UTN.BA
UNIVERSIDAD TECNOLÓGICA NACIONAL
FACULTAD REGIONAL BUENOS AIRES

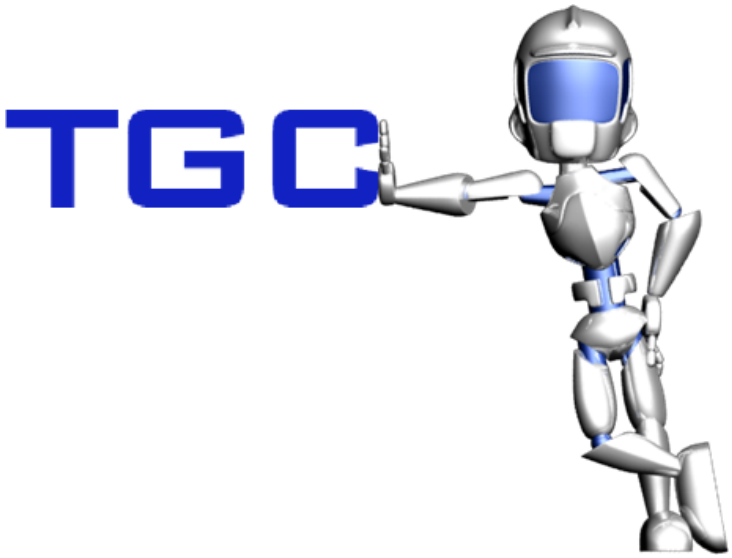
Técnicas de Gráficos por Computadora

Enunciado del Trabajo Práctico

Última revisión: 05 de Abril de 2022

Contenido

Trabajo Práctico



Introducción

El trabajo práctico de este cuatrimestre consiste en desarrollar un *Ejemplo Creativo* usando la siguiente plantilla. En el mismo, el alumno debe integrar todas las técnicas que vio en la cursada, cumpliendo con ciertas restricciones básicas pero aplicando su imaginación para elegir la temática global.

Tendrán que seleccionar el tema del desarrollo de una lista de opciones disponibles, donde cada una presentará los lineamientos y condiciones que deberán ser cumplidos y evaluados.
La cátedra provee un conjunto de ejemplos que sirven de base al alumno. Estos ejemplos utilizan muchos modelos 3D y texturas que se ubican en la carpeta Content. El alumno puede hacer uso de todos estos modelos para su ejemplo como así también puede incorporar otros.

Muchas de las técnicas a utilizar son desarrolladas en forma individual en cada uno de los ejemplos provisto por la cátedra. **Es tarea del alumno comprenderlos, dominarlos y lograr su integración para desarrollar su propio ejemplo**.

Creación del ejemplo

Deben hacer un fork del proyecto TGC.MonoGame.TP que les servirá como plantilla para desarrollar su propio ejemplo. Una vez realizado el fork deben cambiarle el nombre (TGC.MonoGame.TP) por el nombre de su grupo. De esta manera se podrá identificar el proyecto de cada grupo.
El formato especificado por la cátedra para el nombre del proyecto es:
<Año>-<Cuatrimestre>-<Curso>-<NombreDelGrupo>, donde:

- El año deberá tener cuatro dígitos.
- Cuatrimestre deberá ser ‘1C’ o ‘2C’, según corresponda.
- El curso será el código del curso, sin la letra K antecedente.
- NombreDelGrupo deberá tener como máximo 20 caracteres.

P.E. 2016-2C-3072-LosBorbotones.
Deben respetar el formato.

Ejemplo Creativo

Para el *Ejemplo Creativo*, los alumnos deberán elegir alguna de las ideas básicas propuestas por la cátedra. Cada una plantea una serie de objetivos y desafíos a cumplir, junto con los requisitos mínimos para poder aprobar el TP. El equipo deberá implementar una única idea, que una vez seleccionada, no podrá ser cambiada.
Cada alternativa también enuncia la funcionalidad obligatoria que debe ser implementada para poder aprobar el trabajo práctico, así como también los ítems opcionales que ayudan a elevar la nota del TP. Además, es posible que el alumno agregue otros ítems bajo su propia iniciativa.

