

REVISTA

Blitz 3D

Comentarios y proyectos del
Foro de Videojuegos
Foros 3dPoder
www.foro3d.com

Programación
Videojuegos
Modelado en 3D
Y mucho mas...

Antes de "Pensar" en
crear un juego pasate por aca ...

Que engine uso?

INTRODUCCION AL BLITZ 3D

- * LOS PRIMEROS PASOS
- * TUTOR: NUESTROS PRIMEROS PROGRAMAS



evolucion...



Presentación

Hoy en Dia La oferta en el mercado para realizar videojuegos en 3D es muy alta, cada programa tiene sus pros y contras, pero sin duda uno de los mas sencillos a utilizar es el Blit Basic.

Hoy en día la programación de videojuegos necesita mas que nunca del modelado 3D y ahí también existe una gran cantidad de programas entre ellos el 3DSMax y su contraparte Blender, ambos com muy buenas razones para conocerlos.

Espero que esta revista sea de mucha ayuda para todos nosotros los que nos iniciamos en este mundo

Gracias

El Material de esta revista proviene del foro:
Blitz Basic 3D
en Videojuegos de 3DPoder

Participantes:

Cibertornic	El aprendiz 3d
Caronte	Pepius
Recc45	Estilo_sur
Crashon182	Rompecabezas
Whaitie	Juanyang
Krateos_29	Darkn

Hasta el momento de hacer esta edición.

CIBERTRONIC



Resident Evil

1 Antes de "Pensar" en crear un juego pasate por aca ...

Bueno, todo están muy ilusionados con crear juegos, pues aquí un par de cosas muy útiles a tener en cuenta

basic trabaj mejor con formato .X (yo uso drack basic poreso)

*Todo debe estar en tu mente pre diseñado o en un papel.

*Un Buen juego lleva mucho tiempo, si te fijas los créditos, cuanta gente estuvo para diseñar juegos como el nfs carbon u otros. (ellos tardan 1 año, solos tardamos de 5 a 20 años si no tenemos ayuda)

*Como dije antes, el tiempo, para hacer algo realmente profesional en 3d, necesitas mucho tiempo por cada pj, auto, o lo que sea, y darle una animación, para que cuando ocurra algo tome esa animación y parezca mas realista.

bueno suerte a todos!!!

*Aun no se que programa toma formato .max, 3ds creo que Dark basic pro, pero = es dudoso, y para colmo el DArk

Por: whaitie

La más importante: antes de emprender el difícil camino del desarrollo de videojuegos, aprende a escribir lo mejor que puedas; la comunicación será fundamental para que cualquier proyecto llegue a buen puerto.

Tomar como primer proyecto un clon del NFS Carbon equivale a intentar construir la Torre Eiffel. Como probablemente no seas arquitecto, pregúntate si no es mejor empezar intentando montar la caseta para el perro.

El equipo que reúnas difícilmente va a ser de élite (ni siquiera uno es de élite). Lo verdaderamente importante en un equipo que comienza es que todos se comprometan a dedicar una parte constante de su tiempo al proyecto.

Aléjate del 3D tanto como puedas en tus primeros proyectos; aprender la dinámica de un juego es de por sí complicada, e incluso reduciendo los gráficos a pequeños bitmaps (como en el pong) te vas a encontrar problemas. Si son imágenes animadas, tendrás una dimensión de problemas extra. Y ya con el 3D, ¿para qué quieres dos dimensiones extra de problemas?

Empieza por lo simple. Sigue con lo simple. Y termina con lo simple antes de pasar a otra cosa algo menos simple.

Cuando se comienza, olvídate del dinero.

Por: Mars Attacks



Si se desea hacer video juegos para Ganar dinero lo del tiempo si sera un gran problema. Pero si Tú al igual que Yo, estas en esto por aprender, el tiempo que demore solo dependerá de cuantas ganas le pongas y aun así, mientras mas tiempo te tome mas tiempo sera el placer de explorar

Hace unos años atrás hubo un juego llamado **ANOTHER WORLD** (buscalo en youtube) todo un juego de lo mas adictivo y con una calidad revolucionaria para su época, historia, gráficos jugabilidad, y sorpresa al ver los créditos solo una persona **Erick Chachi**, si el pudo hacerlo quien dice que nosotros no podamos, animo y a seguir adelante.

Obviamente Marz Tiene mucha razón, pero sera cosa nuestra hacer que esa verdad no lo sea para nosotros, si estamos aquí es porque queremos hacer las cosas en 3D, y según las ganas que le pongamos lograremos el objetivo de Iniciarnos en el 3D sin haber pasado por el 2D



Que engine uso?

1.- Que es un ENGINE?

Engine: Se usa en el lenguaje de los videojuegos para definir el tipo de entorno gráfico de un juego.

