

Please do all work on these pages.

1. You are given the following pair of three dimensional line segment endpoints.

Line Segment 1 -  $[5 \ 4 \ 10]$   $[3 \ 7 \ 1]$

Line Segment 2 -  $[2 \ 1 \ 0]$   $[7 \ 8 \ 3]$

- (a) Write parametric equations for each of these line segments.
- (b) Do the lines passing through these segments intersect? If so what is the intersection point? If not, explain *algebraically* why not.
- (c) Give parametric equations for the orthographic projections of these two line segments onto the  $x - z$  plane.
- (d) Do the lines passing through the projected segments intersect? If so, what is the intersection point? If not, explain *algebraically* why not.

2. You have a triangle represented as the following sequence of vertices.

$$\begin{bmatrix} 0 & 0 & 0 \\ 5 & 5 & 5 \\ 5 & 5 & 0 \end{bmatrix}$$

You have adopted the convention that the “front” of the polygon is the side from which the vertices appear to be listed clockwise, and that the normal to the plane of the polygon points toward the front.

(a) Give the equation for the plane passing through this polygon with the normal described above.

(b) Is the point  $[2 \ 4 \ 10]$  in front of the polygon or behind it? Show why *algebraically*, not with a picture.

3. Consider the following matrix.

$$\begin{bmatrix} 0.8 & 0 & -0.6 & 0 \\ 0.4243 & 0.707 & 0.5657 & 0 \\ 0.4243 & -0.707 & 0.5657 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Is this an orthogonal matrix? If it is, give a sequence of transformation matrices for rotations about primary coordinate axes which produce the same result as this matrix. If it is not, explain how you determine that it is not.

4. Suppose you are displaying scenes composed only of translucent polygons in 3 dimensions. You are now working on the design of a hidden-surface algorithm optimized to work rapidly on these scenes. You have already implemented a z-buffer algorithm for opaque polygons, and for the sake of ease of implementation, you are considering whether or not to use a modification of this algorithm for the all-translucent scenes. You are also considering two alternative intensity models for computing the color of a pixel overlapped by one or more translucent polygons. The first is

$$I = I_1 + I_2 + \dots + I_n$$

and the second is

$$I = I_1 + (1/d_1)I_2 + (1/d_2)I_3 + \dots + (1/d_{n-1})I_n$$

where  $I$  is the current intensity,  $I_1$  is the color of the closest of the polygons to your eye,  $I_2$  is the color of the next polygon farther away, and so on, and  $d_i$  is the distance in  $z$  between polygon  $i$  and polygon  $i + 1$  in  $z$  order.

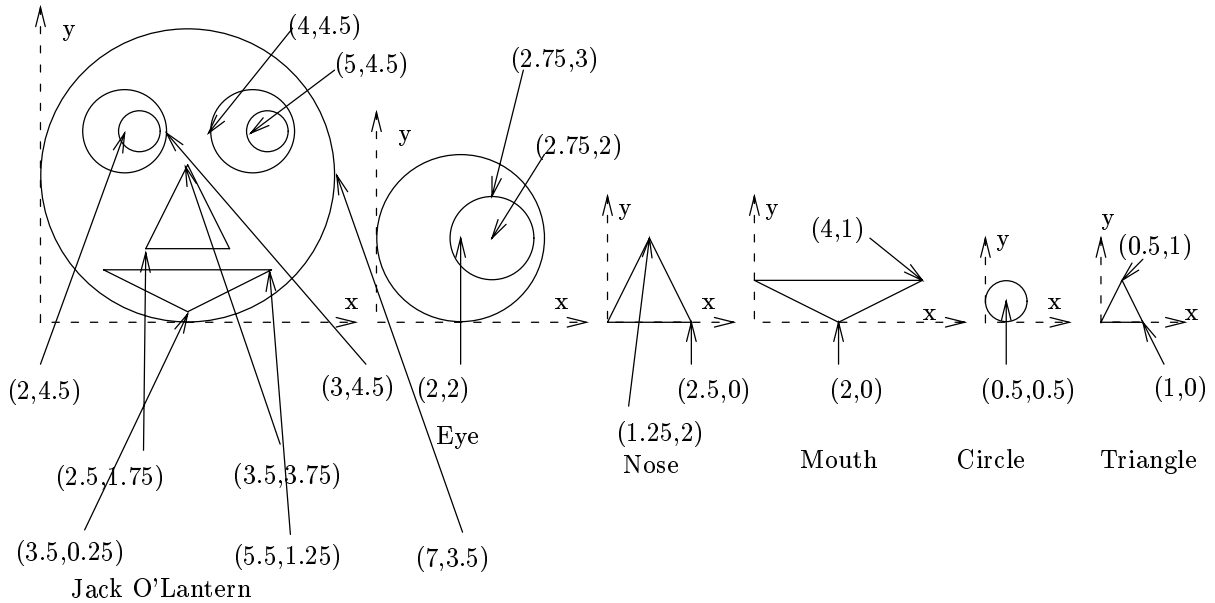
- (a) Using the first intensity model, is there a simple modification of the z-buffer algorithm that will work correctly in computing the intensities of pixels?

- (b) Using the first intensity model, is there any reason to use the z-buffer algorithm or is there an obvious way to speed up the computations at a pixel?

- (c) Answer (a) using the second intensity model.

- (d) Answer (b) using the second intensity model.

5. You are modeling the Jack-o-lantern drawn below hierarchically, using the object definitions given below in terms of their master coordinate systems.



- (a) Draw a directed acyclic graph (DAG) structure for a model which allows both eyes to be rotated **independently** as well as allowing separate transformations of the nose and mouth, all with respect to the face coordinate system. Give the specific parameters of the transformation matrix in all instance transformations in your graph. Only the circle and triangle objects can be leaves of the DAG and each named object above can appear in only a single node of the DAG.
- (b) Draw a DAG which allows the eyes to be modified independently so that they can appear to be different from each other, e.g. one eye can wink while the other eye stays open. This will require changes to the internal structure of the object model for the winking eye. You should define a new object for the winking eye with new instances of circle, and provide the exact instance transformation matrices as in (a). The same constraints on leaves and DAG structure apply as in (a).