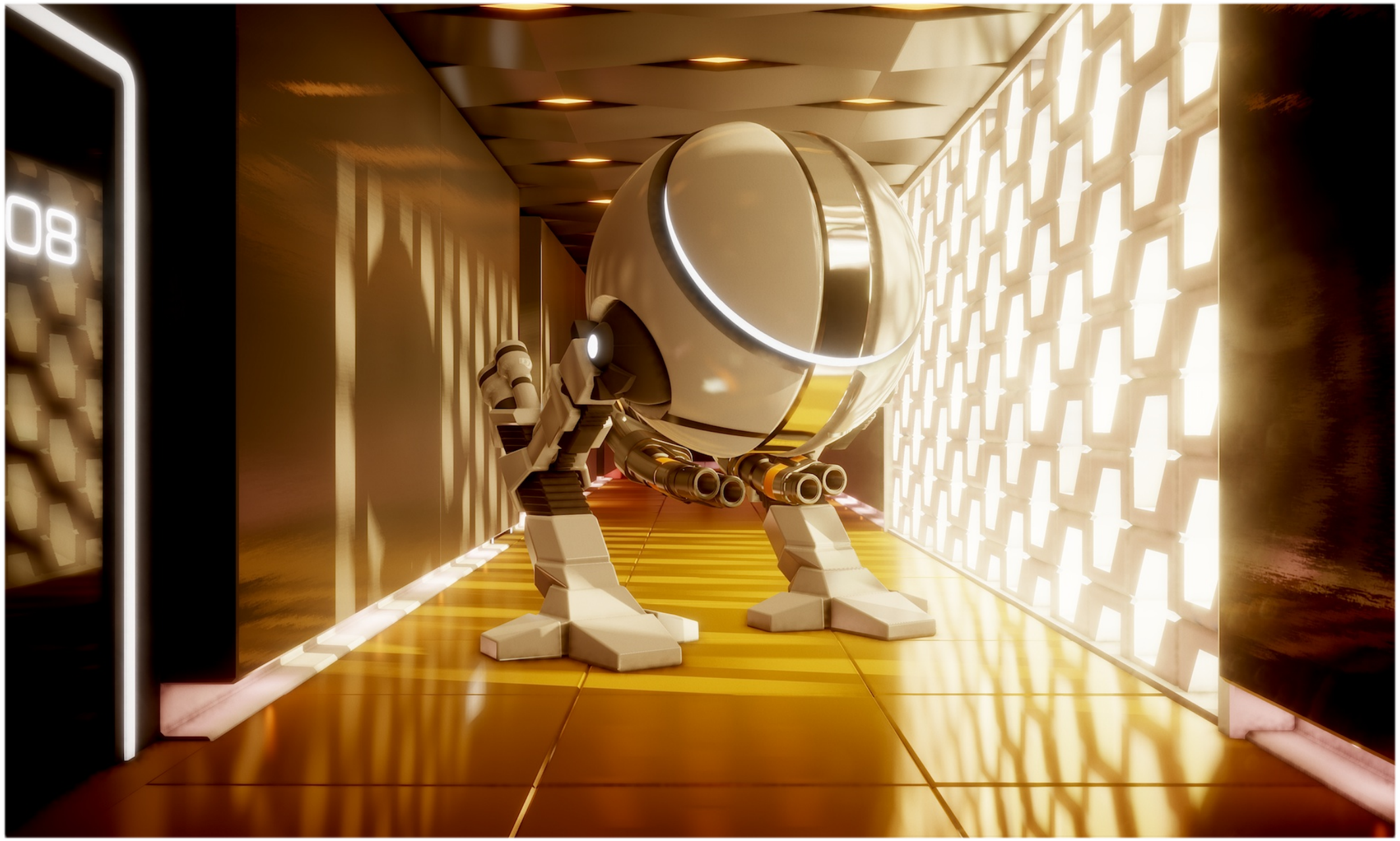



# Dynamic Reflections with Cubemap&FBO

Autor: Dolors Serra Almor

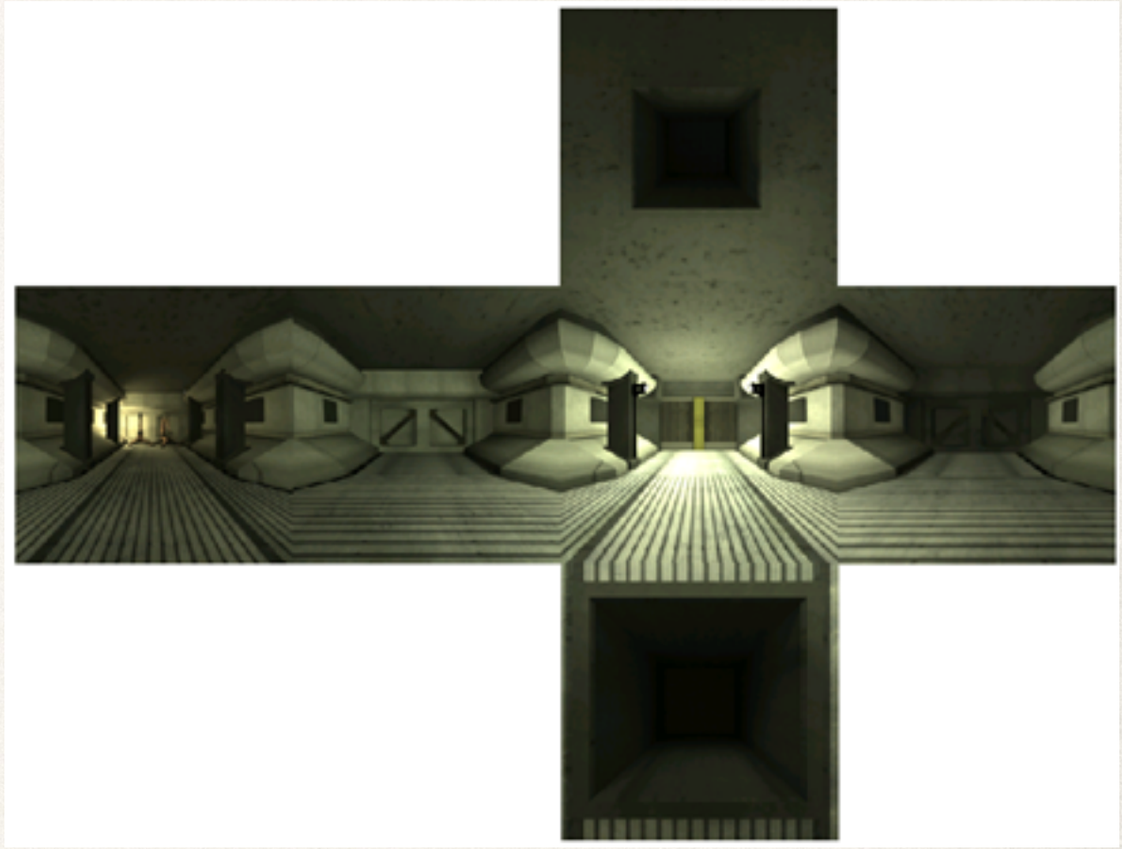
Programación Avanzada de Tarjetas Gráficas

*19 - 6 - 2015*

# The state of the Art

- Para crear reflejos en un objeto reflectante el **trazado de rayos** obtiene resultados perfectos pero requiere costosos cálculos computacionales.

- La opción al trazado de rayos que ya dimos en este curso son los **Mapas de Entorno**, que se basan en la técnica del Image Based