CC92- FUNDAMENTOS DE LA TEORIA DE JUEGOS

Trabajo Final 2016-02

Profesor: Alexis Sotelo Guzmán.

Secciones: SWA1

Indicaciones

- El trabajo podrá ser realizado en parejas o de manera individual.
- El trabajo solo podrá ser desarrollado en Visual Studio 2013, queda bajo su responsabilidad el uso de cualquier otro compilador.
- El trabajo tendrá los siguientes hitos:
    - Primer Hito: Entrega de documentación de sistema elegido. La documentación incluye la investigación y diseño del sistema.
    - Segundo Hito: Avance del trabajo. El motor gráfico debe de cumplir con los requerimientos mínimos descritos.
    - Tercer Hito: Entrega final del trabajo y presentación.
- Para cada entrega se ha definido una rúbrica de evaluación. Al final del presente documento podrá encontrar la rúbrica.

Enunciado

El alumno debe analizar y diseñar el desarrollo de un subsistema de un motor de videojuegos en base a los requerimientos que le serán brindados en este documento. Una vez realizada esta etapa, el alumno debe implementar el subsistema en el lenguaje de programación C++. El Proyecto propone el desarrollo de uno de los siguientes sistemas: **Sistema de Partículas** o **Sistema de Animación piel-huesos**. Ambos sistemas deben de partir un **Motor Gráfico base** que todos los alumnos por igual deben de desarrollar.

Student Outcomes

Para la carrera, el curso contribuye a alcanzar los siguientes Student Outcome:
Student Outcome (CIENCIAS DE LA COMPUTACION - C) Diseña un sistema e implementa un programa, para alcanzar las necesidades deseadas, en un motor de juegos."
Student Outcome (CIENCIAS DE LA COMPUTACION - C) Define especificaciones y objetivos claros para un subsistema de juego, considerando restricciones realistas en un tiempo dado.

Los sistemas creados por el alumno partirán de la solución FTJ Engine vista en clase. El alumno deberá aplicar los conceptos aprendidos en clase a lo largo del desarrollo de sus sistemas. Una vez terminada, la tecnología creada por el alumno debe ser utilizada para crear una pequeña aplicación interactiva; en lo posible un videojuego.

Motor Gráfico
A continuación se explican a detalle el sistema de renderizado base que deben realizar **todos** los alumnos.

Manejador de Recursos
El motor debe ser capaz de interpretar archivos 3D en formato FBX. Estos deben ser luego convertidos en archivos binarios, los cuales deben ser cargados de manera asíncrona en las siguientes oportunidades.

Manejador de Render
El motor debe ser capaz de renderizar un mínimo de 12 objetos 3D en un mundo abierto con al menos 6 geometrías distintas. Estos objetos deben tener un iluminado básico con al menos una luz direccional.
El motor debe permitir manipular el movimiento de una cámara a través del Sistema de Input realizado anteriormente en clase.

Sistema de Partículas
Este es uno de los dos sistemas que el alumno debe elegir desarrollar.
El sistema debe implementar el patrón Flyweight y el patrón Object Pool. Además debe utilizarse un Geometry Shader para procesar la información de renderizado más rápido. El sistema debe ser capaz de crear múltiples emisores de partículas, todos mezclando pixeles de manera adecuada.

