
Implementando OpenGL con Oculus Rift SDK

Jaime Margolin¹, Daniel Monzalalala¹ and Juan Carlos León¹

¹ Instituto de Estudios Superiores del Tecnológico de Monterrey Campus Santa Fe

May 14, 2017

El propósito del paper esta enfocado en dar una nueva perspectiva de como se puede incluir el uso de básicas y viejas funciones de OpenGL con la tecnología más sofisticada y moderna como lo es el SDK de Oculus y las librerías que utiliza esta última, y como estas tecnologías pueden ir de la mano para crear nuevas y mejores soluciones en el campo de las gráficas computacionales. Para esto se utilizaron librerías de modelos que el mismo SDK provee, logrando así una conjunción más armonica para el resultado final. Este paper ayudará a todos aquellos que en su momento quieran explorar el mundo de "VR" y quieran implementarlo de manera fácil y que quieran utilizar OpenGL envez del estandar para VR de DirectX.

1 Introducción

En la actualidad, hay campos que han surgido dentro del desarrollo de las tecnologías computacionales, siendo la Realidad Virtual uno de los que parece tener más futuro, el poder simular ambientes completos y poder hacer que el mismo usuario puede estar inmerso en el mismo, haciendole pensar que esta en un ambiente como en el que esta acostumbrado a interactuar.

Ahora una de las prácticas que se ha vuelto de las más comunes es el uso de Oculus Rift, el cual es un dispositivo que puede simular en forma 3D ambientes, los cuales son controlados por el mismo dispositivo, este dispositivo es un tanto parecido a unos lentes, haciendo asi que la combinacion entre los lentes y un buen equipo de sonido portátil (audífonos) ó estático (bocinas) puedan hacer que el usuario se sienta en el lugar que se esta simulando, un lugar que puede ver

ante sus propios ojos e inclusive en algunas ocasiones y con algunos otros aditamentos, hacer que se pueda interactuar de manera "*normal*".

2 Trabajo Previo

El trabajo previo en el campo del Oculus SDK esta desarrollado en DirectX y los mismos desarrolladores del SDK recomiendan utilizarla. Encontramos muy poco realizado con OpenGL en el campo del oculus rift....

2.1 Paper 1

2.2 Paper 2

3 Desarrollo

Nosotros intentamos crear una pecera con distintos animales marinos utilizando el SDK del oculus rift **DK2** para OpenGL. Por las funciones proveidas por el SDK para OpenGL optamos por realizar este acuario con *pixel art* que es una especie de representacion de los objetos en formas mas cuadradas al estilo de los videojuegos de **8 bit**.

El desarrollo consistio en tomar como base el (unico) proyecto ejemplo que tenia el SDK en OpenGL. Este ejemplo consistia de un pequeño cuarto con una mesa una silla un cubo que gira y un mueble de pared. lo que se realizo fue investigar como estaba el codigo compuesto. Se encontro la funcion

```
AddSolidColorBox(float x1, float y1,
float z1, float x2, float y2,
float z2, DWORD c)
```

que lo que hace es agregar una "caja a la escena". esta funcion recibe como parametros la x, y, z iniciales , la x, y, z finales y el color de la caja en hexadecimal.

3.1 armado del mundo

3.1.1 La pecera

El mundo consta de 4 paredes 1 piso y 1 techo. estos forman la pecera en la que los distintos animales marinos estaran nadando. Cada una de las paredes y el techo son creados agregando un cubo con 2 de sus aldos grandes y el 3 muy pequeño para que asi se asemeje a una pared delgada. El piso es creado de igual forma. por ultimo los colores que se le asignaron a las paredes y al techo fue el mismo tono de azul, esto se eligio para lograr un efecto de continuidad. este efecto evita que se pueda diferenciar el momento en el que empieza una pared y otra y donde conectan con el techo, asi dando un mayor sentido de profundidad. Al piso se le asigno un color amarillento para que se parezca a la arena del mar. se intento agregar alguna de las texturas que ya estan incluidas en el SDK al piso para hcerlo parecer mas real pero las texturas del SDK daban la impresion de ser mosaicos y esto restaba un poco de realismo a la escena.

