Efficient and realistic volumetric cloud rendering

Dragos-Vlad, Tatar

2022

Summary

Classification

- AMS Mathematical Subject Classification
  78A10. Physical optics
- ACM Computing Reviews Categories and Subject Descriptors
  I.3.7. Three-Dimensional Graphics and Realism, I.4.10 Image Representation

Abstract

1 Introduction

1.1 Overview

Nowadays, due to the improvement in the quality of graphics processing units, the demand for high-quality graphics in video games, movies, and any multimedia application is growing incredibly fast. Whether we talk about high-detail terrain, shadows, water, realistic character animations, or weathering systems, the potential for advancement in graphics is never-ending, and rendering real-time realistic clouds is a small part of it.

Over the years there have been many methods of rendering clouds in video games, but many lack quality. Most of the existing ways of drawing clouds imply drawing some overlapped 2D textures and applying some functions to those textures to try to mimic real clouds. Due to the improvement in computational power of the new GPUs, volumetric rendering came back to the attention of graphics programmers who started to think that volumetric rendering, a very time-consuming technique in the past, could be used to render more realistic natural elements such as fire, fog, water foam in real-time, using the volumetric rendering technique that is used in medical imaging.

1.2 Goals

The goal of this article is to see if we could use volumetric rendering to draw realistic real-time clouds. If we manage to prove that volumetric rendering can be used to draw clouds in real-time and that it gives better visual results than the other techniques used nowadays, this technique could be used in many incoming multi-media applications and video games to improve the visual aspect of the scene.

2 Related work

2.1 Cloud types

Clouds are of many different types in the real world, which have been categorized as low clouds, middle clouds, high clouds, and unusual clouds.

- Low clouds: Surface - 2 km
  Cumulus, Stratus, Nimbostratus
- Middle clouds: Surface - 2 - 7 km
  Altocumulus, Altostratus
- High clouds: Surface - 5 -13 km
  Cirrus, Cirrocumulus
- Unusual clouds
  Lenticular, Mammatus

2.2 Clouds lighting model

When light enters clouds, it bounces around inside for a while before exiting the clouds through a method called scattering. Light is deflected and sent in all directions, appropriately named diffuse light. Cloud droplets scatter all wavelengths of visible radiation, more or less equally. Large clouds reflect more and less light can penetrate, making the underside darker. Larger drops will likely absorb rather than scatter, causing the cloud to be even darker. While this effect varies depending on whether it is high, middle, or low clouds, they all travel similarly, causing different effects.

2.3 Ray marching

A popular technique for clouds rendered in real-time is something called Ray marching, being an extension of Ray tracing. In Ray tracing, to render a primitive a function is needed that given a primitive such as a sphere and input ray, tells you exactly where the ray intersects with a primitive, picks the closest intersection, and renders it into the scene. With Ray marching, however, there is no simple intersection function. Given a point on the ray, it is possible to estimate how close the point is to the surface, but it is not known how far the ray needs to be extended to hit the surface. So, in short, it "marches" one step at a time.

- Start at the beginning of the ray - the near plane for scene rendering, or the intersection with the bounding volume if it's just one object in the scene(P0).
- Evaluate the distance function to get an estimate from P0 to the surface
- Move forward along the ray according to the estimate. The move should be conservatively short, so there is no chance to skip the surface anywhere.
- The previous step will give a new point(P1), get a new estimate and repeat the same steps until either the step is within a threshold distance of the surface or the maximum number of steps is reached.

2.4 Volumetric Rendering

Volumetric rendering is a set of techniques used to display a 2D projection of a 3D discretely sampled data set, typically a 3D scalar field. A typical 3D data set is a group of 2D slice images acquired by a CT, MRI, or MicroCT scanner. Usually, these are acquired in a regular pattern and usually have a regular number of image pixels in a regular pattern. Volumetric rendering enables the creation of realistic materials that interact with light in a complex way, such as fog, smoke, water, and glass.

