# Chapter 1

# Geometric classes

Geometric classes are represents objects in three-dimensional space. These include points, directions, and 3D volumes. Points are implemented as `Eigen::Vector3d` objects (the `3d` here means the vector contains 3 `doubles`). Directions are represented by the `UnitVector3d` class, which encapsulates a normalized `Eigen::Vector3d` member. In this section, `Point` is an alias for `Eigen::Vector3d` and `Direction` is an alias for `UnitVector3d`.

3D volumes are derived from the abstract `Shape` class. They must have the following functions overridden in order to be instantiated:

1. `xMin()`, `xMax()`, and so on: locate the extreme points in each of the three Cartesian coordinates.
2. `surfaceArea()`
3. `volume()`
4. `surfaceContains(const Point&)`: checks if the Point is strictly on the surface.
5. `encloses(const Point&)`: checks if the point is strictly inside `*this` (the `Shape` in question). This means the point is not on the surface or outside.
6. `overlaps(const Shape&)`: checks if the two shapes have a positive volume of intersection. That means if the intersection is a point, a line, or a plane, it returns `false`.
7. `distanceToSurface(const Point&, const Direction&)`: calculates the smallest, positive distance that a point has to travel to reach the surface.
    - If the point is inside the shape, the distance equation will have positive and negative roots, and only the positive root is returned.
    - If the point is on the surface, returns 0 if it leaves the surface, and the distance to the other end if it enters the surface.

- If the point is outside the shape, returns NAN if it never enters the shape, else the smallest distance to the surface.

8. `normal(const Point&)`: checks if the point is on the surface, and returns the outward normal vector if it is.

9. `print(std::ostream& os)`: outputs the shape into `std::cout`.

`Shape` also has an overloaded `encloses(const Shape&)` that checks if the other shape is completely contained within `*this`. This function evaluates to `true` even if the two `Shape`s intersect on the surface. Because of the tree design, this is made a function within `Shape`, but will always throw an exception unless `*this` is of type `Box`, and practically enclosure check is always done on `Box` objects.

## 1.1 Box

The `Box` class represents an axis-aligned rectangular prism, where its edges align with the axes in Cartesian coordinates. It is defined by a lower vertex $\boldsymbol{p}_0$ and upper vertex $\boldsymbol{p}_1$. The edge lengths are $a = p_{x1} - p_{x0}$, $b = p_{y1} - p_{y0}$, and $c = p_{z1} - p_{z0}$, conventially called the length, width, and height, respectively.

### 1.1.1 Extreme points

The extreme points are $p_{x0}$ and $p_{x1}$ in the x-direction, $p_{y0}$ and $p_{y1}$ in y, and $p_{z0}$ and $p_{z1}$ in z.

### 1.1.2 Surface area

The surface area is $2(ab + ac + bc)$.

### 1.1.3 Volume

The volume is $abc$.

### 1.1.4 Checking if a point on the surface

A point $\boldsymbol{r}$ is on the one of its six faces if one coordinate is equal to the minimum or maximum coordinate of the box, and the other two coordinates are not outside of their respective extrema. For example, if $\boldsymbol{r}$ is on the face defined by $x = p_{x0}$, then $x = p_{x0}$, $p_{y0} \leq y \leq p_{y1}$, and $p_{z0} \leq z \leq p_{z1}$.

### 1.1.5 Checking if a point is inside

A point $\boldsymbol{r}$ is inside the box if all coordinates are strictly in between their respective extrema. Mathematically, $p_{x0} < x < p_{x1}$, $p_{y0} < y < p_{y1}$, and $p_{z0} < z < p_{z1}$.

### 1.1.6 Checking if another volume overlaps

#### Box

Two boxes overlap with for all three coordinates, the maximum value of one box is greater than the minimum value of the other.

#### Sphere

Refer to Section 1.2.6.

#### Ellipsoid

Refer to Section 1.3.6.

### 1.1.7 Distance to surface

Given a point $\boldsymbol{r}$ on the box surface with direction $\hat{\boldsymbol{\omega}}$, it is travelling away from the box if any of the following conditions is true:

1. $x = p_{x0}$ and $\Omega_x \leq 0$,
2. $x = p_{x1}$ and $\Omega_x \geq 0$,
3. $y = p_{y0}$ and $\Omega_y \leq 0$,
4. $y = p_{y1}$ and $\Omega_y \geq 0$,
5. $z = p_{z0}$ and $\Omega_z \leq 0$,
6. $z = p_{z1}$ and $\Omega_z \geq 0$.

The distance to surface should be 0 in these cases according to the function definition.

Otherwise, the distance $s$ to the plane (not the box face) $x = p_{x0}$ is

$$s = \frac{x - p_{x0}}{\Omega_x}, \tag{1.1}$$

and similar formulae can be applied to the other planes containing a box face.

Even if negative or undefined values of $s$ are neglected, the smallest positive value of $s$ is still not guaranteed to be the distance to the box surface. This is because the calculated $s$ is the distance to the plane containing a face, not the face itself. An additional check to ensure that after traveling a distance $s$ to reach a plane, the remaining two coordinates are still in bound. For example, if $s$ is the distance to the plane $x = p_{x0}$, the additional checks are

$$p_{y0} \leq y + \Omega_y s \leq p_{y1} \tag{1.2}$$

$$p_{z0} \leq z + \Omega_z s \leq p_{z1} \tag{1.3}$$

The smallest positive value of $s$ that also ensures the point is still in bound is the distance to the box surface.

### 1.1.8 Outward normal vector

For a point $\boldsymbol{r}$ on the box surface, the normal vector is undefined if $\boldsymbol{r}$ lies on a vertex or an edge (the intersection of multiple faces). Otherwise, the normal vector is only dependent on which face the point lies:

1. If $x = p_{x0}$, $\hat{\boldsymbol{n}} = -\hat{\boldsymbol{\imath}}$
2. If $x = p_{x1}$, $\hat{\boldsymbol{n}} = +\hat{\boldsymbol{\imath}}$
3. If $y = p_{y0}$, $\hat{\boldsymbol{n}} = -\hat{\boldsymbol{\jmath}}$
4. If $y = p_{y1}$, $\hat{\boldsymbol{n}} = +\hat{\boldsymbol{\jmath}}$
5. If $z = p_{z0}$, $\hat{\boldsymbol{n}} = -\hat{\boldsymbol{k}}$
6. If $z = p_{z1}$, $\hat{\boldsymbol{n}} = +\hat{\boldsymbol{k}}$

### 1.1.9 Checking if another volume is inside

This function is only defined for the `Box` class, which will be useful in a few tree functions. Given a volume with calculated extreme points $\boldsymbol{r}_0$ and $\boldsymbol{r}_1$, respectively, the box will completely enclose this volume if the following 6 conditions are true

1. $p_{x0} \leq x0$
2. $p_{y0} \leq y0$
3. $p_{z0} \leq z0$
4. $p_{x1} \geq x1$
5. $p_{y1} \geq y1$
6. $p_{z1} \geq z1$

## 1.2 Sphere

The `Sphere` class requires information of the center, $\boldsymbol{r}_0 = [x_0, y_0, z_0]^T$ and of the radius $R$. In Cartesian coordinates, the sphere surface is described by the implicit equation:

$$(x - x_0)^2 + (y - y_0)^2 + (z - z_0)^2 = R^2, \tag{1.4}$$

or in vector terms,

$$\|\boldsymbol{r} - \boldsymbol{r}_0\|^2 = R^2. \tag{1.5}$$

### 1.2.1 Extreme points

The $x$-extrema are $x_0 \pm R$, the $y$-extrema are $y_0 \pm R$, and $x$-extrema are $z_0 \pm R$.

