Let c and d be the distances to the plane:

$$c = \|P - P'\|$$

$$d = \|Q - Q'\|$$

The point R will lie between P' and Q' . Let a be the distance from P' to R and b the distance from Q' to R . That is

$$a = \|P' - R\|$$

$$b = \|Q' - R\|$$

Define $\beta = \|P' - Q'\|$, then

$$b = \beta - a$$

We shall use Snell's law

$$\sin(\theta_1)n_1 = \sin(\theta_2)n_2$$

We define α by

$$\alpha = \frac{n_2}{n_1} = \frac{\sin(\theta_1)}{\sin(\theta_2)} = \frac{a\sqrt{b^2 + d^2}}{b\sqrt{a^2 + c^2}}$$

Squaring, we have

$$\alpha^2 b^2 (a^2 + c^2) = a^2 (b^2 + d^2)$$

b is a function of a , so we have a fourth degree polynomial equation

$$f(a) = \alpha^2 b^2 (a^2 + c^2) - a^2 (b^2 + d^2) = 0$$

For $a = 0$, we have $b = \beta$ and

$$f(0) = \alpha^2 \beta^2 c^2 > 0$$

For $a = \beta$, we have $b = 0$ and

$$f(\beta) = -\beta^2 d^2 < 0$$

So f has a root between 0 and β . Now $f(a) = 0$ has four roots, but only one will be between 0 and β . Indeed, from the geometry of the problem, when $a = 0$, then the refraction angle θ_1 and the refraction angle θ_2 are both zero. As θ_1 increases, θ_2 also increases, so the refracted ray sweeps out in a constant direction and clearly can meet Q only once. There is only one root of

$$f(a) = 0$$

in the interval $(0, \beta)$. The root can be found by solving the fourth degree polynomial, but perhaps it is easier to use a root finding method such as bisection.

51 The Intensities of Refracted and Reflected Rays

Fresnel formulas (Born and Wolf):

$$T_{\parallel} = \frac{2n_1 \cos(\theta_i)}{n_2 \cos(\theta_i) + n_1 \cos(\theta_t)} A_{\parallel}$$

$$T_{\perp} = \frac{2n_1 \cos(\theta_i)}{n_1 \cos(\theta_i) + n_2 \cos(\theta_t)} A_{\perp}$$

$$R_{\parallel} = \frac{n_2 \cos(\theta_i) - n_1 \cos(\theta_t)}{n_2 \cos(\theta_i) + n_1 \cos(\theta_t)} A_{\parallel}$$

$$R_{\perp} = \frac{n_1 \cos(\theta_i) - n_2 \cos(\theta_t)}{n_1 \cos(\theta_i) + n_2 \cos(\theta_t)} A_{\perp}$$

52 Geometrical Derivation of the Thin Lens Equation

Consider the cross section of a thin lens lying in the xy plane. Let the focal points be $(-f, 0)$ and $(f, 0)$. Let a source of light be at the left side of the lens at $(-a, c)$, to the left of the focal point $(-f, 0)$. Let a ray pass from this point through the origin to a focused point (b, d) on the right side of the lens. Let a second ray pass through the focus $(-f, 0)$ to the lens, where it is refracted into a ray parallel to the x axis, meeting the focused point (b, d) . Then by similar triangles

$$\frac{c}{a-f} = \frac{d}{f}$$

so that

$$\frac{c}{d} = \frac{a-f}{f}.$$

Also

$$\frac{c}{a} = \frac{d}{b}$$

implies that

$$\frac{a}{b} = \frac{c}{d}$$

Combining these two equations, we have

$$\frac{a}{b} = \frac{a-f}{f},$$

and

$$\begin{aligned}fa &= ba - bf, \\f(a+b) &= ba.\end{aligned}$$

Then

$$f = \frac{ba}{a+b}$$

Inverting, we get the thin lens equation

$$\frac{1}{f} = \frac{1}{a} + \frac{1}{b},$$

which relates the distance a from the object to the lens, to the distance b from the lens to the image. We write this as

$$\frac{1}{f} = \frac{1}{d_i} + \frac{1}{d_o},$$

where d_o is the distance from the object to the lens, and d_i is the distance from the lens to the image. See also a later section on computing with the thin lens equation.

