Projector Texture Mapping

A projector is at \((5, 3, 3)\) looking at \((5, 3, -3)\). The near plane is at \(z = 2\). The left and right of the rectangle in the eye frame are at \(x = -1\) and \(x = 1\). The top and bottom of the rectangle are at \(y = 2\) and \(y = -2\). Construct the model-view matrix and the projection matrix. If the texture in Figure 1 to be projected is shown in the picture, what is the colour to be projected on the point at \((9, 4, -10)\)?

![Figure 1](image)

Model-view matrix in this case transforms world to projector frame:

\[
\begin{bmatrix}
1 & 0 & 0 & -5 \\
0 & 1 & 0 & -3 \\
0 & 0 & 1 & -3 \\
0 & 0 & 0 & 1
\end{bmatrix}
\]

Projection matrix (from (10.7) in the textbook):
Notice the near plane is at \( z_c = 2 - 3 = -1\) in the projector frame.

\[
\begin{bmatrix}
-\frac{2n}{r-l} & 0 & \frac{r+l}{r-l} & 0 \\
0 & -\frac{2n}{r-b} & \frac{r+b}{r-b} & 0 \\
- & - & - & - \\
0 & 0 & -1 & 0
\end{bmatrix}
= \begin{bmatrix}
\frac{b}{4} & 0 & 0 & 0 \\
0 & \frac{b}{4} & 0 & 0 \\
- & - & - & - \\
0 & 0 & -1 & 0
\end{bmatrix}
\]

Coordinates on the near plane projected on \((9, 4, -10)\):
\[
\begin{bmatrix}
\frac{7}{2} & 0 & 0 & 0 \\
0 & \frac{7}{4} & 0 & 0 \\
- & - & - & - \\
0 & 0 & -1 & 0
\end{bmatrix}
\begin{bmatrix}
1 & 0 & 0 & -5 \\
0 & 1 & 0 & -3 \\
0 & 0 & 1 & -3 \\
0 & 0 & 0 & 1
\end{bmatrix}
= \begin{bmatrix}
\frac{4}{4} \\
\frac{1}{4} \\
-10 \\
13
\end{bmatrix}
\]

clip coordinates = \(\frac{1}{13}(4, \frac{1}{2}) = (\frac{4}{13}, \frac{1}{26})\)

\Rightarrow\text{projected colour is yellow}

**Interpolation**

The control points for a Bézier curve are: \( C_0 = (0, 0, 0) , C_1 = (2, 5, 3) , C_2 = (5, 1, 3) , C_3 = (0, 2, 3) \). What is the point at \( t = 0.5 \)?

\[ p = (1 - 0.5)^3 C_0 + 3 \cdot 0.5(1 - 0.5)^2 C_1 + 3 \cdot 0.5^2(1 - 0.5) C_2 + 0.5^3 C_3 = (2.625, 2.5, 2.625) \]

**Depth**

The near plane is at \( z = -5 \), the far plane is at \( z = -20 \), the top, bottom, left and right of the near plane are at \( y = 6 \), \( y = -6 \), \( x = -10 \), \( x = 10 \). Construct the projection matrix. What are the clip coordinates of the points \( P_1 = (2, 2, -6) \), and \( P_2 = (3, 3, -15) \)? What is the depth value that would be stored in the depth buffer, for each point?

From (11.2) in the textbook:

\[
\begin{bmatrix}
-\frac{2n}{r-l} & 0 & \frac{r+l}{r-l} & 0 \\
0 & -\frac{2n}{r-b} & \frac{r+b}{r-b} & 0 \\
0 & 0 & \frac{k}{f_n} & -\frac{2f_n}{f_n} \\
0 & 0 & -1 & 0
\end{bmatrix}
= \begin{bmatrix}
\frac{10}{20} & 0 & 0 & 0 \\
0 & \frac{10}{20} & 0 & 0 \\
0 & 0 & \frac{-25}{15} & \frac{200}{15} \\
0 & 0 & -1 & 0
\end{bmatrix}
\begin{bmatrix}
2 \\
0 \\
-6 \\
0
\end{bmatrix}
= \begin{bmatrix}
1 \\
0 \\
-10/3 \\
-5/3
\end{bmatrix}
\]