### 1.2.2 Surface area

The surface area is $4\pi R^2$.

### 1.2.3 Volume

The volume is $\frac{4}{3}\pi R^3$.

### 1.2.4 Checking if a point on the surface

A point with coordinates $\boldsymbol{r}$ is on the sphere surface if it satisfies Equation 1.5.

### 1.2.5 Checking if a point is inside

A point with coordinates $\boldsymbol{r}$ is inside the sphere if $\|\boldsymbol{r} - \boldsymbol{r}_0\|^2 < R^2$.

### 1.2.6 Checking if another volume overlaps

#### Sphere

If the other sphere is centered at $\boldsymbol{r}_1$ and has radius $R_1$, then it intersects with the current sphere if

$$\|\boldsymbol{r}_0 - \boldsymbol{r}_1\| < R + R_1, \tag{1.6}$$

in other words, the two spheres intersect if the line joining their centers is shorter than the sum of the two radii.

#### Box

A box with lower vertex $\boldsymbol{p}_0 = [p_{x0}, p_{y0}, p_{z0}]^T$ and upper vertex $\boldsymbol{p}_1 = [p_{x1}, p_{y1}, p_{z1}]^T$ overlaps with the sphere if

$$\|\boldsymbol{r}_0 - \boldsymbol{p}\| < R, \tag{1.7}$$

where $\boldsymbol{p}$ is defined as

$$\boldsymbol{p} = \begin{bmatrix} \max(p_{x0}, \min(x_0, p_{x1})) \\ \max(p_{y0}, \min(y_0, p_{y1})) \\ \max(p_{z0}, \min(z_0, p_{z1})) \end{bmatrix} \tag{1.8}$$

#### Ellipsoid

Refer to Section 1.3.6.

### 1.2.7 Distance to surface

Given a point $\boldsymbol{r}$ travelling at direction $\hat{\boldsymbol{\omega}}$, the distance $s$ to the sphere surface satisfies the equation

$$\|\boldsymbol{r} + \hat{\boldsymbol{\omega}} s - \boldsymbol{r}_0\|^2 = R^2, \tag{1.9}$$

which is a quadratic equation in $s$. It has the explicit form:

$$s^2 + 2Bs + C = 0, \tag{1.10}$$

where the coefficients $B$ and $C$ are defined as

$$B = (\boldsymbol{r} - \boldsymbol{r}_0) \cdot \hat{\boldsymbol{\omega}} \tag{1.11}$$

$$C = \|\boldsymbol{r} - \boldsymbol{r}_0\|^2 - R^2. \tag{1.12}$$

### 1.2.8 Outward normal vector

The outward normal vector of a point $\boldsymbol{r}$ on the sphere surface is $\dfrac{\boldsymbol{r} - \boldsymbol{r}_0}{R}$.

## 1.3 Ellipsoid

The `Ellipsoid` class represents a shape centered at $\boldsymbol{r_0}$, of which the surface is given as

$$(\boldsymbol{r} - \boldsymbol{r}_0) \cdot \boldsymbol{M}(\boldsymbol{r} - \boldsymbol{r}_0) = 1, \tag{1.13}$$

where $\boldsymbol{M}$ is a symmetric positive definite (SPD) matrix in $\mathbb{R}^3$ that holds the orientation and magnitude of the ellipsoid. Its unit eigenvectors – denoted as $\hat{\boldsymbol{e}}_0$, $\hat{\boldsymbol{e}}_1$, and $\hat{\boldsymbol{e}}_2$ – form an orthonormal basis that corresponds to the direction of the three primary axes. The corresponding eigenvalues are the inverse square of the respective semi-axis. In other words, if $a$, $b$, and $c$ are the semi-axes, and $\lambda_i$ is the $i^{\text{th}}$ eigenvalue of $\boldsymbol{M}$, then $\lambda_0 = \dfrac{1}{a^2}$, $\lambda_1 = \dfrac{1}{b^2}$, and $\lambda_2 = \dfrac{1}{c^2}$.

If the three axes are aligned to Cartesian coordinates, then Equation 1.13 can be expanded as

$$\left(\frac{x - x_0}{a}\right)^2 + \left(\frac{y - y_0}{b}\right)^2 + \left(\frac{z - z_0}{c}\right)^2 = 1, \tag{1.14}$$

and the matrix $\boldsymbol{M}$ is

$$\boldsymbol{M} = \begin{bmatrix} \frac{1}{a^2} & 0 & 0 \\ 0 & \frac{1}{b^2} & 0 \\ 0 & 0 & \frac{1}{c^2} \end{bmatrix}. \tag{1.15}$$

Such ellipsoid is called axis-aligned, similar to the `Box` class.

A sphere can also be represented by the `Ellipsoid` class, in which case $a = b = c = R$, and $\boldsymbol{M} = \dfrac{1}{R^2}\boldsymbol{I}$.

### 1.3.1 Extreme points

This is a constrained optimization problem. Take the $x$ coordinate for example, the problem is to maximize or minimize $f(\boldsymbol{r}) = x = \hat{\boldsymbol{\imath}} \cdot \boldsymbol{r}$ subject to the constraint in Equation 1.13, i.e. $g(\boldsymbol{r}) = (\boldsymbol{r} - \boldsymbol{r}_0) \cdot \boldsymbol{M}(\boldsymbol{r} - \boldsymbol{r}_0) = 1$. This problem is solvable through the Lagrange multipliers method, where $\nabla f$ and $\nabla g$ are required:

$$\nabla f(\boldsymbol{r}) = \hat{\boldsymbol{\imath}} \tag{1.16}$$

$$\nabla g(\boldsymbol{r}) = 2\boldsymbol{M}(\boldsymbol{r} - \boldsymbol{r}_0). \tag{1.17}$$

The following a system of 4 equations is formulated to solve for $\boldsymbol{r}$ and $\lambda$:

$$\begin{aligned} 2\lambda \boldsymbol{M}(\boldsymbol{r} - \boldsymbol{r}_0) &= \hat{\boldsymbol{\imath}} \\ g(\boldsymbol{r}) &= 1. \end{aligned} \tag{1.18}$$

Equation 1.18 is used to solve for $\boldsymbol{r}$ in terms of $\lambda$:

$$\boldsymbol{r} = \boldsymbol{r}_0 + \frac{1}{2\lambda}\boldsymbol{M}^{-1}\hat{\boldsymbol{\imath}}, \tag{1.19}$$

and this expression can be substituted back to the constraint in Equation 1.13 to solve for $\lambda$:

$$\frac{1}{2\lambda}\boldsymbol{M}^{-1}\hat{\boldsymbol{\imath}} \cdot \frac{1}{2\lambda}\boldsymbol{M}\boldsymbol{M}^{-1}\hat{\boldsymbol{\imath}} = 1 \tag{1.20}$$

$$\frac{1}{4\lambda^2}\hat{\boldsymbol{\imath}} \cdot \boldsymbol{M}^{-1}\hat{\boldsymbol{\imath}} = 1 \tag{1.21}$$

$$\lambda = \frac{\pm 1}{2}\sqrt{\hat{\boldsymbol{\imath}} \cdot \boldsymbol{M}^{-1}\hat{\boldsymbol{\imath}}}. \tag{1.22}$$

