Fundamentos de la Computación Grafica - TP Final

Bernardo Tuso

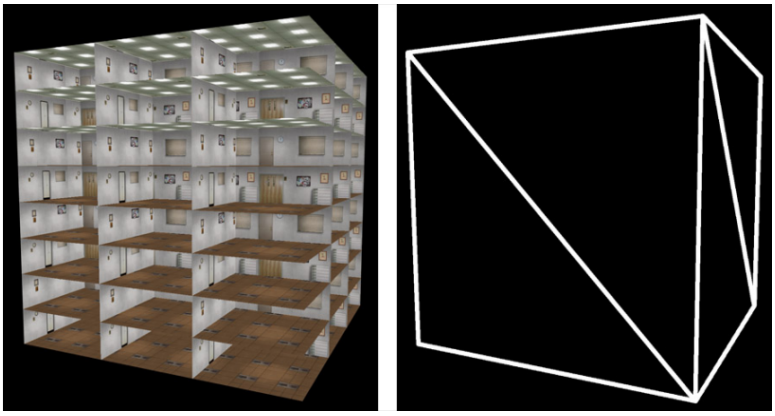
LU: 792/14

Email: btuso.95@gmail.com

Objetivo Principal

Para mi trabajo practico final elegí implementar un shader de simulación de interiores.

El shader busca utilizar la perspectiva de la cámara para generar una ilusión óptica que de la impresión de que hay un espacio 3D dentro del objeto. La manera mas sencilla de ver esto es con un ejemplo visual.



En la imagen podemos ver la geometría real del objeto del lado derecho y la “geometría” generada a partir de la ilusión óptica del lado izquierdo.
Fuente: Interior Mapping por Joost van Dongen

Para orientarme utilice uno de los primeros papers al respecto, titulado "Interior Mapping - A new technique for rendering realistic buildings” por Joost van Dongen. El mismo se puede encontrar con los archivos de la entrega.

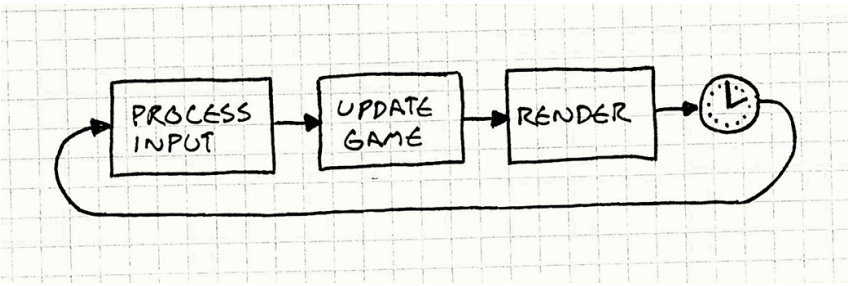
En lineas generales, el paper únicamente introduce la idea de aprovechar la dirección y perspectiva de la cámara para simular planos dentro del objeto. Segun la distancia de la cámara a la intersección con cada plano, se elige cual corresponde ser visto primero por la cámara. Una vez que se tienen estos planos para cada fragmento, simplemente se aplica una textura correspondiente a la pared o techo a mostrar.

En la practica veremos que hay algunos obstáculos y consideraciones a la hora de lograr un resultado creíble.

Outline del proyecto

Siguiendo con la modalidad de la materia de llegar al nivel mas bajo posible, me parecio acorde no utilizar ningun tipo de libreria extra como three.js sino simplemente utilizar lo visto en clase de WebGL y

Javascript.
Esto significo tener que tomar el control de principio a fin del flujo del proyecto, resultando en la tipica arquitectura de motor de juego.



Fuente: Game Programming Patterns por Robert Nystrom

Esto significo hacer polling de los inputs del usuario, actualizar el estado del “juego” en base a eso, y finalmente renderizar la escena resultante con los shaders aplicados.

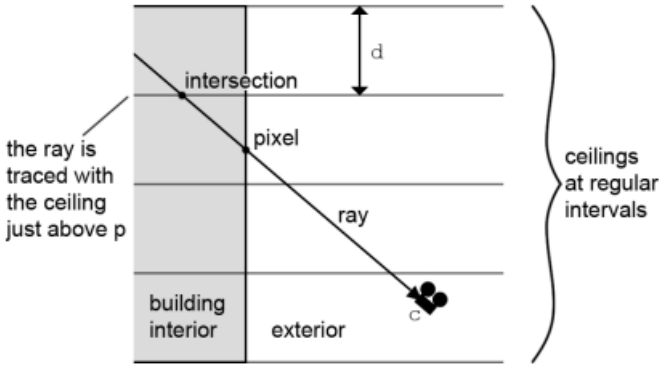
Un tema interesante a tratar en esta instancia es el de la cámara, la cual representa al “jugador”.

