CPSC 314
Assignment 2

Due 4PM, Friday, Nov 4, 2005

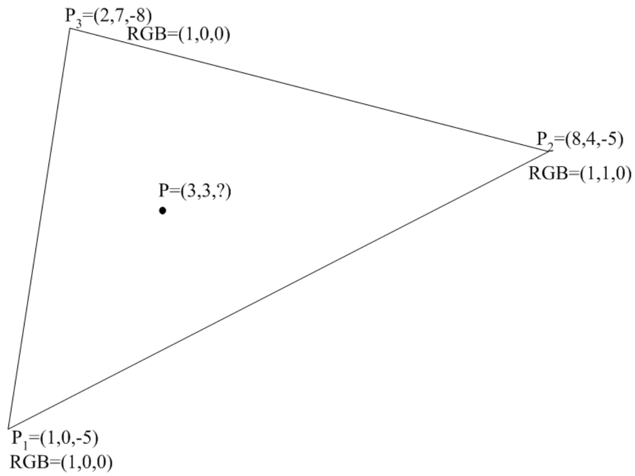
> Answer the questions in the spaces provided on the question sheets. If you run out of space for an answer, use separate pages and staple them to your assignment.

Name: ______________________________

Student Number: ______________________________

Question 1	/ 12
Question 2	/ 3
Question 3	/ 3
Question 4	/ 7
Question 5	/ 5
Question 6	/ 70
TOTAL	/ 100

1. Scan Conversion and Interpolation



(a) (2 points) Give the implicit plane equation for the triangle shown above. I.e., $Ax + By + Cz + D = 0$. Show your work.

First compute the normal N

$$N = \frac{(P_2 - P_1) \times (P_3 - P_1)}{\|(P_2 - P_1) \times (P_3 - P_1)\|}$$

$$N = \begin{pmatrix} -0.2349 \\ 0.4111 \\ 0.8808 \end{pmatrix}$$

This gives is the following coefficients.

$$A = N_x, B = N_y, C = N_z$$

Next we plug in the coordinates for P_1 and solve for D

$$D = -(A * x + B * y + C * z)$$

$$D = 4.639$$

This gives us the equation

$$(-0.2349)x + (0.4111)y + (0.8808)z + 4.639 = 0$$

(b) (1 point) Compute the value of z for point P using your plane equation.

Plug in the x and y coordinates into the plane equation and solve for z

$$A * 3 + B * 3 + C * z + D = 0$$

$$z = -\frac{(A * 3 + B * 3 + D)}{C}$$

$$z = -5.8667$$

(c) (1 point) Compute the barycentric coordinates for point P using the standard areas formula.

The barycentric coordinates for P are (a_1, a_2, a_3) where

$$A = \frac{1}{2} * \|(P_3 - P_1) \times (P_2 - P_1)\| = 25.5441$$

$$a_1 = \frac{\frac{1}{2} * \|(P - P_2) \times (P_3 - P_2)\|}{A} = 0.4667$$
$$a_2 = \frac{\frac{1}{2} * \|(P - P_3) \times (P_1 - P_3)\|}{A} = 0.2444$$
$$a_3 = \frac{\frac{1}{2} * \|(P - P_1) \times (P_2 - P_1)\|}{A} = 0.2889$$

(d) (1 point) Compute the barycentric coordinates for point P using the alternative (bilinear) formula. Show your work.

Though the equations below may look formidable, the idea is very simple. You first interpolate the values along the left and right edges of the triangle that P lies between using only the y coordinates. Then interpolate between these values using only the x coordinates between those edges at that y.

$$a_1 = \frac{c_2}{c_1 + c_2} * \frac{d2}{d_1 + d_2} + \frac{c_1}{c_1 + c_2} * \frac{b_2}{b_1 + b_2} = 0.4667$$

$$a_2 = \frac{c_1}{c_1 + c_2} * \frac{b1}{b_1 + b_2} = 0.2444$$

$$a_3 = \frac{c_3}{c_1 + c_2} * \frac{d1}{d_1 + d_2} = 0.2889$$

Where b_1 and b_2 are the split edge lengths on the edge to the right of P and d_1 and d_2 are the split edge lengths on the edge to the left of P. c_1 and c_2 are the split edge lengths on the horizontal line containing P. So

$$\frac{b_1}{b_1 + b_2} = \frac{P_y - P_{1y}}{P_{2y} - P_{1y}} = 0.75$$

$$\frac{b_2}{b_1 + b_2} = 1 - \frac{b_1}{b_1 + b_2} = 0.25$$

$$\frac{d_1}{d_1 + d_2} = \frac{P_y - P_{1y}}{P_{3y} - P_{1y}} = 0.4286$$

$$\frac{d_2}{d_1 + d_2} = 1 - \frac{d_2}{d_1 + d_2} = 0.5714$$

$$P_R = (P_2 - P_1) * \frac{b_1}{b_1 + b_2} + P_1 = (6.25, 3, -5)$$

$$P_L = (P_3 - P_1) * \frac{d_1}{d_1 + d_2} + P_1 = (1.4286, 3, -6.2857)$$

$$\frac{c_1}{c_1 + c_2} = \frac{P_x - P_{Lx}}{P_{Rx} - P_{Lx}} = 0.3259$$

$$\frac{c_2}{c_1 + c_2} = 1 - \frac{c_1}{c_1 + c_2} = 0.6741$$

(e) (1 point) Compute z and r, g, b for point P using the Barycentric coordinates.