There are two possible values for $\lambda$, as expected, as the minimum and maximum $x$ points are collinear with the ellipsoid center. Subtituting the expression for $\lambda$ in Equation 1.22 back to Equation 1.19 to obtain

$$\boldsymbol{r} = \boldsymbol{r}_0 \pm \frac{\boldsymbol{M}^{-1}\hat{\boldsymbol{\imath}}}{\sqrt{\hat{\boldsymbol{\imath}} \cdot \boldsymbol{M}^{-1}\hat{\boldsymbol{\imath}}}}, \tag{1.23}$$

the objective function $f(\boldsymbol{r}) = \hat{\boldsymbol{\imath}} \cdot \boldsymbol{r}$ can finally be optimized:

$$x = \hat{\boldsymbol{\imath}} \cdot \boldsymbol{r} = x_0 \pm \sqrt{\hat{\boldsymbol{\imath}} \cdot \boldsymbol{M}^{-1}\hat{\boldsymbol{\imath}}} \tag{1.24}$$

Since $\boldsymbol{M}$ is required to be SPD, its inverse can be easily computed. The eigendecomposition of $\boldsymbol{M}$, which has to be carried out regardless in order to obtain the principal axes and their respective length, is

$$\boldsymbol{M} = \boldsymbol{E}\boldsymbol{\Lambda}\boldsymbol{E}^T, \tag{1.25}$$

and from there the inverse matrix $\boldsymbol{M}^{-1}$ is

$$\boldsymbol{M}^{-1} = \boldsymbol{E}\boldsymbol{\Lambda}^{-1}\boldsymbol{E}^T, \tag{1.26}$$

where $\mathbf{\Lambda}^{-1}$ is a diagonal matrix whose entries are the reciprocal of those in $\mathbf{\Lambda}$. In terms of the semi-axes, the entries of $\mathbf{\Lambda}^{-1}$ are $a^2$, $b^2$, and $c^2$.

For the purpose of finding the extreme points, the entire inverse matrix is not needed. Note that $\hat{\boldsymbol{\imath}} \cdot \boldsymbol{M}^{-1}\hat{\boldsymbol{\imath}}$ is simply the first-row, first-column entry of $M^{-1}$. Using the decomposition in Equation 1.26, this entry can be obtained by:

$$\begin{aligned}\hat{\boldsymbol{\imath}} \cdot \boldsymbol{M}^{-1}\hat{\boldsymbol{\imath}} &= \hat{\boldsymbol{\imath}} \cdot \boldsymbol{E}^T\mathbf{\Lambda}^{-1}\boldsymbol{E}\hat{\boldsymbol{\imath}} \\ &= \left(\boldsymbol{E}^T\hat{\boldsymbol{\imath}}\right) \cdot \mathbf{\Lambda}^{-1}\boldsymbol{E}^T\hat{\boldsymbol{\imath}} \\ &= a^2E_{0,0}^2 + b^2E_{0,1}^2 + c^2E_{0,2}^2\end{aligned}$$

Similar approaches can be carried out to find the minima and maxima in the other coordinates. The results are as follows:

$$x_{\min/\max} = x_0 \pm \sqrt{a^2E_{0,0}^2 + b^2E_{0,1}^2 + c^2E_{0,2}^2} \tag{1.27}$$

$$y_{\min/\max} = y_0 \pm \sqrt{a^2E_{1,0}^2 + b^2E_{1,1}^2 + c^2E_{1,2}^2} \tag{1.28}$$

$$z_{\min/\max} = z_0 \pm \sqrt{a^2E_{2,0}^2 + b^2E_{2,1}^2 + c^2E_{2,2}^2} \tag{1.29}$$

For an axis-aligned ellipsoid, since $E_{i,j} = \delta_{i,j}$, Equations 1.27 through 1.29 do simplify to $x_0 \pm a$, $y_0 \pm b$, and $z_0 \pm c$, respectively.

### 1.3.2 Surface area

There are no analytic forms of an ellipsoid surface area; it generally involves computing elliptic integrals. The exact formulation is given in [3]:

$$S = 2\pi c^2 + \frac{2\pi ab}{\sqrt{\delta}}\left[(1-\delta)F\left(\sqrt{\delta}\,\middle|\,\frac{\epsilon}{\delta}\right) + \delta E\left(\sqrt{\delta}\,\middle|\,\frac{\epsilon}{\delta}\right)\right], \tag{1.30}$$

where $F(x|k)$ and $E(x|k)$ are the incomplete ellpitic integrals of the first and second kind, respectively. The parameters $\delta$ and $\epsilon$ are defined assuming that $c \le b \le a$:

$$\delta = 1 - \frac{c^2}{a^2} \tag{1.31}$$

$$\epsilon = 1 - \frac{c^2}{b^2}. \tag{1.32}$$

The elliptical integrals are undefined when $a = b = c$, since both $\epsilon$ and $\delta$ are in an indeterminate form. Obviously the spherical case is a useful benchmark, so a different integral has to be derived.

The surface of an ellipsoid octant where all coordinates are non-negative can be parametrized as

$$\begin{aligned}\boldsymbol{r}(u,v) &= a\sqrt{1-u^2}\cos(v)\hat{\boldsymbol{e}}_0 + b\sqrt{1-u^2}\sin(v)\hat{\boldsymbol{e}}_1 + cu\hat{\boldsymbol{e}}_2 \\ &0 \le u \le 1, 0 \le v \le \frac{\pi}{2}.\end{aligned}$$

Using these parametrizations, the surface area of the entire ellipsoid (which is 8 times that of an octant) can be computed as

$$S = 8abc \int_0^{\frac{\pi}{2}} \int_0^1 \sqrt{\frac{(1-u^2)\cos^2(v)}{a^2} + \frac{(1-u^2)\sin^2(v)}{b^2} + \frac{u^2}{c^2}} du dv \qquad (1.33)$$

### 1.3.3 Volume

The volume is $\frac{4}{3}\pi abc$.

### 1.3.4 Checking if a point on the surface

A point with coordinates $\boldsymbol{r} = [x, y, z]^T$ is on the ellipsoid surface if it satisfies Equation 1.13. For an axis-aligned ellipsoid, Equation 1.13 is simplified to Equation 1.14.

### 1.3.5 Checking if a point is inside

A point with coordinates $\boldsymbol{r}$ is inside the ellipsoid if it satisfies the inequality

$$(\boldsymbol{r} - \boldsymbol{r}_0) \cdot \boldsymbol{M}(\boldsymbol{r} - \boldsymbol{r}_0) < 1.$$

The case of an axis-aligned ellipsoid is defined similarly.

### 1.3.6 Checking if another volume overlaps

#### Ellipsoid and sphere

The collision detection test between two ellipsoids (which can also extend to spheres) is implemented from the algorithm proposed by Gilitschenski [2], with a minor modification (see below). Let $\Delta\boldsymbol{r}$ be the vector connecting the two centers, and $\boldsymbol{A}$ and $\boldsymbol{B}$ be the matrices that define the two ellipsoids. A convex function $K(\lambda) : (0, 1) \to \mathbb{R}$ can be defined as

$$K(\lambda) = 1 - \Delta\boldsymbol{r} \cdot \left(\frac{1}{\lambda}\boldsymbol{A}^{-1} + \frac{1}{1-\lambda}\boldsymbol{B}^{-1}\right)^{-1} \Delta\boldsymbol{r}. \qquad (1.34)$$

The two ellipsoids are non-overlapping if there exists a $\lambda \in (0, 1)$ such that $K(\lambda) \leq 0$. Because of the convexity, it also means that $\min K(\lambda) \leq 0$. In the special case of $\min K(\lambda) = 0$, the region of intersection is a single point, and the two shapes are also considered non-overlapping per the definition of this function. Instead of finding where $\min K$ occurs, this problem is approached as a root-solving problem.