Ítems comunes a todas las ideas

Para cualquier idea que se elija se deberán implementar funcionalidades básicas que son obligatorias:

- **Buena calidad gráfica**: El trabajo práctico es una aplicación gráfica 3D, por lo tanto debe “verse bien”. Esto significa que:
  - Debe tener un escenario creado acorde a la necesidad del TP, con la mayor cantidad posible de objetos, distribuidos y ubicados correctamente según la temática elegida. Por ejemplo si se trata de un escenario del tipo outdoor, ubicar una gran cantidad de árboles, plantas, arbustos, pasto, piedras, etc. Si es posible agregar movimientos complejos a dichos objetos, como el efecto del viento sobre el pasto. La calidad gráfica depende del entorno logrado: cuantos más objetos y animaciones se logren, mejor ambientado quedará el escenario. Se deben utilizar los mesh provistos por la cátedra para, mediante técnicas de optimización, lograr dibujar la mayor cantidad de objetos posible buscando la sensación de escenarios complejos. **No es requerido el diseño de nuevos modelos (NO se evalúa el diseño. Somos programadores, no somos diseñadores)**. Es condición suficiente la utilización de los mesh existentes o los que puedan obtener de los recursos provistos por la cátedra.
  - Todas las texturas del escenario, objetos y personajes deben estar adecuadamente seleccionadas y con la proporción adecuada. Por ejemplo, si estamos modelando el suelo de una cancha de fútbol es necesario utilizar una textura de pasto, pero su apariencia no puede estar toda estirada o deformada por no concordar con la proporción del piso. Tampoco deben tener marcas de agua ni otros defectos que arruinen la ilusión gráfica.
- **HUD**: scoreboards, mapas, previews, indicadores de energía, gadgets, etc. Además dichos textos, menús y controles del TP deben estar correctamente centrados y ubicados en la pantalla, con fuentes y colores que sean compatibles con la temática del TP.
- **Shaders**: El ejemplo debe hacer uso intensivo de los shaders. Esto implica codificar nuevos shaders, utilizar los existentes e integrar las diferentes técnicas que se estudian en la materia, como por ejemplo efectos de postprocesado, modelos de iluminación, sombras, efectos de partículas, etc. El shader utilizado debe implementarse en lenguaje HLSL con Shader Model 3.0.

- **Performance**: el TP debe funcionar al menos a 60 FPS durante toda su ejecución (a excepción de la carga inicial). El alumno deberá hacer principal hincapié en lograr una buena performance. Si un TP cumple toda la funcionalidad pero no alcanza los requisitos mínimos de performance no será aprobado. Los 60 FPS deben lograrse en una computadora con las siguientes características promedio:
    - Notebook de cuatro núcleos de 1.8 ghz.
    - Placa de video Intel integrada con soporte de Shader Model 3.0
    - 4 GB de memoria RAM.
- **Estructura**: El TP deberá contar con una cierta estructura lógica: como mínimo una presentación, luego un menú principal en donde se puedan configurar las distintas opciones, un desarrollo propiamente dicho y un final (se gana, se pierde o se termina el tiempo). Los comandos deben estar disponibles en la pantalla de presentación, o en alguna opción de ayuda. El TP se tiene que poder usar sin leer ningún instructivo externo: simplemente ejecutando el mismo, el usuario debe ser capaz de seguir todos los pasos.
- **Jugabilidad:** Los ejemplos, además de aplicaciones 3D, incluyen la noción de ser juegos en sí mismos, por lo que la fluidez de la aplicación y una clara condición de victoria son esenciales para la evaluación del ejemplo de aplicación. En cada entrega debe haber una descripción de jugabilidad (que es lo que se puede hacer en esa entrega) y cómo se controla el personaje para mejor entendimiento del tutor.
- **Modo God**: para facilitar la presentación es conveniente que se pueda activar un modo "god" o pasar directamente a distintas etapas o niveles del juego.
- **Adaptable:** Teniendo en cuenta que el proyector puede tener distinta resolución (800x600 los más antiguos), el TP tiene que ser "relativamente responsive" y adaptarse a varias resoluciones. Esto quiere decir que no hay que poner posiciones fijas en píxeles que dependan de una resolución particular. Por ejemplo, imaginemos que queremos ubicar una imagen 2D que indique la cantidad de puntos que acumuló nuestro personaje en el juego. Queremos ubicar esta imagen arriba en el medio de la pantalla. Una primera opción sería utilizar valores de posición clavados:

  ```
  spritePuntos.Position = new Vector2(200, 0);
  ```

