TRANSFORMS AND TEXTURE MAPPING 3

CS 184: FOUNDATIONS OF COMPUTER GRAPHICS

1 Basic Transforms

1. Isometric transformations preserve the distances between every pair of points on an object. Which transformations are isometric? Which transformations aren't?

   Solution: Rotations, translations, and reflections (and their combinations) all fulfill this property. They are what we call isometries or rigid transformations (as if we were manipulating an object that cannot be bent or broken).

   Scaling and shearing are examples of transformations that are not isometric.

2. Using homogeneous coordinates, define a point, $\mathbf{p} = \begin{pmatrix} 2 \\ 5 \\ 1 \end{pmatrix}$, a vector, $\mathbf{v} = \begin{pmatrix} 2 \\ 5 \\ 0 \end{pmatrix}$, and a translation transformation, $\mathbf{T} = \begin{pmatrix} 1 & 0 & t_x \\ 0 & 1 & t_y \\ 0 & 0 & 1 \end{pmatrix}$. What is the result of applying $\mathbf{T}$ to $\mathbf{p}$?

   Solution:

   $$\begin{pmatrix} 1 & 0 & t_x \\ 0 & 1 & t_y \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} 2 \\ 5 \\ 1 \end{pmatrix} = \begin{pmatrix} 2 + t_x \\ 5 + t_y \\ 1 \end{pmatrix}$$

   Point $\mathbf{p}$ is shifted t_x to the right and t_y up.

3. What is the result of applying $\mathbf{T}$ to $\mathbf{v}$? Justify this result.

   Solution:

   $$\begin{pmatrix} 1 & 0 & t_x \\ 0 & 1 & t_y \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} 2 \\ 5 \\ 0 \end{pmatrix} = \begin{pmatrix} 2 \\ 5 \\ 0 \end{pmatrix}$$

   A vector represents a direction, not a point in space, so it should be unaffected by translation transformations. (Translating a *direction* should not change anything about the *direction*.)

4. Write the transformation matrix for a 2D object that is reflected across the y-axis, translated up by 1 unit and right by 3 units, then rotated around the origin by $90°$ counterclockwise.

Solution:

An important thing to remember here is that the transformation matrices are multiplied right to left. Keeping the order correct is important!

$$\begin{pmatrix} 0 & -1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} 1 & 0 & 3 \\ 0 & 1 & 1 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} -1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} = \begin{pmatrix} 0 & -1 & -1 \\ -1 & 0 & 3 \\ 0 & 0 & 1 \end{pmatrix}$$

2 Rebecca Takes a Look

In world space coordinates, a teapot sits at $(0, 0, 0)$. Rebecca holds her camera at position $(1, 0, 2)$ — the eye point, **e**, of the camera.

1. Rebecca points her camera at the teapot, intending to take a picture. What is the view direction, **v**, of her camera?

   Solution: $\begin{pmatrix}0\\0\\0\end{pmatrix} - \begin{pmatrix}1\\0\\2\end{pmatrix} = \begin{pmatrix}-1\\0\\-2\end{pmatrix}$. To normalize, $\frac{1}{\sqrt{1+0+4}}\begin{pmatrix}-1\\0\\-2\end{pmatrix} = \frac{1}{\sqrt{5}}\begin{pmatrix}-1\\0\\-2\end{pmatrix}$.

2. In world space coordinates, in what direction does the positive z-axis of Rebecca's "standard" camera space point?

   Solution: $\frac{1}{\sqrt{5}}\begin{pmatrix}1\\0\\2\end{pmatrix}$

3. The up vector, **u**, points in the direction of the positive y-axis of "standard" camera space. In world space coordinates, $\mathbf{u} = \begin{pmatrix}0 & 1 & 0\end{pmatrix}^\mathsf{T}$. Calculate the right vector, **r**, which points in the direction of the positive x-axis of "standard" camera space.

   Solution: Perform the cross-product $\mathbf{u} \times -\mathbf{v}$. This is essentially $\mathbf{j} \times \mathbf{k}$ to find $\mathbf{i}$.

   $\frac{1}{\sqrt{5}}\begin{vmatrix}\mathbf{i} & \mathbf{j} & \mathbf{k}\\0 & 1 & 0\\1 & 0 & 2\end{vmatrix} = \frac{1}{\sqrt{5}}(2\mathbf{i} - \mathbf{k})$.