Gilitschenski proposes that instead of finding what the minimum is, a root-finding problem can be used to find if $K(\lambda)$ has two distinct real roots on the interval (0, 1) [2]. This is equivalent to $\min K(\lambda) \leq 0$, since $K(\lambda)$ is a convex function on (0, 1). The algorithm implemented in this code involves finding

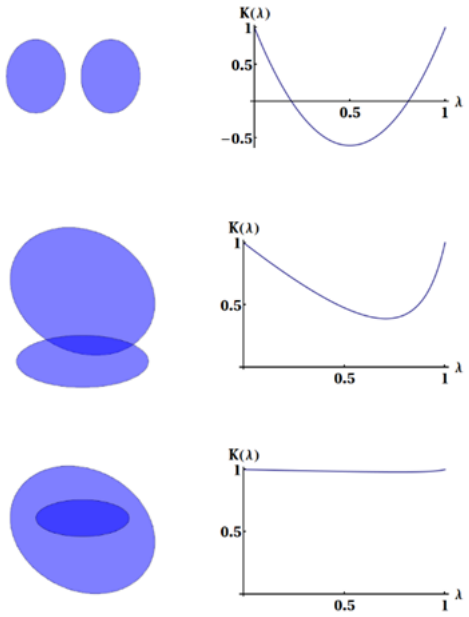


Figure 1.1: Ellipsoid collision test demonstrated $K(\lambda)$ plotted on the right. It can be seen that the two ellipsoids are non-overllaping if and only if $\min K(\lambda) \leq 0$. Figure from [2].

only one of two possible roots, called $\lambda_0$, and if it is found, the two ellipsoids do not overlap. This is also true for the special case of a double root, since the intersection is a single point and the two ellipsoids are still considered non-overlapping.

### Box

The collision detection test between an ellipsoid and a box is implemented from the algorithm proposed by Eberly [1]; however, the edge and face tests described below are slightly modified. The main idea is still to check whether the ellipsoid intersects with the box centroid, if not then any of its vertices, then any of its edges, and finally any of its faces. The point checks can be refered back to Section 1.3.5.

An edge can be represented parametrically by its two endpoints $\boldsymbol{r}_0$ and $\boldsymbol{r}_1$ through the equation $\boldsymbol{e}(t) = \boldsymbol{p}_0 + (\boldsymbol{p}_1 - \boldsymbol{p}_0)t$ for $t \in [0, 1]$. Define $K(t)$ as

$$K(t) = (\boldsymbol{e}(t) - \boldsymbol{r}_0) \cdot \boldsymbol{M}(\boldsymbol{e}(t) - \boldsymbol{r}_0), \tag{1.35}$$

which expands into

$$K(t) = At^2 + 2Bt + C, \tag{1.36}$$

where

$$A = (\boldsymbol{p}_1 - \boldsymbol{p}_0) \cdot \boldsymbol{M}(\boldsymbol{p}_1 - \boldsymbol{p}_0) \tag{1.37}$$
$$B = (\boldsymbol{p}_1 - \boldsymbol{p}_0) \cdot \boldsymbol{M}(\boldsymbol{p}_0 - \boldsymbol{r}_0) \tag{1.38}$$
$$C = (\boldsymbol{p}_0 - \boldsymbol{r}_0) \cdot \boldsymbol{M}(\boldsymbol{p}_0 - \boldsymbol{r}_0). \tag{1.39}$$

Since $\boldsymbol{p}_0$ and $\boldsymbol{p}_1$ are both outside the ellipsoid (otherwise this test would not be needed), $K(0) > 1$ and $K(1) > 1$. If $\boldsymbol{e}(t)$ does intersect the ellipsoid, and is not just tangential to it, there also must exist a $t$ where $K(t) < 1$, and by the Intermediate Value Theorem, the intersections exist and satisfy $K(t) = 1$. In fact, the minimum value of $K(t)$ must be less than 1 in order for the edge to intersect the ellipsoid. This optimization problem is easy to solve for since the function of interest is quadratic. The derivative is given by:

$$\frac{1}{2}\frac{dK}{dt} = At + B = 0. \tag{1.40}$$

It is worth noting that the second derivative of $K(t)$ – $2A$ – is strictly positive since $\boldsymbol{M}$ is an SPD matrix. Therefore, $K(t)$ has a global minimum at the critical point $t = \dfrac{-B}{A}$, $K(t)$. It is also important to check that this critical point is bounded by the domain $[0, 1]$.

In summary, the conditions for an edge to intersect the ellipsoid are as follows

$$-1 \leq \frac{B}{A} \leq 0 \tag{1.41}$$
$$K\left(\frac{-B}{A}\right) < 1 \tag{1.42}$$

Similarly, a face can be defined by three of its four endpoints $\boldsymbol{p}_0$, $\boldsymbol{p}_1$, and $\boldsymbol{p}_2$, where $(\boldsymbol{p}_1 - \boldsymbol{p}_0) \cdot (\boldsymbol{p}_2 - \boldsymbol{p}_0) = 0$. The parametric equation for a face is

$$\boldsymbol{f}(u, v) = \boldsymbol{p}_0 + (\boldsymbol{p}_1 - \boldsymbol{p}_0)u + (\boldsymbol{p}_2 - \boldsymbol{p}_0)v$$
$$(u, v) \in [0, 1]^2.$$

Define

$$K(u, v) = (\boldsymbol{f}(u, v) - \boldsymbol{r}_0) \cdot \boldsymbol{M}(\boldsymbol{f}(u, v) - \boldsymbol{r}_0), \tag{1.43}$$

which expands into

$$K(u, v) = Au^2 + 2Buv + Cv^2 + 2Du + 2Ev + F, \tag{1.44}$$

where

$$A = (\boldsymbol{p}_1 - \boldsymbol{p}_0) \cdot \boldsymbol{M}(\boldsymbol{p}_1 - \boldsymbol{p}_0) \tag{1.45}$$
$$B = (\boldsymbol{p}_1 - \boldsymbol{p}_0) \cdot \boldsymbol{M}(\boldsymbol{p}_2 - \boldsymbol{p}_0) \tag{1.46}$$
$$C = (\boldsymbol{p}_2 - \boldsymbol{p}_0) \cdot \boldsymbol{M}(\boldsymbol{p}_2 - \boldsymbol{p}_0) \tag{1.47}$$
$$D = (\boldsymbol{p}_1 - \boldsymbol{p}_0) \cdot \boldsymbol{M}(\boldsymbol{p}_0 - \boldsymbol{r}_0) \tag{1.48}$$
$$E = (\boldsymbol{p}_2 - \boldsymbol{p}_0) \cdot \boldsymbol{M}(\boldsymbol{p}_0 - \boldsymbol{r}_0) \tag{1.49}$$
$$F = (\boldsymbol{p}_0 - \boldsymbol{r}_0) \cdot \boldsymbol{M}(\boldsymbol{p}_0 - \boldsymbol{r}_0). \tag{1.50}$$

