**ESCUELA SUPERIOR POLITÉCNICA DEL LITORAL**

****

**Write Up**

**Gráficos por computadora**

**INTEGRANTES:**

**Karen Lilibeth Borbor Moreira**

**Iván Alejandro Mera Maldonado**

**José Antonio Viteri Cuenca**

**PROYECTO:**

**Implementación de una escena con figuras geométricas utilizando la herramienta POV Ray**

**Descripción del proyecto**

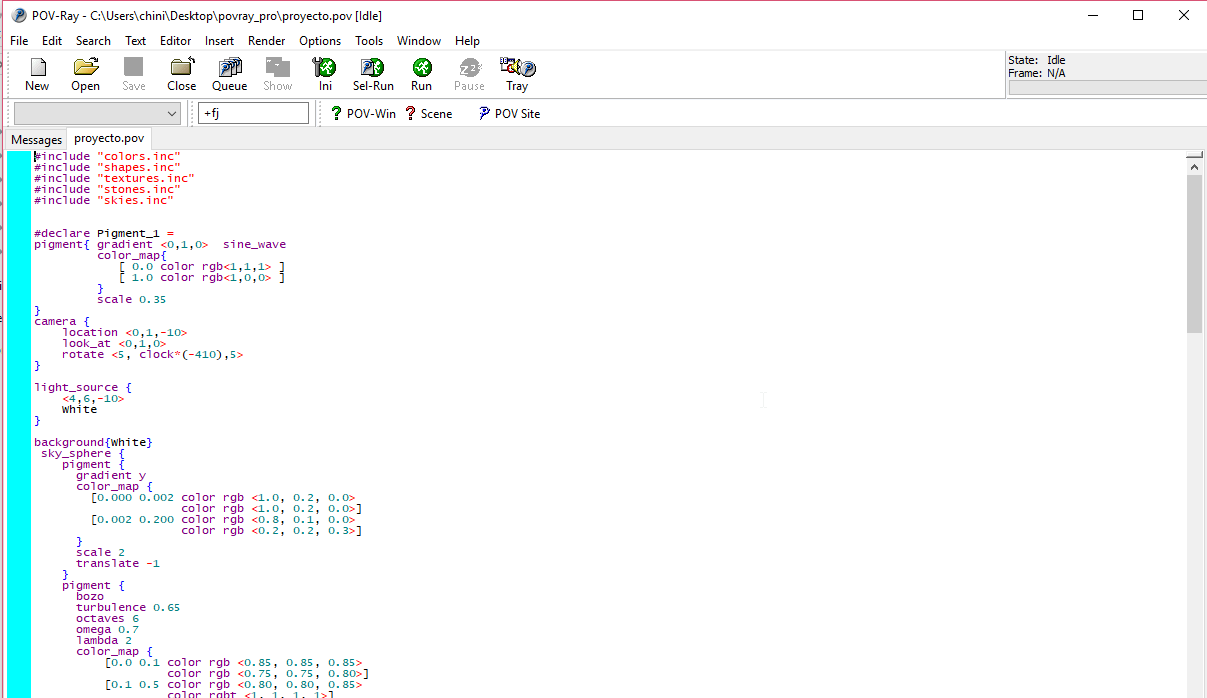
**POV-Ray**

POV-Ray (Persistence of Vision Raytracer) es una herramienta gratuita para producir imágenes fotorrealistas de alta calidad empleando la técnica de renderizado Ray Tracing. Los objetos e iluminación que conforman escenario son descritos en un archivo de texto, así como la posición de la cámara.

La técnica de Ray Tracing consiste en simular la forma en que los rayos de luz viajan en el mundo real. Los rayos se emiten desde alguna fuente de luz y chocan contra los objetos, luego algunos de estos rayos llegan a nuestros ojos. Sin embargo muchos de estos nunca colisionan con objetos observables o son absorbidos por otros objetos, por lo que el proceso es realizado de forma inversa; se dispara un rayo desde el punto de visión (la cámara) por cada pixel que conforma la imagen final y se verifica si dichos rayos intersecan algún objeto, el cual es entonces renderizado, permitiendo ahorrar tiempo y recursos.

**Proceso de creación del proyecto**

POV-Ray puede ser descargado desde su página oficial (<http://www.povray.org/>). Al instalarlo tendremos acceso a un editor de texto que nos permitirá describir el escenario como una serie de comandos.



