La Trobe University
Bundoora, Victoria, Australia
Faculty of Science, Technology & Engineering
Department of Computer Science and Computer Engineering

Ray tracing clouds using a stochastic photon-particle duality

Author:	Joseph Hura
Student No:	99527505
Course:	B. Comp. Sci. (Hons)/ B. Elec. Eng.
Submission Date:	October 2006
Supervisor:	Dr. Richard Hall

Abstract

Particle systems approaches to modelling clouds currently decouple light particles from cloud particles. Consequently, multiple scattering radiance needs to be calculated, irrespective of whether a particle receives any light. In our approach, we generate random photon-cloud particles (a pseudo-duality) located at the scattering positions of photons within a volume of interest. Thus, the inscattered radiance is directly related to the density of photons per particle, à la photon mapping. Consequently, unnecessary radiance calculations are eliminated during Monte Carlo rendering, as we only include areas where multiple scattering events occur. We rendered different types of cloud formations and compared them to qualities found in photographs of real clouds. We evaluated our model based on memory usage and computational cost. Applications could include photorealistic rendering of outdoor scenes for computer generated movies using high-performance computing.

Publications

"Design of a simulation of atmospheric sunbeams," WSEAS Transactions on Computers, vol. 5, no. 10, pp. 2466-2471, 2006. *(See Appendix 5.1)*

Acknowledgments

The journey to the clouds has finally reached a plateau, and I am glad there were many people supporting me as I approached this milestone in my academic life. I would like to thank:

The Department of Computer Science and Computer Engineering at LaTrobe University, for the support and opportunity given to me to write this thesis. They have also provided me with support as I made my first international publication, and for this I am most grateful. I would also like to make a special mention to the technical staff, who have made my life easier, as I made it harder for them.

My supervisor, Dr. Richard Hall. You have helped me push past my limits and occasionaly made me laugh. Thank-you for your guidance, honesty, understanding and above all your patience. It is an honour to have collaborated with you on a publication, and to have shared ideas throughout the year. Your teachings will last forever.

The following academics from outside this university who generously gave up their time to answer my emails: Professor Henrik Wann Jensen at UC, San Diego - your emails and book have been an immense help. Professor Fredo Durand at MIT - for your explanations and ideas on tackling aspects of this problem. Professor Nelson Max at LLNL - for clarifying some finer points on optical models. Dr. Stephen Cheeney at the University of Wisconsin - for suggesting readings. Matt Pharr at Neoptica - for suggesting

readings. Michael Nielsen at the University of Aarhus - for making available ray marching notes from his course.

Simon Flannery for his guidance in all things graphics. Michael Robinson and Hemang Rathod, who have pointed out sleep deprived mistakes on more than one occasion.

Alicia Grech, for coffee, clarity, LaTeX wizardry, encouragement and proof reading this thesis.

My parents, for buying an Amstrad in 1987 that started this entire journey.

My family, especially my mother, for being there for me all year.

My friends, for understanding, not understanding, and helping me have a laugh at all the right times.

Finally, Maria Spinella, for her patience as I pursued my perfect pixels.

Contents

List of Figures

List of Tables

Glossary

2D	Two Dimensional
3D	Three Dimensional
CG	Computer Graphics
DS	Diffuse Surface (Lambertian)
LOD	Level Of Detail
LTE	Light Transport Equation
LTM	Light Transport Method
PDE	Partial Differential Equation
VRE	Volume Rendering Equation
VR	Volume Region

Nomenclature

σ_a	Absorption coefficient
σ_s	Scattering coefficient
σ_t	Extinction coefficient
Λ	Single scattering albedo
$\rho(x)$	Density of a medium
τ	Optical depth of a medium
T	Transparency of a medium
α	Opacity of a medium
s	Start of a line segment
s'	End of a line segment
$\vec{\omega}$	Incoming directional vector
$\vec{\omega}'$	Outgoing directional vector
$\vec{x}$	Three dimensional point in space
Φ	Radiant flux
ξ	Canonical uniform random variable
$L(x, \vec{\omega})$	Irradiance at a point in space
$P(x, \vec{\omega}', \vec{\omega})$	Phase function

1 Introduction

Clouds are an atmospheric body (that exist at one of several altitudes) consisting of particles of aerosols and associated water droplets [1, 2, 3, 4]. When light particles (photons) intersect with cloud particles, the salient effect is multiple scattering due to the high albedo nature of clouds [5, 6, 7, 8, 2]. Computer graphics (CG) researchers have made significant advances in modelling pseudo-realistic clouds [9, 10, 11, 12] and their interactions with light [13, 14, 15, 16]. Their cloud models can be categorised as surface or spatial, with many hybrids also amongst past works. The modelling of light and cloud particles is generally achieved using radiosity [17, 18] which represents interactions between light and diffuse surfaces, and is derived from physics models of radiative heat transfer [19, 20, 5].