3.1.2 Las algas

Las algas constan de un tallo de altura semi aleatoria. La altura del tallo es de 0.8 mas un numero aleatorio que va del 0.0 al 0.7 esto nos da una altura aleatoria de 0.8 a 1.5. Tambien se agregan hojas a la algas. Las hojas constan de 4 cubos pequeños acomodados en diagonal a partir del tallo. a una altura aleatoria de 0.2 a 0.3. cada alga puede tener de 0 a 4 hojas y esto tambien se decide de forma aleatoria. Y por ultimo la posicion de las hojas tambien se decide de forma aleatoria. Esto significa que sin importar la cantidad de algas que se agreguen a la escena habra una gran variedad de algas con apariencias ligeramente diferentes para darle un toque de mayor realismo a la pecera.

3.1.3 Los peces

Los peces estan compuestos por 3 elementos principales: el cuerpo, la cara, las aletas. el cuerpo del pez consiste de un cubo grade de $1x1x1$. Los demas elementos del pez van "pegados" al cuerpo. la cara del pez esta compuesta por una boca que es un rectangulo pequeño en el centro en el eje X y a $1/3$ del eje Y del lado frontal del cubo, los ojos son 2 cuadrado negros a $1/3$ y $2/3$ del eje X y a $2/3$ del eje Y del lado frontal del pez.

Las aletas estan divididas de la siguiente forma: 2 aletas laterales, 1 aleta dorsal y 1 aleta trasera. Las aletas laterales constan de 2 rectangulos delgados. uno mas pequeño que esta directamente pegado al pez y otro mas grande conectado al primero. los colores de

las aletas son diferentes al del cuerpo del pez y cada uno de los rectangulos tiene su propio color.

La aleta dorsal esta compuesta de un rectangulo horizontal largo y de otro mas corto. esta aleta esta en la parte superior del pez. ambos rectangulos estan directamente pegados al cuerpo del pez, primero el rectangulo grande seguido del chico. los colores son los mismos que los de las aletas laterales.

la aleta trasera, o cola, esta compuesta de 3 rectangulos de tamaño creciente y ancho decreciente. el mas pequeño y grueso esta pegado al pez mientras que el mas grande y delgado es el mas alejado. el primer rectangulo va pegado al pez de forma vertical, el segundo va pegado a este y el tercero al segundo. los colores de estos rectangulos son, el primero y tercero tienen el color del primer rectangulo de las aletas laterales y el 2 rectangulo el color del segundo rectangulo de las aletas laterales.

3.1.4 tiburon

El tiburon sigue la misma logica de contruccion que los peces unicamente se modifiko el largo del cuerpo que para el tiburon es de $1x1x3$ y el grosor de las aletas, estas se hicieron mas gruesas para darle un toque de realismo. tambien se adecuo el posicionamiento de las aletas para que se mantuviera la proporcion del cuerpo. el tiburon es completamente negro con tonos grises en las aletas y ojos blancos.

3.1.5 Erizo

Para darle variedad a la vida marina en la pecera se decidio agregar un erizo de mar. el erizo tiene un cuerpo cuadrado de $0.5x0.5x0.5$ y 5 espinas. su cuerpo y espinas son de color gris. las espinas fueron creadas de forma en que todas estuvieran distribuidas por el cuerpo del erizo y tuvieran longitudes ligeramente diferentes para dar un toque de realismo.

3.2 Construccion del mundo

Con todos los elementos del mundo armados se tienen que agregar a la escena. Para dar un toque de realismo se decidio que las posiciones de los peces como de las algas deberian ser aleatorias.

3.2.1 Tiburon y Erizo

Por sus características y funciones dentro de la escena se opto por dejar al tiburon y al erizo en una posicion de inicio fijas.

3.2.2 Algas

Las algas de decidio que deberia de tener un posicion aleatoria dentro de la pecera para dar un toque de realismo. cada vez que se corre el programa las algas apareserane n lugares distintos. Como se agrega una

gran cantidad de algas a la escena el darles una posición aleatoria ayuda a distribuirlas. las algas se crean en un rango de -10 a 10 en X y de -20 a 20 en Z , esto para abarcar la totalidad de la pecera.

