Cloud Modeling, Rendering and Animation in Computer Graphics

Joana Isabel Afonso Gomes, a84912
Visualization and Illumination Essay
Computer Graphics - MSc in Software Engineering
University of Minho

Figure 1: Landscape with computer generated clouds [6]

ABSTRACT

Clouds are a very relevant topic when it comes to realism of landscapes in Computer Graphics, being a very complex and a very expressive visual feature of skies. With this in mind, their realist display comes with great challenges, making generation of clouds a topic of reference it what concerns research in this area. To get close to a convincing real-time exhibition of clouds, one first have to come across many steps and make them all go thowards a realistic display.

Thus, we need methods to model, render (from different points of view) and animate clouds, resulting in countless techniques suggested to those ends. Simultaneously, cloud shapes, difference in altitudes, dynamic lightining, shadows and appropriate transitions and motion rules help creating the expected visual effects that contribute to lifelike clouds. Realistic and convincing cloud scenes are not only the result of light scattering in participating media, but also the result of dynamic clouds that can evolve over time, cast shadows and interact with its environment.

This survey is meant to cover several studies that envolve visual simulation of clouds, as well as reviewing and summarazing notorious progresses of techniques in this area, covering efforts to simulate clouds with a minimal amount of computation (resorting to graphics hardware) and fast image generation.

KEYWORDS

computer graphics, CG, clouds, modeling, rendering, animation

ACM Reference Format:
Joana Isabel Afonso Gomes, a84912 and Visualization and Illumination Essay. 2021. Cloud Modeling, Rendering and Animation in Computer Graphics: .

1 INTRODUCTION

Many methods have been proposed for visual simulation of clouds in the history of computer graphics. These methods are used in many applications such as flight simulators, movies, computer games, and so on.

The first goal is to do the modeling of the clouds so that they look authentic. Their shape and color need to change depending on the position of the sun and the looking point. One can make them look menacing, representing incoming storms, epic or discreet, thin or massive, etc.

Therefore, on the one hand, the density distribution of water droplets (cloud particles) should be defined in three-dimensional space to obtain extremely realistic images.

On the other hand, once we have this realistic distribution, we need to consider an effective illumination model to capture the light transport on the generated shapes (considering its interaction with the cloud particles).

Focusing on static scenes, we may obtain photorealistic images where the clouds can convey a convincingly real static appearance. However, for applications such as movies, videogames or flight simulators, there are dynamic scenes, where they change shape and color. Clouds usually move slowly, but their dynamic aspect is essential in open world scenes with progressive weather changes.

For these reasons, animation techniques are required to produce a natural evolution of the modeled clouds as a function of time and possible complex motions.

Accordingly, in this essay, will be reviewed some of the previous work on modeling, rendering, and animating clouds realistically, and the attempts and struggles to improve them.

2 MODELING

Cloud modeling in computer graphics is creating shapes so that they look like authentic clouds. Various methods have been advanced for this purpose, divided in different categories, according to the used approach. These techniques embrace generating the shapes in a procedural approach (where normally is used some kind of noise function or they're just generated from a texture) or based on captured images from real clouds.

A **procedural approach** is the common way to go. In what concerns techniques with this concept, it must be considered a method proposed by Geoffrey Y. Gardner [28] using **textured** ellipsoids to model clouds. The original studies of modeling clouds shapes in computer graphics apply 2D and 3D textures, and in this particular work, ellipsoids are employed to model a 3D cloud structure and apply a Fourier series to model the details, shading and translucence. To create the cloud layers, the same parameters are applied on a single textured plane instead of the ellipsoids.

Hinging on their appearance, clouds can be classified mainly into cirrus, stratus or cumulus clouds. **Cirrus** clouds are characterized by thin, wispy strands, while **stratus** clouds are low-level clouds with horizontal layering and uniform base. But the **cumulus** clouds and its variants are the most interesting and challenging types due to the various patterns they form, being formed because of vertical ascending currents. Because of this, the great majority of the research in computer graphics directly or indirectly aims these types of clouds.

