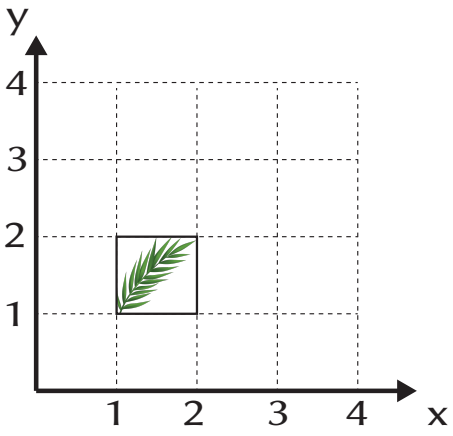
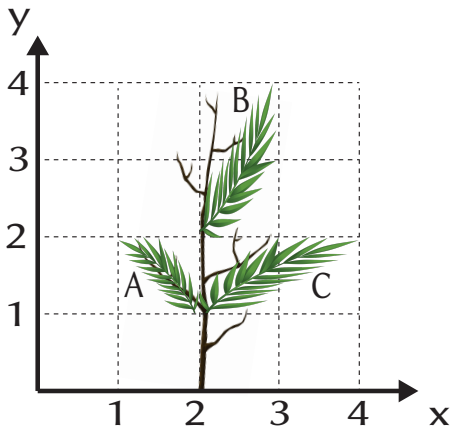
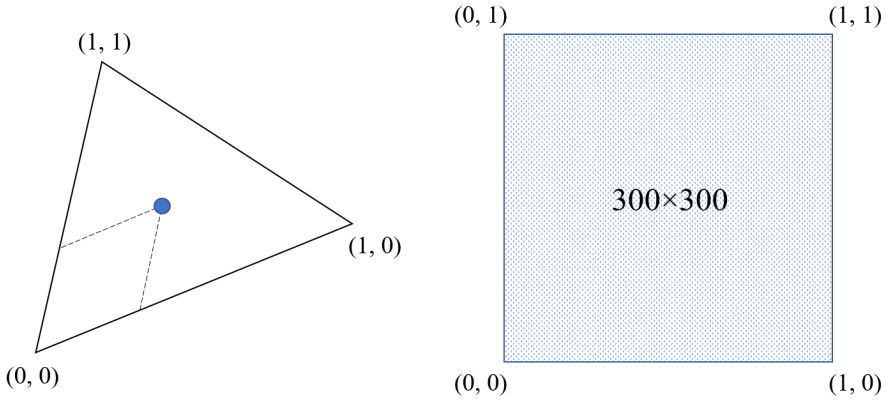
Stanford CS248: Interactive Computer Graphics
Participation Exercise 2

Problem 1: Haotian took a picture of a leaf of his favorite house plant and plans to use different transformations on three copies of the leaf to create a tiny tree (right). For each of the three leaves in the out image (A, B C), please list the transformations (in the correct order!) needed to position them on the trunk. Be careful about rotations and reflections since the result can look similar. You may only use the following basic transformations: $\mathbf{translation}(\mathbf{x}, \mathbf{y})$, $\mathbf{rotation}(\theta)$ (counter-clockwise rotation around the origin), $\mathbf{reflectionX}$, $\mathbf{reflectionY}$ (about x/y axis) and $\mathbf{scale}(\mathbf{x}, \mathbf{y})$.





Problem 2: Consider the triangle below, whose texture coordinates (u,v) at each vertex are shown (Note these are texture coordinates, not vertex positions). You want to sample the texture's color at the location given by the centroid of the triangle. (The position of the triangle's centroid is given by barycentric coordinate (α=1/3, β=1/3, γ=1/3).) **What is the interpolated texture coordinate at this location?** Given this interpolated UV value, and assuming that the texture map is 300×300 pixels in size and bilinear filtering is used during texture sampling, **please provide the indices (i,j) of the four texture pixels used as inputs to bilinear interpolation (no mipmapping is used).** ***Please assume that the bottom left pixel in the texture map is (0,0) and the top right is (299,299).***



Problem 3: In your own words explain what the perspective projection transform does to points in homogeneous space? Why does the perspective projection transform mimic the behavior of a pinhole camera?

Problem 2: Texture Mapping

Consider a 1024×1024 texture map whose value at pixel (x,y) is white if x mod 2 = 0, and black otherwise. This texture is used to texture a single triangle with vertices p0=(0,0), p1=(1,0), and p2=(0,1) and uv texture coordinates uv0=(0,0), uv1=(1,0), and uv2=(0,1)

Consider rendering this triangle to a 512×512 image, where the **background color is 50% gray**. The scene viewport is set up so that scene coordinate (0,0) is in the bottom left of the image, and (1,1) in the top-right corner.

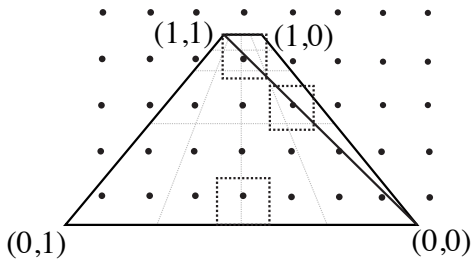
Also assume that screen and texture sample points are at pixel centers (as was the case in assignment 1), and that texture mapping uses **nearest neighbor filtering WITHOUT a MIPMAP.**

A. Describe the image that you will see when you render the scene. Describe both the position of the triangle on screen and what the triangle looks like. (A simple sketch would suffice.)

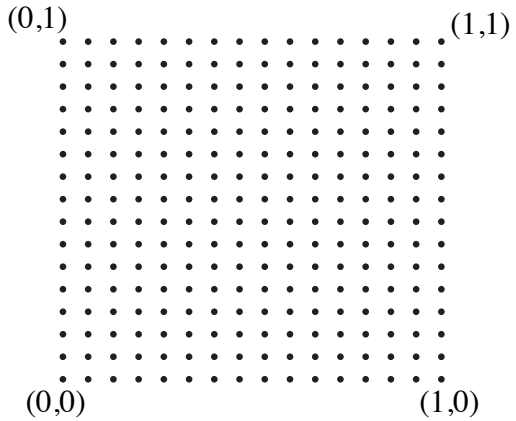
B. Now assume that the rendering mode is changed to **bilinear interpolation** and that the rendered image size is changed to 1024×1024. Describe how you might move the camera (aka pan the viewpoint) to make the triangle *disappear against the 50% gray background!*

C. **This problem is unrelated to parts A and B.**

Consider rendering the two triangles under perspective projection shown at left in the figure below. Per-vertex texture coordinates for the four vertices are given, and the dots indicate the position of screen sample points during rasterization. Now consider the computation to compute the color of the scene at the highlighted screen sample point, which requires a texture lookup into the 16×16 texture shown at right.



Rendered Image



16 x 16 texture (color samples stored at dots)

Three sample points are highlighted in the left side of the above figure, along with dotted boxes showing the extent of the corresponding pixel. In the figure at right, draw the corresponding polygons that correspond to the texture space extent of these screen regions. **BE CAREFUL! Pay attention to the texture coordinate values.**

D. Assume that texture mapping is performed using *bilinear filtering with a mipmap*. Will texture mapping operations to compute the color of the triangles near the top of the rendered image access higher levels of the mipmap (lower resolution textures) or lower levels of the mipmap? Why?

E. Consider the compute cost of texture mapping operations (using mipmapping and bilinear filtering as in part D) for samples at the top of the image or the bottom of the rendered image. Is the cost of texture mapping higher at the top or bottom, or the same? Why?