$$z = a_1 * P_{1z} + a_2 * P_{2z} + a_3 * P_{3z} = -5.8667$$

$$r = a_1 * P_{1r} + a_2 * P_{2r} + a_3 * P_{3r} = 1$$

$$g = a_1 * P_{1g} + a_2 * P_{2g} + a_3 * P_{3g} = 0.2444$$

$$b = a_1 * P_{1b} + a_2 * P_{2b} + a_3 * P_{3b} = 0$$

(f) (1 point) Write down the OpenGL perspective matrix for the following frustum dimensions: $right = 2$,$left = -2$,$top = 2$,$bottom = -2$, $near = .1$, $far = 10$.

$$M_p = \begin{pmatrix} \frac{2*near}{right-left} & 0 & \frac{right+left}{right-left} & 0 \\ 0 & \frac{2*near}{top-bottom} & \frac{top+bottom}{top-bottom} & 0 \\ 0 & 0 & \frac{near+far}{near-far} & \frac{2*far*near}{near-far} \\ 0 & 0 & -1 & 0 \end{pmatrix}$$

Therefore

$$M_p = \begin{pmatrix} 0.05 & 0 & 0 & 0 \\ 0 & 0.05 & 0 & 0 \\ 0 & 0 & -1.0202 & -0.2020 \\ 0 & 0 & -1 & 0 \end{pmatrix}$$

(g) (1 point) Compute the coordinates for P_1, P_2, P_3 after the perspective transformation is applied. Show your work.

First transform P_1, P_2, P_3 into homogeneous coordinates. Then multiply each vector by M_P from the previous part. This yields

$$P_1 = \begin{pmatrix} 0.05 \\ 0 \\ 4.899 \\ 5 \end{pmatrix} \qquad P_2 = \begin{pmatrix} 0.4 \\ 0.2 \\ 4.899 \\ 5 \end{pmatrix} \qquad P_3 = \begin{pmatrix} 0.1 \\ 0.35 \\ 7.9596 \\ 8 \end{pmatrix}$$

Notice that the w coordinate is not 1. We must "homogenize" our coordinates by dividing each vector by its w value giving us the following useful x, y, z values.

$$P_1 = \begin{pmatrix} 0.01 \\ 0 \\ 0.9798 \end{pmatrix} \qquad P_2 = \begin{pmatrix} 0.08 \\ 0.04 \\ 0.9798 \end{pmatrix} \qquad P_3 = \begin{pmatrix} 0.0125 \\ 0.0438 \\ 0.995 \end{pmatrix}$$

(h) (1 point) Compute a new plane equation $Ax + By + Cz + d = 0$ for the triangle after the transformation. Show your work.

Given our new coordinates, we repeat the process from part a. We must recompute the normal because the perspective transformation does not preserve normals. This gives us the following equation

$$-(0.1891)x + (0.3309)y - (0.9245)z + 0.9077 = 0$$

(i) (1 point) Compute the value of z for point P using the new plane equation.

We can't just plug in the old points from P. We must compute a new P following the process from part g. Given this new x and y, we then can apply the process from part b to get the z component. This yields

$$z = 0.9858$$

This is the same z coordinate we get from multiplying P by M_P and homogenizing.

(j) (2 points) Compute the barycentric coordinates for point P after transformation.

We now have new coordinates for P_1, P_2, P_3, and P. We can simply apply the same formula from part c to these points. This yields the following barycentric coordinates

$$a1 = 0.3977$$

$$a2 = 0.2083$$

$$a3 = 0.3939$$

2. (3 points) Clipping

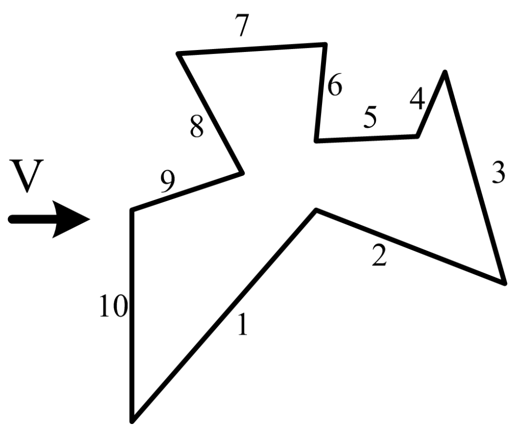
Use the definition of convexity to prove that the intersection of two convex objects is convex.

The definition of convexity is that if for any two points $P_1 \in C$ and $P_2 \in C$, if $(\lambda * P_1 + (1-\lambda) * P_2) \in C$ for $\lambda \in [0, 1]$ then C is convex. We want to show that the intersection C_3 of two convex regions C_1 and C_2 is convex.