   $\mathbf{r} = \frac{1}{\sqrt{5}}\begin{pmatrix}2\\0\\-1\end{pmatrix}$

4. Write the matrix that transforms coordinates from "standard" camera space to world space. The inverse of this matrix is the *Look-At* matrix. What coordinate system transformation does the *Look-At* matrix perform?

   Solution:

   $$\begin{pmatrix}2/\sqrt{5} & 0 & 1/\sqrt{5} & 1\\0 & 1 & 0 & 0\\-1/\sqrt{5} & 0 & 2/\sqrt{5} & 2\\0 & 0 & 0 & 1\end{pmatrix}$$

Inverse performs world space to camera space transformation.

3 Barycentric Coordinates

The first step in mapping a texture onto a triangle is to convert the screen pixel to barycentric coordinates. These coordinates (α, β, γ) can be thought of as the weights assigned to each vertex. The weighted average of the vertices is a screen-space coordinate: $(x, y) = \alpha A + \beta B + \gamma C$.

1. What happens if α, β, or γ is less than zero?

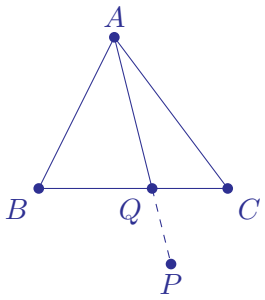
   Solution: If any of the coordinates is less than zero, then the point will lie outside the triangle. As long as the coordinates all sum to one, however, the point should still lie in the same plane as that formed by the three barycentric coordinates.

2. What is the range of (x, y) if $\alpha + \beta + \gamma = 1$ and $\alpha = 0$?

   Solution: Since we are only interpolating between B and C, (x, y) must lie on line BC.

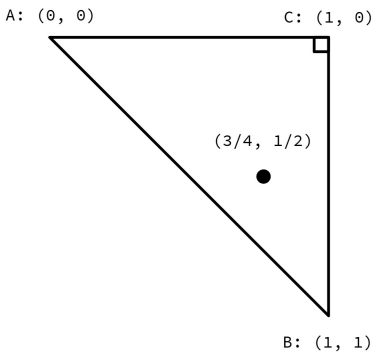
3. What is the range of (x, y) if $\alpha + \beta + \gamma = 1$ and $\alpha < 0$?

   Solution: If a coordinate is less than zero, then the point will lie outside the triangle, since it is being extrapolated away from the corresponding vertex.

   More formally, we can consider $P = (x, y)$ the result of two interpolations: First, interpolate between B and C to obtain a point $Q = \frac{\beta}{1-\alpha} B + \frac{\gamma}{1-\alpha} C$ on line BC. Then, interpolate between A and Q to obtain $P = \alpha A + (1 - \alpha) Q$ on line AQ. Since $\alpha < 0$, points A and P are on opposite sides of point Q. Therefore, P must lie on the opposite side of line BC as point A.

4. What are the barycentric coordinates of the point corresponding to the screen-space coordinates $(x, y) = (\frac{3}{4}, \frac{1}{2})$ in the following diagram? (Hint: Use the definition of barycentric coordinates, not the closed-form solution.)



Solution: The x and y coordinates provide two constraints, and the third constraint is that barycentric coordinates sum to 1. Thus, we have 3 equations and 3 unknowns.

$$0\alpha + 1\beta + 1\gamma = \frac{3}{4} \implies \beta + \gamma = \frac{3}{4} \qquad (x\text{-coordinate})$$

$$0\alpha + 1\beta + 0\gamma = \frac{1}{2} \implies \beta = \frac{1}{2} \qquad (y\text{-coordinate})$$

$$\alpha + \beta + \gamma = 1.$$

Solving, we obtain $(\alpha, \beta, \gamma) = \left(\frac{1}{4}, \frac{1}{2}, \frac{1}{4}\right)$.

5. If the RGB values of A, B, C are `(1, 0, 0)`, `(0, 1, 0)`, `(0, 0, 1)`, respectively, then what is the interpolated color at the selected point above?

Solution: The interpolated color can be computed as

$$\begin{aligned} V &= \alpha V_A + \beta V_B + \gamma V_C \\ &= \frac{1}{4}\texttt{(1, 0, 0)} + \frac{1}{2}\texttt{(0, 1, 0)} + \frac{1}{4}\texttt{(0, 0, 1)} \\ &= \texttt{(0.25, 0.5, 0.25)}. \end{aligned}$$

4 A Texture and Mipmap Adventure

In this problem, we explore texture coordinates and mipmaps. A pixel may cover more or less than one texel, leading to aliasing. To handle this, we use mipmaps—pre-filtered, downscaled versions of the texture.

