Ray Tracing

Venkatesh Tantravahi

What is Ray Tracing ?

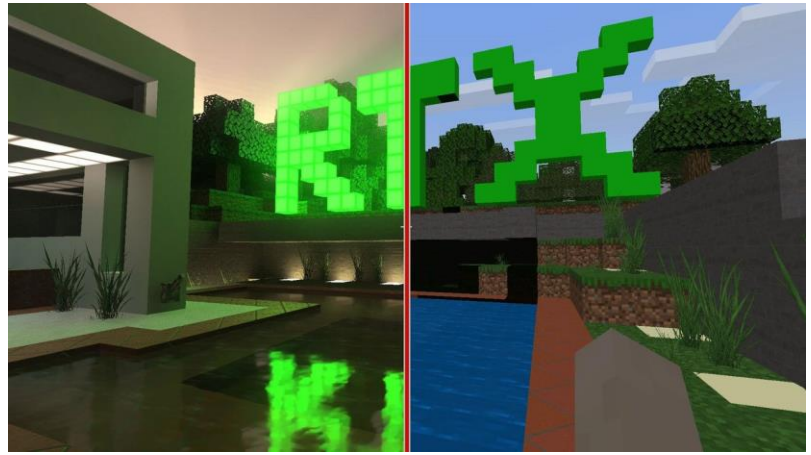
- It is a technique for modeling light transport for use in a wide variety of rendering algorithms for generating digital images.
- ray tracing-based rendering techniques, such as ray casting, recursive ray tracing, distribution ray tracing, photon mapping and path tracing, are generally slower and higher fidelity than scanline rendering methods.

Dürer woodcut of Jacob de Keyser's invention. With de Keyser's device, the artist's viewpoint was fixed by an eye hook inserted in the wall. This was joined by a silk string to a gun-sight style instrument, with a pointed vertical element at the front and a peephole at the back. The artist aimed at the object and traced its outline on the glass, keeping the eyepiece aligned with the string to maintain the correct angle of vision.

Some Applications

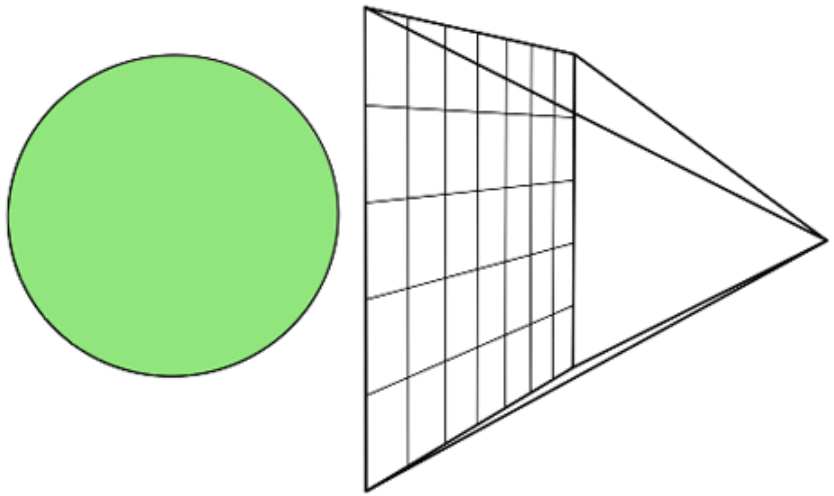
Image Rendering

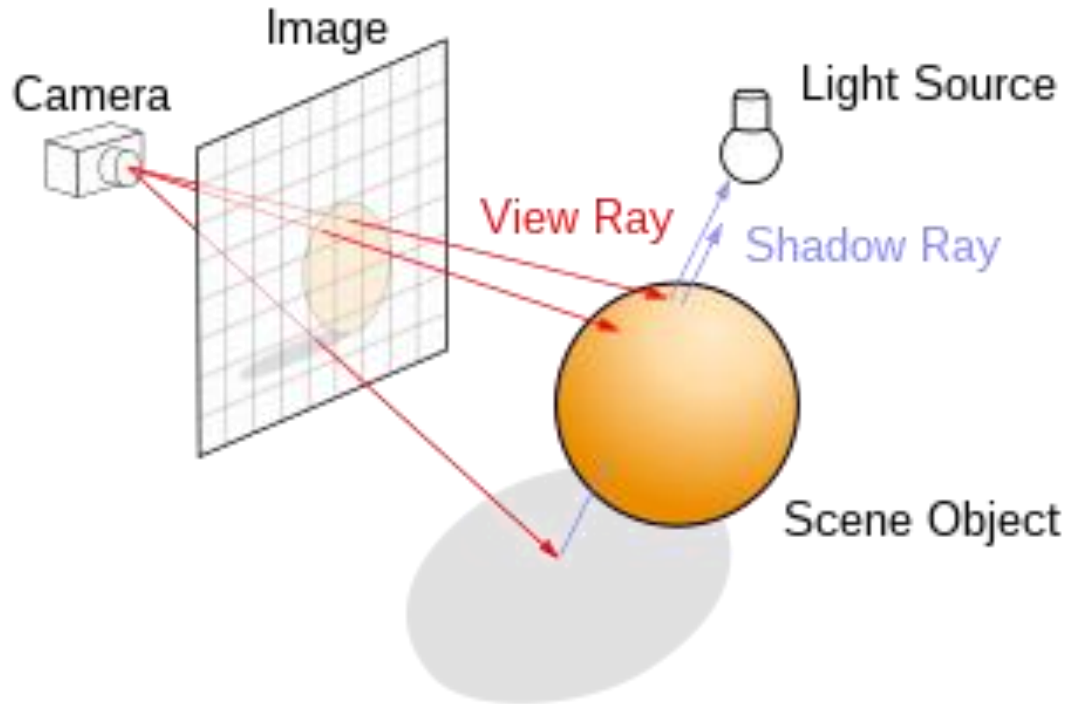
Animation



Gaming Graphics

How it happens?