  En nuestra pantalla de la PC puede parecer que “200” es justo la posición del medio deseada, pero en otra pantalla (por ejemplo la de un proyector) no será así. La forma correcta es ubicarlo respecto a las dimensiones de la pantalla:

  ```
  var width = GraphicsDevice.Viewport.Width;
  spritePuntos.Position = new Vector2(width / 2, 0);
  ```

Ideas posibles

Third Person Platform - Marble It Up!

Video: Third Person Platform - Marble It Up!

- **Funcionalidades obligatorias para 2ra entrega:**
  - El jugador va a ser una bola que avanza, retrocede, se mueve a los lados y salta. Tener en cuenta que es una esfera que rueda sobre una superficie, esta tiene que rodar dependiendo la dirección de movimiento.
  - La cámara es en tercera persona enfocada en la esfera y deberá orbitar alrededor de la misma utilizando el mouse.
  - Crear un nivel, pueden ser más, un circuito con puntos de control que el jugador debe seguir con límites fuera de estos el jugador cae y deberá volver al último punto de control. Debe estar lo suficientemente poblado de elementos.
  - El circuito deberá tener obstáculos fijos y móviles.
  - Debe tener varios power ups de velocidad y vuelo que incrementa momentáneamente la velocidad de movimiento de la bola o le permite desplazarse a mayor distancia en el aire.
  - Deben existir colisiones entre el jugador, obstáculos y el entorno, tener en cuenta la gravedad y a qué objetos afecta.
  - Debe haber por lo menos tres tipos de bola: piedra, metal, plástico, goma, etc. cada una debería tener propiedades particulares por ejemplo: la de goma salta más que el resto, la de metal es la más rápida.
  - El entorno debe tener un skybox.

- **Funcionalidades obligatorias para 4ta entrega:**
    - Cada superficie debe recibir luces, y cada material deberá reaccionar de manera correcta a la iluminación.
    - El checkpoint, el arranque y el final deberá tener un efecto que muestre dónde está cada uno.
    - Cada tipo de bola tiene que tener distintos tipos de apariencia por ejemplo la bola de goma tiene que ser más áspera.
    - La mayoría de las bolas deben, dependiendo del tipo, aplicar un Enviroment Map.
    - Agregar un Shadow Map.

- **Funcionalidades opcionales:**

    - Agregar algún efecto de Blur momentáneo o alguna distorsión en pantalla al agarrar un power up o cruzar el final.

    - El powerup deberá tener un efecto de rotación y estar contenido en una esfera translúcida.

WarShip - World of Warships / Barquitos

Video: World of Warships (2020) - Gameplay (PC HD) [1080p60FPS]

- **Funcionalidades obligatorias para 2ra entrega:**
    - Construir un escenario en donde un bote se encuentra en el medio del océano, rodeado de agua, en medio de una tormenta.
    - El bote se tiene que poder desplazar por el agua, con los siguientes movimientos:
        - Aceleración

        - ▪ Desaceleración
        - ▪ Virar
    - El agua debe tener manera, con olas grandes que suben y bajan en tiempo real.
    - El bote debe adaptarse en tiempo real a la marea. Deberá inclinarse correctamente para adaptarse a la superficie del agua en donde se encuentra.
    - La velocidad de desplazamiento del bote deberá variar según qué tan inclinado se encuentre en el agua:
        - ▪ Cuesta arriba debe avanzar más lento.
        - ▪ Cuesta abajo debe avanzar más rápido.