53 The Apparent Position of a Source Point In Medium 1, Viewed From Medium 2

Suppose two media are separated by a plane. Suppose a point source of light is located a distance d_1 from the plane in medium 1. Suppose a ray from this source makes an angle θ_1 with the plane normal. At the surface the refracted ray makes an angle θ_2 with the plane normal so that

$$\sin(\theta_1)n_1 = \sin(\theta_2)n_2.$$

Construct a normal line through the source point in medium 1 and through the plane. Extend the refracted ray backwards into medium 1 so that it

intersects the normal line at a point, which is a distance d_2 from the plane. Then we see that

$$\sin(\theta_1) = \frac{h}{d_1},$$

and

$$\sin(\theta_2) = \frac{h}{d_2}.$$

We have

$$d_2 = \frac{h}{\sin(\theta_2)} = \frac{hn_2d_1}{hn_1} = d_1 \frac{n_2}{n_1}.$$

Therefore, in medium 2 the rays appear to originate from a point on the normal line at distance

$$d_2 = d_1 \frac{n_2}{n_1}.$$

For example, suppose medium 1 is diamond, and medium 2 is air. Then

$$n_1 = 2.42, n_2 = 1$$

Hence a source point in the diamond, when viewed from the air side of the plane appears to be closer to the surface than it is by a factor

$$\frac{1}{2.42}.$$

The object appears to be about 1/2 of the distance to the surface from where it really is. For water, because $n_2 = 4/3$, the factor is about 3/4.

54 Snell's Law as an Extremum of Time

Referring to the **Snell's Law** figure, let v_1 be the velocity in the upper plane and v_2 the velocity in the lower plane, with $v_2 < v_1$. We are to find the position of the point $Q = (x, 0)$ to minimize the travel time from point P to point R . The length of the path in the upper plane is

$$\ell_1 = \sqrt{d^2 + x^2},$$

and the length of the path in the lower plane is

$$\ell_2 = \sqrt{e^2 + (c - x)^2}$$

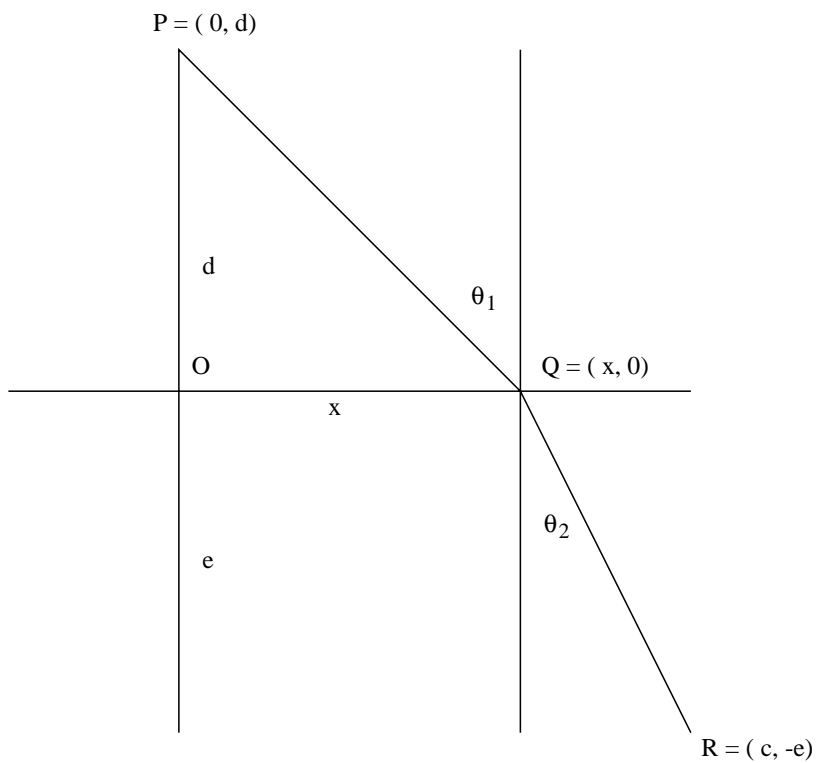


Figure 11: **Snell's Law.** *Let two media be separated by the horizontal line. Light in the upper media travels at velocity v_1 , in the lower at velocity v_2 , with $v_2 < v_1$. The travel time from P to R is minimized when x the coordinate of Q is selected to satisfy Snell's law.*

The travel time as a function of x is

$$t = \frac{\ell_1}{v_1} + \frac{\ell_2}{v_2}.$$

The derivative of the time is

$$\begin{aligned} \frac{dt}{dx} &= \frac{(x/\sqrt{d^2 + x^2})}{v_1} - \frac{((c-x)/\sqrt{e^2 + (x-c)^2})}{v_2} \\ &= \frac{\sin(\theta_1)}{v_1} - \frac{\sin(\theta_2)}{v_2}. \end{aligned}$$