<http://es.geocities.com/sixerx/diccionarioxixerx.htm>

Game Engine o Motor de Juego hace referencia a una serie de rutinas de programación que permiten el diseño, la creación y la representación del juego.

La analogía con el motor de un automóvil es ilustrativa: el motor debajo del capot no es visible pero le da la funcionalidad al automóvil que es la de transportar. La misma analogía permite explicar algunos de los aspectos que generalmente maneja un motor de juego: las texturas y los modelos 3D serían la carrocería, pintura e interiores.

Del mismo modo en que carrocería, pintura y exteriores no andan sin un motor, el arte y los guiones del juego no funcionan sin un motor de juego. Un ejemplo de motor de juego sería el motor gráfico del juego Doom

Http://es.wikipedia.org/wiki/Motor_de_juego

Esta Claro?

Claro!

Aun no lo tengo realmente claro, es decir que podemos hacer nuestros juegos usando esos motores... Como se hace, como puedo usar el motor de Doom3 para hacer mis videojuegos.....

El día que lo sepa seguro que lo grito a los cuatro vientos, si tú lo sabes por favor escíbeme diciéndomelo.

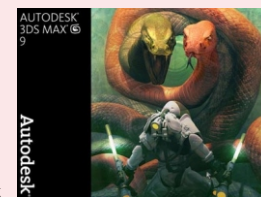


Que usaremos

Este Engine (como es llamado en su pagina oficial), es sencillo de usar y incluso puede servir de base para programar en C++, nosotros usaremos este lenguaje de programación por ser fácil y permitir resultados 3D desde los primeros ejercicios.

Este programa en definitiva es el mejor modelador 3d masivo, más orientado a videojuegos, con el que se han hecho enteramente juegos como la saga Tomb Raider y la saga Splinter Cell, y la mayoría de los juegos de Ubisoft. Incluido el magnífico Crysis y FarCry. Con él también se crearon películas enteras de animación como ToyStory, Los Increíbles, Buscando a los Robinson, y parte de Final Fantasy.

Http://es.wikipedia.org/wiki/3D_Studio_Max



2



ACTIVIDAD 1

Rotando una estrella

Objetivo: Controlar las cámaras y objetos utilizando las teclas del cursor

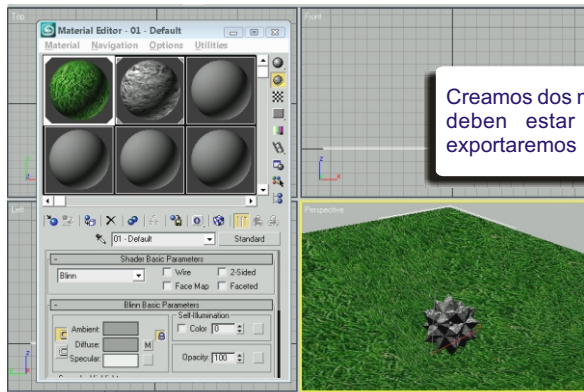
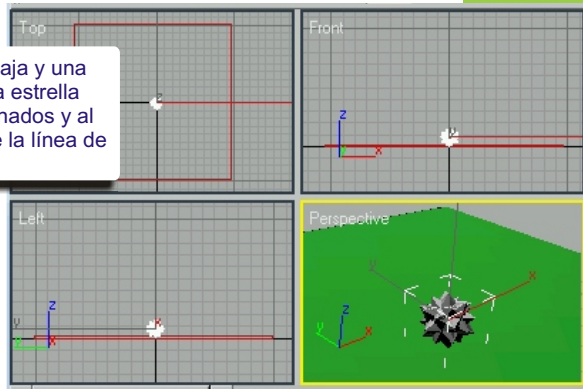
Escenario: Un campo verde

Actor: Una estrella

Dinámica: cuando pulsemos la tecla de cursor, la estrella empezara a girar desplazándose hacia adelante, al pulsar las teclas izquierda o derecha la estrella girara. en un primer momento nosotros veremos esto con una cámara estática situada de tal manera que nos permita ver todo el campo verde, y los movimientos que en el hagamos con la estrella

A. Creando el Escenario

En 3DS max creamos una caja y una estrella, luego ubicamos a la estrella en el centro de los ejes coordenados y al plano un poco mas abajo de la línea de la superficie.



Creamos dos materiales, no olvidar que las imágenes deben estar guardadas en la carpeta donde exportaremos el escenario y crearemos el juego

Luego exportas objeto por objeto (file>Exported Selected). El Box como **piso.3ds** y la estrella como **estrella.3ds**

En la misma carpeta donde están las texturas que sera la misma donde guardaremos el programa y crearemos el ejecutable.

▲ Asegura te tener a la estrella en el mismo origen de coordenadas.

B. El Programa: Camara estática

La ventana 3D

Graphics3D 640,480,16,2

Graphics3D crea una ventana de las dimensiones indicadas (640x480), con una profundidad de color de 16 bits, el ultimo parámetro 2 indica en modo ventana.