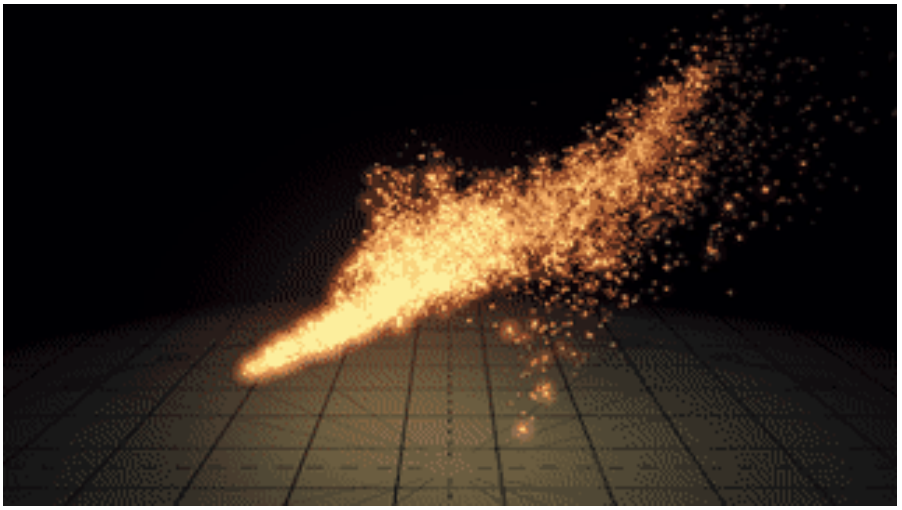

Figura 1 – Emisor de Partículas

Sistema de Animación

Este es el otro sistema que los alumnos pueden elegir desarrollar.

El sistema debe ser capaz de reproducir animaciones basadas en la técnica piel-huesos. Para ello se debe mejorar el Manejador de Recursos para poder leer información de animación en los archivos FBX. Luego la información debe ser guardada en archivos binarios para ser utilizada en ocasiones posteriores.

Por otra parte, la animación debe utilizar un Geometry Shader para procesar la información de renderizado más rápido. Asimismo se deben de mostrar al menos 3 animaciones distintas en simultáneo.

Entrega del Trabajo

Para que los alumnos demuestren con éxito el desarrollo de sus proyectos, se considera necesario que lleven un control de versiones utilizando Git.

Los alumnos pueden alojar sus repositorios en Github o Bitbucket según prefieran.

Es imperativo que realicen *commits* constantemente y suban su progreso al servidor para que el profesor pueda hacer un *Pull Request* y pueda hacer la corrección en cada uno de los hitos.

Hitos

El trabajo se ha dividido en 3 hitos.

1. **Primer Hito:**

Fecha: **Semana 11**

Sesión: Segunda sesión de clase de esa semana.

Objetivo:

Para este primer hito, el alumno deberá cumplir con lo siguiente:

- Diagrama UML del sistema escogido.
- Entrega de la documentación del sistema escogido, explicando las razones por las cuales se ha diseñado de esa manera.
- Link al repositorio creado por el/los alumno(s), para que el profesor puede realizar el seguimiento.

2. **Segundo Hito**

Fecha: **Semana 13**

Sesión: Primera sesión de clase de esa semana.

Objetivo:

Para este segundo hito, el alumno deberá cumplir con lo siguiente:

- Haber terminado con el hito 1.
- Tener, como mínimo, lo siguiente:
  - Tener un Manejador de Recursos que lee archivos FBX, los convierte a binario y luego utiliza en próximas ocasiones. La lectura de los archivos se hace de manera asíncrona.
  - El Manejador de Render es capaz de mostrar múltiples (mínimo 12) modelos 3D al mismo tiempo con geometrías distintas.
  - Es posible manipular la cámara con el teclado y mouse.

3. **Tercer Hito**

Fecha: **Semana 15**

Sesión: Segunda sesión de clase de esa semana

Objetivo:

Para este tercer hito, el alumno deberá cumplir con lo siguiente:

- Haber terminado con el hito 1 y 2
  - Completar su Sistema de Partículas o Animación según corresponda.
  - Presentar su trabajo y explicar la tecnología creada.

