

Name: _____ Student ID: _____

- 1) Write down how to test if a 2D point (x, y) is inside the triangle with vertices (x_0, y_0) , (x_1, y_1) , and (x_2, y_2) .

We can evaluate the edge functions (derived from triangle areas):

$$\begin{aligned} F_{01}(x, y) &= (x_1 - x_0)(y - y_0) - (x - x_0)(y_1 - y_0) \\ F_{12}(x, y) &= (x_2 - x_1)(y - y_1) - (x - x_1)(y_2 - y_1) \\ F_{20}(x, y) &= (x_0 - x_2)(y - y_2) - (x - x_2)(y_0 - y_2) \end{aligned}$$

The point is in the triangle if and only if all three edge functions have the same sign.

- 2) Give an **explicit** description of the circle with centre $(0, 0)$ and radius 1.

One example formula is to plug a parameter $\theta \in [0, 2\pi)$ into $(\cos \theta, \sin \theta)$ to get all points on the circle.

- 3) How do you use barycentric coordinates to linearly interpolate colour across a triangle?

If the colours (RGB vectors) at the vertices are \vec{c}_0 , \vec{c}_1 and \vec{c}_2 and the barycentric coordinates for an interior point are α , β and γ , then interpolate the colour as $\alpha\vec{c}_0 + \beta\vec{c}_1 + \gamma\vec{c}_2$.

- 4) Describe how translation transformations can be implemented with matrix multiplication.

Introduce a fourth homogeneous coordinate, normally set to 1, for all points. Then the fourth column of the matrix can be used to implement 3D vector addition i.e. translation. To translate by (a, b, c) with a matrix multiply, use this matrix:

$$\begin{pmatrix} x+a \\ y+b \\ z+c \\ 1 \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 & a \\ 0 & 1 & 0 & b \\ 0 & 0 & 1 & c \\ 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} x \\ y \\ z \\ 1 \end{pmatrix}$$

- 5) Determine a model-view transformation corresponding to a camera with world space coordinates $(0, -3, 3)$ pointing at the origin of world space. You can express this as a sequence of named transformations (e.g. rotate around this axis by this angle) or if you prefer as a 4×4 matrix.

One solution is to first translate by $(0, 3, -3)$ so as to move the camera from world space coordinates $(0, -3, 3)$ to the origin in camera space. Then rotate by 45° around the x -axis in a right-handed way to point the camera down the negative z -axis.

- 6) Write down an expression for a unit-length vector orthogonal to a triangle with vertices (x_0, y_0) , (x_1, y_1) , and (x_2, y_2) .

The question is tricky because the coordinates are 2D. Assuming the 3rd coordinates are assumed to be zero, a unit-length orthogonal vector is $(0, 0, 1)$. More generally in 3D you can use the cross-product of two of the edges, divided by its length.

- 7) Write down how camera x and y coordinates are transformed in a perspective projection.

Name: _____ Student ID: _____

At its most basic, they are just divided by the camera z coordinate to get projected x and y .

8) How could you make a perspective view frustum as close as possible to a given orthographic view volume? (in terms of field of view etc.)

(a diagram is the easiest way to answer this question, but in words, here goes...) An orthographic view volume has parallel sides, but a frustum's sides are at an angle equal determined by the field of view. To get it close, we'd make the field of view as small as possible (it breaks down at zero, so it has to be slightly positive). The camera would have to be very far away from the view volume to be able to match the width/height of the orthographic's near clipping plane.

9) Why do we clip triangles against the near clipping pane before rasterization?

In a perspective projection case, the homogenization that causes the z -divide can flip some vertices that are behind the camera in camera space to being in front of the camera in normalized device coordinates. The rasterizer therefore can get the wrong result if it tries to clip a triangle (say that has one vertex behind the camera and others in front of it) in n.d.c.

10) Write down pseudocode for the Z-buffer algorithm.

Clear the image and set the Z-buffer values all to 1.

For every triangle:

- For every pixel in the clipped bounding box of the projected triangle:
 - Evaluate edge functions to check if it's in the triangle; if not, skip to the next triangle.
 - Get the barycentric coordinates from the edge functions.
 - Interpolate the projected z value from the vertices of the triangle to this pixel.
 - If the z value is less than what's stored in the Z-buffer, overwrite with the colour and z value from this triangle.