

[Practica 4: Aumento de realismo]

[Álvaro Gómez Haro y Alejandro Pérez Álvarez]



02 de octubre de 2015

**Parte obligatoria**

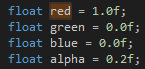
**Control del Motion Blur a través de teclado:**

Es posible controlar las componentes R.G.B y alfa del blending con las teclas (en orden) q, w, e, r para aumentar valores y a, s, d, f para disminuirlos.

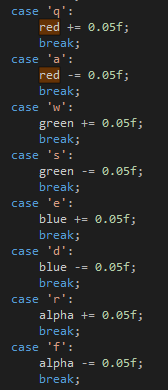
Los valores están normalizados.

Se modifican los valores introducidos en la función glBlendColor perteneciente al método renderFunc() a través de la keyboardFunc.

Inicialización de las variables (globales).



Se implementa el control de teclado a través de un switch.



Se suben los valores actualizados al shader desde el método renderFunc().



**Control del DOF por teclado:**

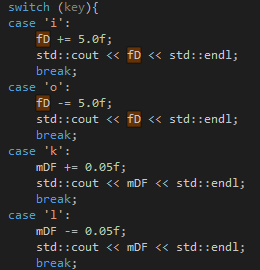
Se controlan la distancia focal y el límite de desenfoque con las teclas i, o respectivamente para aumentar valores y las teclas k, l para disminuirlos.

Se han sustituido las variables const por unas manejables por el usuario. Se pueden controlar asi valores que antes eran constantes.

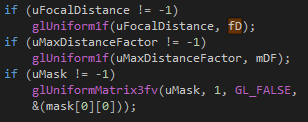
Los valores se inicializan como variables globales.



Se actualizan los valores desde la keyboardFunc() desde un switch.



En el método renderFunc() se suben los valores actualizados a los shaders.



**Uso del buffer de profundidad para controlar el DOF:**

Se determina el desenfoque se sufren los objetos en función de su posición en el buffer de profundidad.

Se inicializan las variables zNear y zFar y se crean las variables uniform que se subirán al shader de post proceso.







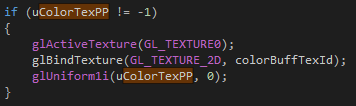
Actualizamos la matriz de proyección con los nuevos valores de zNear y zFar.

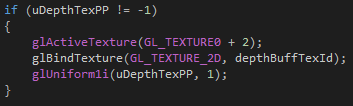


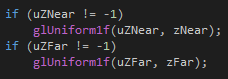
Inicializamos el shader con las variables uDepthTexPP y uColorTextPP.



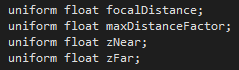
Se actualizan en el shader las variables uniform.







Se declaran en el shader de post proceso las variables.





Calculamos el desenfoque a aplicar a la coordenada de textura en función de los valores near y far de zBuffer.



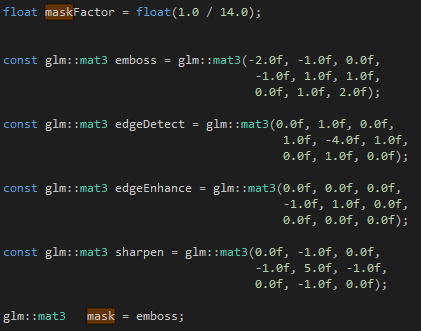
**Subida de nuevas máscaras de convolución a través de variables uniform y selección de las mismas a través de teclado.**

Se controla el cambio de máscara de convolución con las teclas z, x, c, v. Las máscaras implementadas son Emboss, Sharpen, Edge Detect y Edge Enhance respectivamente.

Es posible controlar el valor del factor de máscara con las teclas n, m para aumentarlo o disminuirlo.

Se inicializan las máscaras de convolución bajo forma de matriz 3x3 y el maskFactor.

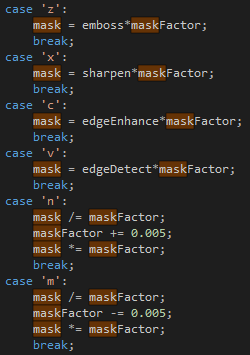
La matriz mask será la que se suba al shader. Se inicializa a la matriz emboss de forma arbitraria.



Se crea la variable uniform de mask, será la que se suba al shader.



Con el switch de la función keyboardFunc() se actualizan los valores de mask, igualándola a cualquiera de las cuatro matrices disponibles. Tambien se incrementa o decrementa el valor de maskFactor.



Al inicializar el shader se le indica como se va a subir la matriz mask.



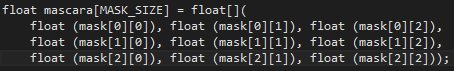
Se sube al shader de post proceso de fragmentos la matriz mask.



Ya en el shader se declara la variable mask.



Convertimos la matriz mask a array, para implementarla de la misma manera que se implementó el filtro gaussiano.



Se aplica la matriz mask a la escena.

