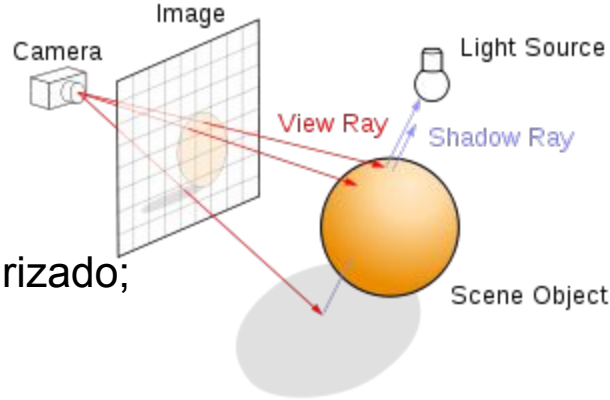
Paralelización de transformaciones y pruebas de intersección para Raytracing

Karen Jaqueline Reyes Flores
Alejandra Guadalupe Esquivel Guillén
Lic. Tecnologias para la Información en Ciencias.
Materia: Cómputo de alto rendimiento
Profesor: Dr. Ulises Olivares Pinto

Introducción



Raytracing es una técnica para la generación de imágenes o animaciones 3D, que aplica técnicas de iluminación, reflejo, texturizado; utilizada en proyectos como:

- Animación
- Simulación

El Raytracing modela el comportamiento de la interacción entre los objetos y la luz.Originalmente fue llamado algoritmo de Ray Casting y fue propuesto en 1980 por Turner Whitted.

El algoritmo de Ray Casting determina las superficies que son visibles en la escena por medio del trazado de rayos desde el observador (cámara) hasta la escena a través del plano de la imagen, calculando así las intersecciones más cercanas al observador determinando con ello el objeto visible.

Una de las principales ventajas de esta técnica es que realmente te enfocas en los rayos de importancia ya que no exploras todos los rayos que se generarían; sino en aquellos que son de relevancia para el observador.

Trabajos Relacionados

- Películas Pixar
- Animación en tiempo real

Video:

El 12 de junio de 2008 Intel demostró una versión especial de Enemy Territory: Quake Wars, titulada $Quake Wars: Ray Traced. La demostración se desarrolló en un sistema Xeon Tigerton de 16 núcleos (4 zócalos, 4 núcleos) que funcionaba a 2,93 GHz.

En SIGGRAPH 2009, Nvidia anunció OptiX , una API gratuita para el seguimiento de rayos en tiempo real en las GPU de Nvidia. Nvidia ha enviado más de 350,000,000 GPU compatibles con OptiX a partir de abril de 2013.

Trabajo a realizar

El Raytracing utilizado fue el localizado en el siguiente repositorio:

- https://github.com/bnaveenkr/raytracer

El objetivo principal sobre la implementación se basó en las transformaciones contenidas en el mismo raytracer:

- Rotación
- Escalación
- Traslación

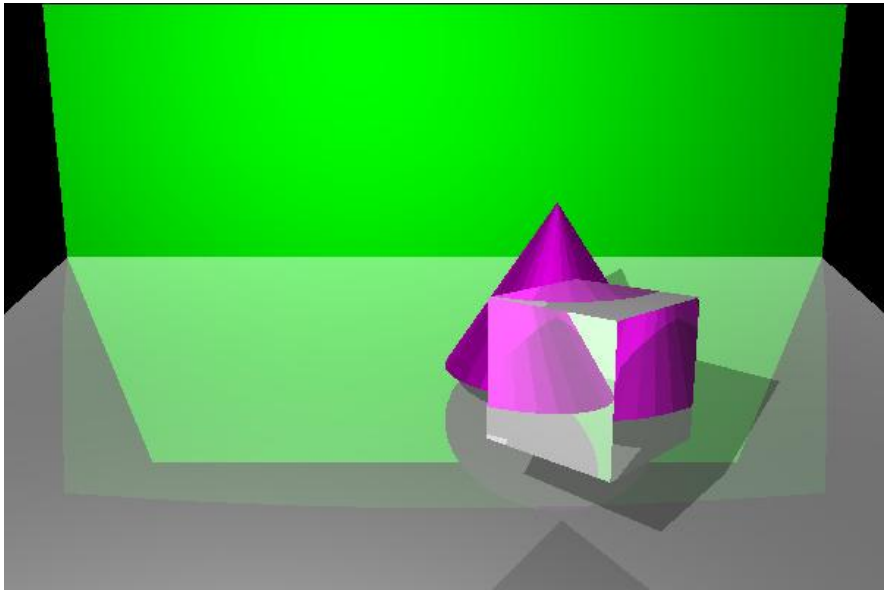
Anexando la paralelización de las pruebas de intersección entre rayos y la geometría poligonal para la producción de una imagen de salida.