Cargar y crear los objetos

Piso=LoadMesh("piso.3ds")
Estrella=LoadMesh("estrella.3ds")

Camara=CreateCamera()
Light=CreateLight()

LoadMesh, nos permite cargar un objeto en este caso un objeto 3ds, los resultados son guardados en sus alias Piso y Estrella

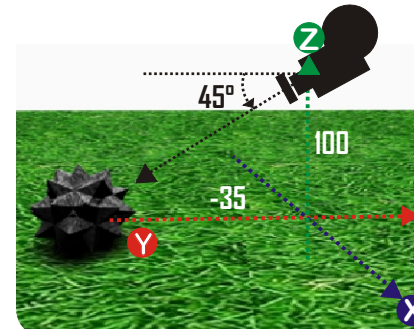
CreateCamera y Createlight crean la cámara y una luz.

Posicionando la camara

PositionEntity camara,0,100,-35
TurnEntity camara,45,0,0

PositionEntity mueve la cámara:
0 en X0 hacia los costados
100 en Y100 arriba
-35 en Z 35 atrás(-35)

TurnEntity Gira la cámara en el eje X 45° ocasionado un efecto de agachar la cámara



Graphics3D 640,480,16,2

piso=LoadMesh("piso.3ds")
estrella=LoadMesh("estrella.3ds")

camara=CreateCamera()
light=CreateLight()

PositionEntity camara,0,100,-35
TurnEntity camara,45,0,0

While Not KeyHit(1)

If KeyDown(200) Then
MoveEntity estrella,0,0,0.5
RotateMesh estrella,5,0,0
EndIf

If KeyDown(208) Then
MoveEntity estrella,0,0,-0.5
RotateMesh estrella,-5,0,0
EndIf

If KeyDown(203) Then
TurnEntity estrella,0,1,0
EndIf

If KeyDown(205) Then
TurnEntity estrella,0,-1,0
EndIf

UpdateWorld
RenderWorld
Flip

Wend

End

Actualiza y realiza un volcado del contenido a pantalla

Moviendo la Estrella

Si se presiona la tecla Up(flecha arriba):

If KeyDown(200) Then
MoveEntity estrella,0,0,0.5
RotateMesh estrella,5,0,0
End

KeyDown(tecla): Retorna Verdadero si se presiona la tecla.

MoveEntity objeto, lado, altura, avance

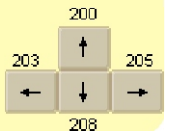
Lado: -izquierda, +derecha

Altura: -abajo, +arriba

Avance: -retroceder, + Adelantar

RotateMesh objeto, enX, enY, enZ: Rota al objeto sobre si el ángulo indicado para cada coordenada sin cambiar el sistema coordenado del objeto

TurnEntity Objeto, enX, enY, enZ:
Gira el objeto con todo y su sistema coordenado.



C. El Programa: Camara en movimiento

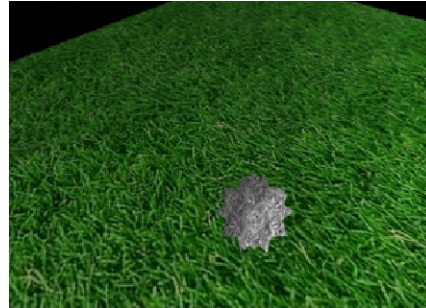
Para conseguir esto solo bastara modificar 2 lineas en nuestro código:

Camara=CreateCamera(bola)

Al momento de crear la camara le indicamos a que objeto deberá seguir de forma automática, esto solo poniendo el nombre del objeto entre los paréntesis del comando create camera.

PositionEntity camara,0,50,-25

En el código anterior la cámara estaba lejos del objeto para poder ver todo su movimiento, con este cambio acercamos la camara al objeto,



D. Asignando Variables

```
Graphics3D 640,480,16,2
piso=LoadMesh( "piso.3ds" )
bola=LoadMesh( "estrella.3ds" )
camara=CreateCamera(bola)
light=CreateLight()
PositionEntity camara,0,50,-25
TurnEntity camara,45,0,0
```

```
VelocZ#=0.5
Giray#=1
RotaX#=velocZ# * 10
```

```
While Not KeyHit(1)
  If KeyDown(47) Then ;Tecla v
    velocZ#=velocZ#+0.1
    If velocZ#>=1.5 Then velocZ#=0
    rotaX#=velocZ# * 10
  EndIf
  If KeyDown(200) Then
    MoveEntity bola,0,0,velocZ#
    RotateMesh bola,rotaX#,0,0
  EndIf
  If KeyDown(208) Then
    MoveEntity bola,0,0,-velocZ#
    RotateMesh bola,-rotaX#,0,0
  EndIf
  If KeyDown(203) Then TurnEntity bola,0,giraY#,0
  If KeyDown(205) Then TurnEntity bola,0,-giraY#,0
EndWhile
```

```
UpdateWorld
RenderWorld
Text 10,10,"velocidad:"
Text 10,25,velocZ#
```

```
Flip
Wend
End
```

Es mucho mas ventajosos asignar a todos los números una variable y definirla su valor al inicio del programa(inicializarla).