Cloud and light modelling in computer graphics presents an interesting fusion of mathematics, physics and computer visualisation. The first basic cloud visualisations in computer graphics [9] have evolved into a many forms of models that are constantly improved. One facet of these models is the term "photo-realism" which can be thought of as the resulting image evoking a similar visual reponse to that evoked by a photograph of the same scene [21, 22]. Not all computer graphics models of clouds and the interactions they make with light, truly represent the physical nature of the interaction. In this thesis, we investigate what physical aspects of these two phenomena we need to represent in order to visualise a physically-faithful photo-realistic cloud model. As such, it is important for us to understand the basic physical properties that govern these interactions, before reviewing previous research in this field.

The organisation of this chapter provides an introduction into the physical nature of clouds, and the interactions and effects that they exhibit due to light passing through them. It introduces terms used in physics and CG research that describe the physical interactions between cloud and light particles, with respect to absorption and scattering coefficients, optical density, scattering phenomena and cloud albedo. We then pose several research questions that

relate directly to the fundamental nature of cloud and light interactions. An overview of the organisation of this thesis concludes this chapter.

1.1 Clouds

Clouds form from the condensation of rising air parcels. As air parcels ascend into areas of lower pressure, they expand and cool, using their internal potential heat to supply the energy for the expansion process. Descending air parcels will encounter increasing air pressure and will experience compressional warming. Both of these processes occur at an independent, self-contained rate, termed *adiabatic*. This process leads to condensation, the formation of water droplets. The collection of water droplets at differing altitudes combined with the ascent of air parcels leads to cloud formation. Formation of water droplets in clouds could not exist without atmospheric aerosols [1, 2].

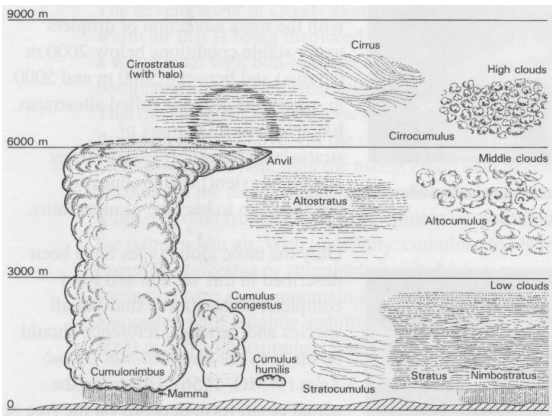


Figure 1: Principal cloud forms and main sub-types [23].

Due to differing altitudes, clouds are categorised into three broad categories:

1. *Stratiform clouds*, which develop from warm moist air that overrides cold heavier air along a warm front.

2. *Cumuliform clouds*, which develop through heat transfer from the surface and latent heat released to the air during condensation.

3. *Cirriform clouds*, which develop from the two forms of lifting stated above. They are generally found at high altitudes.

Each category contains many sub-types as shown in Figure 1, however the formation of water droplets that form clouds could not exist without atmospheric aerosols.

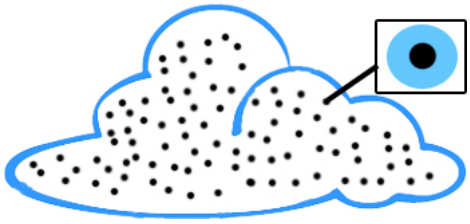


Figure 2: Hygroscopic aerosols act as cloud condensation nuclei which help water droplets form around them.

Atmospheric aerosols consist of many small solid and liquid particles that are either soluble in water, *hygroscopic*, *wettable* but insoluble, or water resistant, *hydrophobic* [1]. The formation of water droplets on aerosols is based on *Kelvin's formula* and the *Gibbs' free energy function*.[1] Clouds consisting of many particles (aerosol and associated water droplets) are analogous to CG particle systems (see section 2), whereby each cloud particle exhibits similar characteristics of radius, density, and life-cycle. The interaction of light with cloud particles is described by Mie Theory, discussed in the next section.