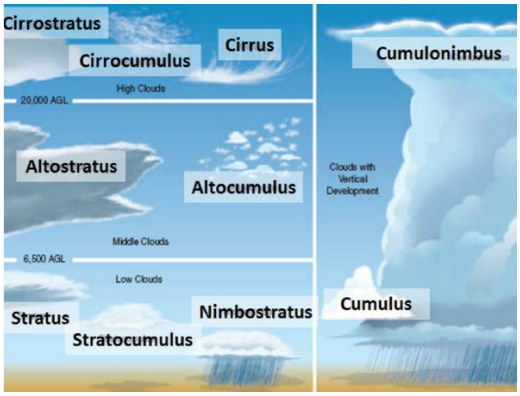

Figure 2: Various classification of clouds based on their elevation and appearance. [25]

The method proposed by Gardner is able to create impressive clouds from all the types referred before, talking of course in terms of the technology available at that time (80's).

Later, in 1990, David S. Ebert and Richard E. Parent [9] developed a method where controlling the transparency of the objects defines the space occupied by the gaseous substance, using the solid texturing of the polygonal mesh to model the clouds (and other gaseous phenomena).

Having brought to light these works, is still unquestionable that texture mapping on 3D shapes can be a lot complicated.

Consequently, it's usually easier to generate shapes recurring to the use of procedural **noise**. A lot of work has been developed resorting to one or more kinds of noise to model clouds.

David S. Ebert developed a method combining metaballs and a noise function in 1997 [11], using a procedural volumetric implicit function volumetric modeling to generate cumulus clouds.

Previously, in 1993, Georgios Sakas [27] modeled clouds by using spectral synthesis, motivated by turbulence theory (chaotic changes in pressure and flow velocity), in a study where texture is defined in the frequency domain and transformed to the Euclidean space.

In 2003, Joshua Schpok et al. [22] developed a real-time system for the procedural modeling of clouds, providing a cloud modeling framework that uses volumetric implicits to define the density, transparency and shadowing of the cloud. The user interactively outlines the cloud shape using ellipsoids and describes the low-level details from a collection of preset noise filters.
These methods based on procedural modeling can in fact generate realistic clouds, but many parameters are required to be specified by trial and error.
There is also the hypothesis to generate clouds by physically based simulation of the cloud formation process, as proposed by Kajiya and Herzen and Miyazaki et al. By using their methods, realistic clouds can be created, and these can also be used for animating clouds. Nonetheless, their computational cost is very high.

The above problems can be resolved by employing an **image-based** approach, that is, modeling cloud shapes from photographs of real clouds.
Since taking multiple photographs of clouds from different directions is usually difficult, the image-based methods for modeling clouds use a single image as input.

The purpose of these image-based modeling methods is not to reconstruct the exact shapes of clouds in an image, but to use it as a guide to generate clouds that look similar to those in the image.

Recently, Chunqiang Yuan et al. [8] suggested a method to model clouds from an image by assuming a symmetric cloud structure. The image is segmented into various subregions, and the height field is computed using the pixels labeled as clouds. A propagation procedure is devised that constructs cloud geometry from the cloud pixels progressively. To improve the generated appearance, the front part of the cloud is refined by adding shape constraints and

the back portion geometry is simplified. Finally, the cloud surface obtained is filled with the particles via an adaptive sampling process.

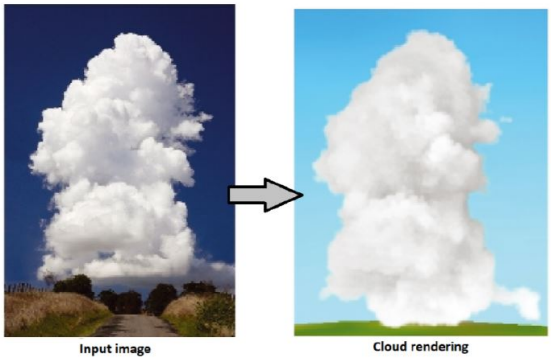

Figure 3: Cloud modeled from a single image by Yuan et al. [17]

Way before this suggestion, still in the 90's, Yoshinori Dobashi et al. proposed a cloud modeling in 3D using metaball by making use not of photographs taken from the ground but from **satellite images**. Cues are taken from the satellite image by classifying each pixel as either cloud or background, and metaballs are generated in pixels with the maximum intensity identified as cloud regions of the satellite image. After each such addition, the center and radius of the metaball are approximated. The termination criterion is based on the difference of the synthetic image with that of the satellite image, given an error threshold.