Esto para realizar cambios de forma mas fácil, por ejemplo para cambiar la velocidad, si no se inicializan las variables tendremos que revisar todas las lineas de código, buscando donde se realiza esto.

Pero si en cambio están inicializadas, solo bastara cambiar su valor al inicio y ya el resto del código trabajara con el valor indicado.

En el código se puede observar:

VelocZ#=0.5: para controlar la velocidad de desplazamiento(en Z).

GiraY#=1: Para el giro del objeto a la izquierda o derecha(en Y)

RotaX#=VelocZ# * 10: Para que a más velocidad, gire más rápido el objeto(en la formula gira 10 veces la velocidad del objeto)

Si se presiona la tecla V:

Velocidad aumenta en 0.1
Y si llega hasta 1.5 o mas regresa a 0

Text: Permite poner un texto, para ver en pantalla el valor de la variable Z, puede darse como parámetros las coordenadas y el texto ó variable a mostrar.



Código enviado por Cibertronic
Gracias especiales ha Estilo Sur
Foros 3DPoder

5

6 ACTIVIDAD II

Un nuevo Escenario

Objetivo: Mover la es la estrella por un escenario mas elaborado y activar el reconocimiento de colisiones



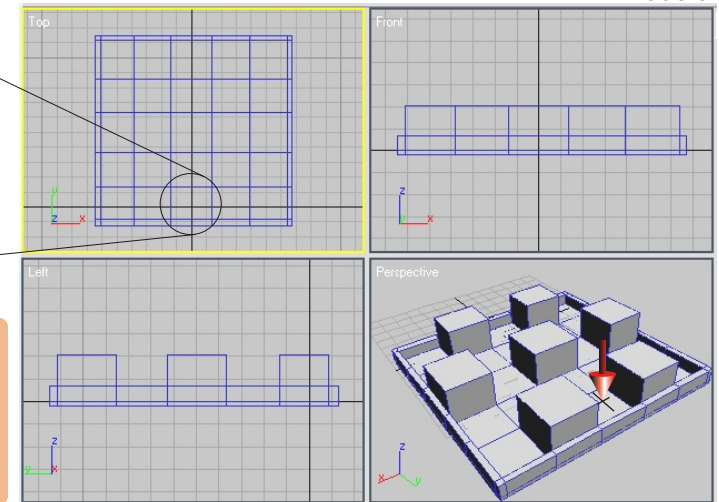
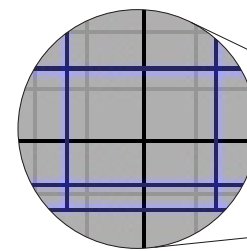
Escenario: Una ciudad

Aktor: Una estrella

Dinámica: La estrella deberá moverse por las calles de una ciudad y deberá chocar(colisionar) con las paredes de los edificios.

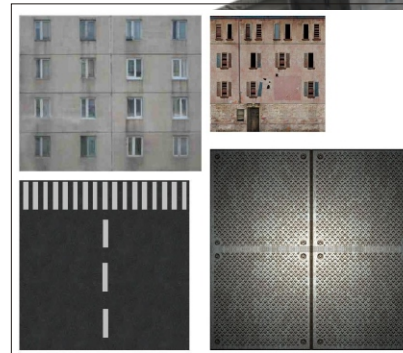
A. Creando el Escenario

El modelo



Al momento de dibujar y antes de exportar deberán de poner el centro de coordenadas en la posición indicada porque sera ahi donde salga la estrella.

La textura



Exportar a 3ds en la misma carpeta donde tenemos la stella y la textura.

B. El código

A Crear la ventana 3d

B Cargar y/o crear los objetos para el juego

C Asignar números a los tipos de objetos, esto para no tener que crear una colisión para cada objeto

Asignamos a cada objeto el tipo de objeto al cual pertenecerá.

D Inicializar variables.

Giray# y Rotax# son funciones directas de VelocZ#, es decir que a mayor velocidad de translación, mas rápidamente girara tanto sobre si mismo como sobre el eje y(doblar esquinas)

Inicializamos Xc, Yc, Zc, que será la posición de la cámara con respecto al objeto que sigue esta.

E Colocar la cámara detrás del objeto, girar la cámara(agachar) y definimos el radio de colisión del objeto(la distancia a la que chocara)

F Si se presiona la tecla V, aumentara la velocidad de translación, y con ello rotaX y GiraY, si la velocidad llega a 1 o mayor esta se reinicializa desde cero.

G Mover el objeto con las teclas del cursores

H Detectar colisiones entre los objetos tipo actor y escenario, tipo polígono y resbalar cuando se produce la colisión.