For any two points $P_1 \in C_3$ and $P_2 \in C_3$, we know they are both in C_1 and C_2 by definition of intersection. Therefore, since C_1 and C_2 are convex the point $(\lambda * P_1 + (1-\lambda) * P_2)$ for $\lambda \in [0, 1]$ is in both C_1 and C_2. Since it is both these regions, it is also in C_3, therefore C_3 is convex.

3. (3 points) Backface Culling

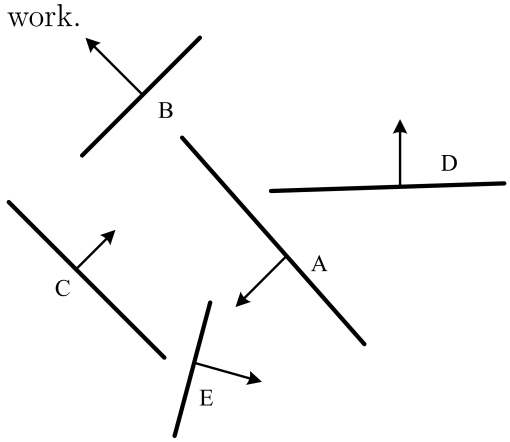
The edges shown below represent faces forming a closed solid. Given the view direction V which faces will be removed by backface culling? Show your work.



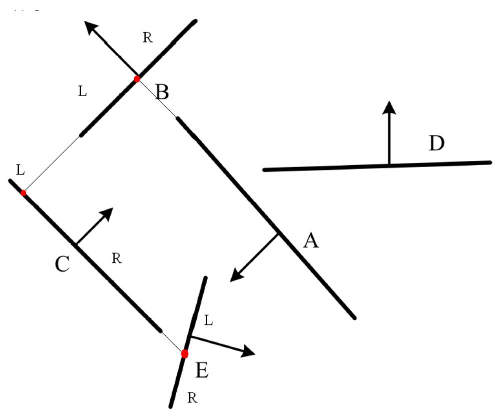
The culled faces will be 1, 3, and 6 because their surface normals point away from the view direction.

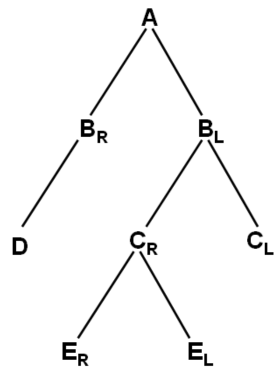
4. BSP Trees

   (a) (4 points) Construct the BSP tree for the segments shown below (use alphabetical insertion order for the segments, when possible). Use the convention where the right subtree (child) is located on the side that the normal points to. Show your work.

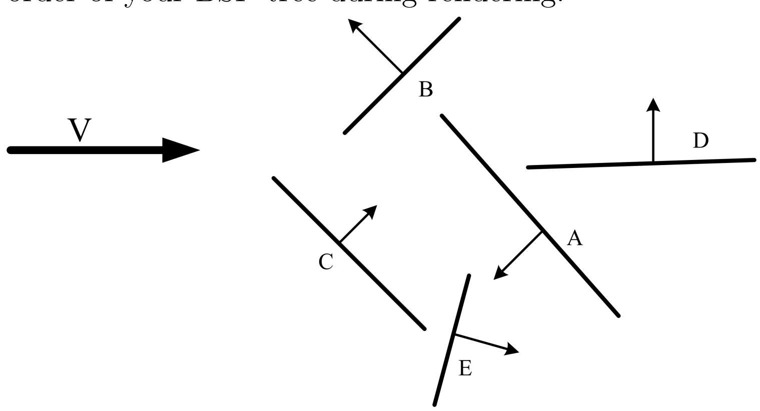


Use the following guide to see how the line segments split the other line segments in two during the construction of the tree.





(b) (3 points) Given the view direction as shown below, describe the complete traversal order of your BSP tree during rendering.



$$(A-)A(A+)$$

$$(B_R-)B_RA(B_L-)B_L(B_L+)$$
$$DB_RA(C_R+)C_R(C_R-)B_LC_L$$
$$DB_RAE_LC_RE_RB_LC_L$$

At any node, we check the view vector agains the surface normal of the node's corresponding surface. If it is negative we traverse to the left and draw when we return, then traverse to the right. If it is positive we traverse to the right and draw when we return, then traverse to the left. If we follow this system with the tree from part a, then we get the following draw ordering.

5. (5 points) Bresenham

Write the Bresenham algorithm for rasterizing a line from (x_1, y_1) to (x_2, y_2) where $x_1 \geq x_2$, $y_2 > y_1$ and $x_1 - x_2 < y_2 - y_1$.

```
y = y1
x = x1
dx = x2 - x1
dy = y2 - y1
d = -2 * dx - dy
W = -2 * dx
NW = -2 * (dx + dy)
while(y < y2)
        if(d ≤ 0){
                d = d + W
                y = y + 1
        }
        else{
                d = d + NW
                y = y + 1
                x = x - 1
        }
        SetPixel(x, y)
}
```