Setting this to zero, we find that the condition for a minimum is

$$\frac{\sin(\theta_1)}{v_1} = \frac{\sin(\theta_2)}{v_2}.$$

In the case of optics we have the indices of refraction

$$n_1 = \frac{c}{v_1}, n_2 = \frac{c}{v_2}.$$

So we have Snell's law

$$n_1 \sin(\theta_1) = n_2 \sin(\theta_2).$$

55 Indices of Refraction

Here are a few indices of refraction.

water	1.3330
borosilicate glass	1.470
ice	1.31
diamond	2.419
ethanol	1.36
crown glass	1.50
flint glass	1.523 - 1.925
silicon	4.01
carbon disulfide	1.628
fused quartz	1.46

56 The Focal length of a Pair of Thin Lenses

One can show using the matrices for thin lenses, which are separated by t that

$$\frac{1}{f} = \frac{1}{f_1} + \frac{1}{f_2} - \frac{t}{f_1 f_2}$$

So the power is additive neglecting the last term.

57 Computing With The Thin Lens Equation

The sign convention used here is consistent with most elementary physics books. But conflicts with the convention we have used for lens matrices. The thin lens equation used here is

$$1/f = 1/d_o + 1/d_i,$$

where f is the focal length, d_o is the object distance and d_i is the image distance. We use a special notation for this simple case that may differ a bit from the more general case. So certain conventions apply to this formula. f is positive for a converging lens and negative for a diverging lens. d_o is positive for a real object, and negative for a virtual object. d_i is positive for a real image, and negative for a virtual image. A converging lens produces a real inverted image for an object located outside of the focal length. A converging lens produces a virtual erect image for an object located inside of the focal length. A diverging lens produces only virtual erect images.

The **Lens Maker's Equation** for the thin lens case may be written as

$$1/f = (n - 1)(1/r_1 + 1/r_2),$$

where r_1 and r_2 are the radii of the lens surfaces, and n is the index of refraction. This comes from the more general lens makers equation

$$\frac{1}{F} = (n - 1)\left(\frac{1}{R_1} - \frac{1}{R_2} + \frac{(n - 1)t}{nR_1R_2}\right),$$

by neglecting the last term because t is small, and changing the sign in front of $1/r_2$. To do this we specify that for the thin lens version of the equation,

the radii of curvature are positive for convex surfaces and negative for concave surfaces.

For a thin lens the focal length f is the distance from the lens to the point where parallel rays are focused. Because the lens is thin, this distance may be measured approximately from a lens surface or from the center of the lens. A focal length is negative for a negative diverging lens, so the virtual focus will be located left of the lens.

Here is a Matlab script called thinlens.m for computing with this formula:

```
The thin lens equation is
% 1/f = 1/do + 1/di
% using the following sign convention:
% f is the focal length, positive for a converging lens and
% negative for a diverging lens
% d0 is the distance of the object from the lens
% di is the distance from the image to the lens, and is
% positive for a real image, negative for a virtual image
% r1 is the radius of the first lens surface.
% r2 is the radius of the second lens surface.
% the supplied data need not satisfy the equation,
% each value is computed from the other two supplied.
f=37.5
do=48
di=102
% fdi, fdo, and ff are computed values of di, do, and f using the
% thin lens equation on the supplied data.
% fdi is computed from do and di
% fdo is computed from di and f
% ff is computed from di and do
%
%display(' Computing the thin lens equations for di, do, f ')
imagedistance=1/(1/f - 1/do)
objectdistance=1/(1/f - 1/di)
focallength=1/(1/do + 1/di)
% The linear magnification is m.
% The object is inverted if the magnification is negative.
%display(' Magnification')
fdi=imagedistance;
fdo=objectdistance;
magnification = -fdi/fdo
```

Example 1 Suppose an object is located 5 cm in front of a positive lens of focal length 7.5 cm. Determine the image location and its magnification. We compute

$$d_i = 1/(1/f - 1/d_o) = -15.$$

Because the sign is negative, the image is virtual. The magnification is proportional to the ratio of the distances,

$$m = \frac{-d_i}{d_o} = 3.$$

Because the magnification is positive, the image is not inverted.

Example 2 A lens has a convex surface of radius 20 cm and a concave surface of radius 35 cm. Let the index of refraction be $n = 1.5$. What is the focal length of the lens?