Collisions type1, type2, method, response

I Ubicar la cámara detrás del objeto, actualizar la pantalla, también se incluye mostrar el valor de VelocZ



Graphics3D 640,480,16,2 **A**

piso=LoadMesh("ciudad.3ds")
bola=LoadMesh("estrella.3ds") **B**

camara=CreateCamera(bola)
light=CreateLight() **C**

type_escenario=1
type_actor=2

EntityType piso,type_escenario
EntityType bola,type_actor
EntityType camara,type_actor **C**

VelocZ#=0.1
Giray#=velocZ# * 5
RotaX#=velocZ# * 10

Xc=0
yc=5
zc=-7 **D**

PositionEntity camara,xc,yc,zc
TurnEntity camara,15,0,0
EntityRadius bola,3 **E**

While Not KeyHit(1)

If KeyDown(47) Then
velocZ#=velocZ#+0.05
If velocZ#>=1 Then velocZ#=0
rotaX#=-velocZ# * 10
Giray#=velocZ# * 5
EndIf **F**

If KeyDown(200) Then
MoveEntity bola,0,0,velocZ#
RotateMesh bola,rotaX#,0,0
EndIf
If KeyDown(208) Then
MoveEntity bola,0,0,-velocZ#
RotateMesh bola,-rotaX#,0,0
EndIf **G**

If KeyDown(203) Then
TurnEntity bola,0,giraY#,0
EndIf
If KeyDown(205) Then
TurnEntity bola,0,-giraY#,0
EndIf **H**

Collisions type_actor,type_escenario,2,2 **H**

PositionEntity camara,xc,yc,zc

UpdateWorld
RenderWorld
Text 10,10,"velocidad:"
Text 10,25,velocZ#
Flip **I**

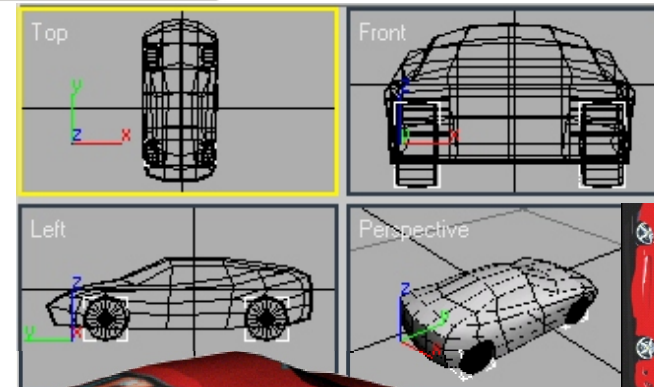
Wend
End

ACTIVIDAD III

Paseando en Coche

Objetivo: Poder animar un modelo de un coche y conducirlo por la ciudad, con las funciones propios de este tipo de programa acelerar, frenar, y desde luego colisionar con las estructuras.

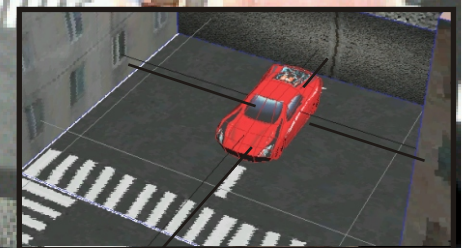
A. El Coche



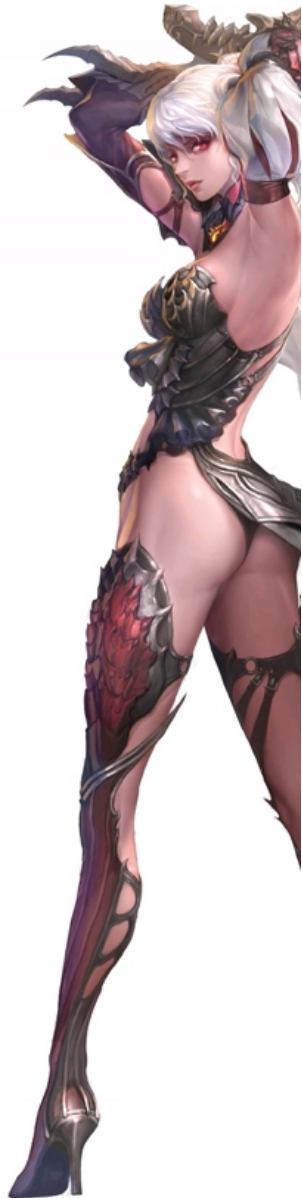
Aquí podemos observar el modelo del coche que utilizaremos, cedido por Recc45, lo importamos y texturizamos en la ciudad de la actividad II.



No olvidar que el coche deberá estar en las coordenadas 0,0,0 en el 3dsmax.



