Tema 5 - Física y colisiones. Efectos especiales.

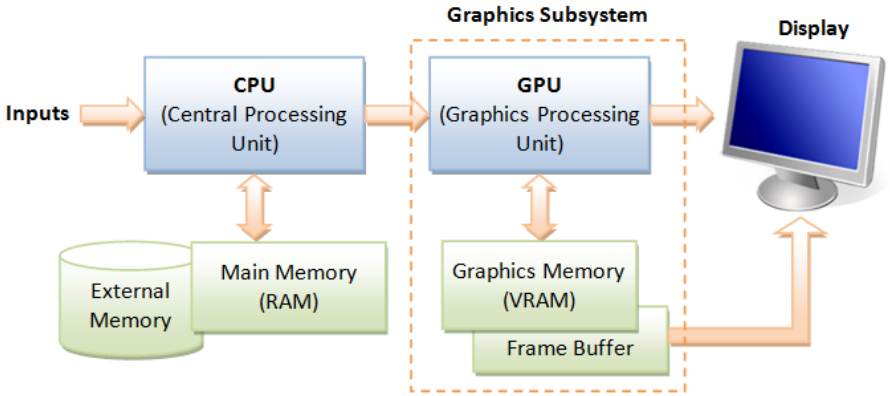
5.6 Shaders de vértices y técnicas avanzadas.

Germán Arroyo, Juan Carlos Torres

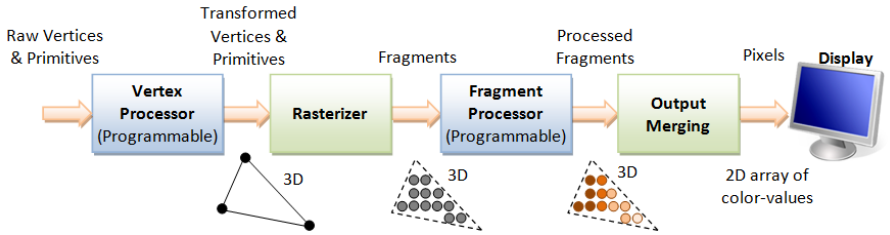
20 de mayo de 2021

Contenido del tema

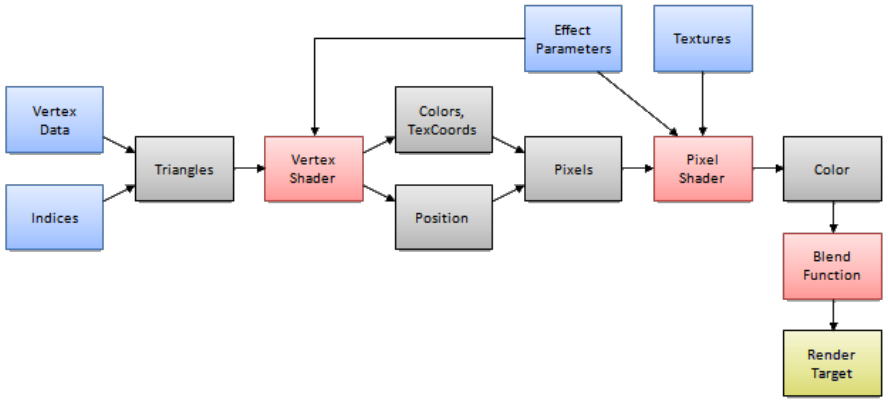
5.6 Shaders de vértices y técnicas avanzadas.



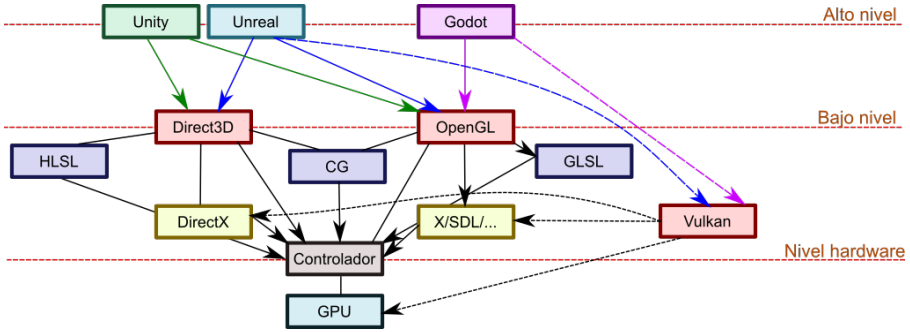
Esquema general de la GPU (I)