Lighting, en la cual construíamos un Cubemap en el que proyectábamos una textura sobre cada una de sus caras la cual correspondía al ambiente envolvente de la escena.



# Limitaciones de los Mapas de Entorno

- Asumimos que el entorno está muy lejos
- No hay reflexiones entre los objetos
- No existen autoreflejos

# ¿Cómo conseguimos seguir usando Mapas de Entorno recortando limitaciones?

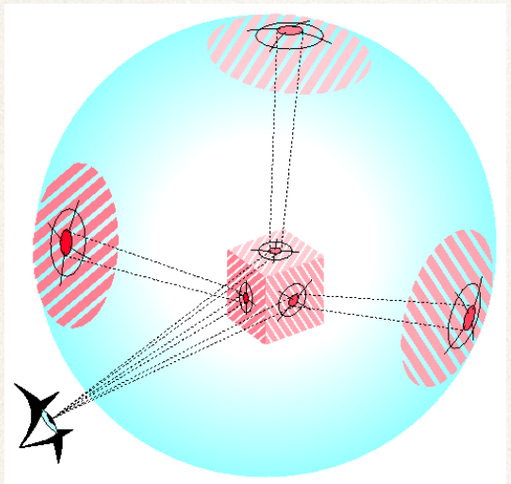
- Los mapas de entorno a diferencia del trazado de rayos nos permite Realtime.