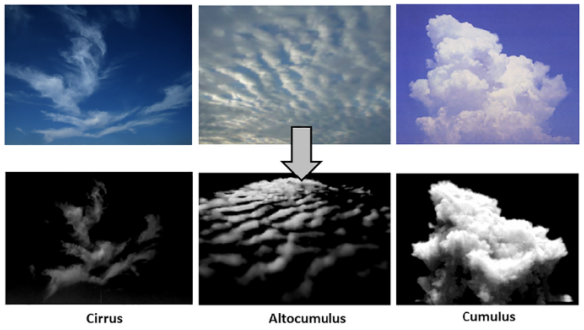

Figure 4: Cirrus, altocumulus and cumulus cloud texture and the corresponding density distribution, generated from single images by Dobashi et al. [17]

3 RENDERING

Real-time rendering of realistic and convincing cloud scenes has always been a desired feature in computer graphics. Many cloud rendering techniques have been developed over the years and are still being researched.

To obtain realistic colors of clouds, we need to simulate optical phenomena that happen inside clouds. One of the very relevant factors it that the lighting captures the illumination or interaction of cloud particles (small water droplets) with the surrounding impinging light, that may come from the sun, sky, ground and or even other cloud particles.

We need to consider the factors that determine the color of clouds and so allow us to compute realistic colors, as close to reality as possible.

An important phenomenon that illuminates both the atmosphere and the clouds is scattering. **Scattering** is the redirection of the incident light due to interactions with the molecules of the medium. The albedo for the clouds is close to 1, which implies that there is very little absorption of light. This leads to a high degree of re-emitting of received light by the cloud particles in the forward direction, commonly known as anisotropic scattering.

When the light reaches a point inside clouds, the light is scattered by the cloud particle and reaches the viewpoint. If the light is scattered only once before reaching the viewpoint, it is called single scattering component. However, inside the clouds, the light is scattered multiple times as shown in the figure. The multiple scattering component is also important for clouds. Furthermore, since the clouds are surrounded by the atmosphere, the light is also scattered and attenuated by the atmospheric particles. Such atmospheric effects are also important for the appearance of the clouds. Finally, the light behind the clouds also reaches the viewpoint through the clouds.

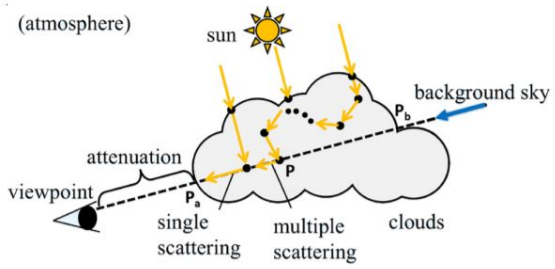

Figure 5: Important factors that affect the intensity of clouds before reaching the viewpoint

The nature of scattering effect on a particle is dependent on the particle size, the refractive index and the wavelength of the visible light.

Rayleigh scattering, named after the nineteenth-century British physicist Lord Rayleigh (John William Strutt), is responsible for giving the sky its blue color due to scattering by its tiny particles. The scattering phase function gives the angular distribution of light intensity scattered by a given particle for a given wavelength. The scattering equation combines these terms to capture the scattering behavior. Considering the case of single scattering as shown in Figure 5, the intensity of light reaching at P_a (Figure 5) is given by

In Figure 6,

- λ is the wavelength of the light;
- I_s is the intensity of sky light in the viewing direction;

$$I_a = I_s(\lambda) exp(-\tau(P_a P_b, \lambda)) + \int_{P_a}^{P_b} I_p(\lambda)\beta_\rho(l)F(\theta)exp(-\tau(PP_b, \lambda))dl$$

Figure 6: Light transport equation

- P_b is the other end of cloud falling in the view direction when connected to P_a;
- θ is the scattering angle;
- F_θ is the scattering phase function;
- τ is the optical depth obtained by integrating the attenuation coefficient (β_ρ); along the path
- gathered density ρ is a function of the path length.

In several researches in CG, the interaction of clouds with the sunlight and its environment is calculated using single scattering model into account. In such a model, the light is assumed to be scattered only once before it reaches the viewpoint. While single scattering assumption simplifies the computation, in reality the light inside a cloud is scattered multiple times.

Hence, a significant amount of research work has been dedicated to designing efficient rendering models and approximations to capture multiple scattering.

