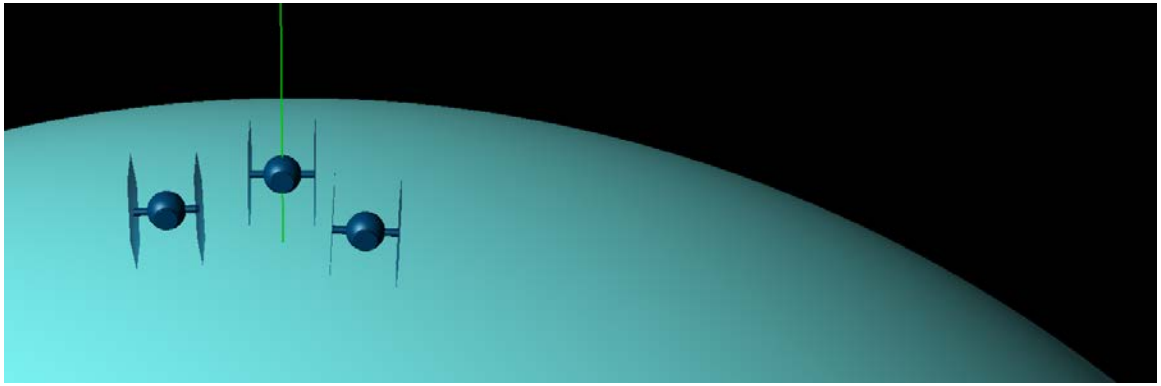
Indicaciones para la realización de los apartados 23-34

A. Gavilanes
Departamento de Sistemas Informáticos y Computación
Facultad de Informática
Universidad Complutense de Madrid

Apartado 23

- La formación de combate se obtiene situando tres **TIE**'s encima del Polo Norte del planeta y sometiendo a dos de ellos a, al menos, una rotación adecuada
- No basta con hacer solamente traslaciones
- Se ha intentado mostrar esto en la captura siguiente en la que se ve que cada par de alas de cada **TIE** no es paralelo a ningún otro par



- Para mover fácilmente los **TIE**'s es conveniente hacer que la escena tenga una entidad compuesta formada por los tres **TIE**'s situados como se pide para esta escena. Es muy sencillo mover esta entidad compuesta alrededor del planeta o hacerla rotar sobre sí misma. En términos de modelo jerárquico, lo que hacemos así es introducir lo que se llama un **nodo ficticio**

Apartado 26

- El **`render()`** de la clase **`Esfera`** debe hacer esta distinción de casos:
  - Si el atributo **`Material* material`** no es nulo, se carga en GPU con **`material->upload()`**
  - Si es nulo y no existe material, se activa el color material con **`glEnable(GL_COLOR_MATERIAL)`**
  - Se renderiza la malla
  - Se desactiva el color material si se activó antes
- Debe darse prioridad al material frente al color material, es decir, preguntar primero si hay material, porque si se llegara a activar el color material, actuará el color en la renderización, haya o no haya material

Apartados 27, 28 y 29

- Las luces de estos apartados son únicas para una escena dada. Se declaran como atributos de la clase **`Scene`**
- Hay que tener cuidado si se construyen escenas auxiliares porque se construirán luces para ellas y el número total de luces creadas está limitado a 8
- Las luces no se encienden con el comando **`glEnable(GL_LIGHT0)`**, … correspondiente, sino mediante el método **`enable()`** de la clase **`Light`**. Por ejemplo, para dar la luz direccional **`dirLight`** de la escena mediante la tecla **`q`**, al **`switch`** del **`keyPressed()`** de **`IG1App`** se añade el caso:

```
case 'q':
    mScene->dirLight->enable();
    break;
```

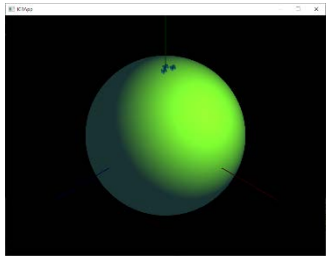
- **NO** se debe hacer:

```
case 'q':
    glEnable(GL_LIGHT0);  //GL_LIGHT0 es la base de dirLight
    break;
```

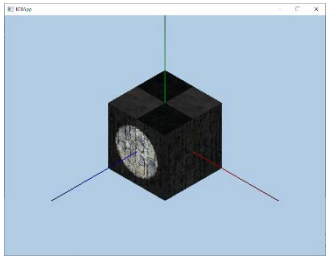
- Análogamente, con el apagado se usa **`disable()`** de **`Light`**

Apartados 27, 28 y 29

- Ten cuidado a la hora de elegir la componente difusa de la luz posicional **`posLight`** para que la esfera azul se muestre de color verde al iluminarla con ella



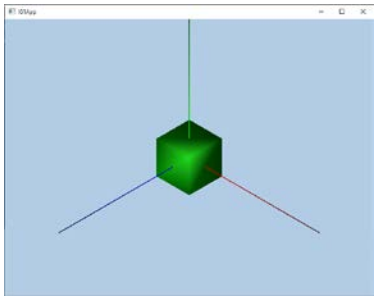
- Las tres luces de la clase **`Scene`** deben poder iluminar las distintas escenas de la Práctica por eso se muestran distintas capturas iluminadas por una misma luz



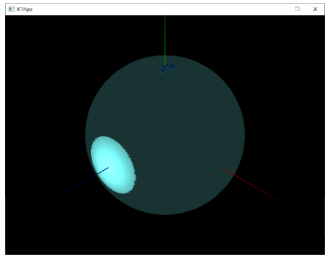
- Ojo a la captura que muestra el **`GridCube`** iluminado por el foco