1.2 Interactions of Light with Clouds

The path taken by a light particle (*photon*) in the atmosphere and the interaction it makes with cloud particles is important in the calculation of its energy, Φ. All photons that interact with a medium are either emitted, absorbed or scattered (Figure 3), whilst those that don't interact are *transmitted* [5, 20]. Media that scatters or absorbs light is called *participating media*. In the case of photons interacting with clouds, the dominant interactions are absorption and scattering.

[1]The equations are beyond the scope of this thesis. Further reading refer to [1, 2, 4, 3].

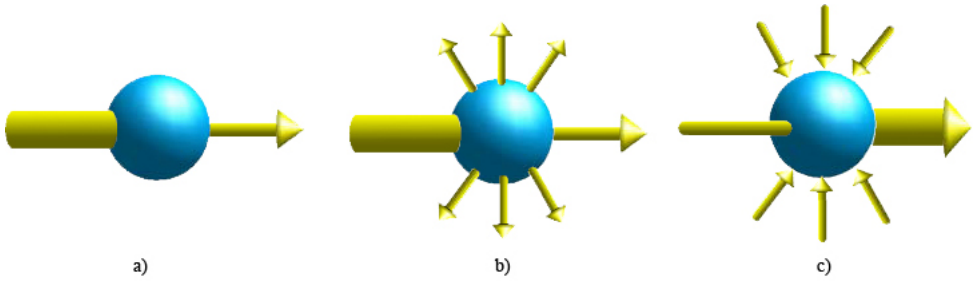


Figure 3: Phenomena affecting light propagation [24]. a) Absorption. b) Out-Scattering. c) In-Scattering.

Absorption occurs when the photon intensity is transformed into other forms of energy upon interacting with a medium (Figure 3b). In contrast, *scattering* is when a photon collides with a medium and the energy is distributed in such a way that the direction of the photon changes. Absorption and scattering attenuate the intensity of a photon as it passes through a medium. The total attenuation of a photon is described by *extinction*. The extinction coefficient of a medium describes the amount of attenuation a photon will have when it interacts with a particle [25]. The extinction coefficient is:

$$\sigma_t = \sigma_a + \sigma_s \tag{1}$$

where σ_a and σ_s are the absorption and scattering coefficients respectively. The coefficients may be interpreted as the probability of absorption or scattering per unit length, travelled by a photon in the medium [5]. Coefficients are used to describe the optical properties of a given medium because, calculating the absorption and scattering of each particle in the medium would be impractical and computationally costly. If the medium is homogenous, its scattering and absorption coefficients are constant [26].

Further, $\sigma_a = \sigma_a\rho(x)$ and $\sigma_s = \sigma_s\rho(x)$, where $\rho(x)$ is the density of the medium, while σ_a and σ_s are the average absorption and scattering coefficients respectively. *Single scattering* is the scattering of light by a single particle in a medium, and occurs in media that are very thin or transparent. Media such as clear air are termed *optically thin*. Unlike clear air, clouds have an *optically thick* density consisting of many aerosol particles, which leads to

dominant multiple scattering within the media. *Multiple scattering* is scattering of light from many particles in succession. Physical approximations for single scattering use Rayleigh scattering [5], while multiple scattering use Mie Theory [8].

Scattering may be further subdivided into two cases. The first, is light energy lost on the path of a ray in the media, which is called *out-scattering* (Figure 3c). The second, is the light energy that has been scattered by the media in the direction of observation, this is called *in-scattering* (Figure 3d).

The scattering albedo $\Lambda = \frac{\sigma_s}{\sigma_t}$, represents the fraction of light lost from scattering, while $(1-\Lambda)$ represents the remaining fraction which has transformed into other forms of energy [5, 20]. Perfect absorption occurs when $\Lambda = 0$, while perfect scattering is when $\Lambda = 1$.