Many methods have been proposed for rendering clouds, considering the previously exposed physical phenomena.

One of the main objectives when rendering in real time computer graphics is to accelerate the computation involved in the rendering process. To meet this objective, real-time methods compute in graphics processing units (GPUs).

The preponderance of the methods proposed uses volumetric representation of clouds. Apart from this, several variations have been suggested for rendering, including the well-known ray casting and rasterization, that can take advantage of the mentioned inbuilt hardware capability of the GPUs for a achieving that more efficient rendering.

Ray Casting is a rendering technique used in computer graphics that is capable of creating a 3D perspective in a 2D map. Developed in the 60s, it is considered one of the most basic graphics-rendering algorithms.

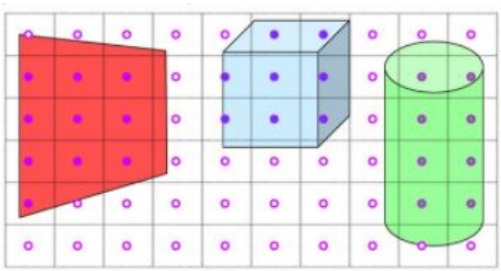

Figure 7: Representation of the uses of Ray Casting [12]

The main principle behind ray casting is that rays can be cast and traced in groups based on certain geometric constraints. In this technique, a ray from the pixel through the camera is obtained and the intersection of all objects in the picture is computed. Next, the pixel value from the closest intersection is obtained and is further set as the base for the projection. Ray casting is distinct from ray tracing, with ray casting being a rendering algorithm which would never recursively trace secondary rays, while ray tracing is capable of doing so. Ray casting is also simple to use compared to other rendering algorithms such as ray tracing.

Ray casting is fast, as only a single computation is needed for every vertical line of the screen. Compared to ray tracing, ray casting is faster, as it is limited by one or more geometric constraints. This is one of the reasons why ray casting was the most popular rendering tool in early 3D video games.

Billboards are texture-mapped polygons which always face the viewer. Dobashi et al. suggested in [13] a cellular automata simulation that, in order to render clouds from it, a continuous density field expressed as metaballs is constructed at each cell by taking a weighted interpolation with all its neighboring cells. Billboards are placed at the centers of metaballs, sorted based on their distance to the sun or viewpoint and projected onto the image plane.

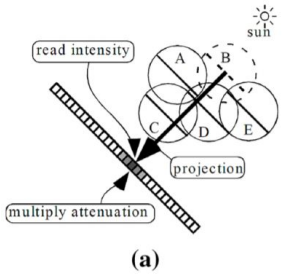

Figure 8: Cellular automata by Dobashi et al. [13]

The projection from the point of view of the light source provides light intensity reaching the clouds. At the same time, the projection from the camera achieves blending cloud densities with the remaining scene. In their work, Dobashi et al. account for single scattering of light and considered sunlight, transmitted color from the sky and attenuation from the cloud particles to compute illumination. By taking into account atmospheric scattering effects, this method can produce highly realist images, as shown in Figure 9.

The light shafts are also rendered efficiently by scattering the sunlight passing through the cloud gaps. This can be seen in Figure 10.

Notwithstanding the excellence of this study, it takes into consideration just the single scattering of light. Mark J. Harris and Anselmo Lastra proposed in 2001 a cloud shading algorithm for flight simulators that renders clouds realistically and partly considers the multiple scattering of light. The more expensive multiple forward scattering.

Figure 9: Efficient rendering of clouds by Dobashi et al. [13] method.

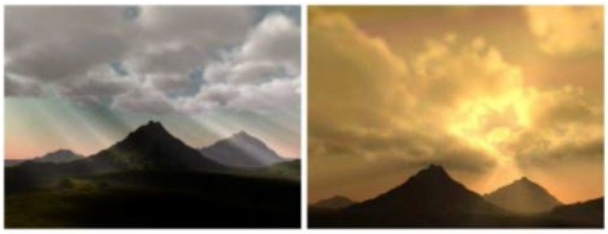

Figure 10: Efficient rendering of shafts of light by Dobashi et al. [13] method.

Their rendering is split into two stages. The multiple forward scattering (with more costs) is approximated in a preprocessing step, and the single scattering of light is done at runtime. The space is assumed to be filled with particles, and view-oriented textured polygons (impostors) are dynamically generated for efficient rendering.