- **Funcionalidades obligatorias para 4ta entrega:**
    - Hacer principal hincapié en lograr realismo en la calidad del agua:
        - ▪ Utilizar iluminación dinámica para el sol.
        - ▪ Aplicar reflejos sobre el agua.
    - Debe haber un efecto de lluvia simulando una tormenta fuerte.
    - Agregar otro bote con Inteligencia Artificial que persigue al jugador, dispara y hace daño.
    - Los barcos deben tener un efecto metálico.
    - Se debe poder ver el fondo del mar, el mar debe tener transparencia o refracción.

- **Funcionalidades opcionales:**
    - Agregar un efecto de truenos que ponen en blanco la pantalla momentáneamente.
    - Agregar islas, y que el agua del mar golpee contra la orilla de la misma.

Derby de Demolición Isométrico - Crash of Cars

Video: Demolition Derby Destruction | Crash of Cars

- **Funcionalidades obligatorias para 2da entrega:**
    - Construir un escenario en donde hay un auto manejado en vista isométrica por el usuario dentro de un escenario grande y cerrado. Los autos no pueden salir de este.
    - El auto debe poseer los siguientes movimientos:

        - Acelerar, frenar e ir en reversa.
        - Doblar.
        - Saltar
        - Las ruedas del auto deben rotar acorde al movimiento del auto.
    - El auto debe tener detección de colisiones contra los objetos del escenario (paredes, otros autos, etc). La colisión deberá tener en cuenta los siguientes aspectos:
        - Ser precisas considerando la vista isométrica.
        - Adaptarse de la mejor manera posible al volumen de la malla del auto.
        - Tener en cuenta la rotación de la malla.
        - No podrá ser utilizada una solución de detección de colisiones simple con AxisAlignedBoundingBox o BoundingSphere (los obstáculos sí pueden ser implementados con AxisAlignedBoundingBox)
        - Al chocar contra un obstáculo, se debe dar una respuesta de colisión apropiada, según la velocidad y ángulo de choque del auto.
    - La cámara debe seguir al coche principal.
    - Agregar power ups (items) que le den al jugador distintas habilidades temporales y armamento:
        - Turbo
        - Misiles
        - Ametralladora

- **Funcionalidades obligatorias para 4ta entrega:**
    - Incorporar otro auto con Inteligencia Artificial que persigue al primero e intenta chocarlo.
    - Los power ups y armas deberán tener una animación de rotación y rebote además de un brillo intermitente característico de cada elemento.
    - Se debe mostrar el tiempo y el marcador por pantalla.
    - Se debe agregar iluminación en los faros delanteros, puede ser una iluminación tipo Point Light.
    - El auto debe tener efectos gráficos.
        - Reflejos (Shader de Environment Map).
        - Aplicar un efecto de brillo sobre la superficie del auto Bloom.

- **Funcionalidades opcionales:**

    - Agregar un “estadio” con gradas y personas para dar una mayor inmersión al entorno.
    - Crear algún sistema de partículas que muestren humo, fuego, chispas, etc.

Consideraciones adicionales

Guía de temas

Cada idea de trabajo práctico requiere conocer distintos temas que se tratan en la materia. Los temas son tratados en las clases a lo largo de todo el cuatrimestre. El alumno no debe esperar a

que llegue la clase de un tema puntual que necesita. Deberá avanzar por su propia cuenta en los temas que deba incursionar para poder desarrollar el TP.
A continuación se brinda un panorama de conceptos generales requeridos en todos los TP junto con la unidad que los explica:

- **Terrenos con Heightmap**: es una forma de crear escenarios de un gran tamaño a partir de una imagen de escala de grises. El concepto es desarrollado en la Unidad 7 “Técnicas de Optimización” en la sección de escenarios “Outdoor”.