B. EL CODIGO



```

A Graphics3D 640,480,16,2

Light=CreateLight()
B piso=LoadMesh( "ciudad.3ds" )
    ferrari=LoadMesh( "carro.3ds" )
    camara=CreateCamera(ferrari)

    type_escenario=1 : type_actor=2

C EntityType piso,type_escenario
    EntityType ferrari,type_actor
    EntityType camara,type_actor

    EntityRadius ferrari,3

D TurnEntity camara,15,0,0
    Veloc#=0
    Acelera#=0.001

    While Not KeyHit(1)
        giro#=0

E     If KeyDown(200) Then
E         If Veloc#<=0.4 Then Veloc#=Veloc#+Acelera#
        EndIf

F     If KeyDown(208) Then Veloc#=Veloc#-Acelera#

    If Veloc#<>0 Then
G         If KeyDown(203) Then giro#=Veloc# * 3
        If KeyDown(205) Then giro#=Veloc# * -3
        EndIf

        If KeyDown(200)=0 And KeyDown(208)=0 Then
H             If Veloc#>0 Then
                Veloc#=Veloc#-0.01
            Else
                Veloc#=0
            EndIf
        EndIf

I     MoveEntity ferrari,0,0,Veloc#
        TurnEntity ferrari,0,giro#,0

J     PositionEntity camara,0,8,-10

K     Collisions type_actor,type_escenario,2,2

    UpdateWorld : RenderWorld : Text 10,10, Veloc# : Flip
    Wend
    End
    
```

- A** Graphics3D 640,480,16,2 Iniciar el modo 3D, en este caso una pantalla de 640x480 pix, con una profundidad de color de 16 bit y en modo de pantalla(2)
- B** Light=CreateLight()
piso=LoadMesh("ciudad.3ds")
ferrari=LoadMesh("carro.3ds")
Camara=CreateCamera(ferrari)
Crea y carga el escenario y los elementos de esté-
- D** TurnEntity camara,15,0,0
Veloc#=0
Acelera#=0.001
Ubica la camara e Inicializa las variables para Velocidad y Aceleración.
- C** type_escenario=1 : type_actor=2 Crear dos tipos de objetos para las colisiones.

EntityType piso,type_escenario
EntityType ferrari,type_actor
EntityType camara,type_actor Asignar a cada objeto un tipo

EntityRadius ferrari,3 Asigna al ferrari un radio de colisión igual a tres(depene del tamaño del objeto), menor se incrustaría dentro de las paredes, mayor antes de impactar a las paredes chocara con estas.
- E** If KeyDown(200) Then
If Veloc#<=0.4 Then Veloc#=Veloc#+Acelera#
EndIf Si se presiona felcha arriba
Si hay velocidad(se esta moviendo el coche), esta aumenta en el valor que tenia + la aceleración

If KeyDown(208) Then Veloc#=Veloc#-Acelera#
Si presiona flecha abajo, la velocidad disminuye, logrando con esto una aceleración en retroceso.
- F** Si hay velocidad(diferente de cero)
Si se presiona flecha izquierda la variable giro sera 3 veces el valor de la velocidad
Si se presiona flecha derecha, giro sera 3 veces la velocidad pero negativo.
Esto para conseguir giros mas rápidos cuando tangamos mas velocidad.
- G** If KeyDown(200)=0 And KeyDown(208)=0 Then
If Veloc#>0 Then
Veloc#=Veloc#-0.01
Else
Veloc#=0
EndIf
EndIf
Si no se presiona ni arriba ni abajo
Si hay velocidad, esta disminuye, sino (si la velocidad es negativa) queda en cero.

Con esto se consigue que al soltar las teclas el coche frene suavemente.
- I** MoveEntity ferrari,0,0,Veloc#
TurnEntity ferrari,0,giro#,0 Mueve la ferrari en el eje Z, el valor de la variable velocidad, (si es alta el auto se mueve rápido,)
- J** PositionEntity camara,0,8,-10 Pone la camara atrás de ferrari, esto por si se movio en alguna colición
- K** Collisions type_actor,type_escenario,2,2 Revisa si existnen coliciones y configura la respuesta
UpdateWorld : RenderWorld : Text 10,10, Veloc# : Flip Actualizar la pantalla.

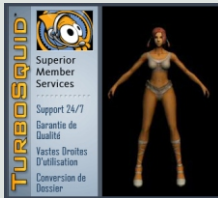
Crear y exportar modelos con Animación al Blitz3D

A. El modelo

Al momento de crear un modelo hacerlo siempre con los brazos y piernas ligeramente abiertos, esto para que después al ponerle los huesos nos resulte mas fácil ligar los vértices a cada hueso, en 3dsMax para empezar utilizaremos el modificador Skin.

Si se está recién empozando con el 3ds Max y no se tiene mucha destreza, mejor conseguir un modelo en internet, por ejemplo en Turbosquid, en esta pagina hay muchos modelos algunos tiene precios pero en hay muchos Free.