$K(u, v)$ is minimzed if

$$\frac{1}{2}\nabla K = \begin{bmatrix} A & B \\ B & C \end{bmatrix} \begin{bmatrix} u \\ v \end{bmatrix} + \begin{bmatrix} D \\ E \end{bmatrix} = \mathbf{0}, \tag{1.51}$$

which occurs at

$$\begin{aligned} \begin{bmatrix} u \\ v \end{bmatrix} &= - \begin{bmatrix} A & B \\ B & C \end{bmatrix}^{-1} \begin{bmatrix} D \\ E \end{bmatrix} \\ \begin{bmatrix} u \\ v \end{bmatrix} &= \frac{1}{B^2 - AC} \begin{bmatrix} C & -B \\ -B & A \end{bmatrix} \begin{bmatrix} D \\ E \end{bmatrix} \\ \begin{bmatrix} u \\ v \end{bmatrix} &= \frac{1}{B^2 - AC} \begin{bmatrix} CD - BE \\ AE - BD \end{bmatrix} \end{aligned} \tag{1.52}$$

Therefore, the conditions for a face to intersect the ellipsoid are as follows

$$B^2 - AC \neq 0 \tag{1.53}$$

$$0 \leq \frac{CD - BE}{B^2 - AC} \leq 1 \tag{1.54}$$

$$0 \leq \frac{AE - BD}{B^2 - AC} \leq 1 \tag{1.55}$$

$$K\left(\frac{CD - BE}{B^2 - AC}, \frac{AE - BD}{B^2 - AC}\right) < 1 \tag{1.56}$$

If a box does not overlap with the ellipsoid, it has overgone the centroid containment test, 8 vertex containment tests, 12 edge overlap tests, and 6 face overlap tests. Eberly proposes a closest-features testing

### 1.3.7 Distance to surface

Given a point $\boldsymbol{r}$ travelling at direction $\hat{\boldsymbol{\omega}}$, the distance $s$ to surface of the ellipsoid satisfies the equation

$$(\boldsymbol{r} + \hat{\boldsymbol{\omega}} s - \boldsymbol{r}_0) \cdot \boldsymbol{M} (\boldsymbol{r} + \hat{\boldsymbol{\omega}} s - \boldsymbol{r}_0) = 1. \tag{1.57}$$

Equation 1.57 can be written in a quadratic form in $s$:

$$As^2 + 2Bs + C = 0, \tag{1.58}$$

where the coefficients $A$, $B$ and $C$ are defined as

$$A = \hat{\boldsymbol{\omega}} \cdot \boldsymbol{M} \hat{\boldsymbol{\omega}} \tag{1.59}$$

$$B = \hat{\boldsymbol{\omega}} \cdot \boldsymbol{M} (\boldsymbol{r} - \boldsymbol{r}_0) \tag{1.60}$$

$$C = (\boldsymbol{r} - \boldsymbol{r}_0) \cdot \boldsymbol{M} (\boldsymbol{r} - \boldsymbol{r}_0) - 1. \tag{1.61}$$

### 1.3.8 Outward normal vector

Given the explicit form $f(\boldsymbol{r}) = (\boldsymbol{r} - \boldsymbol{r}_0) \cdot \boldsymbol{M}(\boldsymbol{r} - \boldsymbol{r}_0)$, the normal vector at a point $\boldsymbol{r}$ on the surface is parallel to $\nabla f(\boldsymbol{r})$. Therefore,

$$\hat{\boldsymbol{n}}(\boldsymbol{r}) = \frac{\nabla f(\boldsymbol{r})}{\|\nabla f(\boldsymbol{r})\|} = \frac{\boldsymbol{M}(\boldsymbol{r} - \boldsymbol{r}_0)}{\|\boldsymbol{M}(\boldsymbol{r} - \boldsymbol{r}_0)\|} \tag{1.62}$$

For an axis-aligned ellipsoid, Equation 1.62 simplifies to:

$$\hat{\boldsymbol{n}}(\boldsymbol{r}) = \frac{1}{\sqrt{\left(\frac{x-x_0}{a^2}\right)^2 + \left(\frac{y-y_0}{b^2}\right)^2 + \left(\frac{z-z_0}{c^2}\right)^2}} \left[\frac{x - x_0}{a^2}\hat{\boldsymbol{\imath}} + \frac{y - y_0}{b^2}\hat{\boldsymbol{\jmath}} + \frac{z - z_0}{c^2}\hat{\boldsymbol{k}}\right] \tag{1.63}$$

## 1.4 Shape defined by a closed parametric surface

A parametric surface is described by a function $\boldsymbol{r}(u, v) : \mathbb{R}^2 \to \mathbb{R}^3 = x(u, v)\hat{\boldsymbol{\imath}} + y(u, v)\hat{\boldsymbol{\jmath}} + z(u, v)\hat{\boldsymbol{k}}$, defined over $u \in [u_0, u_1]$ and $v \in [v_0, v_1]$. It is assumed that $\boldsymbol{r}(u, v)$ is of class $C^2$, so the partial derivatives $\boldsymbol{r}_u = \dfrac{\partial r}{\partial u}$, $\boldsymbol{r}_v = \dfrac{\partial r}{\partial v}$, $\boldsymbol{H}_x = \nabla \otimes \nabla x$, $\boldsymbol{H}_y = \nabla \otimes \nabla y$, and $\boldsymbol{H}_z = \nabla \otimes \nabla z$ exist. They can be provided by the user or calculated numerically.

Because of the extensive use of derivative and integral operators, repeatedly calculating some properties of similar shapes can be expensive. All parametric surfaces have shared pointer to a common base shape $\boldsymbol{r}(u, v)$, provided that they are defined by a set related parametric equations $\boldsymbol{r}'(u, v) = \boldsymbol{M}\boldsymbol{r}(u, v) + \boldsymbol{r}_0$, where $\boldsymbol{M}$ is an invertible linear transformation matrix in $\mathbb{R}^{3\times3}$, and $\boldsymbol{r}_0$ is a translation vector.

### 1.4.1 Extreme points

Determining the extreme points in any Cartesian direction is a constrained optimization problem, since they must occur at either at the boundary or a critical point within. Take the x-coordinate of a base parametric surface for example, this involves locating the global minimum of $x(u, v)$ on the box interval $[u_0, u_1] \times [v_0, v_1]$.

Because of the low dimensionality of the functions (2), computing the Hessian (of $x(u, v)$, for example) is cheap. The method of choice is the projected Newton's method. Similar to the Levenberg–Marquardt least-square algorithm, an addition of $\alpha\boldsymbol{I}$ (for an $\alpha \geq 0$) to the Hessian may be needed to ensure that it is positive definite. At every iteration, a backtracking line search along the search direction $(\boldsymbol{H} + \alpha\boldsymbol{I})^{-1}\nabla x$ is needed to ensure that the solution stays constrained while satisfying the Armijo condition.

For a transformed shape, the x-component and its derivative functions are given as

$$x'(u,v) = \boldsymbol{M}_0 \boldsymbol{r}(u,v) + x_0 \tag{1.64}$$

$$(\nabla x')^T = \boldsymbol{M}_0 \begin{bmatrix} \boldsymbol{r}_u & \boldsymbol{r}_v \end{bmatrix} \tag{1.65}$$

$$\boldsymbol{H}_{x'} = M_{00}\boldsymbol{H}_x + M_{01}\boldsymbol{H}_y + M_{02}\boldsymbol{H}_z \tag{1.66}$$

where $\boldsymbol{M}_0$ is the row vector corresponding to the first row of $\boldsymbol{M}$.

If $\boldsymbol{M}$ is a scaling matrix that scales the x-coordinate by a scalar $k_x$, then the extrema points of $x'$ are $x'_{\min} = k_x x_{\min} + x_0$ and $x'_{\max} = k_x x_{\max} + x_0$.

Similar procedures can be followed for the y and z coordinates.

