Shaders en THREE-js

Nicolás David Rincón Pinzón - 6000358

Julián Andrés Villarraga Sarmiento - 6000303

Andrés Felipe Hernandez Lopez - 6000346

Proyecto Computación Gráfica

Gabriel Eduardo Ávila Buitrago

Universidad Militar Nueva Granada

Computación gráfica

23 / 09 / 2020

Introducción

En el presente informe se pretende usar la función ShaderMaterial para la creación de un código en **three.js** que permite simular el videojuego conocido como "Minecraft", por medio de una investigación que se evidenciará en el transcurso de esta documentación.

Para esto, se buscarán distintas fuentes que permitirán un cómodo aprendizaje de este material usado en JavaScript para la comprensión de Shaders.

Objetivos

General:

- Desarrollar un código en el lenguaje de JavaScript imitando las funciones básicas del videojuego "Minecraft".

Específicos:

- Estudiar, replicar y modificar líneas de códigos necesarias para desarrollar funciones que permitan el uso del Shaders Material como herramienta de texturas.
- Utilizar el entorno de threeJS como base para la creación de geometrías y estructuras requeridas.

Marco teórico

ShaderMaterial

Es un código escrito en GLSL que se usa para generar sombras sobre las texturas de un material. Un ShaderMaterial sólo será representado correctamente por WebGLRenderer. Es útil si se desea mejorar el rendimiento en el momento de combinar varios objetos en una misma geometría. **(1)**

Vertex shaders y fragment shaders. Ambos son dos tipos de sombreado en donde:

- *Vertex shaders recibe* atributos, modifica valores y pasa valores necesarios a fragment shaders.
- *Fragment shaders* se efectúa después del vertex shaders, donde establece el color de cada pixel.

Las variables que utiliza ShaderMaterial son las siguientes. **(1)**

- *Uniformes*: Son variables que permanecen iguales para cada vértice. Se puede acceder a este desde Vertex shaders y fragment shaders.
- *Atributos*: Pueden ser la posición del vértice, la cara normal y el color del vértice. Se asocia a cada vértice y solo se puede acceder desde fragment shaders.
- *Variables*: Aquellos que pasan del Vertex shaders al fragment shaders.

Constructor: ShaderMaterial(parámetros : Object). un objeto con una o más propiedades que definen la apariencia del material.

Propiedades: posee las mismas de la clase material, ejemplo: ".fog (niebla), .lights (luces), .vertexColors (color del vértice), etc.

Métodos: .clone () : ShaderMaterial this : ShaderMaterial Genera una copia superficial de este material.

RawShaderMaterial

Realiza las mismas funciones de ShaderMaterial pero con las definiciones de uniformes y atributos integrados no se anteponen automáticamente al código del sombreado GLSL. **(4)**

Ejemplo: https://threejs.org/examples/#webgl_custom_attributes_lines

```
var shaderMaterial = new THREE.ShaderMaterial( {

        uniforms: uniforms,
        vertexShader: document.getElementById( 'vertexshader' ).textContent,
        fragmentShader: document.getElementById( 'fragmentshader' ).textContent,
        blending: THREE.AdditiveBlending,
        depthTest: false,
        transparent: true
```

Imagen 1. Uso del ShaderMaterial en código

Imagen 2. ShaderMaterial implementado

Referencias

1. Tomado de: https://threejs.org/docs/index.html#api/en/materials/ShaderMaterial
2. Imagen 1 tomada de: https://github.com/mrdoob/three.js/blob/master/examples/webgl_custom_attributes_lines.html
3. Imagen 2 tomada de: https://threejs.org/examples/#webgl_custom_attributes_lines
4. Tomado de : https://threejs.org/docs/#api/en/materials/RawShaderMaterial