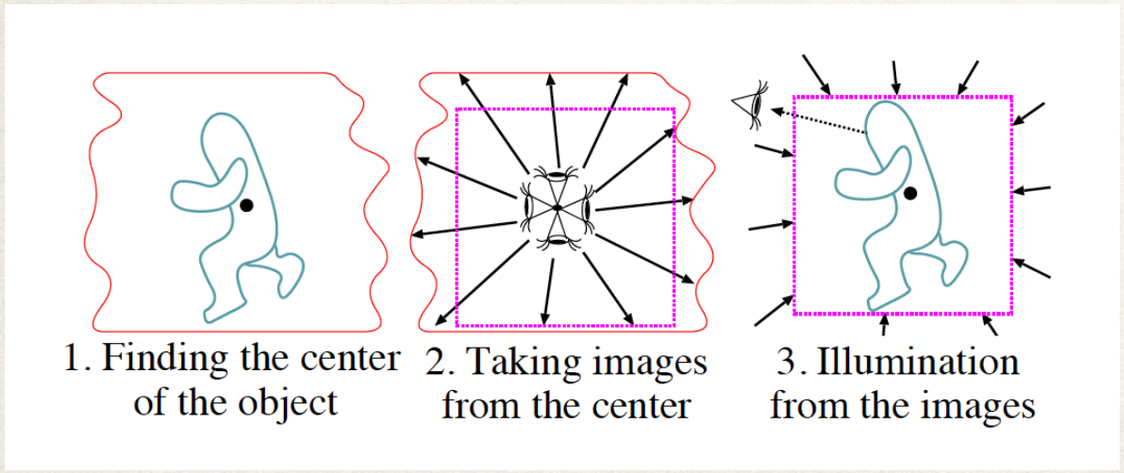
- Entre las limitaciones que tienen el que no existe reflexión entre los objetos es la que visualmente más se nota .

- Para solucionarlo usaremos una técnica llamada Local Cubemap o Dynamic Cubemap, para incluir objetos animados.

# ¿Qué es Dynamic Cubemap?

- Utilizando Framebuffer Objects (FBO) y un cubemap de texturas conseguiremos reflexiones entre los objetos incluso aunque los mismos se muevan, en Tiempo Real. Trasladando en cada frame la cámara al centro del objeto reflectante y consiguiendo desde ese punto de vista una imagen esférica de 360º de todo lo que reflejará el objeto, esta textura se creará en el buffer de profundidad excluyendo el propio objeto y se proyectará dicha textura sobre el Cubemap.






# ¿Cómo lo hacemos?

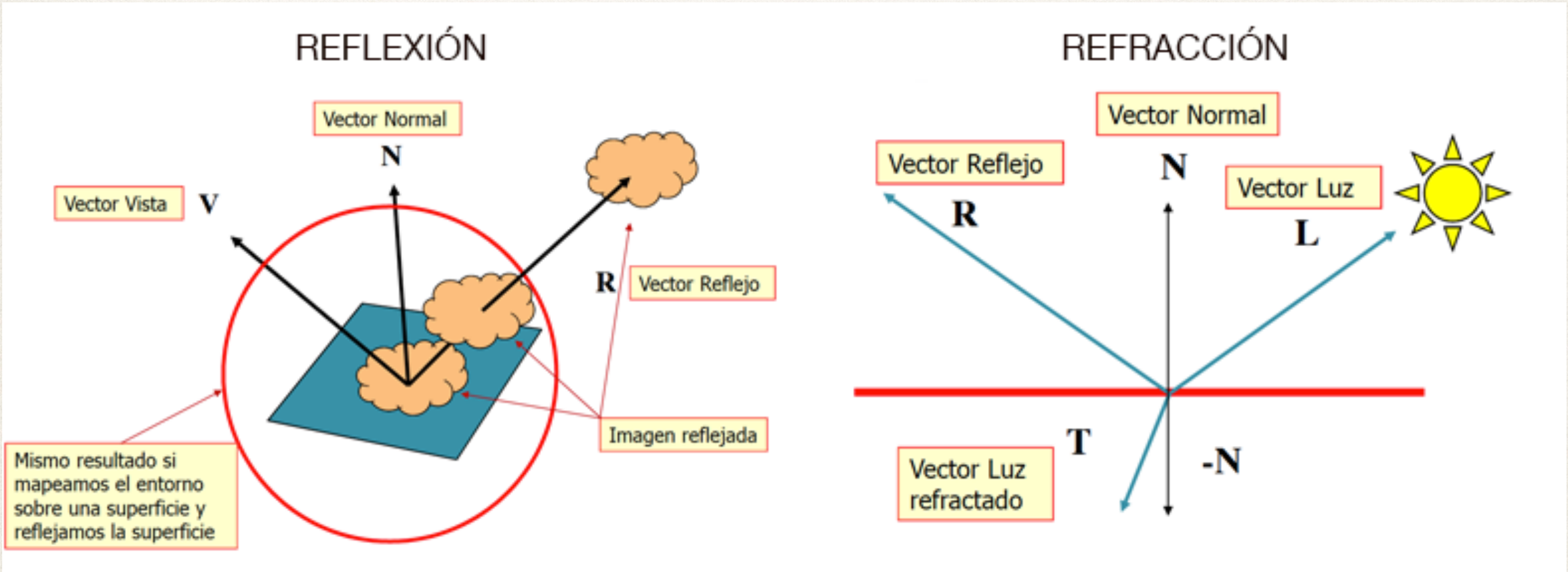
- En el Init:
    - Creamos y vinculamos un FBO
    - Creamos y vinculamos un Cubemap
    - Creamos 6 texturas vacías para el Cubemap
    - Creamos y adjuntamos la primera textura del cubo al FBO como Colour Attachment
    - Desvinculamos el cubemap
    - Creamos una textura vacía y la adjuntamos al FBO como Depth Attachment
    - Desvinculamos el FBO

