Algorithm 4: Intersect

Comp175: Introduction to Computer Graphics – Spring 2016

Due: **Monday March 28th** at 11:59pm

```
Your Names: ________Matt_Asnes___________

            ________Adam_Plumer__________

Your CS Logins: ____masnes01_____________

                ____ap___________________
```

1 Instructions

Complete this assignment only with your teammate. You may use a calculator or computer algebra system. All your answers should be given in simplest form. When a numerical answer is required, provide a reduced fraction (i.e. 1/3) or at least three decimal places (i.e. 0.333). Show all work; write your answers on this sheet. This algorithm handout is worth 3% of your final grade for the class.

2 Generating rays

For this assignment, you need to shoot a ray from the eye point through the center of each pixel.

[1 point] Given a pixel with screen-space coordinates x and y, and the width and height of the screen x_{max} and y_{max}, what is the corresponding point, p_{film}, on the normalized film plane (note that this means that the film plane is at the distance of 1 from the eye point)? Assume that this is taking place after all of the perspective viewing transformations have been applied except for the unhinging transformation. Remember that a pixel y-value of 0 corresponds to the top of the screen.

$$p_{film} = \left(-1 + \frac{2x}{x_{max}}, -1 + \frac{2y}{y_{max}}, -1\right)$$

[1 point] In `Camera`, you transformed a point from world space to screen space by using the normalizing transformation. In `Intersect` and `Ray`, you need to transform p_{film} on the normalized film plane into an untransformed world-space point, p_{world}, by performing the viewing transformation. Using only the components of the viewing transformation (and/or their inverses) write the equation for $M_{film-to-world}$, the composite transformation matrix that transforms p_{film} to p_{world}, such that $p_{world} = M_{film-to-world} * p_{film}$.

$$M_{film-to-world} = T_{uvw}^{-1} R_{uvw2xyz}^{-1} S_{xyz}^{-1}$$

[1 point] Given your eye-point p_{eye} and the world-space point on the normalized film plane p_{world} give the equation for the world-space ray you want to shoot into the scene. Specify your ray in the format $p + t\vec{d}$, where p is a point and $\vec{d}$ is a normalized vector.

$r(t) = p_{eye} + t \cdot (p_{world} - p_{eye})$

3 Plane-Ray Intersection

Write out the plane-ray intersect equation in terms of t. Use the definition of a ray used in question 1, i.e. $p + t\vec{d}$. To get you started you might want to define an intersection point as $(x, y, z) = \langle p_x + \vec{d}_x t, p_y + \vec{d}_y t, p_z + \vec{d}_z t\rangle$, where p is the eyepoint, and $\vec{d}$ is the direction of the ray we are shooting.

For this exercise, show how you would compute the intersection for two sides of the cube, namely the planes where x=0.5 and x=-0.5.

[1 points] **Plane x = 0.5**

$$0.5 = p_x + d_x t$$
$$t = \frac{0.5 - p_x}{d_x}$$

We then check that y and z are between -0.5 and 0.5 to make sure the intersection is with the face of the cube and not just with the plane that that face lies on.

[1 points] **Plane x = -0.5**

$$-0.5 = p_x + d_x t$$
$$t = \frac{-0.5 - p_x}{d_x}$$

We then check that y and z are between -0.5 and 0.5 to make sure the intersection is with the face of the cube and not just with the plane that that face lies on.

4 Cone-Ray Intersection

Special Instructions: For this problem, show all your work and circle, box, or bold your final answers.

Write out both of the cone-ray intersect equations in terms of t. There are two equations: one for the body of the cone, and one for the bottom cap. For your cone, use the same dimensions that you did in `Shapes`. Use the definition of a ray used in question 1, i.e. $p + t\vec{d}$. To get you started you might want to define an intersection point as $(x, y, z) = \langle p_x + \vec{d}_x t, p_y + \vec{d}_y t, p_z + \vec{d}_z t\rangle$, where p is the eyepoint, and $\vec{d}$ is the direction of the ray we are shooting. Looking over the the derivation of the implicit equations for the cylinder in the Raytracing lecture might prove to be useful.

Recall that the equation of a circle on the 2D XZ coordinate plane is $x^2 + z^2 = r^2$. Think of our canonical unit cone as an infinite number of "differential" circles in the XZ plane stacked on top of one another in the Y direction; the bottommost circle has a radius of 1/2 and the topmost circle has a radius of 0. Then the equation of the unit cone is $x_2 + z_2 = k(y)^2$, where k linearly interpolates the radius of the differential circle from 1/2 at the base to 0 at the top (see the lecture notes for a slightly more intuitive equation for the cone).

The intersection points you compute are possible intersection points and need to be examined further (such as the $-0.5 \le y \le 0.5$ restriction for the body of the cylinder in the lecture notes). However for this problem you are NOT required to list these restrictions.

Note that in your program you will need to find intersection points by finding a value for t. If you do not find an explicit formula for t (ie. t =some value(s)) for both the cone and the cap then you will have a very hard time writing the program.

Finally, the equations you write should not use vectors but should be functions of the individual components of the vectors. By reducing your equations after deriving them, you eliminate unnecessary computation and thereby optimize your code before you even write it!

[3 points] **Cone Body** First, we want to check that y is within bounds; $-0.5 \le y \le 0.5$. If we know that this is true, then we can check whether the ray intersects with the infinite cone described by $x^2+z^2 = (apex-y)^2 \cdot \frac{r^2}{height^2}$ within those bounds. In our case $apex = 0.5$, and $height = 1$, so this is $x^2 + z^2 = (0.5 - y)^2 \cdot r^2$. We can substitute the ray's x, y, and z formulations into the cone equation, to get:

$$(p_x+d_xt)^2+(p_z+d_zt)^2 = (0.5-(p_y+d_yt))^2 \cdot r^2$$

$$(p_x^2 + 2p_xd_xt + d_x^2t^2) + (p_z^2 + 2p_zd_zt + d_z^2t^2)$$
$$= (p_y^2 + 2p_yd_yt - p_y + d_y^2t^2 - d_yt + 0.25) \cdot r^2$$
$$= (d_x^2 + d_z^2 - d_y^2r^2)t^2$$
$$+(2p_xd_x + 2p_zd_z - 2p_yd_yr^2 + d_yr^2)t$$
$$+(p_x^2 + p_z^2 - p_y^2r^2 + p_yr^2 - 0.25r^2) = 0$$

Then plugging into the quadratic formula of $At^2 + Bt + C = 0$, with:
$A = d_x^2 + d_z^2 - d_y^2r^2$
$B = 2p_xd_x + 2p_zd_z - 2p_yd_yr^2 + d_yr^2$
$C = p_x^2 + p_z^2 - p_y^2r^2 + p_yr^2 - 0.25r^2$
we get:

$$t = \frac{-B \pm \sqrt{B^2 - 4 \cdot A \cdot C}}{2 \cdot A \cdot C}$$

This should be sufficient to implement this ray intersection in our code.

[2 points] Cone Cap This cone cap is very similar to a plane intersection for the cube case, where we fix y to be a certain value and solve for t: $-0.5 = p_y + d_yt$ $t = \frac{-0.5-p_y}{d_y}$

We then need to check that $x^2 + z^2 = \frac{1}{2}^2$ to ensure the intersection happens in the circle, and not just in the plane the circle resides in.

5 How to Submit

Hand in a PDF version of your solutions using the following command:

```
provide comp175 a4-alg
```

.