### 1.4.2 Surface area

The surface area of a base parametric surface is

$$S = \int_{v_0}^{v_1} \int_{u_0}^{u_1} \|\boldsymbol{r}_u \times \boldsymbol{r}_v\| du dv. \tag{1.67}$$

Rotation and reflection do not alter the surface area. Uniform scaling by a scalar $K$ scales the surface area by $K^2$. Any other transformation will require a re-evaluation of the integral in Equation 1.67 as

$$S = \int_{v_0}^{v_1} \int_{u_0}^{u_1} \|\boldsymbol{M}\boldsymbol{r}_u \times \boldsymbol{M}\boldsymbol{r}_v\| du dv. \tag{1.68}$$

### 1.4.3 Volume

The volume is only defined for a closed surface, since we can take advantage of the Divergence Theorem

$$\iiint_{\Gamma} (\nabla \cdot \boldsymbol{F})\, dV = \oiint_{\partial\Gamma} \boldsymbol{F} \cdot d\boldsymbol{S}. \tag{1.69}$$

Rather than integrating over a volume, the better alternative is to find an $\boldsymbol{F}$ such that $\nabla \cdot \boldsymbol{F} = 1$ and integrate it over the surface. From Equation 1.67, if we follow the convention

$$S = \oiint_{\partial\Gamma} dS = \oiint_{\partial\Gamma} \hat{\boldsymbol{n}} \cdot d\boldsymbol{S}, \tag{1.70}$$

the unit normal vector $\hat{\boldsymbol{n}}$ and the differential surface area $d\boldsymbol{S}$ can be defined as

$$\hat{\boldsymbol{n}} \equiv \frac{\boldsymbol{r}_u \times \boldsymbol{r}_v}{\|\boldsymbol{r}_u \times \boldsymbol{r}_v\|} \tag{1.71}$$

$$d\boldsymbol{S} \equiv (\boldsymbol{r}_u \times \boldsymbol{r}_v)\, du dv. \tag{1.72}$$

The last task to to find a function $\boldsymbol{F}$ such that $\nabla \cdot \boldsymbol{F} = 1$. The most obvious candidates are $\boldsymbol{F} = x\hat{\boldsymbol{\imath}}$, $\boldsymbol{F} = y\hat{\boldsymbol{\jmath}}$, and $\boldsymbol{F} = z\hat{\boldsymbol{k}}$. Choosing the last candidate, the expression for the volume is:

$$\begin{aligned} V &= \int_{v_0}^{v_1} \int_{u_0}^{u_1} z\hat{k} \cdot (\boldsymbol{r}_u \times \boldsymbol{r}_v)\, dudv \\ &= \int_{v_0}^{v_1} \int_{u_0}^{u_1} z \left( \frac{\partial x}{\partial u}\frac{\partial y}{\partial v} - \frac{\partial x}{\partial v}\frac{\partial y}{\partial u} \right) dudv. \end{aligned} \tag{1.73}$$

The volume of a transformed shape is that of the base shape scaled by $|\det \boldsymbol{M}|$.

### 1.4.4 Checking if a point on the surface

A point $\boldsymbol{p}$ is on the parametric surface $\boldsymbol{r}(u, v)$ (regardless of if it's a base or transformed surface) if there exists a pair $(u, v)$ that satisfies the system of equation

$$\boldsymbol{p} - \boldsymbol{r}(u, v) = \boldsymbol{0}. \tag{1.74}$$

This is a non-linear least square problem, since the system has 3 equations but only 2 unknowns. Given the least-square solution $(u^*, v^*)$ to Equation 1.74, then $\boldsymbol{p}$ is on the parametric surface if

$$\|\boldsymbol{p} - \boldsymbol{r}(u^*, v^*)\| = 0. \tag{1.75}$$

In other words, $(u^*, v^*)$ has to satisfy Equation 1.74 exactly (or within a numerical tolerance) rather than merely minimizing the residual.

### 1.4.5 Checking if a point is inside

The concept of "inside" (and by extension "outside") is only well-defined if the surface is closed. For simplicity, the surface is assumed convex. A point $\boldsymbol{p}$ is inside the volume enclosed by the parametric surface if regardless of the direction at which it travels, there is a positive distance to the surface. This problem reduces to two distance calculation problems, one with a direction $\hat{\boldsymbol{\omega}}$ and the other with $-\hat{\boldsymbol{\omega}}$, and checking whether both distances are positive. The choice of $\hat{\boldsymbol{\omega}}$ is arbitrary. Refer to section 1.4.7 for the actual distance calculation.

### 1.4.6 Checking if another volume overlaps

### 1.4.7 Distance to surface

Given a point $\boldsymbol{p}$ travelling at direction $\hat{\boldsymbol{\omega}}$, the distance $s$ to surface of the sphere satisfies the equation

$$\boldsymbol{p} + \hat{\boldsymbol{\omega}} s = \boldsymbol{M}\boldsymbol{r}(u, v) + \boldsymbol{r}_0.$$

This can be arranged into an objective function of three variables:

$$\boldsymbol{F}(u, v, s) = \boldsymbol{p}_0 + \hat{\boldsymbol{\omega}} s - \boldsymbol{M}\boldsymbol{r}(u, v) - \boldsymbol{r}_0, \tag{1.76}$$

and the goal is to solve for the solution vector $\boldsymbol{s} = [u, v, s]^T$ such that $\boldsymbol{F}(\boldsymbol{s}) = \boldsymbol{0}$.

Since the Jacobian of $\boldsymbol{F}$ is known

$$\boldsymbol{J} = \begin{bmatrix} -\boldsymbol{M}\boldsymbol{r}_u & -\boldsymbol{M}\boldsymbol{r}_v & \hat{\boldsymbol{\omega}} \end{bmatrix}, \tag{1.77}$$

Newton's method can be used to solve for $\boldsymbol{s}$. The solution is permissible if $u$ and $v$ are both in bound, and $s \geq 0$.

### 1.4.8 Outward normal vector

The outward normal vector is defined in Equation 1.71.

## 1.5 Shape defined by an implicit function

# Chapter 2

# Tree classes

The motive for using a Tree data structure is to quickly locate which —Shape— object (potentially out of thousands or millions) that a ray of photon will enter next, which is our version of the nearest neighbor search algorithm. A brute-force approach will have a time-complexity of order $\mathcal{O}(N)$, which in itself is not ideal. Furthermore, distance calculations are expensive because of the use of the square-root function (in case of the `Sphere` class) or some other functions that are potentially more expensive. A tree is a hierarchical data structure that groups objects based on certain common features. In this case, an Octree is used to group 3D shapes into each of the 8 octants of equal volume. Using a tree structure, the average time-complexity is down to $\mathcal{O}(\log N)$, which scales extremely well for large datasets. The majority of distance calculations involving those octants are to a surface of a box, and therefore are linear in nature, leaving a very few expensive calculations at the end.

### `BoundingBox`

The `BoundingBox` class is derived from `Box` and represents an axis-aligned bounding box (AABB). It is the building block of an `Octree` and therefore is constrained to have at most 8 children. A `Node` is defined as a pointer to a `Shape` - more specifically, a `std::unique_ptr<Shape>` - that is contained within a `BoundingBox`. The pointed-to `Shape` itself can be either another `BoundingBox` (which means there are further subdivisions within that particular octant) or a terminal shape (which means they are leaf nodes).