# ¿Cómo lo hacemos?

- En el Display:
    - Dentro de un bucle realizaremos para cada cada una de las 6 caras del cubemap:
        - Vinculamos el FBO creado
        - Limpiamos el buffer
        - Adjuntamos la cara correcta del cubo al FBO como Colour Attachment
        - Renderizamos la escena completamente excluyendo el objeto reflector
        - Desvinculamos el FBO
        - Renderizamos la escena de forma normal utilizando el nuevo cubemap generado para pegar dicha textura sobre el objeto

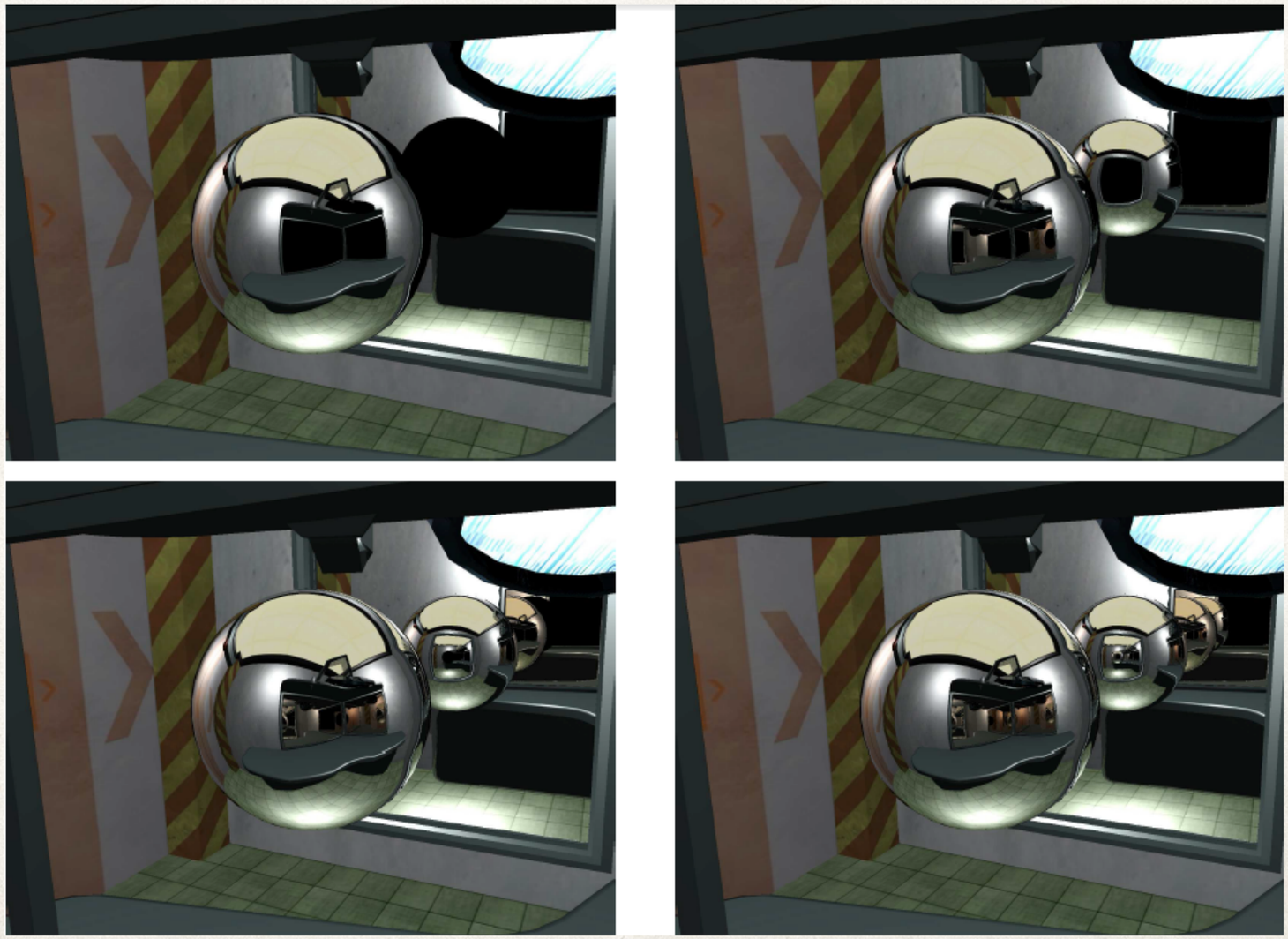