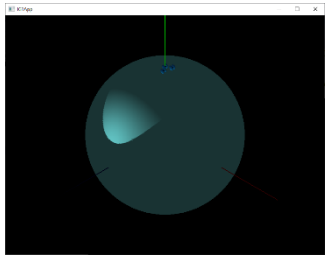
porque se debe ver perfectamente el cono de luz sobre una de las caras del cubo

- Compárese con el **`Cubo`** iluminado por el mismo foco



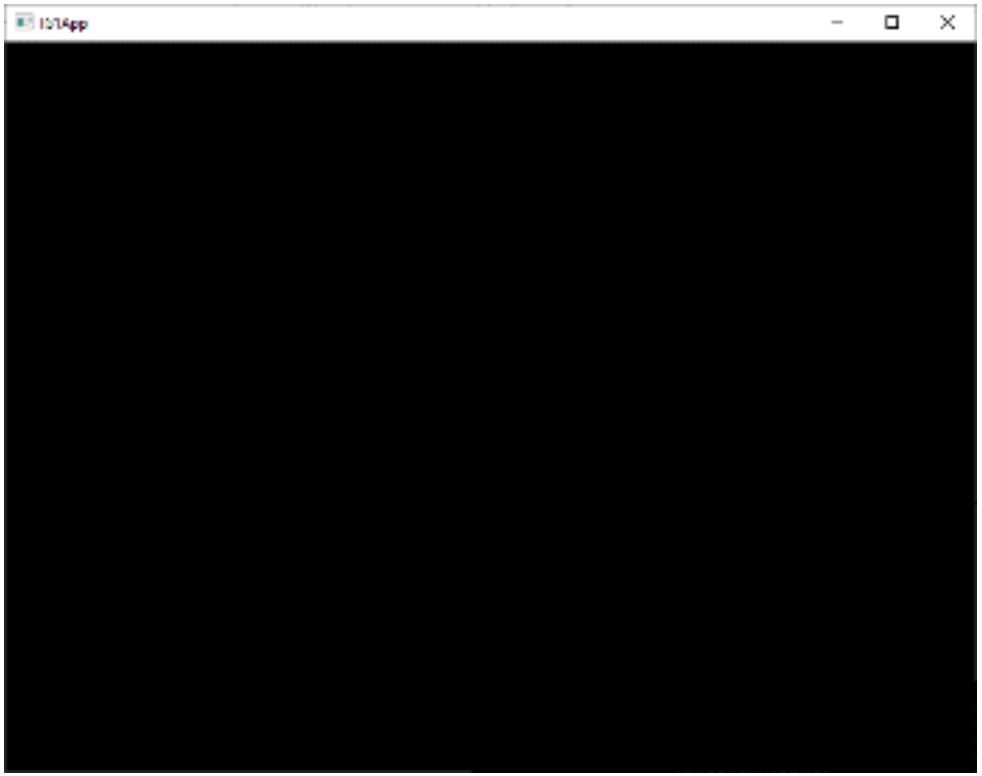
- El cambio entre las dos capturas de más abajo





no se hace por teclado, sino deteniendo la ejecución y levantando, en código, la dirección de emisión del foco.

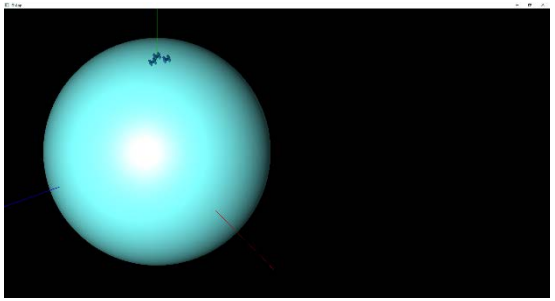
- La tecla **`e`** deja la escena completamente a oscuras
- La captura de abajo contiene los ejes, el planeta y los tres **`TIE`**'s
- Observa que no deben verse las texturas para lo cual tienes que elegir bien el **`mixMode`**

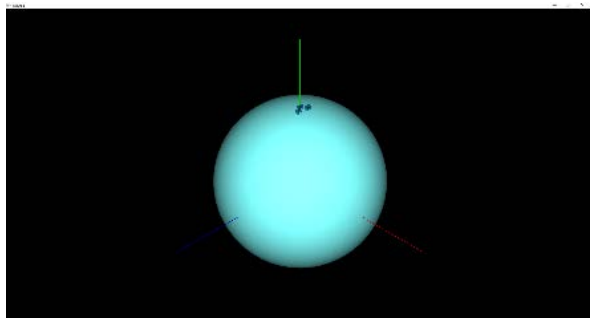


- Cómo poner material (y color material) a un objeto de la escena. Al planeta, por ejemplo:
  - Si **`init()`** es el método de **`Scene`** donde se construyen las escenas, en la escena donde se ha construido la esfera llamémosle **`planeta`**, se construye un material

```
Material* mat = new Material();
```

  - Este material invoca el método **`setCopper()`**, por ejemplo, para "ser cobre"
  - Se asocia **`mat`** a **`planeta`** con el método **`setMaterial(mat)`**
  - En el **`render()`** de **`Esfera`** (ya se ha dicho en la transparencia del apartado 26) se hacen dos preguntas sucesivas
    1. Si el atributo **`material`** no es nulo, **`material`** sube sus características a GPU invocando **`upload()`**
    2. Si el atributo **`material`** es nulo, se activa el color material. Por cierto, no es necesario que tu renderización de la esfera tenga el magnífico brillo que se muestra. Puede ser mate. Compara.





Apartados 32, 33 y 34

- Mira la demo para ver el comportamiento de los **`TIE`**’s, sus movimientos y sus focos