2.5 Texturized primitives

In this option, basic surfaces like skydomes or skyboxes are texturized with an omnidirectional true color capture of the sky. Other implementations use procedural textures applied to spheres and ellipsoids, which was the starting point for cloud generation in the early years. These methods may be considered as obsolete with modern GPUs, however, they are employed in smaller devices that do not have three-dimensional (3D) acceleration. The main limitation of these methods is the inability to approach, turn around, or traverse gaseous bodies and the lack of animation of the different cloud parts.

2.6 Particle system

Researchers have tried to achieve simulations of clouds using basic quads or triangle surfaces with Gaussian textures. There was conceived a new algorithm using particles with efficient multiple-scattering illumination systems. The use of this technology increases performance and speed. This technique is considered useful for physical workload model simulators. However, the overall realism is not accurate.

2.7 Geometry distortion

The basic idea behind this technique consists of drawing a complex cloud mesh with a group of mega particles. Then, a lighting algorithm, like Phong or Gouraud shading, is used on the geometry. After this step, the cloud map is blurred using a Gaussian filter to distort it. To do this, a quad is placed at the center of every cloud and billboard vertex concerning the camera, covering the entire cloud from any angle. This quad is rendered to distort the cloud map and sample a two-channel fractal/noise texture to obtain a distortion offset. Afterward, the blurred cloud map is sampled using these texture coordinates and the distance from the quad to the camera. Optionally, a radial blur may be performed to soften the resulting image. In the final stage, the render target is merged into the back buffer. This method is midway between the particle systems and volumetric rendering techniques, with good performance and easy shading. However, it lacks accurate realism and is not suitable for cloud shapes other than cumulus.

3 Method

3.1 Research Hypothesis

The research hypothesis of this article is that we can render realistic clouds using volumetric rendering in real-time, ensuring the frame rate is sitting at 30 frames/second. Simultaneously, additional features are added to the scene alongside the clouds to push the computational cost while still keeping the frame rate at 30 FPS.

3.2 Mathematical model

The mathematical model for the problem is the following:

LC = $\{c \mid t$ is a low cloud in the scene$\}$,

MC = $\{c \mid t$ is a middle cloud in the scene$\}$,

HC = $\{c \mid t$ is a high cloud in the scene$\}$,

UC = $\{c \mid t$ is a unusual cloud in the scene$\}$,

We will also define a current frame rate:

fr = the current frame rate of the program

We will also define the following functions:

$f_{accuracy}$: $LC, MC, HC, UC \rightarrow R, f_{accuracy}(c)$ = the accuracy of the cloud rendered compared to a real cloud of that type. It will be automatically 0 if fr is under 20 FPS.

To test the accuracy of each technique, we run the program for an hour and compute the overall result given by $f_{accuracy}$ for each cloud rendered once a second.

3.3 Research methodology

The method we will use in this article will consist of volumetric rendering and the cloud lighting model to draw a scene in real-time that contains very good-looking and realistic clouds. To draw the scene I will use C++ and OpenGL. There will be three components in the scene that define a complete version of the clouds: a global directional light that will act like the main source of light in a real environment(the sun for example), a skybox that is affected by the position of the directional light and the clouds themselves. To simulate a real graphics application I will also add some terrain generation and water rendering.

3.4 Contribution

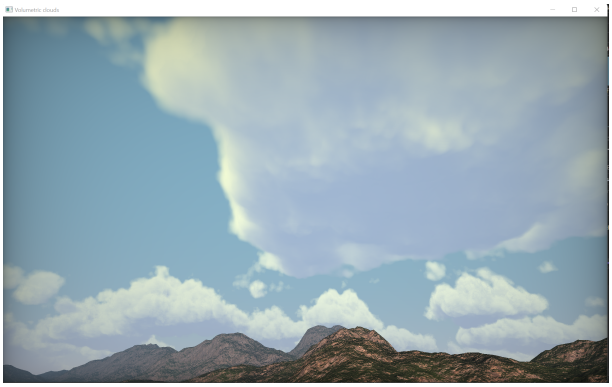
The original contribution of this article is the idea of using volumetric rendering to draw realistic clouds. In the future, this technique could become the new state of the art in the field of cloud rendering. This idea could also be extended to other natural elements such as fire, smoke, fog, etc.