Parámetros

IMAGEN: W=600, H=400

TRIANGLES: 32 P/O , GENERAL 80

RAYS: 240000



Paralelización de las transformaciones de traslación, escalación y rotación

```
#pragma omp parallel for
  for(i = 0; i < (*object).ntris; i++)
  {
      //printf(" Threads: %d\n", i);
      temp.v0 = matVecMult(M, ((*object).tri[i].v0));
      temp.v1 = matVecMult(M, ((*object).tri[i].v1));
      temp.v2 = matVecMult(M, ((*object).tri[i].v2));
      (*object).tri[i].v0 = temp.v0;
      (*object).tri[i].v1 = temp.v1;
      (*object).tri[i].v2 = temp.v2;
  }
```

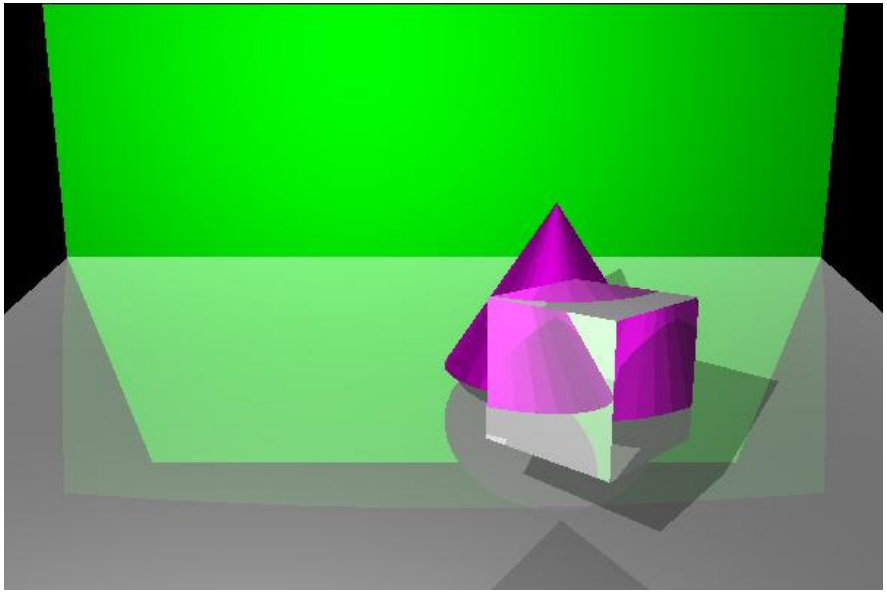
Paralelización sobre la generación de la imagen

```
#pragma omp parallel for collapse(2) private(outcolor, ray, i, j, index) firstprivate(width, height, image) shared(scene, light, recur)
  for(j = 0; j < height; j++){
      for(i = 0; i < width; i++)
      {
          //printf(" ray id: %d , %d\n",i,j);
          ray = generateRay(i, j, cam);
          outcolor = vector2color(trace(ray, scene, light, recur));
          #pragma omp critical
            setPixel(&image, i, j, outcolor, index, width, height);
            //printf("finish  critical section: %d \n", omp_get_thread_num());
      }
  }
```