The albedo of clouds is insensitive to cloud height and very close to unity ($\Lambda \approx 1$), as absorption by water droplets is negligible in the visible spectrum [1, 2]. Nearly all light that enters a cloud exits, but only after multiple scattering events. The dark areas in clouds are caused by the out-scattering of light rather than absorption, where the intensity of light leaving the cloud is attenuated. *Airlight* is responsible for cloud shadows being visible in the air. Airlight is the diffuse scattered sunlight from air molecules, to differentiate it from light that is radiated from the Sun at ground level, sunlight [27].

The interaction of light with clouds is significant when creating images of photo-realistic quality in CG. Characteristic physical descriptions of cloud/light interactions are a basis for realistic image cloud synthesis in CG. Realism is also captured by shadows in synthesised images, thus, we discuss the interactions of shadows with clouds.

1.3 Interactions of Shadows with Clouds

A shadow is a region of space not directly illuminated when a cloud occludes the Sun. Shadows are important as they provide useful cues for the shape and depth of objects in the natural world; hence they are inherent in the perception of realism. The *umbra* is the portion of shadow which receives no source

light, whereas the *penumbra* is the portion of shadow which only partially blocks the source light. Umbrae are made visible by airlight illuminating the volume in shadow.

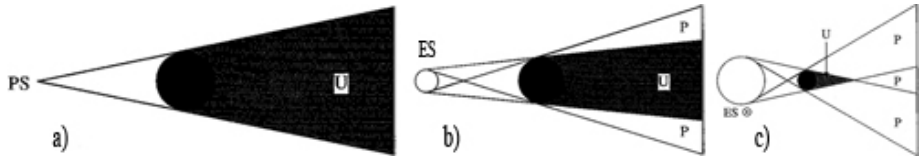


Figure 4: Geometry of Shadows [27]. a) A point source, PS, casts only an umbra U, which is totally hidden from the point source. b) An extended source, ES, like the Sun, casts both an umbra U and a penumbra P. c) When the light source like the Sun $\odot$, is physically larger than the opaque object, the umbra has a finite length. When it is smaller, the umbra extends without limit behind the opaque object as in b).

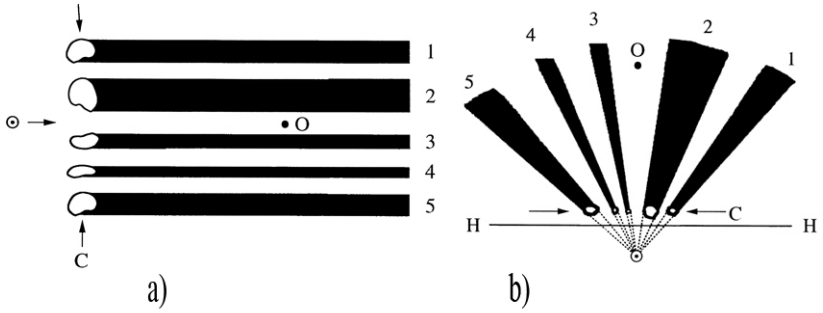


Figure 5: Optics of Sunbeams [27]. a) Looking down on the Observer, O, from the zenith. Clouds, C, cast long straight parallel shadows. b) When viewed in perspective, cloud shadows seem to diverge away from the Sun, $\odot$.

Two kinds of light sources can create shadows, *point sources* and *extended sources*. A point source is always smaller than the object it illuminates, and the emitted rays of light diverge from a single point. Shadows cast by a point source only have an umbra (Figure 4a). Extended sources cast two part shadows consisting of an *umbra* and *penumbra* (Figure 4b,c). Shadows cast by an extended source, such as the Sun, emit rays of light that are near-parallel to each other; they emanate from the entire solar disc.

Cloud shadows are illuminated by airlight, while sunlight permeates between the gaps in clouds, creating a contrast between light and dark volumes in

the sky, resulting in sunbeams. Sunbeams, when viewed by an observer on the ground seem to converge at the solar point, which is due to the optical process of perspective (see Appendix 5.1) [27].

Creating realistic and visually appealing images in CG requires shadows [28]. Shadows provide useful information about the depth of objects and the position of lighting in the real world. By understanding how clouds cast shadows, we are able to better represent them in a synthesised image.

1.4 Research Questions

Having gained a basic understanding of the physical properties of cloud/light interactions, we are able to identify areas of research that will aid in the photo-realistic synthesis of clouds.