Consider an image mapped to a texture with a resolution of $(128, 128)$. We'll work through sampling a pixel and retrieving its value from the appropriate mipmap level. Note that the texture coordinates (u, v) lie in the range $[0, 1]$, not the texture resolution range.

1. **Conceptual Question:** What is the resolution of the mipmap at level 3?

   Solution: Mipmaps are generated by halving the resolution of the original texture in each dimension. Starting with a texture of size $(128, 128)$, we have:

   $$\begin{aligned} \text{Level 0:} &\quad (128, 128) \\ \text{Level 1:} &\quad (128/2, 128/2) = (64, 64) \\ \text{Level 2:} &\quad (64/2, 64/2) = (32, 32) \\ \text{Level 3:} &\quad (32/2, 32/2) = (16, 16) \end{aligned}$$

   Thus, the mipmap at level 3 has a resolution of $(16, 16)$.

2. **Conceptual Question:** Derive an expression for the amount of space needed to store all mipmap levels, relative to the memory needed to store the original texture.

   Solution: At mipmap level i, the texture resolution is given by $(\frac{x}{2^i}, \frac{x}{2^i})$. Halving each dimension reduces the overall size by a factor of 4. Thus, let x be our texture size of the highest resolution. The total storage required is the sum of the mipmaps, which form a geometric series:

   $$x + \frac{x}{4} + \frac{x}{16} + \cdots = x\left(1 + \frac{1}{4} + \frac{1}{4^2} + \cdots\right)$$

   The infinite sum is:

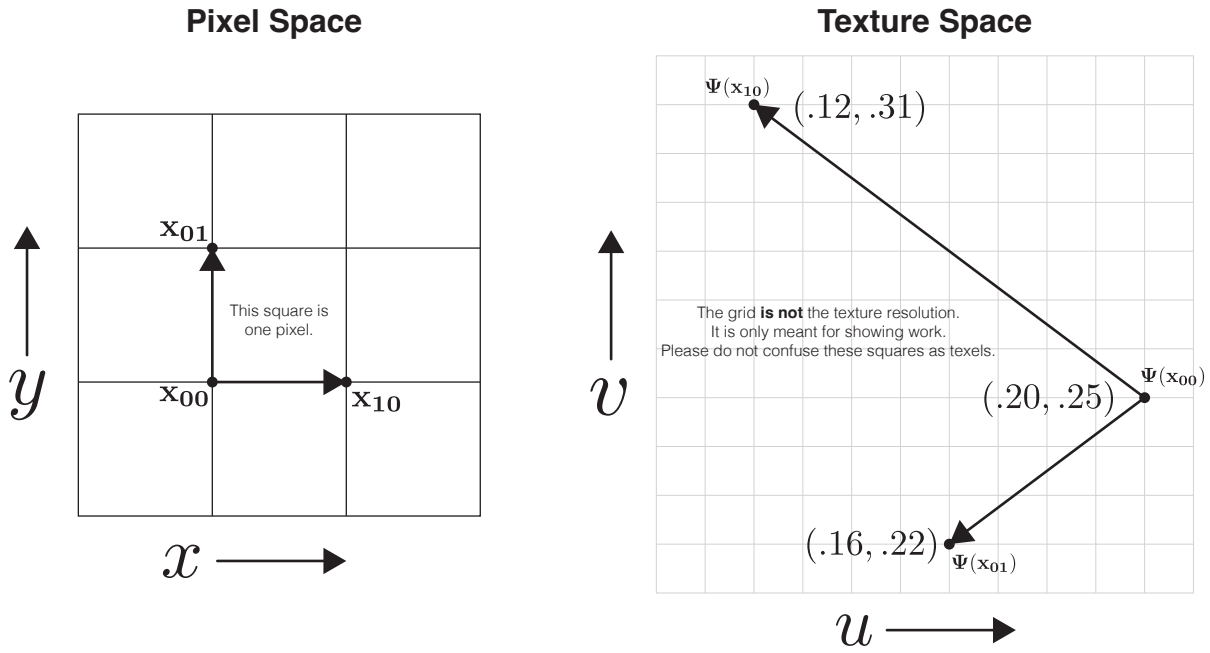
   $$\sum_{i=0}^{\infty} \left(\frac{1}{4}\right)^i = \frac{1}{1 - \frac{1}{4}} = \frac{4}{3}.$$

   Multiplying by x gives a total of:

   $$x \cdot \frac{4}{3} = \frac{4}{3}x.$$

   Thus, storing all mipmap levels requires $\frac{4}{3}$ times the memory of the original texture. In practice, the series is finite and stops after the texture resolution is (1×1), so $\frac{4}{3}x$ is an upper bound.

