11-11-2015

Práctica Opcional 2

Informática Gráfica

Sebastian Guisado Barreras y Susana Pineda De Luelmo

1. **Implementa la técnica del Bump Mapping:**
   El bump mapping es una técnica de gráficos computacionales en 3D que consiste en modificar las normales de un objeto para que este adquiera "rugosidad" en su superficie.

   Para implementar esta técnica contamos con las dos imágenes proporcionadas para la práctica: color y normal a partir de las cuales conseguiremos un textura rugosa.
   Para ello, lo primero que tenemos que hacer es pasar las normales al shader de vértices, ya que es el que tiene el primer acceso a ellos. Mediante las variables de entrada inNormal, inTexCoord e inTangent vamos a obtener una matriz (TBN) con la cual podamos pasar de coordenadas del mapa de normales a coordenadas del objeto. Por otra parte, tendremos que indicar una nueva salida del shader de vértices que será una matriz 3x3 (TBN).

   Una vez indicadas las nuevas entradas y salidas del shader de vértices pasamos a crear la matriz de cambio de base TBN. De forma matemática dicha matriz tendrá la siguiente forma:

$$TBN = \begin{pmatrix} Tan.x & Tan.y & Tan.z \\ Bin.x & Bin.y & Bin.z \\ Nor.x & Nor.y & Nor.z \end{pmatrix}$$

   Para poder implementar la formula lo primero que tenemos que hacer es declarar un nuevo vector de 3 coordenadas que recogerá nuestras tangentes, otro que recoja las normales y otro que tenga las binomiales.

```
vec3 Tangente = normalize (vec3(inTangent));
vec3 Normal = normalize (vec3(inNormal));
vec3 Binomial = normalize (vec3 (cross(Normal, Tangente)));
```

   A continuación pasamos a implementar la matriz:

```
TBN [0].x = Tangente.x;
TBN [0].y = Binomial.x;
TBN [0].z = Normal.x;

TBN [1].x = Tangente.y;
TBN [1].y = Binomial.y;
TBN [1].z = Normal.y;

TBN [2].x = Tangente.z;
TBN [2].y = Binomial.z;
TBN [2].z = Normal.z;
```

   Una vez que tenemos esta parte implementada en el shader de vértices pasamos al shader de fragmentos, donde lo primero que tenemos que hacer es declarar la nueva entrada que va a tener nuestro shader:

```
in mat3 TBN;
```

   Vamos a contar también con las variables unifor normalTex, modelView y model.
   Una vez en el main del shader de fragmentos vamos a cambiar el valor que teníamos definido para la variable N.

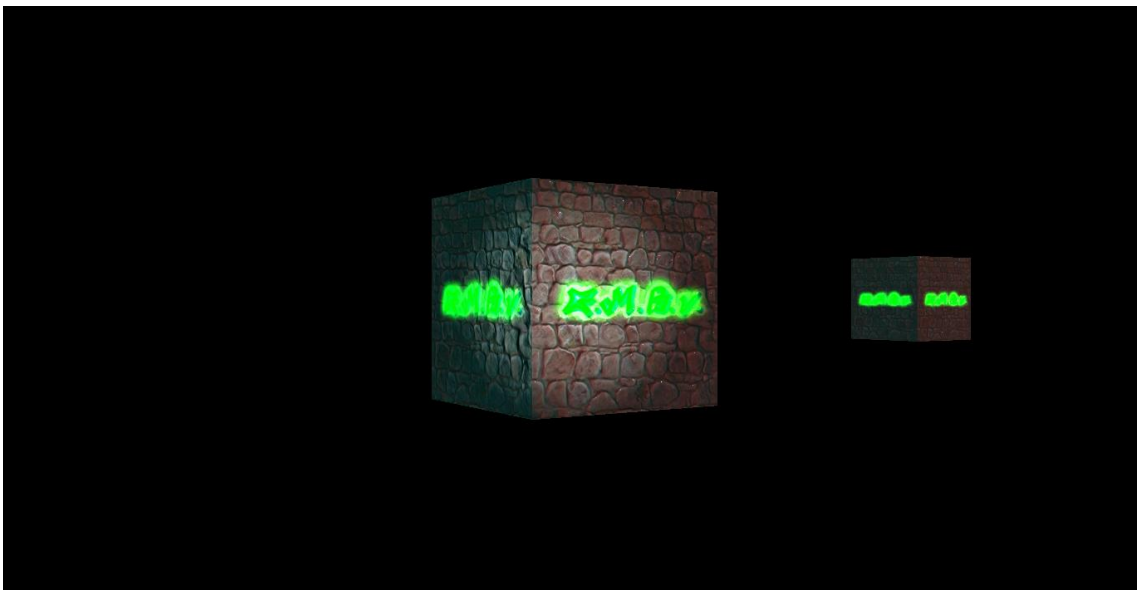
Para calcular las nuevas normales que vamos a asociar al objeto lo primero que tenemos que hacer es extraer las normales del mapa de normales, para ello utilizaremos el siguiente fragmento de código:

```
vec3 coordlocal = 2.0 * Normal.rgb - vec3(1.0);
```

Una vez que tenemos las normales de nuestro mapa de normales lo siguiente que tenemos que hacer es aplicar este cambio de normal a N, pasando de coordenadas del objeto a coordenadas de la cámara. Para ello realizamos las siguientes operaciones:

```
N = normalize (TBN*coordlocal*mat3(modelView));
```

Una vez que tenemos calculadas las normales de esta forma podemos dar por concluida la implementación del Bump Mapping.



NOTA: las capturas de pantalla no coinciden con los objetos realmente implementados en las prácticas por problemas al renderizar de uno de los ordenadores empleados en las prácticas.