3.2.3 Peces

La posición inicial de los peces también es aleatoria para dar un toque de realismo. Pero para mantenerlo visualmente estético se decidió que la altura a la que nadan los peces sería predefinida, así se evita que los peces pasen unos sobre otros. la altura más baja es para los que nada en línea vertical (recorrido largo en el rectángulo que forma la pecera) mientras que los que nadan más alto son los que recorren la pecera de forma horizontal (recorrido corto en el rectángulo que forma la pecera).

3.3 Animaciones

Para dar un toque de realismo se animaron todos los animales marinos de la escena. Las animaciones fueron de las partes más retardadas ya que se trató de mantener una animación congruente con el **pixel art** de la escena pero a la vez de hacerlo parecido a la realidad para mantener el toque de realismo que se le está intentando dar al proyecto.

3.3.1 Erizo

El erizo es el animal con el movimiento más simple. Él se mueve siempre en un círculo alrededor del centro $(x/2, y/2)$ de la pecera, también es el más rápido de todos los animales y el único que se mueve por el piso.

3.3.2 Tiburón

A petición del profesor "Estaría padre que agreguen un tiburón así que vaya como acechando a los otros peces" se agregó el tiburón y la animación tenía que ir de acuerdo a dicha petición. se optó por el movimiento en forma de *8 extendido*, para que entre y salga de la escena como si estuviera acechando a los peces. El tiburón es el más lento para dar un toque de realismo al acechar.

3.3.3 Peces

Los peces son los animales con más movimientos. para dar un toque de realismo se les programó un movimiento hacia el frente mientras que tienen un movimiento ondulatorio ya sea hacia arriba y abajo o de derecha a izquierda.

Además de tener una posición de origen aleatoria los peces tienen movimiento aleatorio, la velocidad con la que avanzan hacia el frente es aleatoria entre 0.22 a 0.44 . esto hace que los peces se muevan a diferentes velocidades y da un toque de realismo a la escena.

Con fines demostrativos para la exposición se decidió fijar el movimiento ondulatorio de los peces de la siguiente forma. lo que se mueven de forma vertical ondulan de arriba abajo mientras que lo que se mueve en forma horizontal ondulan de derecha a izquierda.

4 Complicaciones

4.1 figuras

El primer problema con el que se encontró es que el SDK para el oculus únicamente tiene una figura, **el cubo**, por lo que la primera restricción y la más compleja fue el tener que armar todo con subos. También la función que crea los cubos recibe como parámetros la XYZ inicial y final, a diferencia de las funciones básicas que se vieron en clase que reciben únicamente el tamaño del cubo (y este es creado en la posición en la que se encuentra GL en ese momento). Por lo que todas las posiciones y movimientos de la escena fueron calculados por nosotros.

para lograr las distintas figuras de la pecera se hicieron muchos cálculos para lograr modificar los cubos a que fueran cuadrados y rectángulos que pudiéramos usar. además del cálculo que hicimos para posicionarlos unos con respecto de otros.

4.2 OpenGL SDK vs lo visto en clase

Encontramos diferencias sustanciales entre lo que se usa en el SDK del oculus y lo que se vio en clase. Aquí listaremos las que más nos causaron problemas.

- GL main loop - En este proyecto no hay un main loop que se pases al main loop de GL. la función del main loop tiene funcionalidad similar pero no acepta ninguno de las funciones de GL dentro.
- GL init - no hay una función de init como en GL.
- funciones de GL - en el SDK del oculus no sirven ninguna de las siguientes funciones que se vieron en clase
 - push matrix
 - pop matrix
 - rotate
 - scale
 - translate
 - curvas
 - superficies
- luz - la luz como se vio en clase se agrega 1 (o varias) fuente de luz a la escena la cual interactúa con los materiales de los objetos para ser reflejada. en el proyecto no hay ninguna fuente de luz, si no que cada objeto tiene su propio color, que de cierta forma es la luz que este objeto irradia, pero sin afectar a las demás o reflejarse en otros objetos.

4.3 .OBJ's

Se intento cargar archivos **.OBJ** de muchas maneras y ninugna permitio cargarlos. se probaron varias librerias y metodos entre los que destacan: glm (visto en clase) y assimp.

Inclusive se intento cargar los archivos con Dircx y tampoco se pudo

5 Resultados

6 Trabajo Futuro

7 Conclusiones