3. We are given pixel point $\mathbf{x_{00}} = (x, y)$, as well as $\mathbf{x_{01}}$ and $\mathbf{x_{10}}$, each 1-pixel away from $\mathbf{x_{00}}$. These three points are mapped to their corresponding locations in texture space by the mapping $\mathbf{\Psi}(\mathbf{x})$. *Note that the mapping uses barycentric coordinates!* The texel space is in the range $[0, 1]$. At what mipmap level, L, should we sample to retrieve the texture for point $\mathbf{x_{00}}$?



Solution: We need to compute the length of the vectors in texture space. First, we can compute the derivatives for the unit change in pixel space, $\left(\frac{du}{dx}, \frac{dv}{dx}\right)$ and $\left(\frac{du}{dy}, \frac{dv}{dy}\right)$.

$$\left(\frac{du}{dx}, \frac{dv}{dx}\right) = \mathbf{\Psi}(\mathbf{x_{10}}) - \mathbf{\Psi}(\mathbf{x_{00}}) = (.12, .31) - (.20, .25) = (-.08, -.06) \tag{1}$$

$$\left(\frac{du}{dy}, \frac{dv}{dy}\right) = \mathbf{\Psi}(\mathbf{x_{01}}) - \mathbf{\Psi}(\mathbf{x_{00}}) = (.16, .22) - (.20, .25) = (-.04, -.03) \tag{2}$$

Next, we find the length of these vectors. Notice that both $(-0.08, -0.06)$ and $(-0.04, -0.03)$ are proportional to $(4, 3)$, which is part of a standard Pythagorean triple $(3, 4, 5)$. Hence, each has magnitude 5 times its scale factor. However, below is a more detailed way to compute it.

$$\left\|\left(\frac{du}{dx}, \frac{dv}{dx}\right)\right\| = \sqrt{(-0.08)^2 + (-0.06)^2} = \sqrt{0.0064 + 0.0036} = \sqrt{0.01} = 0.10, \tag{3}$$

$$\left\|\left(\frac{du}{dy}, \frac{dv}{dy}\right)\right\| = \sqrt{(-0.04)^2 + (-0.03)^2} = \sqrt{0.0016 + 0.0009} = \sqrt{0.0025} = 0.05. \tag{4}$$

Next, let's recall that our texture coordinates are in the range $[0, 1]$ but the texture resolution is $(128, 128)$. We need to scale these derivatives by 128 and then compute the following formula:

$$L = 128 \times \max\left(\left\|\left(\frac{du}{dx}, \frac{dv}{dx}\right)\right\|, \left\|\left(\frac{du}{dy}, \frac{dv}{dy}\right)\right\|\right) \tag{5}$$

$$= 128 \times \max(0.10, 0.05) \tag{6}$$

$$= 128 \times 0.10 \tag{7}$$

$$= 12.8 \tag{8}$$

So, our mipmap selection is

$$D = \log_2(L) = \log_2(12.8) \approx 3.68 \tag{9}$$

4. For mipmap levels 0 through 5, the texture value at $\Psi(x_{00})$ is given by:

$$T_0 = 0.38, \quad T_1 = 0.42, \quad T_2 = 0.36, \quad T_3 = 0.40, \quad T_4 = 0.39, \quad T_5 = 0.37.$$

Given our answer to the previous question, what texture value should we use?

Solution: We found that $D \approx 3.68$ in the previous problem. We round this value to the nearest integer level $\text{round}(3.68) = 4$, and we select from the corresponding texture $T_4 = 0.39$.

5. How can we combine values from two neighboring mipmap levels to get an even smoother result?

Solution: We can use *trilinear interpolation* between the two nearest mipmap levels to $D \approx 3.68$,

$$\lfloor D \rfloor = 3, \quad \lceil D \rceil = 4,$$

First, define the fractional part:

$$\alpha = D - \lfloor D \rfloor = 3.68 - 3 = 0.68.$$

Then we blend between $T_3 = 0.40$ and $T_4 = 0.39$:

$$T_{3.68} = (1-\alpha)\,T_3 + \alpha\,T_4 = 0.32 \times 0.40 + 0.68 \times 0.39 = 0.128 + 0.2652 = 0.3932.$$

Hence, the final texture value at $D \approx 3.68$ is 0.3932.