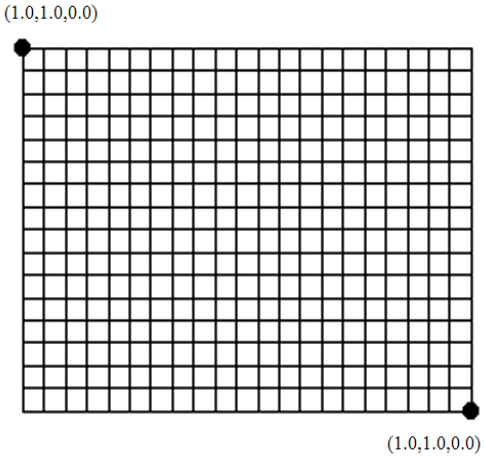
P1 CG 2013.1

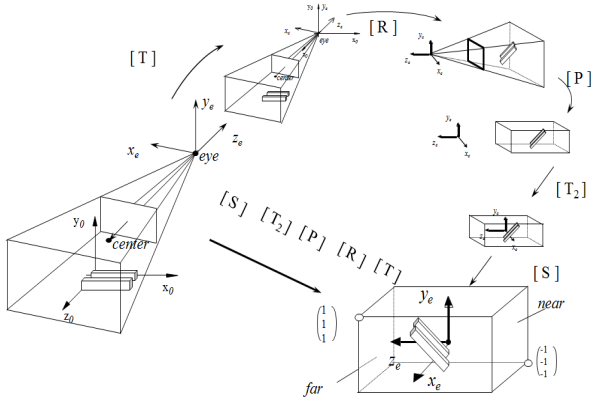
	Computação Gráfica 1 Prof. Rodrigo de Toledo	Data: 3/6/2013 P1 2013.1

1) (2 pontos) semelhante à questão da P1 2011.1 O que faz o vertex shader abaixo, se aplicado a uma malha regular descrita no plano normal a Z com coordenadas x e y no espaço $[-1,1]^2$? (Qual o resultado visual?) Que mudanças devemos fazer no código para que a normal respeite a nova geometria da malha?

INPUT:




```
void main(){
    vec4 vert = gl_Vertex;
    vec3 color;
    float invertx = -vert.x;
          vec3 normalFace;
    if (vert.x > vert.y) {
        if (invertx > vert.y) {
            color = vec3(1.0, 1.0, 0.0); //Y
             vert.z = 1.0-abs(vert.y);
        } else {
            color = vec3(1.0, 0.0, 0.0); //R
            vert.z = 1.0-abs(vert.x);
        }
    } else {
        if (invertx > vert.y) {
```




```
            color = vec3(0.0, 0.0, 1.0); //B
            vert.z = 1.0-abs(vert.x);
        } else {
            color = vec3(0.0, 1.0, 0.0); //G
            vert.z = 1.0-abs(vert.y);
         }      }
    vec3 normal = normalize(gl_NormalMatrix * normalize(normalFace));
    vec3 lightDir = normalize(gl_LightSource[0].position.xyz - vert.xyz);
    float diffuse = max(0.0,dot(lightDir, normal));
```

```
    gl_FrontColor.rgb = (0.2 + 0.8 * diffuse) * color;
    gl_FrontColor.a = 1;
    gl_Position = gl_ModelViewProjectionMatrix * vert;
}
```

2) (3 pontos) (a) Explique o propósito de cada uma das 5 transformações presentes nessa figura. (b) Em sala de aula foram mencionadas duas transformações que acontecem uma antes e outra depois dessas 5, explique quais são.

3) (2,5 pontos) Explique a diferença entre cada par de conceitos (dado o contexto entre parênteses):
Máximo de três linhas por item! (quatro linhas terá metade do ponto, cinco ou mais linhas será desconsiderado).
(a) repeat X clamp (aplicação de textura); (b) bilinear X baricêntrica (interpolação); (c) difusa X especular (iluminação); (d) fragmento x pixel (síntese de imagem); (e) auto-oclusão X auto-sombra (visualização de mesoestrutura)

4) (2,5 pontos) Dado um triângulo ABC de uma malha qualquer com coordenadas A = [8,0,0]; B = [0,4,0]; C = [0,0,4]; e normais N_A = [0.1, 0.1, 0.8]; N_B = [0.4, 0.4, 0.2]; N_C = [0.2, 0.4, 0.4]. Calcule a cor final do ponto P_i pertencente a esse triângulo, cujas coordenadas baricêntricas são: P_i^{coord} = [0.5,0.25,0.25], sabendo que a luz de cor branca está posicionada em L = [4,1,9]. A superfície do triângulo tem cor C, considere apenas a difusa dos modelos (a) Flat; (b) Phong.