Rúbrica de Evaluación - Primer Entregable (3 Puntos)

	Nivel 1	Nivel 2	Nivel 3
Sustentación	Los alumnos pueden explicar, sin mayor inconveniente, el diagrama realizado y responden las preguntas de su profesor sin problemas. Demuestran conocimiento del trabajo realizado.	Los alumnos explican con inconvenientes, el diagrama que han realizado y/o NO responden las preguntas de su profesor. No Demuestran conocimiento del trabajo realizado	Los alumnos no desean explicar su diagrama y/o desconocen el mismo. No hicieron la aplicación. No usaron VS2013
	3	1	0

Rúbrica de Evaluación - Segundo Entregable (5 Puntos)

	Nivel 1	Nivel 2	Nivel 3
Sustentación	Los alumnos pueden explicar, sin mayor inconveniente, el código que han realizado y responden las preguntas de su profesor sin problemas. Demuestran conocimiento del trabajo realizado	Los alumnos explican con inconvenientes, el código que han realizado y/o NO responden las preguntas de su profesor. No Demuestran conocimiento del trabajo realizado	Los alumnos no desean explicar su código y/o desconocen el mismo. No hicieron la aplicación. No usaron VS2013
	0.5	0	0
Manejo de Recursos	El sistema creado por el alumno es capaz de leer archivos 3D en formato FBX y los exporta en archivos binarios. Usa multi-threading para el leer archivos de manera eficiente.	El sistema es capaz de leer archivos FBX, mas no utiliza archivos binarios o multi-threading para mayor eficiencia.	No existe uso de archivos binarios ni de multi-threading en el manejo de archivos. No lee archivos en formato FBX
	2.5	1.5	0
Motor Gráfico	El motor gráfico muestra objetos multiples objetos tridimensionales en una escena de juego. La aplicación posee una cámara que se puede manipular en todas direcciones (primera persona).	Al menos se muestra un objeto tridimensional en una escena. El movimiento de la cámara es limitado.	No es posible mover la cámara. No se pueden visuales objetos 3D.
	2	1	0

Rúbrica de Evaluación - Tercer Entregable (12 puntos)

	Nivel 1	Nivel 2	Nivel 3
Exposición	El alumno puede explicar, sin mayor inconveniente, el código que han realizado y responden las preguntas de su profesor sin problemas. Demuestran conocimiento del trabajo realizado	El alumno explica con inconvenientes, el código que han realizado y/o NO responde las preguntas de su profesor. No demuestra conocimiento del trabajo realizado	Los alumnos no desean explicar su código y/o desconocen el mismo. No hicieron la aplicación.
	2	0	0
Sistema de Párticulas	La aplicación presenta las siguientes características: • Implementación utiliza el patrón Flyweight • Implementación utiliza el patrón Object Pool • Uso de un Geometry Shader para cálculos acelerados con la tarjeta gráfica. • Se pueden apreciar varios emisores de partículas en sola escena sin afectar los cuadros por segundo, FPS. • Los pixeles de las partículas se mezclan adecuadamente con el resto de objetos pintados.	La aplicación presenta alguna de las siguientes deficiencias: • Implementación no utiliza el patrón Flyweight o patrón Object Pooling de manera adecuada. • La implementación no utiliza un Geometry Shader. • Las partículas no se mezclan apropiadamente.	No hizo el sistema o no funciona.
	8*	5*	0
Sistema de Animación	La aplicación presenta las siguientes características: • Uso de un Geometry Shader para cálculos acelerados con la tarjeta gráfica. • Los archivos de animación se guardan en formato binario y se utilizan en ocasiones posteriores. • Es posible visualizar 3 animaciones en simultáneo sin afectar el FPS.	El sistema permite visualizar animaciones, pero estas afectan considerablemente el FPS o no usan un Geometry Shader.	No hizo el sistema o no funciona.
	8*	5*	0
Funcionalidad adicional	El sistema tiene una funcionalidad adicional. La funcionalidad adicional funciona bien el 100% de veces.	El juego tiene una funcionalidad adicional. La funcionalidad adicional no funciona bien el 100% de veces	No hizo la funcionalidad adicional o no funciona. No usaron VS2013
	2	1	0

*Se aplica una sola nota del sistema elegido por el alumno.