**Proyecto Parcial 2**

**Equipo:**

Carlos Enrique López Garcés

Waldemar Quevedo Salinas

**Breve descripción de la aplicación:**

Nuestro proyecto consiste de 2 partes, una parte hecha en C++ y otra en Python. La razón es la siguiente: mientras trabajábamos en el proyecto nos fuimos dando cuenta de que para poder darle a la aplicación la funcionalidad que necesitaba nos faltaban varios conocimientos, cada uno con su propia curva de aprendizaje, (picking, crear clipping volumes, entender bien las los diferented modos de matrices, la relación entre clipping volumes y viewports, entre otros…). Aprender éstas partes de la tecnología mientras aplicábamos al mismo tiempo un patrón MVC y encima en Python y su FFI para OpenGL complicaba más el proceso.

Así es como comenzamos a realizar una versión hecha en C, sin embargo, pronto nos dimos cuenta que para repintar la escena, cada uno de los objetos tendría que mantener su propio estado y propiedades, por lo que pronto nos movimos a realizarlo en C++.

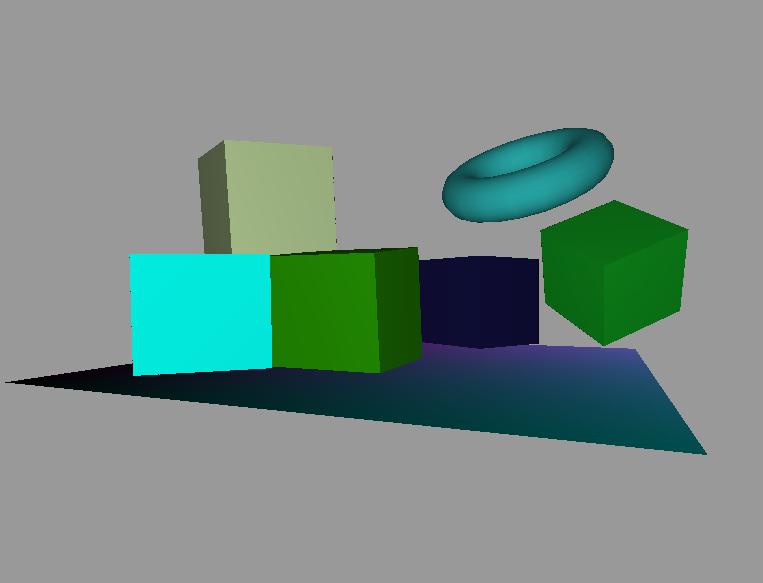
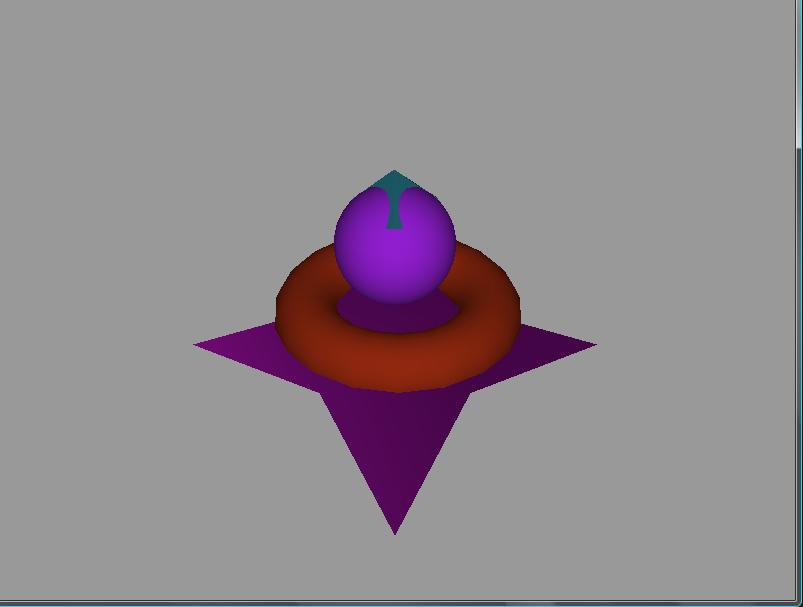
La interacción con la aplicación hecha en C++ es principalmente con el teclado:

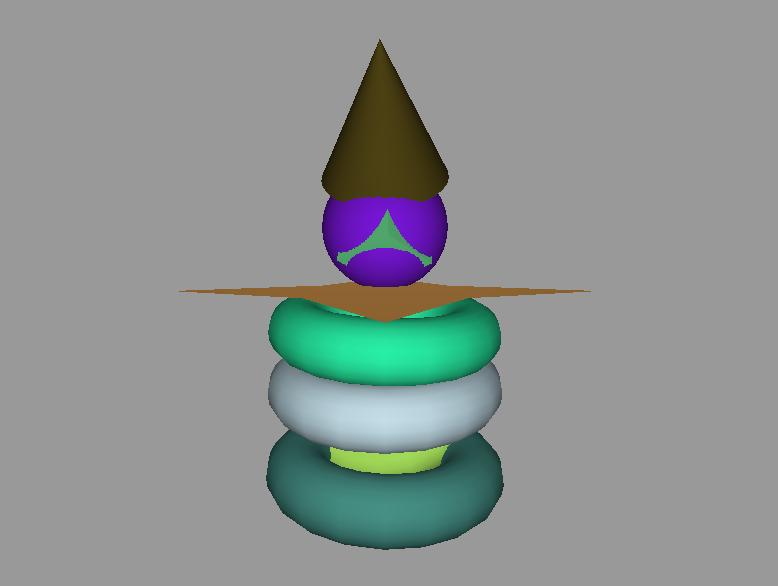
|  |  |
| --- | --- |
| **Teclado** | **Funcionalidad** |
| Q | Salir |
| A | Dibujar estrella |
| S | Dibujar esfera |
| D | Cubo |
| F | Cono |
| G | Toroide |
| T | Cilindro |
| Z | Mover cámara a la izquierda |
| V | Mover cámara a la derecha |
| X | Avanzar con la cámara |
| C | Retroceder con la cámara |
| H | Trasladara a la figura seleccionada hacia +x |
| Y | Trasladara a la figura seleccionada hacia –x |
| J | Trasladara a la figura seleccionada hacia +y |
| U | Trasladara a la figura seleccionada hacia –y |
| K | Trasladara a la figura seleccionada hacia +z |
| I | Trasladara a la figura seleccionada hacia -z |
| O | Mover hacia atrás en el arreglo de objetos creados |
| P | Mover hacia adelante en el arreglo de objetos creados |
| FLECHAS | Las flechas del teclado sirven para rotar a la figura sobre su propio eje. |
| M | Hacer que aparezca el grid ( como en maya) |
| N | Hacer que aparezcan las líneas de origen de la figura |

Ejemplo de su uso:

* Abrimos la aplicación
* Creamos dos figuras (a s d f g h)
* Seleccionamos la segunda presionando P
* La rotamos con las flechas del teclado o la trasladamos con las teclas(hjk yui)
* Activamos el grid con N
* Activamos sus líneas de origen con n
* Presionamos O para escoger la figura anterior
* Activamos sus líneas de origen con n

**Screenshots:**



****

**Acerca de la arquitectura de la aplicación hecha en C++:**

Tiene un arquitectura muy simple. Tenemos una clase abstracta Figura de la cuál heredan los demás polígonos (Sphere, Torus, Cylinder, Cone), en el patrón MVC esto sería el modelo. Después la clase principal Escena es la encargada de realizar todo el pipeline de OpenGL.

**Reflexiones:**

**Waldemar**: Aunque Python es una gran lenguaje (como Peter Norvig dice: “no es más que un Lisp disfrazado donde intercambiaron los paréntesis por la identación”), pienso que no es el ideal para aprender por primera vez OpenGL. Irónicamente, a pesar de que Python es conocido por servir para realizar prototipos rápidamente de aplicaciones, pienso que al tratarse de OpenGL, en realidad es más fácil hacerlo en C o en C++, que en Python y PyOpenGL o alguna otra tecnología que soporte hacer FFIs como por ejemplo CommonLisp + clOpenGL. Asimismo, pienso que es muy importante saber de la teoría antes de programar en OpenGL porque la verdad algunas veces no tengo idea de lo que sucedía.