Produto Vetorial 3D entre a e b: $[a_yb_z-a_zb_y, a_zb_x-a_xb_z, a_xb_y-a_yb_x]$ Produto Escalar 3D entre a e b: $a_xb_x+a_yb_y+a_zb_z$

Gabarito:

1) pirâmide onde cada lado tem uma cor diferente (R,G,B,Y).

```
void main(){
    vec4 vert = gl_Vertex;
    vec3 color;
    float invertx = -vert.x;
       vec3 normalFace = vec3(0.0, 0.0, 1.0);
    if (vert.x > vert.y) {
        if (invertx > vert.y) {
            color = vec3(1.0, 1.0, 0.0); //Y
             vert.z = 1.0-abs(vert.y);
            normalFace = vec3(0.0, -1.0, 1.0);
        } else {
            color = vec3(1.0, 0.0, 0.0); //R
            vert.z = 1.0-abs(vert.x);
            normalFace = vec3(1.0, 0.0, 1.0);
        }
    } else {
        if (invertx > vert.y) {
            color = vec3(0.0, 0.0, 1.0); //B
            vert.z = 1.0-abs(vert.x);
                normalFace = vec3(-1.0, 0.0, 1.0);
        } else {
            color = vec3(0.0, 1.0, 0.0); //G
            vert.z = 1.0-abs(vert.y);
                normalFace = vec3(0.0, 1.0, 1.0);
        }
      }
    vec3 normal = normalize(gl_NormalMatrix * normalFace);
    vec3 lightDir = normalize(gl_LightSource[0].position.xyz - vert.xyz);
    float diffuse = max(0.0,dot(lightDir, normal));
    gl_FrontColor.rgb = (0.2 + 0.8 * diffuse) * color;
    gl_Position = gl_ModelViewProjectionMatrix * vert;
}
```

2a) (2 pontos)

[T]: translação para posição relativa ao olho (ou seja, a câmera é o 0,0,0)
[R]: rotação para o sistema de coordenadas do olho
[P]: projeção da perspectiva
[T_2]: centralizar o eixo de coordenadas

[S]: escala para que o frustum fique entre $[-1,1]^3$

2b) (1 ponto)
Antes: transformação das coordenadas do modelo para as coordenadas do mundo.
Depois: transformação para coordenadas de pixel

3)
a) Para coordenadas de texturas maiores que 1.0 a textura se repetirá indefinidamente no REPEAT. No CLAMP o último texel será estendido quando maior que 1.0.
b) BILINEAR: Interpolação usada para achar um valor no interior de um quadrado (linear em cada eixo). BARICÊNTRICA: interpolação usada no interior de um triângulo.
c) DIFUSA: componente principal de iluminação, resultante da incidência da luz na superfície. ESPECULAR: componente que considera a reflexão da luz na superfície.
d1) Cada triângulo rasterizado na placa de vídeo pode ocupar diversos PIXELS na imagem final, porém, eles ainda podem ser descartados no z-buffer e momentaneamente são chamados de FRAGMENTO.
d2) O fragmento é o pixel com a informação de profundidade. Na imagem final, vários fragmentos podem ser gerados para um único pixel.
e) AUTO-OCLUSÃO: Quanto partes do objeto podem ocludir outras partes do mesmo objeto. AUTO-SOMBRA: Quanto partes do objeto podem projetar sombras em outras partes do mesmo objeto.

4) (Atenção, os pontos ABC estão invertidos em relação a versão impressa da prova)

Cáculos intermediários:
$P_i = [4,1,1]$ (0.5 ponto)
Direção da luz (L_{dir}) em Pi : [4,1,9] - [4,1,1] = [0,0,8],
L_{dir} normalizando: [0,0,1] (0.5 ponto)
Normal do triângulo, N_{tri}: [0.25,0.25,0.5] (Produto vetorial entre os vetores BA e BC) (0.5 ponto)
Normal ponderada em Pi, N_{Pi}: [0.2,0.25,0.55] (0.5 ponto)

(até aqui 2 pontos, a partir daqui +0.5)

norma de N_{tri}: $\sqrt{3/8}$
norma de N_{Pi}: $\sqrt{0.405}$

No Flat: produto escalar entre L_{dir} e N_{tri} (dividido pela norma)
No Phong: produto escalar entre L_{dir} e N_{Pi} (dividido pela norma)

Respostas:
(a) Flat: 0.5 / $\sqrt{3/8}$ C
(b) Phong: 0.55 / $\sqrt{0.405}$C