### 

### **Creando las figuras:**

Todas las figuras en POV-Ray comparten una sintaxis similar:

*nombre\_de\_la\_figura {*

*<coordenada\_1, coordenada\_2, coordenada\_3>,*

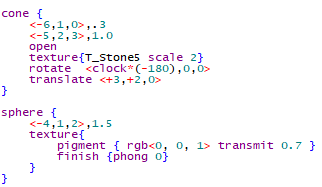
*textura {*

*},*

*rotación*

*}*

Entre otras opciones:



### 

### **Iluminando de la escena:**

Una escena sin iluminación no nos mostrará nada al momento de renderizarla, por lo que POV-Ray nos permite definir fuentes de luz:

*Light\_source {*

*<coordenada\_1, coordenada\_2, coordenada\_3>,*

*color*

*}*

Se debe definir, por lo menos, las coordenadas para colocar la fuente de luz y un color:



**Colocando la cámara:**

También es necesario definir la ubicación de la cámara:

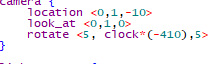
*camera {*

*location <coordenada\_1, coordenada\_2, coordenada\_3>,*

*look\_at <coordenada\_1, coordenada\_2, coordenada\_3>*

*}*

Se debe definir, por lo menos, la ubicación de la cámara, y el punto hacia el cual estará observando la escena:



**Colocando un fondo:**

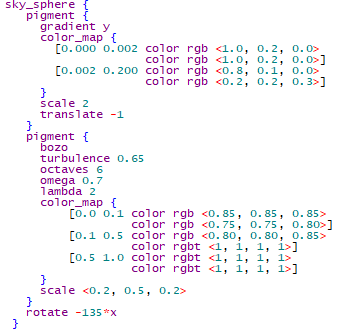
POV-Ray también nos permite definir un color de fondo para el escenario:

*background {*

*color*

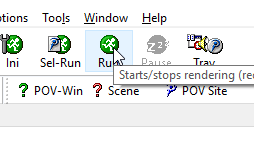
*}*

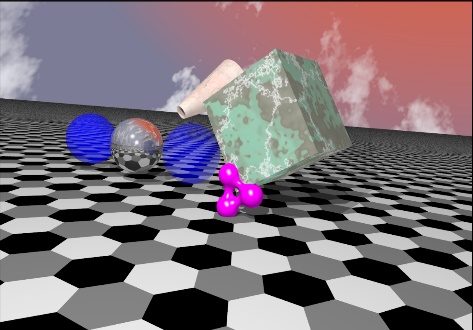
Así como bonitos patrones y combinaciones de colores, como el *sky\_sphere* para crear un bonito fondo realista similar al cielo:



**Luces, cámara, ¡ACCIÓN!:**

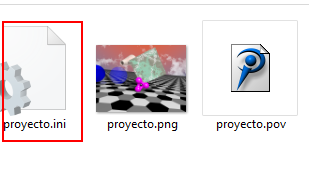
Una vez definido el escenario, estamos listos para renderizar la escena. Sólo debemos dar click en el botón “Run” y POV-Ray se encargará de generar una imagen en formato .PNG en el directorio raíz de nuestro proyecto:

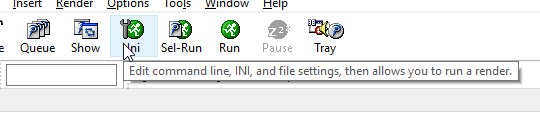
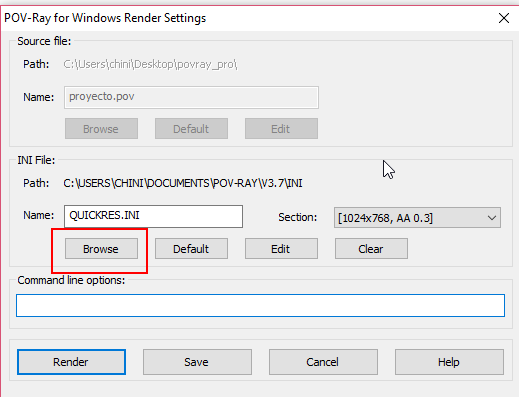




**Animaciones:**

Junto con el archivo del proyecto, se genera un archivo de configuración .INI. En este archivo podemos escribir varios comandos que se ejecutarán durante el proceso de renderizado del escenario.

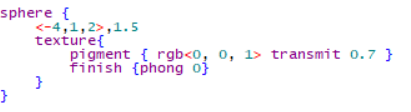


Se debe definir, por lo menos un frame inicial (*Initial\_frame*) y un frame final (*Final\_frame*). Luego debemos escoger el archivo de configuración desde el editor de POV-Ray:

Hecho esto, podemos finalmente pasar a renderizar. POV-Ray generará los frames de la animación en formato .PNG (el formato se puede cambiar a través de la línea de comandos de la ventana de configuración de renderización antes mostrada) y el usuario debe encargarse de compilarlos con ayuda de otro software externo, dado que POV-Ray no renderiaza la animación por sí solo.

**Figuras:**

Para la figura con transparencia se uso dentro de la textura, un factor de Phong de 0, esto permitió que la figura adquiera transparencia.



Para la figura que logre reflejar las cosas de su entorno lo que se hizo fue ponerle una textura de tipo Polished Chrome , y a su vez ponerle una refacción de 1, lo que la vuelve de tipo espejo.