- **Escenario estático**: un escenario puede construirse mediante la carga de modelos 3D estáticos individuales. El concepto es desarrollado en la Unidad 3 “Conceptos básicos de 3D”. Los modelos estáticos a utilizar pueden ser obtenidos de las siguientes formas:
    - Creando un escenario completo en la herramienta de diseño 3D Studio MAX o Blender exportando en .obj o .fbx y luego cargar los modelos. ***Utilizar estas herramientas implica cierto conocimiento de diseño gráfico que no será enseñado en el transcurso de la materia***. También pueden aprovecharse modelos ya hechos que el alumno encuentre para 3Ds MAX o Blender y luego exportarlos.
    - **Animación**: existen distintas formas de lograr animación, movimiento o una combinación de ambos:
        - Animación esquelética: Cargar modelos animados mediante la técnica de animación esquelética. El concepto es desarrollado en la Unidad 5 “Animación”. La cátedra provee algunos modelos animados bajo esta técnica. El alumno puede crear nuevos modelos, o agregar más animaciones a modelos existentes mediante la herramienta 3Ds MAX o Blender, aunque su utilización puede no resultar trivial. Luego los modelos pueden ser exportados a formato fbx por ejemplo.
        - Animación por KeyFrames: Cargar modelos animados mediante la técnica de animación por KeyFrames. El concepto es desarrollado en la Unidad 5 “Animación”. Los resultados obtenidos mediante esta técnica son similares a la animación esquelética, aunque menos flexible y más performantes en algunas situaciones.
        - Transformaciones: aplicar transformaciones como traslación, rotación y escalado a objetos estáticos o animados para lograr movimiento. El concepto es desarrollado en la Unidad 2 “Conceptos avanzados de 2D” y luego profundizado en la Unidad 3 “Conceptos básicos de 3D”.

    - **Detección de colisiones**: para detectar si dos objetos chocan entre sí, determinar la selección del mouse o el impacto de un disparo se utiliza el concepto de detección de colisiones. El concepto es desarrollado en la Unidad 6 “Detección de Colisiones”.

    - **Shaders**: un shader es un programa escrito en lenguaje HLSL que se ejecuta directamente en la GPU. Se puede utilizar para muchas cosas, pero principalmente para lograr efectos especiales, como iluminación, sombras, reflejos, etc. El tema es tratado en profundidad en la Unidad 8 “Adaptadores de Video” y en el apunte “Guía de Shaders” de esa misma unidad.

Ideas avanzadas

La cátedra busca mejorar en caso de querer incursionar en ideas que poseen una dificultad avanzada. Aquellos grupos que deseen implementar una idea avanzada deberán plantearlo en

issues en el repositorio de TGC. Estas ideas tendrán menos soporte de los tutores por tratarse en sí de investigaciones fuera del alcance normal de la materia.
Algunas de ellas son:

- Occlusion culling de una ciudad con muchos edificios.
- Manejar gran cantidad de personajes animados en un mismo escenario, simulación de multitudes.
- Editor de escenarios con geometría constructiva y operaciones booleanas.
- Editor de personajes para poder cambiar aspecto, forma, color, rasgos, etc.
- Editor de animaciones esqueléticas dentro del framework.
- Implementar un Importer del formato Collada para el framework.

También pueden proponerse nuevas ideas que no estén en el listado expuesto anteriormente.

Herramientas de TGC

El link a los repositorios público de la cátedra es:

- https://github.com/tgc-utn

Esto puede ser útil para que aquellos que tengan deseo de aprender cómo se construyó puedan hacerlo, y a su vez aportar nuevas ideas de diseño.
También los repositorios en la parte de **Issues** pueden registrar todos los inconvenientes que detectaron o mejoras que crean posibles.

Entrega del TP

Grupos

Los grupos deberán estar formados por 4/5 integrantes como máximo y 1 como mínimo, sin excepción. Cada grupo debe tener un nombre que lo identifique.
Los grupos se inscriben a través de un formulario web ubicado en la sección de trabajo práctico del sitio de la materia:

- http://tgcutn.com.ar/trabajos/actual/

Los grupos deberán conformarse para antes del día de Cierre de grupos de TP. Ese día se encuentra especificado en la sección de Cronograma del sitio de la materia.
En el formulario web cargarán los integrantes del grupo y además elegirán la idea de TP a desarrollar. Aquellos alumnos que no lleguen a conformar un grupo para la fecha del Cierre de grupos de TP deberán realizar el TP en forma individual.
No habrá posibilidad de cambio de grupo luego del Cierre de grupos de TP, bajo ninguna excepción.