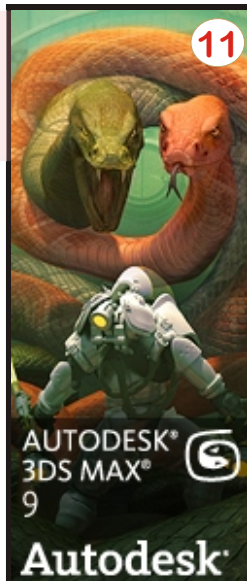
A la hora de utilizar un modelo deberemos tener cuidado con las texturas, las cuales deberemos descargarlas tambien, o aplicarlas al objeto, lo cual requerida cierto dominio del 3dsmax, para esto consultar en el foro la forma de hacerlo o recurrir a algún tutor de los muchos que hay en la red.



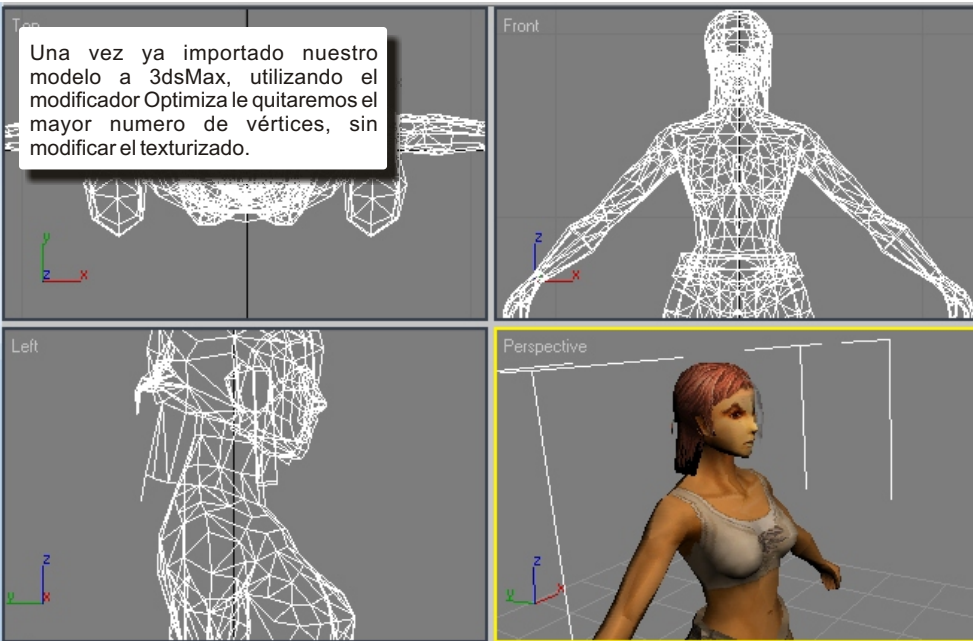
Como ejemplo utilizaremos el modelo mostrado a la izquierda, descargado desde Turbosquid, mas especifica mente en:

[Http://www.turbosquid.com/FullPreview/Index.cfm/ID/359362](http://www.turbosquid.com/FullPreview/Index.cfm/ID/359362)

Para descargarlo debes de crearte una cuenta(gratuita)



11



B. La animación



La secuencia de animación:

- ☞ Cargar el modelo
- ☞ Crear un biped
- ☞ Attach a la malla
- ☞ Animar el biped
- ☞ Ocultar el biped
- ☞ Exportar a b3d

Por ejemplo:

De 1 a 10: mira de un lado a otro

De 15 a 35 camina

Para poder exportar a B3D, debiera estar instalado el plugin Pipeline en el 3dsMax



Graphics3D 640,480,16,2

: AmbientLight 255,255,255

ruby=LoadAnimMesh("ruby.b3d")

ExtractAnimSeq(ruby,1,10)

ExtractAnimSeq(ruby, 15,35)

camara=CreateCamera()
RotateEntity camara,35,0,0

: PositionEntity camara,0,50,-50
: Piso=LoadMesh("suelo.b3d")

Animate ruby,2,0,1,1

While Not KeyHit(1)