### `Octree`

# Chapter 3

# Propogation of light in matter

Light is a manifestation of travelling electric and magnetic field. The two fields oscillate perpendicular to each other and to the direction at which light travels. They both have wavenumber $k$ and frequency $\omega$. Let $x$ be the direction of propogation of the eletric field, the field can be expressed as

$$\mathbf{E}(x,t) = \mathbf{E}_0 \Re\left[e^{i(kx-\omega t)}\right] \tag{3.1}$$

To be precise, the term "wavenumber" refers to the angular wavenumber and is defined as the number of radians per unit length. In a vacuum with no attenuation, it is related to the phase velocity $v_p$ as

$$k = \frac{\omega}{v_p}. \tag{3.2}$$

Since the phase velocity is a fraction $n$ lower than the speed of light in vacuum $c$, the relation between wavenumber and refractive index is

$$k = \frac{\omega n}{c} \tag{3.3}$$

In an unattenuated medium, $k$ is a real number, and the variable $n$ is known as the refractive index. This only holds in vacuum; in any other material, $k$ is a complex number, with its imaginary part accounting for the extent of attenuation. $n$ therefore must also be a complex number. From now on, the variable $n$ refers the real part of $n + i\kappa$, and the imaginary part $\kappa$ is called the extinction coefficient. In other words,

$$k = \frac{\omega}{c}(n + i\kappa) \tag{3.4}$$

To show that attenuation is accounted for by the imaginary component, Equation 3.1 can be expanded as

$$\begin{aligned}\mathbf{E}(x,t) &= \mathbf{E}_0 \Re\left[\exp\left[-\frac{\omega}{c}\kappa x + i\left(\frac{\omega}{c}nx - \omega t\right)\right]\right] \\ &= \mathbf{E}_0 \exp\left(-\frac{\omega}{c}\kappa x\right)\cos\left(\frac{\omega}{c}nx - \omega t\right) \end{aligned} \tag{3.5}$$

Here, the electric field strength falls off exponentially with distance travelled, and the decay extent is characterized by the extinction coefficient. This resembles the linear attenuation of light intensity

$$I(x) = I_0 e^{-\sigma x}, \tag{3.6}$$

and the two equations are in fact related. Since the medium of interest is non-magnetic, $\mathbf{B} \approx \mu\mathbf{H}$, and the magnetic permeability $\mu$ can be readily approximated as its vacuum value $\mu_0 = 4\pi \times 10^{-7}$ H/m. The Poynting vector, defined as

$$\mathbf{S} \equiv \mathbf{E} \times \mathbf{H}, \tag{3.7}$$

can be written in terms of the $\mathbf{B}$ field:

$$\mathbf{S} = \frac{1}{\mu}\mathbf{E} \times \mathbf{B}. \tag{3.8}$$

Thanks to Maxwell's Equations, the $\mathbf{E}$ and $\mathbf{B}$ are in phase in both spatial and temporal domains. In terms of magnitude, the two fields are proportional by a factor of $c/n$, or the wave speed. As a result,

$$\mathbf{S}(x,t) = \frac{n}{\mu c}\|\mathbf{E}_0\|^2 \exp\left(-\frac{2\omega}{c}\kappa x\right)\cos^2\left(\frac{\omega}{c}nx - \omega t\right)\hat{\mathbf{x}}. \tag{3.9}$$

The light intensity is defined as the time-averaged magnitude of the Poynting vector:

$$I(x) = \langle\|\mathbf{S}(x,t)\|\rangle \tag{3.10}$$

$$= \frac{n}{\mu c}\|\mathbf{E}_0\|^2 \exp\left[-\frac{2\omega}{c}\kappa x\right] \cdot \frac{1}{T}\int_0^T \cos^2\left(\frac{\omega}{c}nx - \omega t\right) dt. \tag{3.11}$$

Assume the integrating time $T$ is much larger than the wave period - i.e. $T \gg \dfrac{2\pi}{\omega}$, then the averaging integral in Equation 3.11 will approach the average value over one single period, which is $\dfrac{1}{2}$. Therefore,

$$I(x) = \frac{n}{2\mu c}\|\mathbf{E}_0\|^2 \exp\left[-\frac{2\omega}{c}\kappa x\right]. \tag{3.12}$$

The relation between the attenuation coefficient $\sigma$ and the extinction coefficient $\kappa$ is then

$$\boxed{\sigma = \frac{2\omega\kappa}{c}}. \tag{3.13}$$

# Chapter 4

# Snell's Law and Orientation of the Transmitted Ray

Let $\hat{\boldsymbol{i}}$ be the incident vector, $\hat{\boldsymbol{n}}$ be the (outward) normal vector, and $\hat{\boldsymbol{t}}$ be the transmitted vector. Also the index of refraction of a medium is labeled $n$.

Our job is to find $\hat{\boldsymbol{t}}$, given $\hat{\boldsymbol{i}}$, $\hat{\boldsymbol{n}}$, $n_1$ and $n_2$. The vectors are related through Snell's Law, and in vector notation it reads

$$n_1 \left(\hat{\boldsymbol{i}} \times \hat{\boldsymbol{n}}\right) = n_2 \left(\hat{\boldsymbol{t}} \times \hat{\boldsymbol{n}}\right), \tag{4.1}$$

or

$$\hat{\boldsymbol{t}} \times \hat{\boldsymbol{n}} = \boldsymbol{c}, \tag{4.2}$$

for the new vector $\boldsymbol{c}$ defined as $\boldsymbol{c} = \dfrac{n_1}{n_2}\left(\hat{\boldsymbol{i}} \times \hat{\boldsymbol{n}}\right)$. Note that $\boldsymbol{c}$ and $\hat{\boldsymbol{n}}$ are orthogonal.

We start solving for $\hat{\boldsymbol{t}}$ by decomposing it into two components: one parallel and one orthogonal to $\hat{\boldsymbol{n}}$:

$$\hat{\boldsymbol{t}} = t_{\parallel}\hat{\boldsymbol{n}} + t_{\perp}\hat{\boldsymbol{p}} \tag{4.3}$$

Since the cross product of the parallel component and $\hat{\boldsymbol{n}}$ is the zero vector, we require the cross product of the orthogonal component and $\hat{\boldsymbol{n}}$ be equal to $\boldsymbol{c}$.

$$t_{\perp}\hat{\boldsymbol{p}} \times \hat{\boldsymbol{n}} = \boldsymbol{c} \tag{4.4}$$

Since $\boldsymbol{c}$ and $\hat{\boldsymbol{n}}$ are already orthogonal, one of the possible values for the unit vector $\hat{\boldsymbol{p}}$ is $\hat{\boldsymbol{p}} = \hat{\boldsymbol{n}} \times \hat{\boldsymbol{c}}$ (the other possibility is its additive inverse). Note that the unit vector $\hat{\boldsymbol{c}}$ is $\boldsymbol{c}$ normalized to unit length. Using the vector triple product identity $\boldsymbol{a} \times (\boldsymbol{b} \times \boldsymbol{c}) = (\boldsymbol{a} \cdot \boldsymbol{c})\boldsymbol{b} - (\boldsymbol{a} \cdot \boldsymbol{b})\boldsymbol{c}$, it can be showed that

$$\begin{aligned}
\hat{\boldsymbol{p}} \times \hat{\boldsymbol{n}} &= (\hat{\boldsymbol{n}} \times \hat{\boldsymbol{c}}) \times \hat{\boldsymbol{n}} \\
&= \hat{\boldsymbol{n}} \times (\hat{\boldsymbol{c}} \times \hat{\boldsymbol{n}}) \\
&= (\hat{\boldsymbol{n}} \cdot \hat{\boldsymbol{n}})\hat{\boldsymbol{c}} - (\hat{\boldsymbol{n}} \cdot \hat{\boldsymbol{c}})\hat{\boldsymbol{n}} \\
&= 1\hat{\boldsymbol{c}} - 0\hat{\boldsymbol{n}} \\
\hat{\boldsymbol{p}} \times \hat{\boldsymbol{n}} &= \hat{\boldsymbol{c}}
\end{aligned} \tag{4.5}$$