clip coordinates of \( P_1 \):

\[ \begin{bmatrix}
\frac{10}{20} & 0 & 0 & 0 \\
0 & \frac{10}{20} & 0 & 0 \\
0 & 0 & \frac{-25}{15} & \frac{200}{15} \\
0 & 0 & -1 & 0
\end{bmatrix}
= \begin{bmatrix}
3 \\
0 \\
-15 \\
0
\end{bmatrix}
\]

normalized device coordinates of \( P_1 : \frac{1}{15}(1, \frac{5}{6}) = (\frac{2}{15}, \frac{5}{30}) \), Depth: \( \frac{1}{15} \cdot \frac{10}{7} = \frac{10}{15} \)

clip coordinates of \( P_2 \):

\[ \begin{bmatrix}
\frac{10}{20} & 0 & 0 & 0 \\
0 & \frac{10}{20} & 0 & 0 \\
0 & 0 & \frac{-25}{15} & \frac{200}{15} \\
0 & 0 & -1 & 0
\end{bmatrix}
= \begin{bmatrix}
3 \\
0 \\
-15 \\
0
\end{bmatrix}
\]

normalized device coordinates of \( P_2 : \frac{1}{15}(\frac{3}{10}, \frac{5}{6}) = (\frac{1}{15}, \frac{5}{30}) \), Depth: \( \frac{1}{15} \cdot \frac{-35/3}{7} = -\frac{7}{5} \)
Sampling

A single fragment is shown in Figure 2, along with the colours from a texture image that would map on to it. Suppose we use over-sampling at points $P_1 = (0.4, 0.6)$, $P_2 = (0.3, 0.3)$, $P_3 = (0.2, 0.7)$, what is the output colour? What if the sampling points are 9 points on a 3 by 3 grid at $x = 0.25, 0.5, 0.75$, and $y = 0.25, 0.5, 0.75$? Assume the colours for red, green, blue are $(1, 0, 0)$, $(0, 1, 0)$, $(0, 0, 1)$ respectively.

![Figure 2](image)

(a)
Colour at $P_1 = \text{blue} = (0, 0, 1)$
Colour at $P_2 = \text{blue} = (0, 0, 1)$
Colour at $P_3 = \text{green} = (0, 1, 0)$
\Rightarrow\text{Sampled colour} = \frac{1}{3}(0, 0, 1) + \frac{1}{3}(0, 0, 1) + \frac{1}{3}(0, 1, 0) = (0, \frac{1}{3}, \frac{2}{3})

(b)
Sampled colours: 2 blue, 3 red, and 4 green
\Rightarrow\text{Sampled colour} = \frac{2}{3}(0, 0, 1) + \frac{1}{3}(1, 0, 0) + \frac{4}{3}(0, 1, 0) = (\frac{1}{3}, \frac{4}{3}, \frac{2}{3})

Compositing

On a completely opaque black background, with colour $(0,0,0,1)$, we draw a foreground fragment with the colour $(1,1,1,0.7)$ i.e. white with alpha value 0.7. What is the output colour of the pixel?

Note: the question was a bit ambiguous about whether the colours were "premultiplied" or not. If nothing is mentioned, assume that the colours "premultiplied" by the $\alpha$ value.