# El cálculo del color en los shaders

- El cálculo en los shaders lo realizaremos idéntico a como lo hacíamos con el mapa de entorno



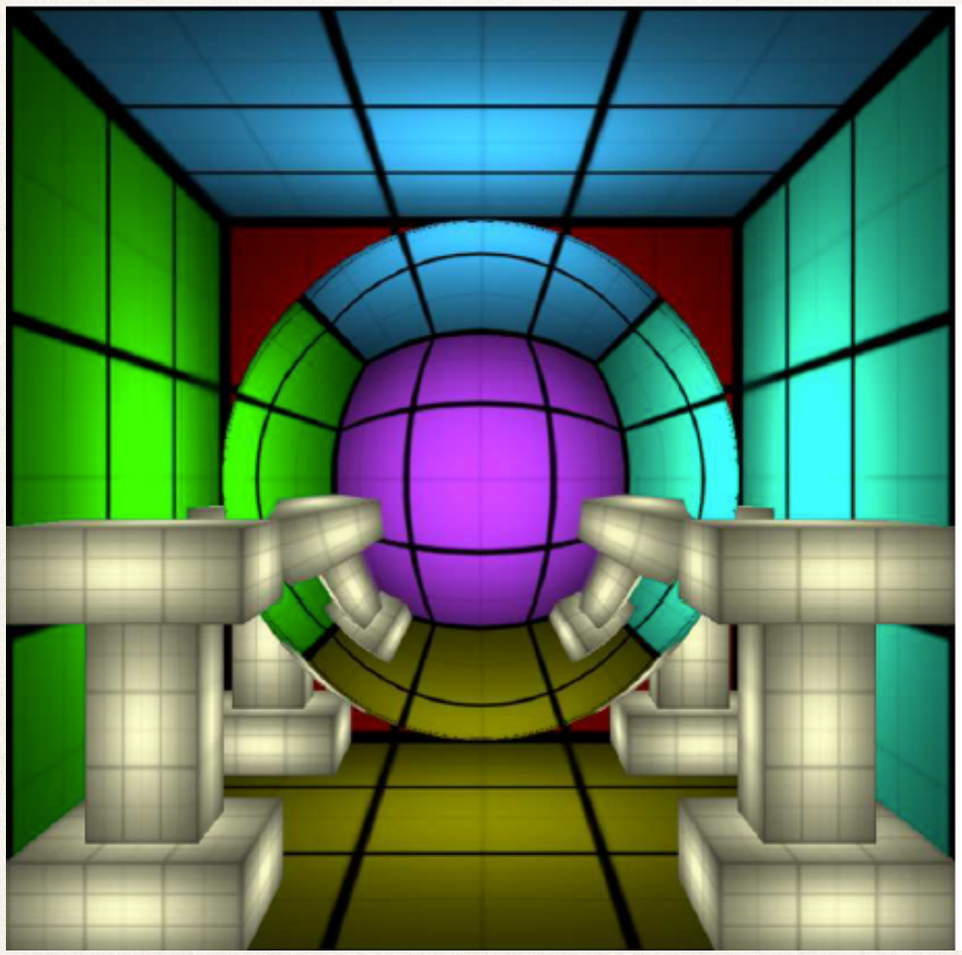
# Permite simular reflexiones infinitas

- Para simular reflexiones infinitas como la del espejo se repite recursivamente un número limitado de veces el fotografiar varias veces las mismas caras habiendo renderizado con el anterior cubemap de reflexión obtenido. Se debe de limitar la recesión a unas 18 veces para que no se sobrecargue.

