P2 CG 2013.1

	Computação Gráfica 1 Prof. Rodrigo de Toledo	Data: 15/7/2013 P2 2013.1

1) (2,5 pontos) As curvas cúbicas paramétricas podem ser representadas matricialmente por $Q(t) = T \cdot M_{4x4} \cdot G$, onde $T = [t^3\ t^2\ t\ 1]$, G é um vetor com os 4 pontos e M é uma matriz 4x4 (calculada em sala de aula). A curva paramétrica, resultante da interpolação quártica de 5 pontos é dada por:

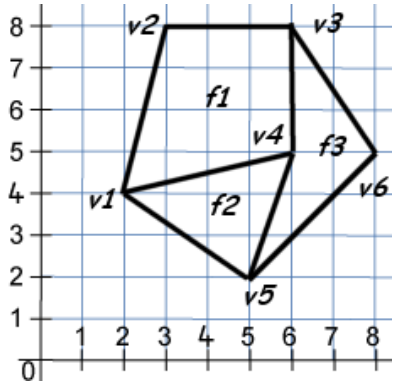
$P_{15}(u) = (1-u)^4 P_1 + 4u(1-u)^3 P_2 + 6u^2(1-u)^2 P_3 + 4u^3(1-u) P_4 + u^4 P_5$

Calcule os 25 coeficientes da matriz M_{5x5}, de $Q(t) = T \cdot M_{5x5} \cdot G$, onde $T = [t^4\ t^3\ t^2\ t\ 1]$ e G são os 5 pontos.

$$\begin{bmatrix} m_{11} & m_{12} & m_{13} & m_{14} & m_{15} \\ m_{21} & m_{22} & m_{23} & m_{24} & m_{25} \\ m_{31} & m_{32} & m_{33} & m_{34} & m_{35} \\ m_{41} & m_{42} & m_{43} & m_{44} & m_{45} \\ m_{51} & m_{52} & m_{53} & m_{54} & m_{55} \end{bmatrix}$$

$M_{5x5} =$

Lembrando que: $(a-b)^4 = a^4 - 4a^3b + 6a^2b^2 - 4ab^3 + b^4$, $(a-b)^3 = a^3 - 3a^2b + 3ab^2 - b^3$, e $(a-b)^2 = a^2 - 2ab + b^2$



2) (2 pontos) Preencha uma tabela com a estrutra Half-edge da malha ao lado. Cada registro (linha) dessa tabela deve conter as seguintes informações (colunas):
- half-edge (representada por he_{xy}, indicando ser uma half-edge que liga o vértice x ao vértice y), vértice de origem, face, half-edge oposta, half-edge seguinte.

Obs1: use a convenção anti-horária. Obs2: a face externa pode ser indicada por λ

3) (1 ponto) Quais outras tabelas são necessárias (com quais informações) para representar completamente essa malha?

4) (2,5 pontos) Ray-tracing (P2, 2012.1)
A função `bool = intersection(vec3 ray_origin, vec3 ray_direction, float radius, vec3 center, float &t1, float &t2)` retorna`verdadeiro/falso` se houve interseção entre uma esfera e um raio, assim como guarda os parâmetros dos pontos de interseção em `t1` e `t2`.
a) Quais linhas de código devem ser acrescentas na função `intersection` para que ela também calcule a normal do ponto de interseção mais próximo (guardando-o normalizado no parâmetro `&normal`), se houver?
b) Implemente a função `bool = intersectionIntersection (vec3 ray_origin, vec3 ray_direction, float radius1, vec3 center1, float radius2, vec3 center2, float &t1, float &t2, &normal)`; que retorna se houve interseção entre o raio e a interseção de duas esferas (CSG). A função também deve calcular o parâmetro do ponto de entrada e saída do raio, assim como a normal do ponto mais próximo).

5) (2 pontos) Explique cada um dos conceitos abaixo em no máximo três linhas por item!
(quatro linhas terá metade do ponto, cinco ou mais linhas será desconsiderado).

a) Continuous LOD (Level Of Detail); b) Tessellator; c) Impostores; d) Back-face culling

Gabarito:

1)

1 -4 6 -4 1
-4 12 -12 4 0
6 -12 6 0 0
-4 4 0 0 0
1 0 0 0 0

2)

HALF-EDGE	V ORIGEM	FACE	HE OPOSTA	HE SEGUINTE
he_{15}	v1	F2	he_{51}	he_{54}
he_{54}	v5	F2	he_{45}	he_{41}
he_{41}	v4	F2	he_{15}	he_{15}
he_{14}	v1	F1	he_{41}	he_{43}
he_{43}	v4	F1	he_{34}	he_{32}
he_{32}	v3	F1	he_{23}	he_{21}
he_{21}	v2	F1	he_{12}	he_{14}
he_{56}	v5	F3	he_{65}	he63
he_{63}	v6	F3	he_{36}	he34
he_{34}	v3	F3	he_{43}	he45
he_{45}	v4	F3	he_{54}	he_{56}
he_{12}	v1	λ	he_{21}	he_{23}
he_{23}	v2	λ	he_{32}	he_{36}
he_{36}	v3	λ	he_{63}	he_{65}
he_{65}	v6	λ	he_{56}	he_{51}
he_{51}	v5	λ	he_{15}	he_{12}

3)

Vértice	X	Y	Half-edge
v1	2	4	he_{15}
v2	3	8	he_{21}
v3	6	8	he_{36}
v4	6	5	he_{43}
v5	5	2	he_{54}
v6	8	5	he_{65}

Face	Half-edge

F1	he_{15}
F2	he_{43}
F3	he_{56}
λ	he_{51}

4)
a) 1.0
vec3 intersection = ray_origin + (t1*ray_direction);
normal = normalize(intersection - center);
b) 1.5

```
bool = intersectionIntersection (vec3 ray_origin, vec3 ray_direction, float radius1, vec3
center1,  float radius2, vec3 center2, float &t1, float &t2, &normal)
{
 bool esfera1 = intersection(vec3 ray_origin, vec3 ray_direction, float radius1, vec3 center1,
float &tmenor1, float &tmaior1, vec3 &normal1);
 bool esfera2 = intersection(vec3 ray_origin, vec3 ray_direction, float radius2, vec3 center2,
float &tmenor2, float &tmaior2, vec3 &normal2);
if (!(esfera1 && esfera2)) return false;
if ((tmaior1 < tmenor2) || (tmaior2 < tmenor1)) return false;
if (tmenor1 < tmenor2)
{
  &t1 = tmenor2;
  &normal = normal2;
}
else
{
  &t1 = tmenor1;
  &normal = normal1;
}
if (tmaior1 < tmaior2)
  &t2 = tmaior1;
else
  &t2 = tmaior2;
}
```

5)
a) Continuous LOD (Level Of Detail);
Malhas em diferentes níveis de detalhes (LOD) são usadas para otimização do rendering (quanto menor a projeção na tela, menor a necessidade de detalhamento). Continuous LOD, é uma técnica para variar o LOD ao longo do domínio do objeto (por exemplo, um terreno).
(Progressive Meshes: a malha altera suavemente no tempo, evita Pop-ups) ((mas aceito como resposta também)).

b) Tessellator;
É uma etapa programável (shader) do pipeline da placa de vídeo. Nessa etapa é possível criar novos vértices diretamente em GPU.

c) Impostores;
Técnica que substitui malhas por representações mais simples. Por exemplo, as árvores pode ser uma imagem com transparência em um billboard que gira na direção do observador.

d) Back-face culling
Técnica de otimização implementada nas placas de vídeo que não renderiza os polígonos vistos por trás.