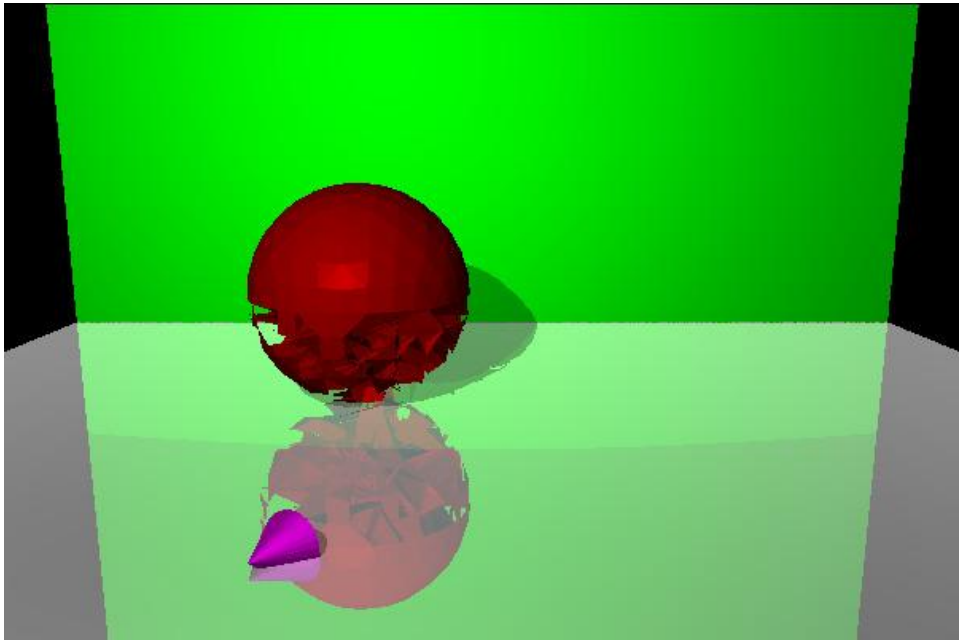
Conclusiones

Se aplicaron las operaciones por separado a una imagen que fue predefinida anteriormente.

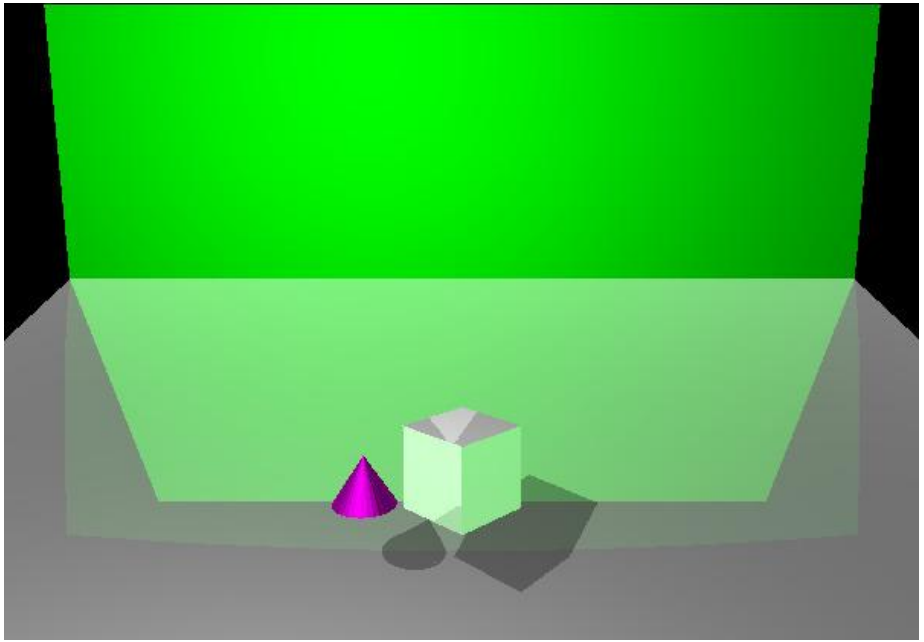
Las transformaciones de las imagenes se muestran en las siguientes imágenes.



Original



Aplicando rotate



Aplicando translate y scale

TIEMPOS Y SPEED-UP

TIEMPO EN SERIAL

```
➜ raytracer-master time ./raytracer
Image: output.ppm, Size: 600x400
Initial Ray Count: 240000, Triangle Count: 580, Recursion Depth: 2
Raytracing...
Done, writing image...
Done
./raytracer  104.70s user 0.36s system 99% cpu 1:45.68 total
```

TIEMPO EN PARALELO

```
➜ raytracer-master time ./raytracer
Image: output.ppm, Size: 600x400
Initial Ray Count: 240000, Triangle Count: 580, Recursion Depth: 2
Raytracing...
Done, writing image...
Done
./raytracer  115.88s user 0.37s system 274% cpu 42.321 total
```

SPEED-UP EN PARALELO

S=Ts/Tp

P=4

S= 1:45:68 / 42:321 = 2.497 s

ESCUELA NACIONAL DE ESTUDIOS SUPERIORES

UNIDAD MORELIA

GRACIAS POR SU ATENCIÓN.

Karen Jaqueline Reyes Flores
karenreyes346@gmail.com
Alejandra Guadalupe Esquivel Guillén
alejandraeg9899@gmail.com
Lic. Tecnologias para la Información en Ciencias.
Cómputo de alto rendimiento, Dr. Ulises Olivares Pinto