Computing f from the lens maker's equation we find

$$f = 93.3333 \text{ cm.}$$

The lens is a positive converging lens.

Example 3. Suppose the measured object distance of a lens is measured as 48 cm, and the image distance is measured as 102 cm, what is the focal length and magnification. Using the MatLab script we find that the focal length is

$$f = 32.64,$$

and the magnification is

$$m = -2.8908.$$

The image is inverted and enlarged.

Example 4 Suppose a farsighted person can not focus on an object closer than 50 cm. Suppose glasses are to be prescribed to allow reading type at 25 cm. So we require a lens that will create an image at 50 cm of an object located at at 25 cm. The object and image are on the same side of the lens. So the image at 50 cm is virtual, hence

$$1/f = 1/25 - 1/50.$$

Solving for f we find

$$f = 50\text{cm},$$

or .5 meters. The power of the needed lens is

$$p = 1/f = 1/.5 = 2 \text{ diopters.}$$

Example 5 Suppose a nearsighted person can not focus on objects farther away then 10 meters. What lens should be prescribed so that he can focus on distant objects. So we want an object at infinity to be brought to an image at 10 meters. So $1/d_O = 0$, and because the image must be virtual, $d_i = -10$. Then

$$1/f = 0 + 1/d_i = -1/10.$$

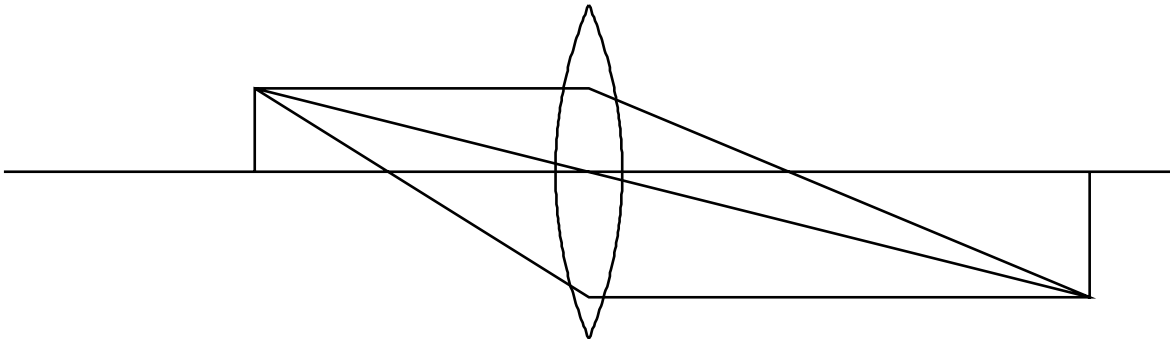


Figure 12: **Computing With the Thin Lens Equation, Example 6.** *The object distance is 2, the image distance 3, the height of the object point is .5, the computed focal length is 1.2, the magnification is -1.5, so -.75 is the height of the image point.*

So the required power is

$$p = 1/f = -.1 \text{ diopters.}$$

A diverging lens is required.

Example 6 Suppose an object point is located at $d_O = 2$ and at a positive distance .5 above the optic axis. Suppose the image plane is located at distance $d_I = 3$. What is the focal length of the lens?. What is the height of the image point?

$$1/f = 1/d_O + 1/d_I,$$

so

$$f = 1.2.$$

The magnification is

$$m = -\frac{d_I}{d_O} = -1.5.$$

So the height of the image point is

$$(.5)(-1.5) = -.75.$$

See the figure for Example 6.

Example 7 Suppose an object point is located at $d_O = 1$ and at a positive distance .25 above the optic axis. Suppose the image plane is located at

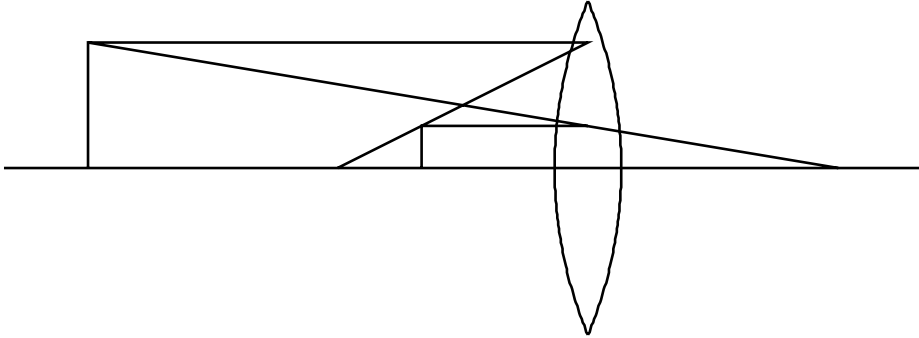


Figure 13: **Computing With the Thin Lens Equation, Example 7.** *The object distance is 1, the image distance -3 (the image is virtual), the height of the object point is .25, the computed focal length is 1.5, the magnification is 3, so .75 is the height of the image point. This is a magnifier, and the relevant magnification is angular magnification.*