Figure 11: Cumulus clouds considering multiple scattering of light (Harris and Lastra) [19]

In 2005, Szirmay-Kalos et al. [23] proposed a real-time rendering method for clouds with multiple scattering of light, based on the particle hierarchies, to render dynamically under changing light conditions. A collection of particles determined from a given direction are grouped together as a block, the image of which replaces the particles. In the next step, a depth impostor is generated for each block, following which the particle blocks are rendered one by one. During this step, separate volume–light interactions with the particles are stored in the textures. Finally, the blocks are sorted

Figure 12: Cloud illuminated by two dynamic directional light sources (left-up and bottom-right) by Szirmay-Kalos et al. [23]

for the viewing direction and rendered in back to front order in the rendering pass.

Bouthors et al. proposed in 2006 [7] a real-time method for rendering **stratiform** (layered) **clouds**. The clouds are rendered by account for single, double, triple and above levels of scattering in addition to transparency. The effects reproduced by their method are labeled in Figure 13, (a) diffuse reflectance, (b) glory, (c) fogbow, (d) pseudo-specular reflectance, (e) diffuse transmittance, (f) ground-clouds inter-reflection and (g) forward scattering.

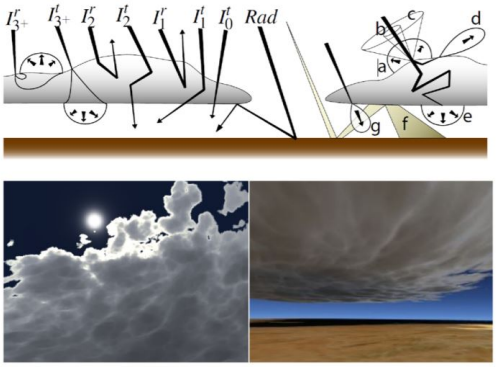

Figure 13: Real-time illumination of the stratiform clouds (Bouthors et al.) [7]

They also proposed an interactive rendering method for clouds with multiple anisotropic scattering, that approximates the light transport between an initial, receiving point on the flat surface and reaching at any point inside the volume using the concept of collector area.

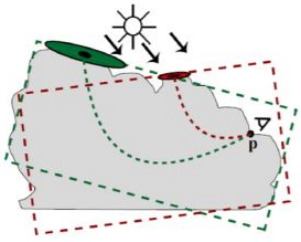

Figure 14: Collector area determined as a function in multiple anisotropic scattering by Bouthors et al. [7]

The collector area is the piece of the surface that receives a very large percentage of light from the source, which reaches the current point rendered. It is determined iteratively as a function of the rendered point and is inspired from the idea of the most probable paths. The method is implemented on the GPU using shaders with several orders of Mie scattering [4] (a solution to Maxwell's equations that describes the scattering of an electromagnetic plane wave by a homogeneous sphere and takes the form of an infinite series of spherical multipole partial waves).

Figure 15: Examples of rendered clouds with Bouthours et al. method for multiple anisotropic scattering rendering of clouds. [7]

4 ANIMATION

In several dynamic scenes in computer graphics, such as in movies or video games, it's not only necessary to model the clouds but also to animate them. Animated clouds may change their shape and color and vary many other factors too, resulting in fascinating and at least reasonably realistic liveliness clouds (which in most situations is sufficient and adequate).

Cloud animation, although useful and captivating, is a very elaborate job, since to animate the clouds themselves we need to consider the environmental characteristics that revolve around their formation.

In computer graphics, the desire is to create animations as simple and as fast as possible. There are two main types of methods in computer graphics to perform cloud animations.

One that is computationally inexpensive is **procedural animation**, based modeling the basic behavior of the physics field through noise, texture or obtaining cues from images, videos and other sources. This procedural approach gives the illusion of animating clouds without the direct use of underlying physics.

Geoffrey Y. Gardner [15] achieved animation of the modeled clouds in 1985 by varying the modeled mathematical texturing parameters in textured plane/ellipsoids with time.

Figure 16: Capture from a video [2] that shows the first clouds particle CGI animation by Gardner

Antoine Webanck also developed modeled (much later, in 2018 [14]) cloudscapes that are animated procedurally by morphing two selected models or density fields. An optimal transport function helps to establish correspondences between the best pair matches in the source and target cloudscapes. The animated primitives are created from the interpolation of the identified pairs such that they follow the shortest trajectory.

