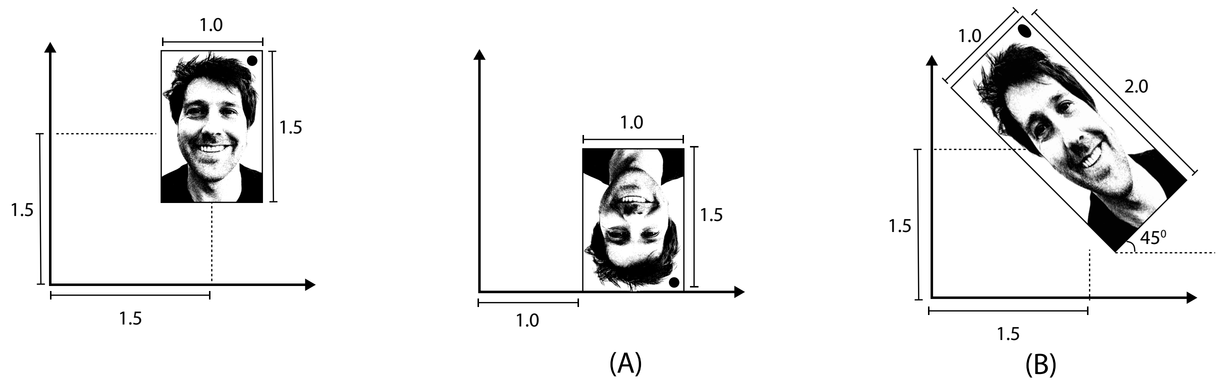
Stanford CS248A: Computer Graphics
Participation Exercise 2

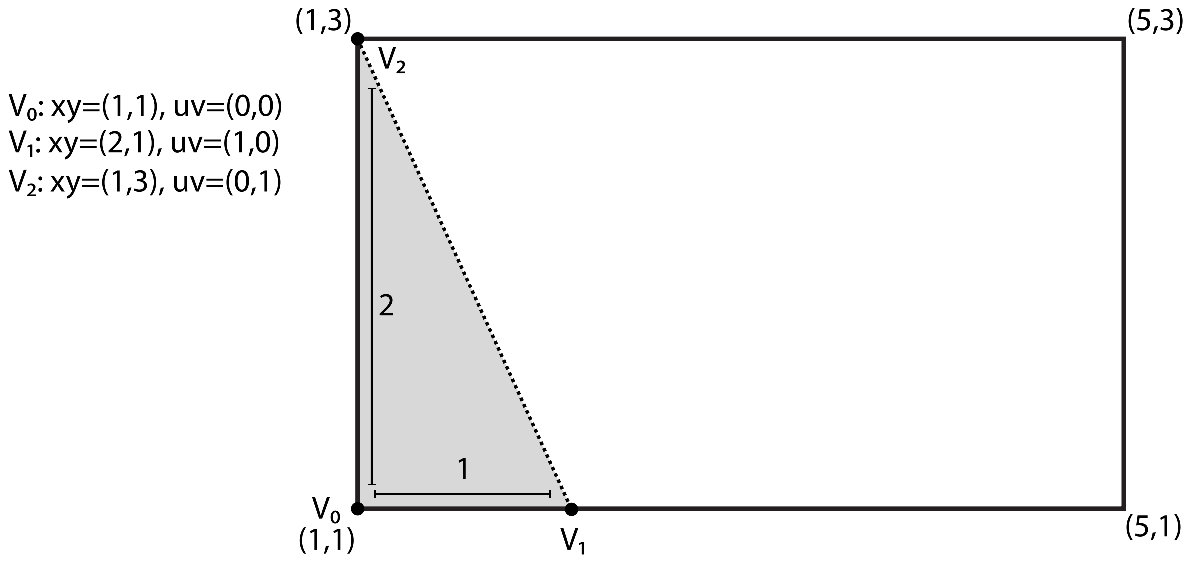
This exercise is graded for CREDIT ONLY. Serious attempts to answer the problems will be given full credit, even if the answers are incorrect.

Problem 1: Please describe a sequence of basic transforms needed to transform the rectangular object on the left (a rectangle of width 1.0 and height 1.5, and centered at the point (1.5, 1.5)) into its position in (A). Also describe the transformations needed to take its position on the left into its position in (B). (These are separate questions). You may assume that basic transforms available to you include: rotate(X), translate(X,Y), scale(X,Y), shear(X,Y). **Be careful, rotations and reflections can sometimes look similar, so we placed a dot on the object to help to disambiguate!**



Problem 2: In your own words explain what the perspective projection transform does to points in homogeneous space? In other words, why does the perspective projection transform mimic the behavior of a pinhole camera?