distance $d_I = -3$. What is the focal length of the lens?. What is the height of the image point?

$$1/f = 1/d_O + 1/d_I,$$

so

$$f = 1.5.$$

The magnification is

$$m = -\frac{d_I}{d_O} = 3.$$

So the height of the image point is

$$(.25)(3) = .75.$$

See the figure for Example 7.

Example 8 Suppose an object point is located at $d_O = 3$ and at a positive distance .75 above the optic axis. Suppose the image plane is located at distance $d_I = -1$ (virtual image). What is the focal length of the lens?. What is the height of the image point?

$$1/f = 1/d_O + 1/d_I,$$

so

$$f = -1.5,$$

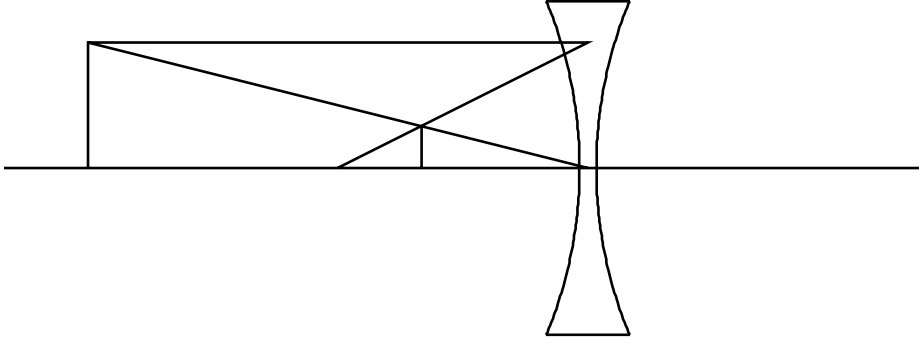


Figure 14: **Computing With the Thin Lens Equation, Example 8.** Suppose the object distance is 3, and the image distance is -1 (the image is virtual). Let the height of the object point be .75. The computed focal length is -1.5 (so the lens is diverging, a negative lens). The magnification is 1/3, so .25 is the height of the image point.

and so the lens is a negative diverging lens. The magnification is

$$m = -\frac{d_I}{d_O} = 1/3.$$

So the height of the image point is

$$(.75)/(3) = .25$$

See the figure for Example 8.

58 The Single Lens Microscope

A single lens microscope is a magnifying glass with a very small radius and very small focus. Such microscopes are capable of producing magnifications on the order of 200 times. A typical diameter of a spherical single lens is about 2 millimeters. So suppose the radius of the spherical lens is .6 mm, and the index of refraction is $n = 1.5$. Then $R_1 = .6$ mm, $R_2 = -.6$ mm, and $t = 1.2$ mm. Then using the lens makers equation

$$\frac{1}{f} = (n - 1)\left(\frac{1}{R_1} - \frac{1}{R_2} + \frac{(n - 1)t}{nR_1R_2}\right),$$

we find the focal length in mm to be

$$f = 1.2,$$

and the angular magnification to be

$$m = 250/1.2 = 208.33.$$

We have taken the near point of the eye to be $d_e = 250$ mm, and the angular magnification to be

$$m = \frac{d_e}{f}.$$

Van Leeuwenhoek made his great discoveries in bacteriology and microscopy using a single lens microscope. See Brian Ford, **the Single Lens**, for more details.

The **Giordano Collection** of single lens microscopes was exhibited at the Linda Hall Library from April 30, 2009 to September 12, 2009. The show was called **Singular Beauty**. The microscopes were from a private collection, first shown at MIT. The exhibition will be moved permanently to Musee des Confluences in Lyon France.

59 Polarized Light

59.1 Circular Polarization

60 Jones Matrices

Jones matrices are used in the analysis of polarized light.

