Computer Science 102
Lab 4

In this lab you will begin the implementation of the sphere*.c* module.

Components that are provided for you include: *main.c, ray.h, rayfuns.h, rayhdrs.h,* and a *makefile.* You must provide your own *camera.c and vector.h* and will build *sphere.c.*

The equation for determining the intersection of a ray with a sphere is derived in the notes. The following parameters determine the point of intersection (or the failure to intersect) .

C = *center of the sphere*
r = *radius of the sphere*
V = *the viewpoint*
D = *ray direction*

We stated that we can simplify the derivation by translating (moving) the coordinate system so that the center of the circle is at the origin. This moves the viewpoint as follows:

$V' = V - C$ = *translated viewpoint*

We showed that if we compute *a, b, and* c as follows:

$$a = D \text{ dot } D$$
$$b = 2\,(V' \text{ dot } D)$$
$$c = V' \text{ dot } V' - r^2$$

then the distance t from the viewpoint to the hitpoint is given by the solution of this quadratic equation.

$$at^2 + bt + c = 0$$

whose solution is the standard form of the quadratic formula:

$$t_h = \frac{-b \pm \text{sqrt}(b^2 - 4ac)}{2a}$$

Recall that quadratic equations may have 0, 1, or 2 real roots depending upon whether the *discriminant:*

$$(b^2 - 4ac)$$

is negative, zero, or positive. These three cases have the following physical implications:

negative => ray doesn't hit the sphere
zero => ray is tangent to the sphere hitting it at one point (we will consider this a miss).
positive => ray does hit the sphere and would pass through its interior(this is the *only* case we consider a *hit).*

Furthermore, the two values of *t* are the distances from the base of the ray to the points(s) of contact with the sphere. We always seek the *smaller* of the two values since we seek to find the “entry wound” not the “exit wound”.

Therefore, the *hits_sphere()* function should return

$$t_h = \frac{-b - \mathrm{sqrt}(b^2 - 4ac)}{2a}$$

if the discriminant is positive and

$$t_h = -1$$

otherwise.

Determining the coordinates of the hit point on a sphere.

The *hits_sphere()* function must also also fill in

- the coordinates of the *hit* and
- a normal vector at the hit point

in the *object_t* structure.

The *(x, y, z)* coordinates are computed as usual.

$$H = V + t_h D$$

Important items to note are:

The actual base of the ray *V* and *not* the translated base *V'* must be used

The vector D must be a *unit vector* in the direction of the ray.

Determining the surface normal at the hit point.

The normal at *any* point *P* on the surface of a sphere is a vector from the *center* to the *point.* Thus

$$N = P - C$$ *(note that N will be a unit vector <==> r = 1)*

Therefore a unit normal may be constructed as follows:

$$N_u = (H - C) / ||(H - C)||$$

The sphere hits function

```
double  sphere_hits(
vec_t    *base,      /* ray base  (the viewpoint) */
vec_t    *dir,       /* ray direction unit vector */
object_t *obj)       /* the sphere object          */
```

This function must:

Determine if the ray hits the sphere in negative z-space

If so, it must

1. store the location of the hit point in the object_t struct and
2. store a unit normal in the object_t structure
3. return the distance from the viewpoint to the hit point

If not, it must

1. return(-1);

In this lab you will submit a compressed tar file named sphere.tar.gz containing all the components needed to build your program.

```
sendlab.102.labsection#  lab#  sphere.tar.gz
```

Since this is lab3 and if you are in section 1 the command you should use is (remember to cd .. because that is where you put your tarfile!)

```
sendlab.102.1 4 sphere.tar.gz
```