Avanza#=0 : gira#=0

If KeyDown(200) Or KeyDown(203) Or KeyDown(205) Or KeyDown(208) Then

If KeyDown(200) Then

If AnimSeq(ruby)=1 Then Animate ruby,1,.5,2

avanza#=0.5

Elseif KeyDown(208) Then

If AnimSeq(ruby)=1 Then Animate ruby,1,.5,2

avanza#=-0.5

EndIf

If KeyDown(203) Then

If AnimSeq(ruby)=1 Then Animate ruby,1,.5,2

gira#=1

Elseif KeyDown(205) Then

If AnimSeq(ruby)=1 Then Animate ruby,1,.5,2

gira#=-1

EndIf

Else

If AnimSeq(ruby)=2 Animate ruby,2,0,1,1

EndIf

MoveEntity ruby,0,0,avanza# : TurnEntity ruby,0,gira#,0

PointEntity camara,ruby

UpdateWorld : RenderWorld : Flip

Wend

En el aula 2

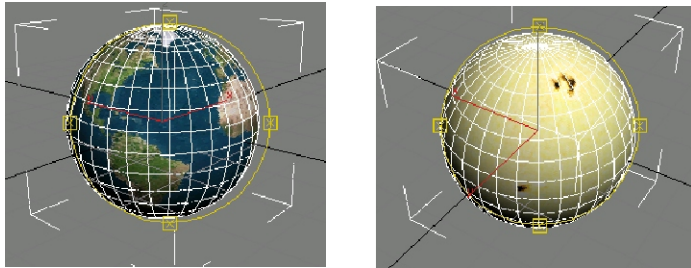
Las Matemáticas Mueven al mundo



A. Introducción

Si algo no me gustaban eran las clases de matemáticas, bueno no me gustaban hasta que pude probarlas en mi computadora y darme cuenta que gracias a ellas podía simular el mundo a mi alrededor, aquí iniciamos un pequeño recorrido probando las formulas y funciones matemáticas en el desarrollo de videojuegos, espero les sea de interés.

B. Sistema Solar

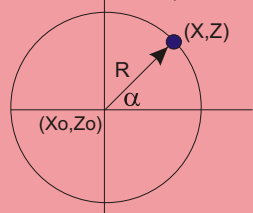


Con este ejercicio simularemos una órbita circular del planeta tierra al rededor del Sol.

Primero Moldeamos dos esferas y las texturizamos para que representen a la tierra y al sol, cada objeto por separado, y teniendolos siempre en las coordenadas 0,0,0 en el 3dmax

C. Un poco de Trigo

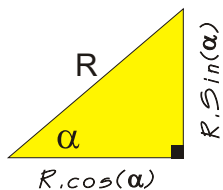
EN EL PANO X,Z



(Xo,Zo) : Coordenadas del centro de la circunferencia
 (X,Z) : Las coordenadas de un punto cualquiera sobre la circunferencia.
 R : Radio de la circunferencia
 α :Ángulo que define la posición de x,z

Para empezar ha programar deberemos tener los datos X,Z en función a los valores constantes de nuestro sistema: Xo , Zo , R y el ángulo que en nuestro sistema ira cambiando gradualmente desde 0 a 360.

Aplicando una conversión en el triangulo obtenemos las siguientes formulas:



$$X = Xo + R \cos(\alpha)$$

$$Z = Zo + R \sin(\alpha)$$

Y constante =0, punto real: X,0,Z

D. El programa

Graphics3D 640,480,16,2

tierra=LoadMesh("tierra.b3d")
 sol=LoadMesh("Sol.b3d")
 camara=CreateCamera()

AmbientLight 255,255,255

PositionEntity camara,0,300,0
 PointEntity camara,sol

xo=0
 zo=0
 r=200
 a=0

While Not KeyHit(1)

x = xo + r * Cos(a)
 z = zo + r * Sin(a)

a=a+1 : If a>=360 Then a=0

PositionEntity tierra,x,0,z,True

RotateMesh sol,0,1,0
 RotateMesh tierra,0,5,0

UpdateWorld
 RenderWorld
 Flip

Wend

End

x = xo + r * Cos(a)
 z = zo + r * Sin(a)

Halla sa nuevas coordenadas de x,z, para el nuevo valor de (actual) a

a=a+1 : If a>=360 Then a=0

Aumenta a de uno en uno: si a es igual o mayor a 360 entonces a regresa a cero.

PositionEntity tierra,x,0,z,True

Mueve a la tierra a las nuevas coordenadas, el ultimo parámetro true, indica que las coordenadas son en gorma global es decir son mediad desde el 0,0 absoluto y no desde la ultima posición de la tierra

Graphics3D 640,480,16,2

Carga el entorno gráfico para trabajar en 3D, una ventana de 640x480 con una profundidad de color de 16 bits.

tierra=LoadMesh("tierra.b3d")
 sol=LoadMesh("Sol.b3d")
 camara=CreateCamera()

Cargar los objetos 3d. Carga la tierra y el sol
 Crea la camara.

tierra=LoadMesh("tierra.b3d")
 sol=LoadMesh("Sol.b3d")
 camara=CreateCamera()

Crea y cargar los objetos 3d.

AmbientLight 255,255,255

Establece la luz ambiental en color blanco

PositionEntity camara,0,300,0
 PointEntity camara,sol

Pone la camara en y=300 (altura)
 Apunta la camara al sol

xo=0
 zo=0
 r=200
 A=0

Inicia las variables:

a ángulo
 Xo,Zo posición del centro
 r radio de giro de la tierra

While Not KeyHit(1)

.....

Wend

Repetir mientras no se presione escape(key 1)