Cuando se mueve el mouse o se presionan las teclas de movimiento, esto resulta en un cambio del angulo y posición de la cámara. El componente de rendering mantiene una referencia a la cámara del jugador y recalcula la matriz de proyección en base al ultimo frame antes de pasarla a cada shader para renderizar.

Los objetos en la escena se separan entre renderizables y no renderizables, con cada objeto renderizable teniendo su shader asociado. En cada frame se recorre el árbol de la escena, ejecutando el shader de cada objeto a renderizar.

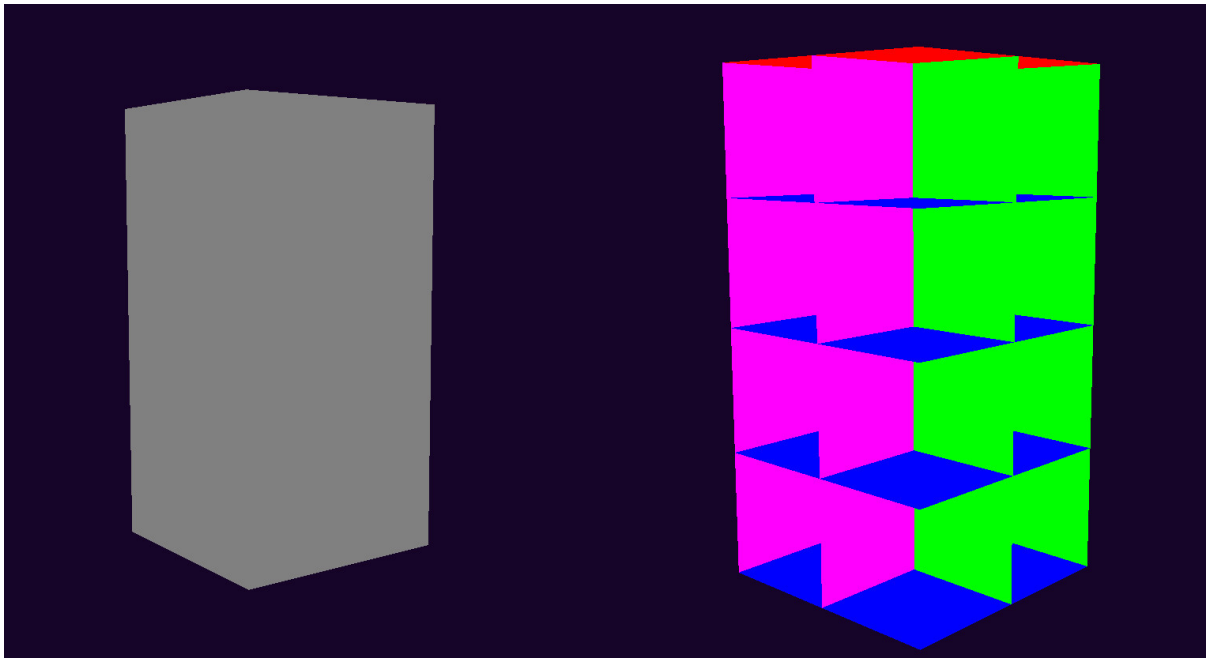
Outline del Shader

Como se menciono en la introduccion, el shader consiste en utilizar algebra lineal simple para simular superficies dentro del objeto al cual el shader se aplica.



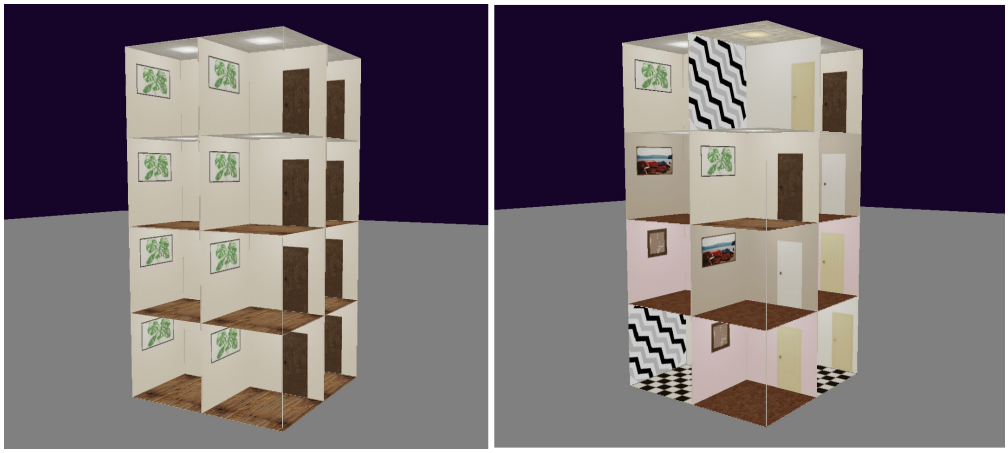
Este ejemplo del paper citado muestra la idea a seguir a la hora de decidir que “techo” y que piso esta viendo la cámara.
Fuente: Interior Mapping por Joost van Dongen

Utilizando la dirección en la cual la cámara se encuentra mirando y tomando la posición global del objeto, podemos trazar planos a intervalos regulares. En base a cual de esos planos se interseca primero, decidimos que “piso” estamos viendo. Esta misma lógica se aplica luego para las paredes en los otros ejes. El resultado de esto se puede ver en la imagen inferior.