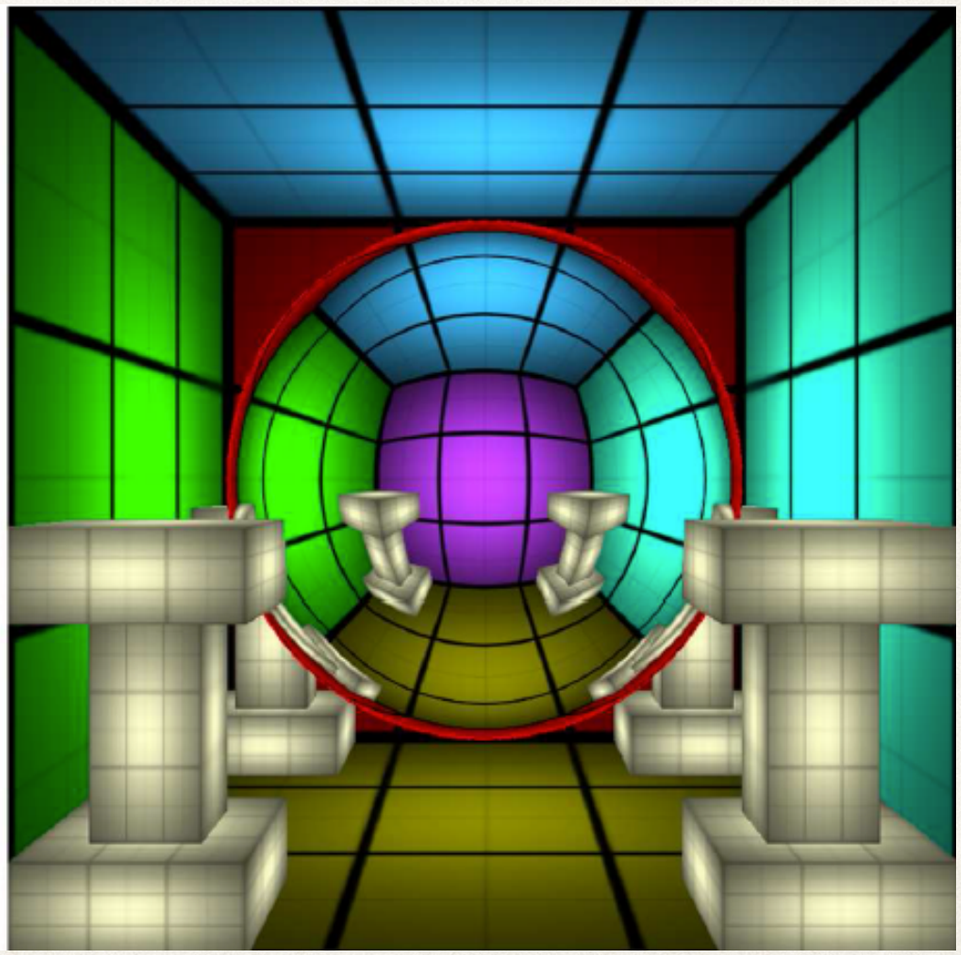


# Limitaciones del método

- No obtiene los mismos resultados que el reflejo de trazado de rayos, aunque compensa por el tiempo de renderizado



TRAZADO DE RAYOS



DYNAMIC CUBEMAP

# Limitaciones del método

- No se pueden obtener autoreflexiones ya que el objeto reflectante no se dibuja en su mapa de reflexión. Aunque existen técnicas añadidas que ya lo pueden conseguir.

- En caso de sobrecargar la escena con muchos objetos reflectantes se puede sobrecargar demasiado y perder el Realtime. Ya que por cada objeto reflectante en cada frame hay que fotografiar desde el centro de cada objeto las 6 visiones de 90º para cada cara de cada cubo

# Resultados obtenidos con Dynamic Cubemap

# Bibliografía


- **Nvidia OpenGL Cube Map Texturing**

  http://www.nvidia.com/object/cube_map_ogl_tutorial.html

- **Opengl Development Cookbook**

  https://www.packtpub.com/game-development/opengl-development-cookbook