Problem 3. Consider rendering the the triangle below onto a screen. In the figure below we show the location triangle vertices (in world coordinates), the texture coordinates of triangle vertices, and the world space coordinates of the corners of the image viewport. (e.g., the point (1,1) maps to the bottom left of the region that is visible on screen.)


V_0: xy=(1,1), uv=(0,0)
V_1: xy=(2,1), uv=(1,0)
V_2: xy=(1,3), uv=(0,1)
(1,3)
(5,3)
V_2
2
1
V_0
(1,1)
V_1
(5,1)

A. Assume that the output image is rendered at 720p HD resolution (1280×720 pixels). Please give the image-space coordinates of vertex V_1 of the triangle. **In this problem assume that image space is defined as follows: the bottom-left corner of the visible image is at image-space coordinate (0,0) and the top-right corner is at coordinate (1280,720)**. This means that the center of pixel (i,j) in image space coordinates is at $(i + 0.5, j + 0.5)$

Note: throughout this problem you can express your answers as fractions. The math is not meant to reduce nicely to integers.

B. Assuming that point-in-triangle coverage sampling is performed at pixel centers in image space, please give the texture coordinate (uv) of the sample associated with pixel (0,0). (First confirm this sample covers the triangle. *It does!* Then compute the value of the texture coordinates at this screen sample location.)

C. Now assume that the texture map is a very high resolution 4096×4096 image. Please describe the region of texture space that corresponds to the image-space region spanned by the pixel (0,0). Make sure your answer describes the number of texture pixels in width and height. *(Hint: if you do have a calulator handy, it might be useful to take fractional answer to a real number to get a sense of the number of pixels spanned.)*

D. Consider a texture map that contains high-frequency detail, such as lines a few pixels in width. Describe why aliasing may be visible in this example.

E. Imagine that this rendering system **DID NOT** support any form of mip-mapping, but does support **SUPERSAMPLING** of triangle coverage. (Supersampling = sampling triangle coverage and triangle's color many times per pixel.) Will supersampling reduce aliasing in the rendered image? Describe why or why not?

F. (OPTIONAL, NOT REQUIRED FOR CREDIT) Now assume that the rendered **DOES NOT** support supersampling, but does support trilinear texture sampling using a mipmap. Describe how use of trilinear filtering can significantly reduce aliasing in this example. **Advanced question: In your answer describe why filtering using a mipmap will result in overblurring in the vertical direction.**

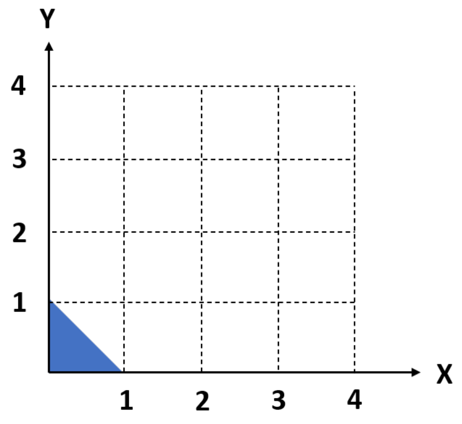
G. (OPTIONAL, NOT REQUIRED FOR CREDIT) There's one type of aliasing in the resulting image that mip-mapped texture sampling WILL NOT remove in this example (hint: think about aliasing during triangle/sample coverage testing). Even with proper texture pre-filtering, can you describe one aliasing artifact that will be noticeable when sampling coverage once per pixel?

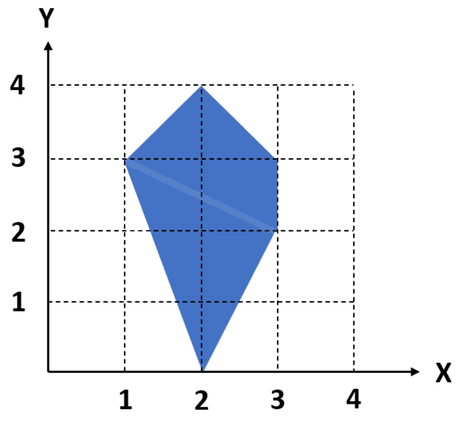
OPTIONAL PROBLEM 4. (PRACTICE ONLY, NOT REQUIRED FOR CREDIT!)