Izquierda: El objeto con un color uniforme.
Derecha: Cada color representa un plano diferente para el shader, no hay geometria extra.

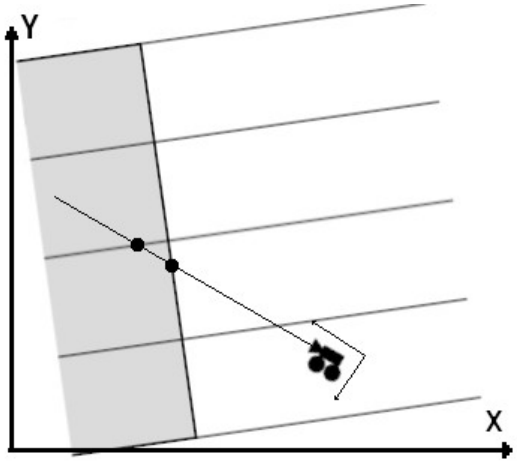
Luego solo necesitamos elegir que parte de la textura a utilizar para cada plano.



Del lado izquierdo: usando la misma textura para cada habitación. Del lado derecho: Utilizando una textura al azar.

Problemas y soluciones

Si bien la idea detrás del shader es bastante sencilla, a la hora de trabajar en 3D se generan casos complejos de resolver, ya que hay muchas variables en juego.



Un caso posible, una cámara invertida y un edificio inclinado.

Los problemas espaciales se solucionan simplemente teniendo un entendimiento fuerte del espacio objeto, espacio mundo, su relación con la cámara y como combinarlos para obtener el resultado deseado.

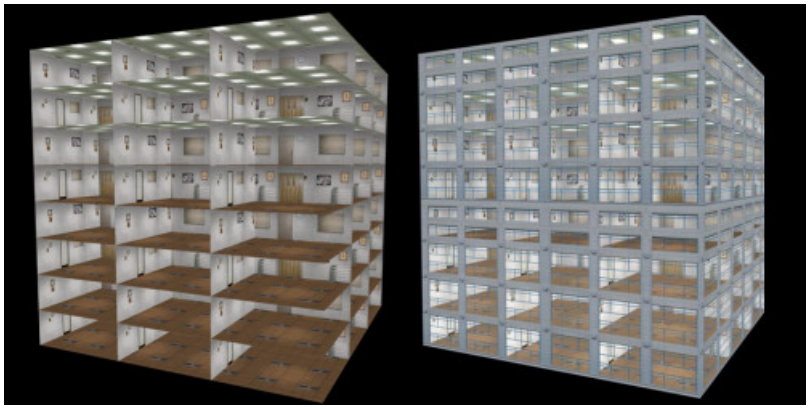
Otros problemas surgen a la hora de darle credibilidad al resultado, por ejemplo utilizando un acercamiento inocente a la hora de introducir azar al elegir que cuarto dibujar.

De cada lado del objeto se elige una textura, resultando en errores.

Para arreglar esto se necesita cambiar la forma de “numerar” cada cuarto, teniendo en cuenta su posición en el edificio resultando así en la misma elección de texturas.

Mejoras y Otros Metodos

En el paper de Joost van Dongen se menciona el uso de mascaras UV para superponer la fachada del edificio por sobre los cuartos generados. Si bien es algo que parece sencillo, en seguida se entra en problemas de realismo, ya que los edificios suelen tener fachadas no uniformes, ademas de tener detalles como puertas en el primer piso, etc. Por estas complejidades extras, decidí no incorporarlo a la entrega actual ya que no agregaba valor significativo.



Ejemplo de fachada en el paper de Joost. Notar como todos los pisos comparten la misma fachada, incluyendo la planta baja.

El paper que se tomo como base es considerado pionero en el area, sin embargo, nuevos papers proponen otras formas de lograr resultados similares, con otros beneficios y contras.

Arriba: Las imágenes proyectadas en los cuartos. Abajo: Las imágenes originales.
Fuente: https://www.youtube.com/watch?v=9zq3mcFcofk

Uno de los métodos mas populares involucra únicamente una imagen para todo el cuarto, deformando la proyección según el angulo de vision. Esto permite incorporar muebles y otros elementos de manera mas creíble a la escena, pero resulta en artefactos visuales mas pronunciados.