Substituting the above expression into Equation 4.4, we arrive at the expression for $t_\perp$:

$$\begin{aligned} t_\perp \hat{\boldsymbol{p}} \times \hat{\boldsymbol{n}} &= \boldsymbol{c} \\ t_\perp \hat{\boldsymbol{c}} &= \boldsymbol{c} \\ t_\perp &= \|\boldsymbol{c}\| \end{aligned} \tag{4.6}$$

The parallel component of $\hat{\boldsymbol{t}}$ does not contribute to the cross product, so theoretically $t_\parallel$ can assume any value and Equation 4.2 will hold true. However, since $\hat{\boldsymbol{t}}$ is a unit vector, we require that

$$\begin{aligned} t_\parallel^2 + t_\perp^2 &= 1 \\ t_\parallel &= \pm\sqrt{1 - t_\perp^2} \end{aligned}$$

The choice of sign on $t_\parallel$ now depends on the orientation of the incident vector $\hat{\boldsymbol{i}}$. When the incident ray **enters** a surface, the transmitted vector $\hat{\boldsymbol{t}}$ will point inwards, away from the normal vector. Conversely, when the incident ray **exits** the surface, $\hat{\boldsymbol{t}}$ will point outwards. Therefore,

$$t_\parallel = \begin{cases} -\sqrt{1 - t_\perp^2} & , \hat{\boldsymbol{i}} \cdot \hat{\boldsymbol{n}} < 0 \\ \sqrt{1 - t_\perp^2} & , \hat{\boldsymbol{i}} \cdot \hat{\boldsymbol{n}} > 0 \end{cases}$$

Therefore, the final form of the transmitted vector $\hat{\boldsymbol{t}}$ is

$$\begin{aligned} \hat{\boldsymbol{t}} &= \|\boldsymbol{c}\| \left(\hat{\boldsymbol{n}} \times \hat{\boldsymbol{c}}\right) + \operatorname{sgn}\left(\hat{\boldsymbol{i}} \cdot \hat{\boldsymbol{n}}\right) \sqrt{1 - \|\boldsymbol{c}\|^2}\hat{\boldsymbol{n}} \\ \boldsymbol{c} &= \frac{n_1}{n_2}\left(\hat{\boldsymbol{i}} \times \hat{\boldsymbol{n}}\right) \end{aligned} \tag{4.7}$$

There are two special considerations. First, if $\|\boldsymbol{c}\| > 1$, the radical in Equation 4.7 is complex-valued, and no refraction occurs. This phenomenon is known as total internal reflection. Second, if the incident ray to tangent to the surface it intersects with ($\hat{\boldsymbol{i}} \cdot \hat{\boldsymbol{n}} = 0$), it neither enters or exits the surface. As a result, there is no refraction or reflection.

# Appendix A

# Linear transformation

A linear transformation can be represented by a matrix multiplication. In this project, such matrices are in $\mathbb{R}^{3\times3}$, meaning they operate on an input on $\mathbb{R}^3$ to produce an output also on $\mathbb{R}^3$. This therefore excludes reduction transformations such as projection. A consequence of this is that the transformation matrix is invertible. Translation is not a linear transformation and therefore cannot be represented by a matrix in $\mathbb{R}^{3\times3}$. However, it is easy to account for since a mere translation does not change the original shape's orientation or size.

Every shape defined by a user-input function has a shared pointer to a base shape, related to the actual shape by a transformation $\boldsymbol{M}$ and a translation vector $\boldsymbol{r}_0$. The transformed shape derived from a parametric surface $\boldsymbol{r}(u,v)$ is $\boldsymbol{M}\boldsymbol{r}(u,v)+\boldsymbol{r}_0$, while that derived from an implicit surface $f(\boldsymbol{r})=0$ is $f(\boldsymbol{M}^{-1}(\boldsymbol{r}-\boldsymbol{r}_0))=0$.

## A.1 Classification of linear transformations

### A.1.1 Rigid transformation

A rotation by an angle $\theta$ about a fixed axis $\hat{\boldsymbol{\omega}}$ is represented by the matrix

$$\boldsymbol{R}_{\hat{\boldsymbol{\omega}}}(\theta)=\boldsymbol{I}+\boldsymbol{\Omega}\sin\theta+\boldsymbol{\Omega}^2(1-\cos\theta), \tag{A.1}$$

where $\boldsymbol{\Omega}$ is the anti-symmetric matrix defined as

$$\boldsymbol{\Omega}\equiv\begin{bmatrix}0 & -\omega_z & \omega_y\\ \omega_z & 0 & -\omega_x\\ -\omega_y & \omega_x & 0\end{bmatrix}. \tag{A.2}$$

$\boldsymbol{R}_{\hat{\boldsymbol{\omega}}}$ is an orthogonal matrix ($\boldsymbol{R}_{\hat{\boldsymbol{\omega}}}\boldsymbol{R}_{\hat{\boldsymbol{\omega}}}^T=\boldsymbol{I}$) with unit determinant.

### A.1.2 Reflection

A reflection about the plane that crosses the origin and has unit normal $\hat{\boldsymbol{n}}$ is represented by the matrix

$$\boldsymbol{R}_{\hat{\boldsymbol{n}}} \equiv \boldsymbol{I} - 2\hat{\boldsymbol{n}} \otimes \hat{\boldsymbol{n}}. \tag{A.3}$$

Similar to $\boldsymbol{R}_{\hat{\boldsymbol{\omega}}}$, $\boldsymbol{R}_{\hat{\boldsymbol{n}}}$ is orthogonal, but it has a determinant of -1.

### A.1.3 Scaling

A scaling transformation is represented by the diagonal matrix

$$\boldsymbol{K} \equiv \begin{bmatrix} K_x & 0 & 0 \\ 0 & K_y & 0 \\ 0 & 0 & K_z \end{bmatrix}. \tag{A.4}$$

If $K_x = K_y = K_z = K$, the matrix is simplified to

$$\boldsymbol{K} = K\boldsymbol{I}, \tag{A.5}$$

and the scaling is said to be uniform.

### A.1.4 Shearing

A shearing transformation is represented by the matrix

$$\boldsymbol{\Sigma} \equiv \begin{bmatrix} 1 & \Sigma_{xy} & \Sigma_{xz} \\ \Sigma_{yx} & 1 & \Sigma_{yz} \\ \Sigma_{zx} & \Sigma_{zy} & 1 \end{bmatrix}. \tag{A.6}$$

## A.2 Singular value decomposition

The motivation of applying the SVD onto $\boldsymbol{M}$ is to identify which transformations is $\boldsymbol{M}$ composed of. In particular, if they are only a sequence of rotations, reflections, and uniform scalings, then quite a few properties can be easily computed from those of the base shape.

# Bibliography


[1] David Eberly. *Robust and Error-Free Geometric Computing*. Taylor & Francis Group, LLC, 2020. URL: https://www.geometrictools.com/Documentation/IntersectionBoxEllipsoid.pdf.

[2] Igor Gilitschenski and Uwe D. Hanebeck. "A robust computational test for overlap of two arbitrary-dimensional ellipsoids in fault-detection of Kalman filters". In: *2012 15th International Conference on Information Fusion*. 2012, pp. 396–401.

[3] Garry J. Tee. "Surface Area of Ellipsoid Segment". In: 2005. URL: https://api.semanticscholar.org/CorpusID:119797279.