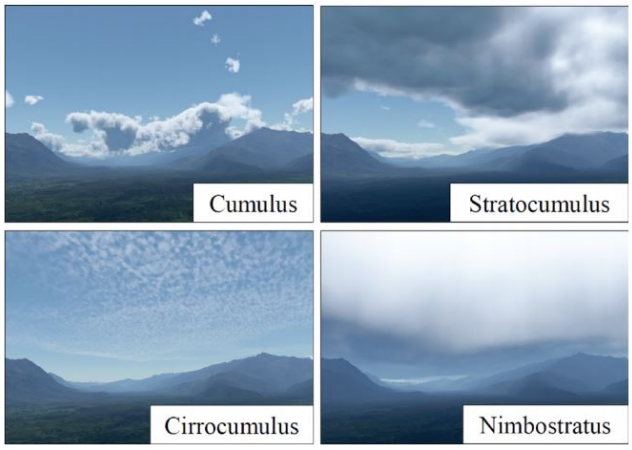

Figure 17: Clouds that are modeled and animated procedurally by Webank et al. [14]

Although these techniques can create realistic images of clouds, they are limited when realistic cloud motion is required. Another way possible to do this is a physics-based simulation, i.e., simulate process of the cloud formation.

The physical processes involved in cloud formation are illustrated in Figure 18. First, ascending air currents are generated due to the heat from the ground (that was previously heat by the sun). At a certain altitude, vapor in the air currents causes a **phase transition**, it transitions from water vapor to water droplets, and the cloud is generated (as shown in the right figure of Figure 18). This phase transition is very important to create realistic animation of clouds.

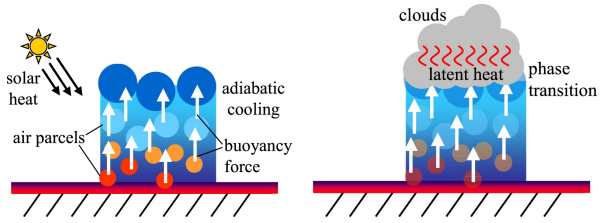

Figure 18: Overview of cloud formation process. [5]

In computer graphics there are many researches that use numerical **fluid analysis** for the visual simulation of clouds. Miyazaki et al. [26] used coupled map lattice (CML) to model and simulate various types of clouds based on the atmospheric fluid dynamics. CML is an extension of the cellular automata and uses a 3D **grid** and the famous Navier–Stokes equations [3] for simulation. The user specifies the type of cloud desired together with the initial and boundary conditions, grid resolution and other variables. All the variables are stored in the grid cells and are implemented through **grid-based operations**.

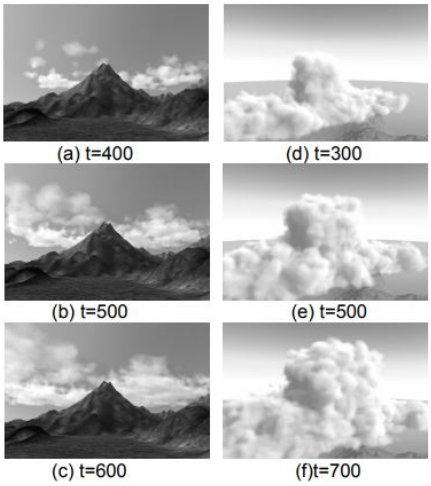

Figure 19: Examples of two different kinds of clouds (left one cumulus and right one cumulonimbus) development, as presented in the work of Miyazakiet al. [26]

Harris et al. [24] later implemented an interactive cloud simulation on the NVIDIA programmable graphics hardware, another work with a grid-based method (Figure 20). A 3D grid-based simulation is structured as layers of 2D textures for easy computation on the GPU, allowing gradual evolution of clouds in calm skies.

Besides grid-based, as the methods showed before, physical methods simulating clouds can also be **particle-based**.