Entregas

El TP posee cuatro entregas formales vía email con tag de entrega y una presentación oral final.
En todos los casos se debe cumplir con la entrega en tiempo y forma para aprobación directa. **La aprobación del TP es condición necesaria para la aprobación de la cursada de la materia.**

- **Primera entrega:** Tendrá el objetivo de tener el escenario armado, sin animaciones y con los modelos seleccionados y ubicados donde corresponda, deberán ser un total de entre 100 a 200 modelos mínimo. Están disponibles los recursos comunes para cada temática.
- **Segunda entrega:** El avance mostrado debe contener todos los puntos obligatorios para esa entrega indicados en este enunciado para cada idea de TP.
- **Tercera entrega:** Tendrá como objetivo abarcar todo la funcionalidad esencial del juego, optimización, jugabilidad, HUD, menú con fondo 3D (una sola ventana) , música y sonidos.

- **Cuarta entrega:** El avance mostrado debe contener todos los puntos obligatorios indicados en este enunciado para cada idea de TP. Esta entrega pone énfasis en los shaders realizados.
- **Presentación oral:** En la última clase del cuatrimestre se hace una puesta en común de todos los trabajos prácticos. Cada grupo muestra el resultado de su TP. La exposición incluye mostrar funcionando en una computadora con proyector el TP desarrollado y explicar a los alumnos las decisiones de diseño y herramientas utilizadas. Esta define la nota del trabajo práctico. Obligatorio presentar la documentación.

Documentación

La presentación oral debe estar acompañada por:

- Video donde se muestra el gameplay.
- Fotos de los integrantes y datos de los mismos.
- Tres capturas de pantalla.

Formato de las Entregas

Para realizar una entrega deben crear una release en su repositorio, que debe respetar el siguiente formato “v1.0” donde el primer dígito significa que es la primera entrega y el segundo dígito la cantidad de veces entregada. Por ejemplo, si están re-entregando la primera entrega el nombre sería “v1.1”.
Recuerden que en el tag no sólo debe estar el código sino también todos los archivos media.
El nombre del proyecto debe corresponder con el nombre del grupo registrado en el sitio de la materia.
En caso de que el nombre posea espacios, se deberá especificar sin espacios o con guiones bajos.
Ejemplo: “Mi Grupo” puede ser “MiGrupo” o “Mi_Grupo”.
Se deben respetar las convenciones de nombres para los proyectos en .Net framework.
Revisar los Namespace de las clases para evitar problemas.
La entrega del trabajo práctico debe **notificarse** que está disponible por mail a la casilla de corrección de la cátedra.

Entregas del TP y exposición oral

La notificación de que la entrega está disponible deberá ser enviada por mail a la casilla de corrección de la cátedra:

- **tgc.entregas@gmail.com**

El asunto del mail deberá respetar la siguiente nomenclatura:

- **[TGC][20221C][<curso>][<NombreGrupo>] Notificación de Entrega <Número>**

Antes de enviar el mail el alumno deberá revisar meticulosamente que se cumplan todos los “Pasos de deploy del TP”
El release debe cumplir a la perfección el formato de entrega especificado anteriormente o será rechazado en forma inmediata.
En caso que algún alumno del grupo haya dejado de cursar deberá ser aclarado en el mail de la entrega del TP.
La entrega debe ser notificada vía mail el día que figura en el cronograma de la materia presentado en la página Web. Ese día se aceptan trabajos prácticos hasta las 23:59 hs. **Pasado ese horario no se aceptará ninguna entrega**.
Se recomienda prever estos temas con anticipación.

Nota del TP

Con la nueva resolución no existe el concepto de nota, se considera Aprobado a quien haya cumplido con todas las entregas en tiempo y forma. Sin embargo la presentación oral forma parte de la nota del alumno, en la misma se tienen en cuenta.

- Cumplir con todos los requerimientos obligatorios.
- Requerimientos opcionales agregados.
- Calidad del Ejemplo Creativo entregado.
- Calidad de la exposición oral.
- Entregar cumpliendo el formato de entrega y todas las normas de la cátedra.
- Entregar en tiempo respetando las fecha y horario de entrega.