In this case we have our foreground colour $I^f = (1,1,1)$, and the background colour $I^b = (0,0,0)$
From (16.4) in the textbook (for premultiplied colour):
\[ I^e = I^f + I^b(1 - \alpha^f) = (1, 1, 1) + 0.3(0, 0, 0) = (1, 1, 1) \]
with the alpha channel
\[ \alpha^c = \alpha^f + \alpha^b(1 - \alpha^f) = 0.7 + 1 \cdot 0.3 = 1 \]
Output colour: (1, 1, 1, 1)

If you thought this was strange, it’s because you may have been expecting the color to be non-premultiplied (see Section 16.4.2). If you use the formula in that section, you will get
Output colour: (0.7, 0.7, 0.7, 1)

**Bilinear interpolation**

If the value at P1 = (1,1) is 0, P2 = (2,1) is 1, P3 = (2,2) is 1, P4 = (1,2) is 1. What is the bilinearly interpolated value at P5 = (1.5,1.5)? What if P5 was (1.25,1.75)? What if the value at P3 is 2?

![Figure 3](image)

For P5 = (1.5, 1.5), the value at P3 is the average over the 4 points.
If the value at P3 is 1, the value at P5 is \( \frac{0}{4} + \frac{1}{4} + \frac{1}{4} + \frac{1}{4} = \frac{3}{4} \).
If the value at P3 is 2, the value at P5 is \( \frac{0}{4} + \frac{1}{4} + \frac{2}{4} + \frac{1}{4} = 1 \).

In a more general situation when P5 = (1.25, 1.75), without easy symmetry, you interpolate along one direction (say X) and then in the other direction (Y).
If the value at point P3 is denoted \( V_i \) the value at P5, then
\[
V_5 = (V_1 \cdot \frac{3}{4} + V_2 \cdot \frac{1}{4}) \cdot \frac{1}{4} + (V_4 \cdot \frac{3}{4} + V_3 \cdot \frac{1}{4}) \cdot \frac{3}{4}
\]

If \( V_3 = 1 \), then \( V_5 = \frac{13}{16} \).
If \( V_3 = 2 \), then \( V_5 = 1 \). Another way to do this is to see that the value \( V_2 \) is the weighted average of the 4 values. The weights are determined by the area of the rectangles. The area of the rectangle with diagonal \( P_1P_5 \) is \( 0.25 \cdot 0.75 \).
Area of the rectangle with diagonal \( P_2P_5 \) is \( 0.75 \cdot 0.75 \).

4
Area of the rectangle with diagonal $P_3P_5$ is $0.75 \cdot 0.25 = \frac{3}{16}$
Area of the rectangle with diagonal $P_4P_5$ is $0.25 \cdot 0.25 = \frac{1}{16}$

The weights are applied to the value on the opposite vertex:
If the value at $P_3$ is 1, the value at $P_5$ is $\frac{3}{16} \cdot 0 + \frac{1}{16} \cdot 1 + \frac{3}{16} \cdot 1 + \frac{9}{16} \cdot 1 = \frac{13}{16}$
If the value at $P_3$ is 2, the value at $P_5$ is $\frac{3}{16} \cdot 0 + \frac{1}{16} \cdot 1 + \frac{3}{16} \cdot 2 + \frac{9}{16} \cdot 1 = 1$

Assignment Related Questions

1. What does the following line of code do?

   ```cpp
glUniform3fv(glGetUniformLocation(w_state->getCurrentProgram(), "gem_pos"), 1, glm::value_ptr(gem_position));
```

   Uniform variables are used to communicate between shaders and the application program.
   This function call sets the value of a uniform variable in the shader called `gem_pos`, to the value of the GLM vec3 variable `gem_position`.

2. In assignment 1, we asked you to deform the armadillo by the following scheme: If a given vertex of the armadillo is within `gem_radius` of `gem_position`, translate it along the vector between it and the gem until it lies on the surface of the sphere. You are given the following:

   ```cpp
   vec4 Position;
   uniform vec4 gem_position;
   uniform float gem_radius;
   
   Fill in the important pieces of the vertex shader below:
   ```

   ```cpp
   //...
   int main()
   {
     vec4 dGem = Position - gem_position;

     if (length(dGem) < gem_radius)
       Position = gem_position + normalize(dGem)*gem_radius;
   }
   //...
   ```