An example of this is Goswami and Neyret methods. In their 2017 work [18], the atmosphere is represented implicitly through curves that provide temperature and humidity values as a function of the altitude. The cost of physics is significantly reduced by carrying out it at the macro-level, on a few large parcels on the CPU itself. The

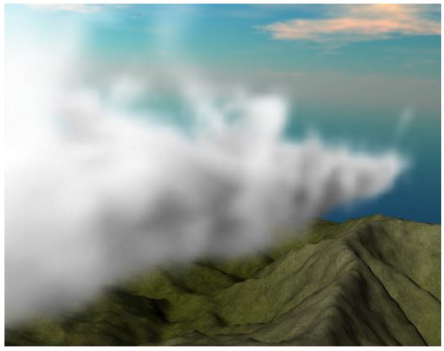

Figure 20: Frame of the GPU-based cloud simulation and rendering by Harris et al. [24]

parcel–parcel interaction is handled through smoothed particle hydrodynamics forces, and a drag force accounts for the friction with the surrounding atmosphere. The parcels also exchange mass with the implicit environment owing to the processes of entrainment and detrainment. The method also allows the user to experiment on the atmospheric profile and select dewpoint altitude, saturation ratio, etc.

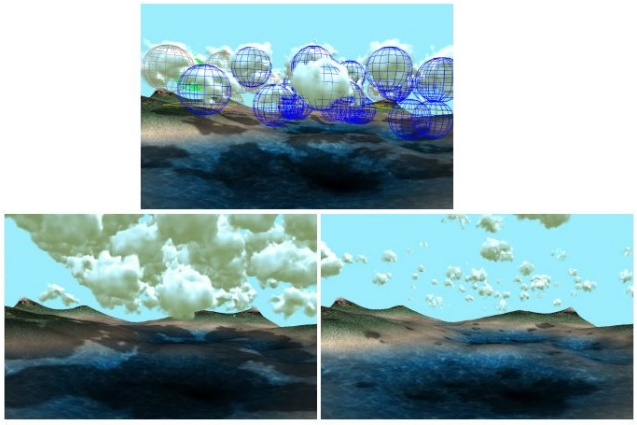

Figure 21: Cloud life cycle demonstrated with Goswami and Neyret method [18]

The original work is refined by Goswami two years ago [16] by incorporating a cloud map for governing cloud shapes and coverage over the landscape. The physics computation is limited to a few parcels within a unit octree. However, instead of generating hypertexture directly inside each of the parcels, the parcel physics attributes are projected onto a texture. In addition, a cloud map containing a precomputed texture of noise is employed. Both these textures are combined at runtime to generate the animated cloud cover through volume ray casting in the shader. The benefits of this approach are higher frame rates, more realistic cloud shapes and better span, especially toward the horizon.

Figure 22: Cumulus cloud animation using Goswami approach for the single-layered cumulus clouds, simulation together with the rendering running at around 200 fps (Goswami, [16])

5 CONCLUSIONS

Clouds are fascinating natural phenomena that always attract our attention, and for that I was intrigued by what would be the process to correctly simulate clouds in computer graphics.

In this essay I have reviewed and analised work on the area of modeling, rendering and animation of clouds in CG. I have come across lots of methods and was impressed by the various techniques that have been used.

Although it has become possible to synthesize realistic images/animations of clouds, there is still a lot of work that can be advanced on this research topic. For example, the resolution of synthetic clouds is not sufficient when compared to clouds in the real world. Efficient rendering/simulation of large-scale clouds covering really large areas should also be resolved.

In any case, I can conclude that it is a really captivating area of research in computer graphics that, whilst most of the time there is no need for complete accuracy in the realism of the simulated clouds, it is really interesting to observe all the progresses and the diversity of used techniques for such a specific topic.

6 BIBLIOGRAPHY

- Convincing Cloud Rendering: An Implementation of Real-Time Dynamic Volumetric Clouds in Frostbite, Rurik Hogfeldt [21]
- Physically Based Sky, Atmosphere and Cloud Rendering in Frostbite, Sébastien Hillaire [20]
- A simple, efficient method for realistic animation of clouds, Yoshinori Dobashia et al. [13]
- A survey of modeling, rendering and animation of clouds in computer graphics, Prashant Goswami [17]
- Visual simulation of clouds, Yoshinori Dobashia, Kei Iwasakib, Yonghao Yuec, Tomoyuki Nishita [28]
- Texturing & Modeling: A Procedural Approach, David S. Ebert et al. [10]
- Volumetric modeling with implicit functions: A cloud is born, David S. Ebert.[11]