61 Lasers and Einstein's Equation for Stimulated Emission

62 To Quickly Determine Whether a Lens is Positive or Negative

A converging positive lens acts as a magnifier. Thus if one uses a magnifying glass to examine an object one finds that when the magnifying glass is moved

horizontally, the image appears to move opposite to the motion of the glass. This is due to the fact that the image is being magnified. Thus using the same magnifying glass when one holds the magnifying glass fixed but moves the object, the image appears to move in the same direction as the object, but at a faster rate. This is because the object is being magnified. This explains the opposite motion of the image when the magnifier is moved.

If when moving a lens while examining an object, the image appears to move in the same direction as the lens, then the lens is a negative diverging lens.

A negative lens is used to correct nearsightedness, while a positive lens is used to correct farsightedness. Thus this test can be used to determine the type of corrective eyeglass lens.

63 Holography

64 Optics Bibliography

- [1]Andonouic Ivan, Uttamchandani Deepak, **Modern Optical Systems**, 1989.
- [2]Bass Michael, **Handbook Of Optics**, Volumes I And II, Optical Society Of America 2nd Ed 1995, JCCC.
- [3]Born Max, Wolf Emil, **Principles Of Optics**, 5th Ed. 1975, Emery library.
- [4]Chandrakumar Narayanan, **Modern Techniques In High Resolution FT-NMR**, 1987 Linda Hall, Fourier Optics.
- [5]Collier R J, **Optical Holography**, AlliedSignal.
- [6]Fowler, **Modern Optics**, Emery library.
- [7]Meyer-Arendt Jurgen R, **Introduction to Classical and Modern Optics**, 2nd Edition, 1984. The nodal slide method is discussed on page 77. Treats quantum optics, Fourier optics, Lasers, Relativistic optics, and Radiometry, Emery library.
- [8]Goodman Joseph W, **Introduction To Fourier Optics**, Linda 1968 287 Pages.
- [9]Jenkins, White, **Fundamentals Of Optics**, Emery Library.
- [10]Kingslake Rudolf, **Optical System Design**, 1983.
- [11]Lipson S G, Lipson H, Tannhauser D S, **Optical Physics**, 3rd Ed 1995 Cambridge, Matrix Formalism Lenses.

- [12]Mandel Leonard, Wolf Emil, **Optical Coherence And Quantum Optics**, 1995 49.95.
- [13]Morgan Joseph, **Introduction To Geometrical And Physical Optics**, 1953, Emery library.
- [14]Nussbaum Allen, Phillips Richard A, **Contemporary Optics For Scientists And Engineers**, 1976 Prentice-Hall Emery library.
- [15]Ogle Kenneth, **Optics: An Introduction For Ophthalmologists**, Charles Thomas Pub. Springfield Ill 2nd Ed 1968, Emery library.
- [16]Pankaj K Das, **Optical Signal Processing**, Ta1632.d337 1991 Linda Acousto-optic Interaction, Gaussian Beam Propagation, Ultrasound Hamilton-cayly Theorem Etc.
- [17]Sliney David, Wolbarsht Myron, **Safety With Lasers And Other Optical Sources**, Plenum Press 1980 Tk1677 .s44 Alliedsignal.
- [18]Smith Warren J, **Modern Lens Design**, OPTICS TOOLBOX, 1992 64.00.
- [19]Smith Warren J, **Modern Optical Engineering**, 2nd Ed., 1990.
- [20]Smith Warren J, **Modern Optical Engineering**, 1st edition 1966, Emery library.
- [21]Stark Henry, **Applications Of Optical Fourier Transformations**, Ta1632 .a68 1982, Linda Hall.
- [22]Sybil Parker, **Optics Source Book**, Raytown and Emery libraries.
- [23]Walther Adriaan, **The Ray And Wave Theory Of Lenses**, 1995, Fourier Optics, Linda Hall Library.
- [24]Wilson Raymond, **Fourier Series And Optical Transforms Techniques In**, 1995 Linda Fourier Optics
- [25]Ronchi Vasco, **Optics, The Science of Vision**, Dover reprint of 1957 edition, Controversial theories, Physiology, Emery library.
- [26]Rossi Bruno, **Optics**, Addison-Wesley, 1956, Emery Library.
- [27]Gerrard A., Burch J.M., **Introduction to Matrix Methods in Optics**, Dover, 1994 reprint of 1976 edition, Emery Library. This book has matrix notation that agrees with the notation in this document in that the height is the first element of the vector and the angle the second element. Some of the other books that treat the lens matrix have this order reversed so that the rows of the matrix are interchanged. This book may be the best reference for this document.