Consider the triangle shape (shown in the left figure) which can be transformed using sequences of the following three transformations:

- **scale**(x, y) (x can be different from y)
- **rotation**(θ) (counter-clockwise rotation around the **origin**)
- **translation**(x, y)

Assuming that you must apply transformations in the order given above, come up with a solution to create the polygon (shown in the right figure) with **only three shapes**. (There's no need to specify the parameters to scales and rotations as real numbers, you may express your answers in terms of sqrt() or arctan() functions. Recall that arctan(X/Y) gives the angle of a triangle with opposite side length X and adjacent side length Y.) *Hint: we drew a white line on the right figure as a hint to help you understand how the three pieces are oriented.*

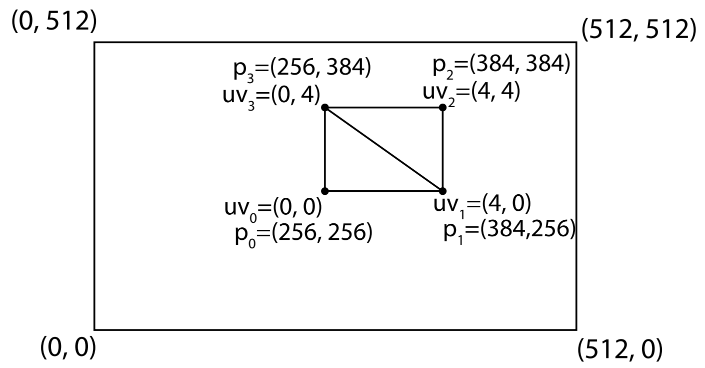




OPTIONAL PROBLEM 5. (PRACTICE ONLY, NOT REQUIRED FOR CREDIT!)

Consider rendering a texture-mapped quadrilateral formed by the two triangles shown below. The quadrilateral is rendered onto a 512×512 pixel screen, and the screen coordinates of the quadrilateral's vertices are given below. (The quad is 128×128 on screen.)

512x512 texture



Notice that the texture coordinates (uv) associated with each vertex are not constrained to be between 0.0 and 1.0. In assignment 1 you implemented texture border behavior of "clamp to edge" but another common behavior when texture mapping is to have texture coordinates "wrap". *In other words, texture coordinates are still interpolated over the surface of a triangle as normal, but if the value of the texture coordinate at a sample point is a, then texture mapping would use the fractional part of a, in other words, $a - floor(a)$, for use in the texture lookup.* (Yes, this means that when bilinear filtering is enabled, a bilerp operation might blend between pixels on the right column of the image and the left column (similarly for the top and bottom row)).

A. Describe the image that you will see when the scene is rendered. In particular how many ice cream cones will you see on the quad?

B. Consider the sample located at the center of pixel (256,256) in the image, its screen-space coordinate is (256.5, 256.5). What is the value of the texture coordinate at this location?

C. Given the geometry and texture coordinates in the scene, what is the size of a pixel in screen space when projected into texture space?

OPTIONAL PROBLEM 6. (PRACTICE ONLY, NOT REQUIRED FOR CREDIT!)

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.

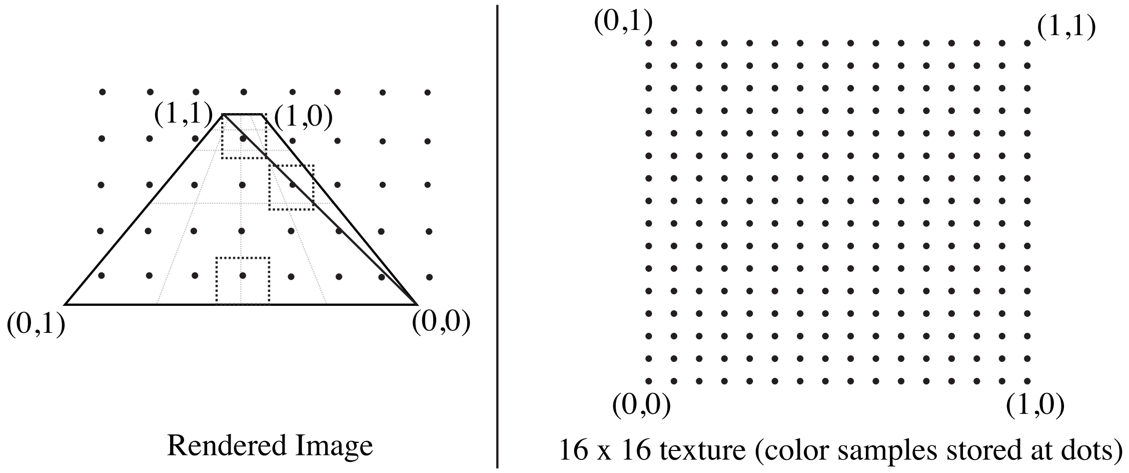
Also assume that screen and texture sample points are at pixel centers (as was the case in assignment 1), and that texture mapping uses **nearest neighbor filtering WITHOUT a MIPMAP.**

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.



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?