1. *Can we develop a physically accurate cloud model that couples the interaction between cloud and light particles?*
2. *Can this model avoid evaluating multiple scattering events for cloud particles that don't receive any light?*
3. *Can this model be used to simulate sunbeams?*

1.5 Organisation

This thesis is organised in the following manner:

In Chapter 2, we review CG research in the areas of cloud and light modelling. We review previous cloud modelling research in terms of its capacity to represent clouds with respect to physical cloud properties (Section 1.1). We review lighting models used to represent the interaction between light and cloud particles, with particular attention to models that adhere to the physical description in section 1.2. We also review shadow techniques used in CG that would add realism to our cloud model, and assist in creating effects

such as sunbeams. We conclude with our motivations for pursuing our design implementation.

In Chapter 3, we detail the fundamental mathematical framework that forms the basis of our cloud/light particle model. We also discuss the light transport method used as part of our model, detailing its methodology and sampling strategies. We conclude the chapter by specifying our design assumptions, defining our implementation and algorithms.

In Chapter 4, we evaluate our design and implementation. We focus on the the computational cost associated with our model in terms of rendering times and memory usage. We present our result of a photo-realistic rendering of a cumulus cloud.

In Chapter 5, we discuss the findings from our results and highlight the significance and contribution of our research. We conclude the chapter by discussing possible applications for our model and highlighting future work.

2 Clouds and Light: Computer Graphics Models

Based on the foundations laid in Chapter 1, we can investigate how clouds, light and shadow interactions with clouds have been modelled by computer graphics (CG) researchers.

Cloud modelling in CG can be formalised into two main categories, *surface* and *spatial* models. Surface models wallpaper a two dimensional (2D) cloud skin onto geometric shapes to produce the appearance of clouds, rather than representing the actual shape of the cloud. In contrast, spatial models attempt to represent the three dimensional (3D) volume of clouds by partitioning space into many objects. These volumes may have rigid or deformable boundaries [29]. To represent clouds with any real physical basis, we must consider models that simulate light transport through participatory media.

Light transport through participatory media commonly uses a radiosity approach to rendering light and cloud particle interactions. Radiosity is derived directly from physics models of radiative heat transfer and models the interaction of light between diffuse surfaces [19]. Sharing similarities with the rendering equation [18], the light transport equation (LTE) is presented here briefly, in order to grasp what the light modelling methods in CG aim to solve. It is described in detail in Chapter 3.

The LTE states, that for any position $\vec{x}$ and direction $\vec{\omega}$ in space, the change in outgoing light intensity $L(x, \vec{\omega})$, is equal to the sum of all incoming light from all directions, minus any light which is absorbed.

$$\frac{dL(\vec{x},\vec{\omega})}{ds} = -\sigma_t(\vec{x})L(\vec{x},\vec{\omega}) + \sigma_s(\vec{x}) \int_{4\pi} P(\vec{x},\vec{\omega},\vec{\omega}')L(\vec{x},\vec{\omega}')d\vec{\omega}'$$

The light transport methods (LTMs) used in CG to solve the LTE can be categorised into irregular or regular techniques. Irregular LTMs have a non-uniform selection process for sampling interacting particles in the cloud, whereas regular LTMs have a uniform selection process. Further, these methods may be termed *homogeneous* or *heterogeneous* in reference to the sample

space per particle. Heterogeneous methods vary the size of the sample space, while homogeneous methods keep the sample space constant. These lighting models are used in generating photo-realistic images, however, shadows also play an important role in the perception realism.

Shadows in CG applications may be categorised as either *hard* or *soft* [30, 31]. These methods correspond to the visualisation of the umbra and penumbra of the shadow. Hard shadows display only the umbra (see Figure 4a) of an object. Hard shadows are determined by a binary approach, whereby a point in a scene lies in shadow of an object or not. This binary approach can be seen as either multiplying the light intensity by a 0 or 1 depending if the object lies in shadow or not. Hard shadows are generally rendered using the ray tracing method [32]. Soft shadows differ from hard shadows by including the penumbra (see Figure 4b) in image synthesis, so the binary approach of evaluation can not be applied. Soft shadow evaluation in its simplicity can be evaluated as the multiplication of the light intensity by a fraction in the range $[0, 1]$, where 0 indicates the umbra, 1 indicates no shadow, and all other values indicate the penumbra [33]. The resultant shadow region has a shape that is dependant on both the occluding object and the light source [28]. By their nature, the LTMs investigated describe the light intensity in the range $[0, 1]$, thus eliminating the need to specifically target shadow generation.