REFERENCES

[1] [n.d.]. . https://www.techopedia.com/definition/21614/ray-casting
[2] [n.d.]. *Gardner ellipsoids sumulation.* https://youtu.be/5MPZVPCM9ZU
[3] [n.d.]. *Navier Strokes Equations.* https://www.grc.nasa.gov/www/k-12/airplane/nseqs.html
[4] [n.d.]. *Scattering Techniques.* https://www.lavision.de/en/techniques/mie-rayleigh-raman/
[5] [n.d.]. *Science Direct.* https://sciencedirectjournal.com/
[6] [n.d.]. *Teahub.* https://www.teahub.io/viewwp/whxhwR_px-cloud-computer-wallpapers-bsnscb-graphics/ Header do documento.
[7] Antoine Bouthors, Fabrice Neyret, and Sylvain Lefebvre. 2006. *Real-time realistic illumination and shading of stratiform clouds.* http://www-evasion.imag.fr/Publications/2006/BNL06
[8] Shiyu Hao Yue Qi Qinping Zhao Chunqiang Yuan, Xiaohui Liang. 2014. *Modelling Cumulus Cloud Shape from a Single Image.*
[9] Richard Earl Parent D. S. Ebert. 1990. *Rendering and animation of gaseous phenomena by combining fast volume and scanline A-buffer techniques.*
[10] Darwyn Peachey Ken Perlin John C. Hart Steven Worley David S. Ebert, F. Kenton Musgrave. [n.d.]. *Texturing Modeling: A Procedural Approach.*
[11] David S. Ebert. 1997. *Volumetric modeling with implicit functions: A cloud is born.*
[12] Aaron Knoll et al. 2019. *Interactive animation of single-layer cumulus clouds using cloud map.*
[13] Yoshinori Dobashia et al. 2000. *A simple, efficient method for realistic animation of clouds.*
[14] A. Webanck Y. Cortial E. Guérin E. Galin. 2018. *Procedural Cloudscapes.*
[15] Geoffrey Y. Gardner. 1985. *Visual simulation of clouds.*
[16] Prashant Goswami. 2019. *Interactive animation of single-layer cumulus clouds using cloud map.*
[17] Prashant Goswami. 2020. *A survey of modeling, rendering and animation of clouds in computer graphics.*
[18] Prashant Goswami and Fabrice Neyret. 2017. *Real-time landscape-size convective clouds simulation and rendering.* https://diglib.eg.org/bitstream/handle/10.2312/vriphys20171078/001-008.pdf
[19] M.J. Harris and Anselmo Lastra. 2001. *Real-Time Cloud Rendering.*
[20] Sébastien Hillaire. [n.d.]. *Physically Based Sky, Atmosphere and Cloud Rendering in Frostbite.* https://media.contentapi.ea.com/content/dam/eacom/frostbite/files/s2016-pbs-frostbite-sky-clouds-new.pdf
[21] Rurik Hogfeldt. 2016. *Convincing Cloud Rendering: An Implementation of Real-Time Dynamic Volumetric Clouds in Frostbite.* https://odr.chalmers.se/bitstream/20.500.12380/241770/1/241770.pdf
[22] David S. Ebert Joshua Schpok, Joseph Simons and Charles Hansen. 2003. *A Real-Time Cloud Modeling, Rendering, and Animation System.*
[23] Mateu Sbert László Szirmay-Kalos and Tamás Umenhoffer. 2005. *Real-Time Multiple Scattering in Participating Media with Illumination Networks.*
[24] Thorsten Scheuermann Anselmo Lastra Mark J. Harris, William V. Baxter III. 2003. *Simulation of Cloud Dynamics on Graphics Hardware.*
[25] Daisuke Matsuoka. 2009. *Extraction, classification and visualization of 3-dimensional clouds simulated by cloud-resolving atmospheric model.*
[26] Y. Dobashi Ryo Miyazaki, S. Yoshida. 2001. *A method for modeling clouds based on atmospheric fluid dynamics.*
[27] Georgios Sakas. 1993. *Modeling and animating turbulent gaseous phenomena using spectral synthesis.*
[28] Yonghao Yuec Tomoyuki Nishita Yoshinori Dobashia, Kei Iwasakib. 2017. *Visual simulation of clouds.*