Initial Code Workout on CPU from Scratchpixel

C++ Source File

Why Choosing a GPU ideal than CPU?

- Parallel Processing and High Thruput
- Efficient Memory Management
- Optimized for Graphics Processing
- Scalability

Results Comparison

```
C:\Users\vtant\Downloads>ray_tracing
Rendering finished in 0.276561 seconds.

C:\Users\vtant\Downloads>raytracing
Rendering finished in 150.543 seconds.

C:\Users\vtant\Downloads>g++ -o raytracing -O3 -Wall ray_tracing.cpp
ray_tracing.cpp:21: warning: "M_PI" redefined
   21 | #define M_PI 3.141592653589793
      |
In file included from C:/MinGW/include/c++/13.2.0/cmath:47,
                 from ray_tracing.cpp:4:
C:/MinGW/x86_64-w64-mingw32/include/math.h:45: note: this is the location of the previous definition
   45 | #define M_PI           3.14159265358979323846
      |
ray_tracing.cpp:22: warning: "INFINITY" redefined
   22 | #define INFINITY 1e8
      |
C:/MinGW/x86_64-w64-mingw32/include/math.h:351: note: this is the location of the previous definition
  351 | #define INFINITY       __builtin_inff()
      |
ray_tracing.cpp:301: warning: ignoring '#pragma omp parallel' [-Wunknown-pragmas]
  301 |     #pragma omp parallel for schedule(dynamic)
      |

C:\Users\vtant\Downloads>raytracing
Rendering finished in 156.369 seconds.
```

```
Rendering a 1200x800 image with 10 samples per pixel in 8x8 blocks.
==7438== NVPROF is profiling process 7438, command: /content/cudart
took 2.35042 seconds.
==7438== Profiling application: /content/cudart
==7438== Profiling result:
            Type  Time(%)      Time     Calls       Avg       Min       Max  Name
 GPU activities:   97.52%  2.38525s         1  2.38525s  2.38525s  2.38525s  render(vec3*, int, int, int, camera**, hitable**, curandStateXORWOW*)
                    1.77%  43.178ms         1  43.178ms  43.178ms  43.178ms  create_world(hitable**, hitable**, camera**, int, int, curandStateXORWOW*)
                    0.69%  16.882ms         1  16.882ms  16.882ms  16.882ms  free_world(hitable**, hitable**, camera**)
                    0.03%  660.27us         1  660.27us  660.27us  660.27us  render_init(int, int, curandStateXORWOW*)
                    0.00%  3.5200us         1  3.5200us  3.5200us  3.5200us  rand_init(curandStateXORWOW*)
      API calls:   89.32%  2.42915s         5  485.83ms  5.0610us  2.38527s  cudaDeviceSynchronize
                    8.22%  223.41ms         1  223.41ms  223.41ms  223.41ms  cudaMallocManaged
                    1.53%  41.609ms         1  41.609ms  41.609ms  41.609ms  cudaDeviceReset
                    0.71%  19.340ms         6  3.2234ms  7.5030us  16.912ms  cudaFree
                    0.20%  5.4777ms         5  1.0955ms  8.2760us  5.2789ms  cudaLaunchKernel
                    0.01%  329.50us         5  65.900us  3.4430us  226.20us  cudaMalloc
                    0.00%  132.78us       114  1.1640us     163ns  51.564us  cuDeviceGetAttribute
                    0.00%  12.501us         1  12.501us  12.501us  12.501us  cuDeviceGetName
                    0.00%  6.0680us         1  6.0680us  6.0680us  6.0680us  cuDeviceGetPCIBusId
                    0.00%  4.7550us         1  4.7550us  4.7550us  4.7550us  cuDeviceTotalMem
                    0.00%  2.0860us         5     417ns     199ns     740ns  cudaGetLastError
                    0.00%  1.5600us         3     520ns     193ns  1.0940us  cuDeviceGetCount
                    0.00%  1.1980us         2     599ns     228ns     970ns  cuDeviceGet
                    0.00%     466ns         1     466ns     466ns     466ns  cuModuleGetLoadingMode
                    0.00%     246ns         1     246ns     246ns     246ns  cuDeviceGetUuid

==7438== Unified Memory profiling result:
Device "Tesla T4 (0)"
   Count  Avg Size  Min Size  Max Size  Total Size  Total Time  Name
      96  117.21KB  4.0000KB  0.9961MB  10.98828MB  1.017988ms  Device To Host
      32         -         -         -           -  3.108054ms  Gpu page fault groups
Total CPU Page faults: 35
```