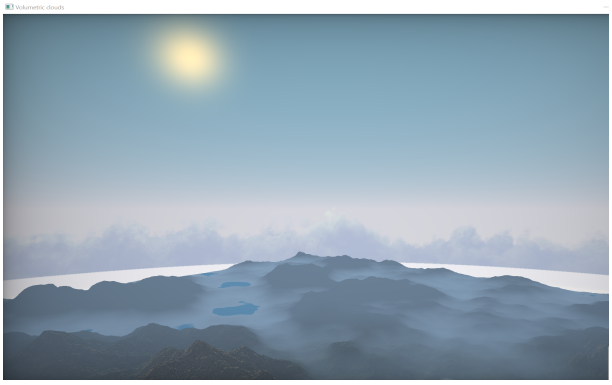
4 Experiments

To prove that the algorithm is efficient and also that it gives better visual results I will create a 3D scene in which I am going to draw different types of clouds. Then, I will take pictures, from different angles, of the result obtained using volumetric rendering and of the one using one of the algorithms that are used nowadays.

The experiments will consist of drawing a 3D scene consisting of directional light, a skybox, and some clouds. Firstly, we will use some algorithms that are used nowadays for rendering the clouds, and then we will use the volumetric rendering approach and compare the resulting images. Also, the frame rate was set to 30 FPS, and the program performed without issues.

Volumetric rendering results:





Texturized primitives results:

We can see clearly that clouds rendered using volumetric rendering look more real than the others. The ones with texturized primitives leak some sense of depth and also is not possible to go through them since they are only 2D textures, so if somebody wants to create a flying simulator will not be able to use this approach, which is one of the state of the art approaches nowadays.

5 Conclusion

6 Links

https://github.com/Dragos123321/ResearchProject - GitHub link

7 Bibliography

- [1] Carlos Jiménez de Parga, Sebastián Gómez Palomo. Efficient Algorithms for Real-Time GPU Volumetric Cloud Rendering with Enhanced Geometry, 2018, pp.3-18.
- [2] Daniel Scherzer, Florian Bagar, Oliver Mattausch. GPU Pro 3: Volumetric Real-Time Water and Foam Rendering, 2012, pp.119-132.
- [3] Miroslaw Jonasz, Georges R. Fournier. Light Scattering by Particles in Water: Theoretical and Experimental Foundations, 2007, pp.87-143.
- [4] Bartlomiej Wronski. GPU Pro 6: Volumetric Fog and Lighting, 2015, pp.217-242.
- [5] Andrew Schneider. GPU Pro 7: Real-Time Volumetric Cloudscapes, 2016, pp.97-126.
- [6] Gang Wang, Zhenzhou Ji, Zexu Zhang. Using instance for large scale volumetric clouds rendering in real-time, 2012, pp.4417-4418.
- [7] Gilbert N. Plass, George W. Kattawar. Monte Carlo Calculations of Light Scattering from Clouds, 1968, pp.415-419.
- [8] Patric Ljung. Efficient Methods for Direct Volume Rendering of Large Data Sets, 2006, pp.15-16.
- [9] Daniel Weiskopf. GPU-Based Interactive Visualization Techniques, 2006, pp.21.
- [10] Robert Houze. Types of Clouds in Earth's Atmosphere, 2014, pp.3-20.
- [11] Oscar Ripolles, Francisco Ramos, Anna Puig-Centelles, Miguel Chover. Real-time tessellation of terrain on graphics hardware, 2012, pp.148-154.
- [12] Corey Packard, Michael L. Larsen, Subin Thomas, Will H. Cantrell. Light Scattering in a Turbulent Cloud: Simulations to Explore Cloud-Chamber Experiments, 2020, pp.2-7.
- [13] Matt Pharr. Programming techniques for high-performance graphics and general-purpose computation, 2005, pp.126-130.
- [14] Adolfo Munoz. Higher Order Ray Marching, 2014, pp.167-177.
- [15] Alain Galavan. Real-Time Ray Tracing, 2019.