1 Theoretical Part

1.1 Simple Scanline Interpolation

(i) Vertices in view space

First, do the interpolation in y direction. For point "m" and "n" respectively:

$$A^{(m)} = \frac{y - y_2}{y_1 - y_2} A_1 + \frac{y_1 - y}{y_1 - y_2} A_2, \tag{1}$$

$$A^{(n)} = \frac{y - y_2}{y_3 - y_2} A_3 + \frac{y_3 - y}{y_3 - y_2} A_2. \tag{2}$$

Then do the interpolation on the x axis using the information given by "m" and "n":

$$A^{(p)} = \frac{x - x_n}{x_m - x_n} A^{(m)} + \frac{x_m - x}{x_m - x_n} A^{(n)}. \tag{3}$$

1.2 Phong Lighting

First, we need to invert the window space coordinates to ndc coordinates. This is done using the viewport transform:

$$x_{window} = \frac{width}{2}(x_{ndc} + 1) + x_{LL},$$
$$y_{window} = \frac{height}{2}(y_{ndc} + 1) + y_{LL},$$
$$z_{window} = z_{ndc}/2 + 1/2.$$

Therefore, the three points in ndc coordinates are given as:

$$v_{1,ndc} = (-0.4,\ 0.6,\ 0.4\ ),\ v_{2,ndc} = (-0.8,\ -0.6,\ 0.0\ ),\ v_{3,ndc} = (0.8,\ -0.8,\ -0.4\ ).$$

From the given parameters of the perspective projection, the projection matrix (symmetric perspective) can be specified. Using the projection matrix, the forth element of clip space coordinate can be speficied as:

$$w_{clip} = \frac{T_2}{z_{ndc} - \frac{T_1}{E_1}} = \frac{-\frac{22}{5}}{z_{ndc} - \frac{6}{5}}, \tag{4}$$

from which we can find the w_{clip} coordinates for the three points as:

$$w_{1,clip} = 5.5,\ w_{2,clip} = 3.67,\ w_{3,clip} = 2.75.$$

Therefore, the clip space coordinates are given as:

$v_{1,clip} = (-2.2,\ 3.3,\ 2.2,\ 5.5)$, $v_{2,clip} = (-2.936,\ -2.202,\ 0.0,\ 3.67)$, $v_{3,clip} = (2.2,\ -2.2,\ -1.1,\ 2.75)$.

Finally, we can revert to the view space, which is done by multiplying by the invert of the projection matrix $v_{view} = M_{proj}^{-1}\ v_{clip}$ with the projection matrix given as:

$$M_{proj} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & -\frac{6}{5} & -\frac{22}{5} \\ 0 & 0 & -1 & 0 \end{pmatrix}.$$

Therefore, the view space coordinates:

$v_{1,view} = (-2.2,\ 3.3,\ -5.5,\ 1)$, $v_{2,view} = (-2.936,\ -2.202,\ -3.67,\ 1)$, $v_{3,clip} = (2.2,\ -2.2,\ -2.75,\ 1)$.

(ii) Color computation – Phong shading

In general, the color at a point is given as:

$$\text{color}_{RGB} = \frac{1}{k_c + k_l d + k_q d^2}(m_{\text{RGB}}^{\text{diffuse}} \cdot l_{\text{RGB}}^{\text{diffuse}} \cdot \max(L \cdot N,\ 0)),$$

For Phong shading, the light calculation has to be carried out for each interpolated point separately. The objects needed to be interpolated are: the material properties, the normal vectors. Both the material diffuse property and the normal vector are in the form of length 3 vector, and the interpolation rule is exactly the same as written in previous derivations for color interpolation. Therefore, the properties of the interpolated points, including the position vectors, distance to the light source, material diffuse properties, normal vectors, and at last the RGB colors, are given in the table below as a summary.

ndc coord	3D position	d	$m_{\text{RGB}}^{\text{diffuse}}$	N	RGB
(2,3)	(-1.39, 1.65, -4.77)	12.1	(0.70, 0.10, 0.20)	(-0.4, 0.27, 0.33)	0
(1,2)	(-2.48, -0.33, -4.26)	13.0	(0.34, 0.62, 0.04)	(-0.61, -0.21, 0.33)	0
(2,2)	(-1.22, -0.11, -4.11)	12.2	(0.38, 0.34, 0.28)	(-0.29, -0.16, 0.33)	0
(3,2)	(0.044, 0.11, -3.96)	11.3	(0.42, 0.06, 0.52)	(0.03, -0.11, 0.33)	0
(1,1)	(-2.3, -2.09, -3.6)	13.3	(0.02, 0.86, 0.12)	(-0.51, -0.64, 0.33)	0
(2,1)	(-1.04, -1.87, -3.45)	12.5	(0.06, 0.58, 0.36)	(-0.19, -0.59, 0.33)	0
(3,1)	(0.22, -1.65, -3.3)	11.7	(0.10, 0.30, 0.60)	(0.13, -0.53, 0.33)	0
(4,1)	(1.48, -1.43, -3.15)	11.0	(0.14, 0.02, 0.84)	(0.45, -0.48, 0.33)	0

(iii) Shader Speed

The case where per-vertex shading involves more calculation could be: when the lighting condition is complex (the lighting calculation is extremely time-consuming), and the triangles are sufficiently fine. For instance, each triangle will only contains 1 grid point inside it. In such case, the number of fragment calculation per triangle is 1, meaning that the lighting calculation needs to be carried out 1 time. While for per-vertex calculation involves lighting calculation 3 times per triangle, which consumes more time.