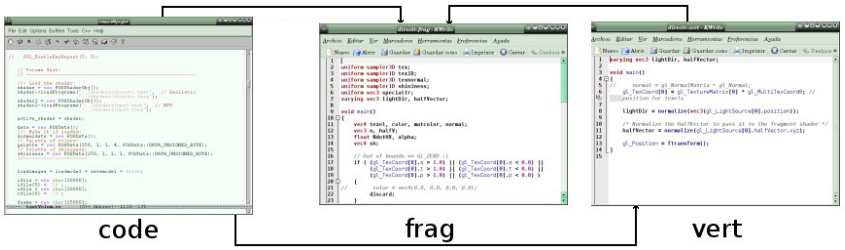
Esquema general de la GPU (II)



Alto y bajo nivel



Shaders a bajo nivel



Shaders a alto nivel

Distintos tipos:

- Material.
- Iluminación.
- Viewport/Canvas.
- Otros: partículas, propósito general, etc.

Ámbito de variables

Distinto ámbito:

Variables por vértices: Variables que se pasan desde CPU al *vertex shader*.
uniform: Variables que se pasan desde CPU.
varying: Variables que se pasan desde el shader de vértices al de fragmentos (interpolación lineal).
Variables locales: Variables que existen solamente en la función definida (típicas funciones estandar son *vertex*, *fragment* y *light*).
in: variables de solo-lectura de los shaders.
out: variables de solo-escritura de los shaders.
inout: variables de lectura y escritura intrínsecas a los algoritmos de shaders.

¡No hay variables de «salida» a CPU!

Tipos de variables

Tipos comunes (y no tan comunes):

vec2,3,4: vectores, vec3 y vec4 también sirven como colores. Se puede acceder a sus componentes: vector.x, vector.rg, vector.xyzw
mat4: matrices 4x4 (transformaciones geométricas).
float,int,bool: $1 \neq 1.0$
sampler1D,2D,3D: un mapa o textura, se puede acceder a ella mediante la función *texture*(sample2d, uv), devuelve un color de 4 dimensiones (rgba).

Lenguaje

Los lenguajes varían pero suelen imitar a C.

¡Cuidado! Los procesadores son poco potentes:

for, **if**, **switch-case**, deberían ser evitados siempre que sea posible…

Procesadores vectoriales:

Usar **vecX**, **matX**, **samplerXD**, siempre que se pueda…

Ejemplo: terreno procedural

Una textura de ruido (Gris) nos indica la elevación.
Una segunda textura (RGB) de ruido sirve para obtener las normales, o calcularla ?…
Una tercera textura (RGB) nos da el color del terreo, o la elevación ?…

https://docs.godotengine.org/en/stable/tutorials/shading/shading_reference/shading_language.html

https://docs.godotengine.org/en/stable/tutorials/shading/shading_reference/spatial_shader.html

Obtener información de las texturas: múltiples pasadas

La única forma de capturar datos es leer la textura resultante (GPU $\rightarrow$ CPU).

Es típico encadenar pasadas:

[PRIMERA PASADA] Preparar datos en CPU, transferir a GPU.
Hacer algoritmo GPU y visualizar textura.
Capturar textura (viewport) y transferir a CPU (o no).
[SEGUNDA PASADA] Utilizar textura como entrada para el nuevo algoritmo.
…

Sistemas de Partículas

Emisor: desde donde sale la partícula. Puede haber sub-emisores.
Función de actualización: donde se actualiza el estado/comportamiento de la partícula.
Vida de la partícula: tiempo en el que la partícula es visible y tiene comportamiento.
Dibujado: cada partícula puede ser cualquier objeto 3D con cualquier material.

https://www.youtube.com/watch?v=4VDNBTF9mu0

https://www.youtube.com/watch?v=aNVviTECNM0

Ejemplo de shaders y partículas

GPU + partículas:

- https://godotengine.org/article/improvements-gpuparticles-godot-40
- https://docs.godotengine.org/es/stable/classes/class_particles.html

Mapas de elevación y colisiones:

- https://godotengine.org/storage/app/media/4.0/particles/window.mp4

Mapas de elevación y hierba:

- https://www.youtube.com/watch?v=uMB3-g8v1B0
- https://github.com/BastiaanOlij/godot-grass-tutorial