We organise this chapter by presenting cloud and light models used in CG. We first discuss cloud models, by categorising them into spatial and surface clouds. We then present LTMs categorised by irregular light and regular light approaches. We conclude this chapter by summarising our findings in CG cloud and light modelling and highlight possible models that would be appropriate for our research questions.

2.1 Spatial Clouds

In CG research, three approaches are currently used to represent clouds in a 3D volume. These methods are voxel grids, particle systems and metaballs.

Voxel grids are a method that is intuitive for modelling cloud densities. A voxel is the 3D analogy of a 2D pixel, and is a portmanteau of the words volumetric and pixel. It defines a six-sided cube in a grid subsection within a volume, and acts as a discrete sample distributed in 3D space (see Figure 6). Voxel grids are commonly used when physically-based simulations are involved [12, 34, 35], with volume rendering receiving much focus as a result [36, 37, 38]. Voxel grids have been used to store results from cloud simulation algorithms based on partial differential equations (PDEs).

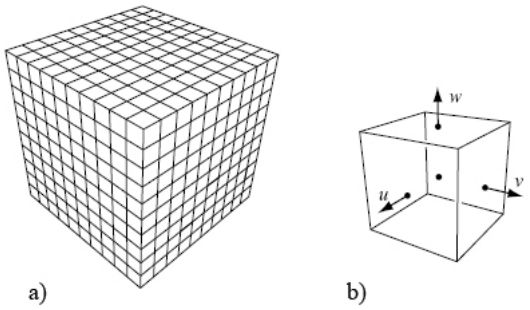


Figure 6: A Voxel Grid [39]. a) A discretised volume domain: a voxel grid. b) A volume element from the grid.

Clouds were first modelled with voxel grids to store solutions from nine PDEs, which described a discretised local volume of cloud [12]. These PDEs were based on the Navier-Stokes equations for simulating fluids, and also included an equation to model the phase transition of water to vapour. A stable model of the Navier-Stokes equations [34] were used for cloud simulations with the resultant data stored on voxel grids [40]. Realistic cumulus and stratus clouds were produced with this method, however little detail about the implementation was discussed. The Navier-Stokes equations were also used to model clouds on a staggered voxel grid [41], discretised for the velocity and pressure components of the PDEs. The pressure, temperature and water content were sampled from the center of the voxels, while the velocity was defined on the faces (see Figure 6b) [39, 42].

Voxel grids may be rendered using either forward or backward ray marching methods, or traditional ray tracing [12]. Advantages to using voxel grids for

density distributions is that physically accurate data can be stored, producing realistic clouds when rendered. Disadvantages are in the computational cost during the rendering phase, as each cell has a complexity of $O(n^3)$; for a grid of $N_1 \times N_2 \times N_3$ the complexity is $O(n^6)$.

Particle systems intuitively model clouds, as particles best fit the model of aerosols within clouds. Particle systems model objects as clouds of primitive particles that define a volume. A particle represents a point in 3D space, with individual attributes that govern its size and color. Particles can exist in a lifecycle, viz. particles are born and die. The model was introduced to model fuzzy objects like clouds [43] and expanded to approximate shading, cater for self-shadowing, external shadows and external light sources [44].

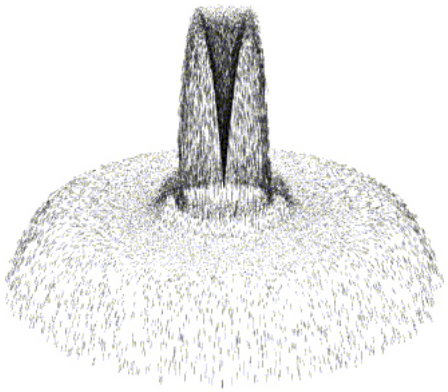


Figure 7: A particle system [45].

Particle systems have been used to model clouds by filling space with particles of varying size and density [46]. Each particle in this method has a center, radius, density, and color. By varying the particle properties in the volume, good approximations of realistic clouds have been attained. Each particle is rendered by blending it into a frame buffer, with the transparency of a particle governing the final pixel color.

Particles have been described as the simplest surface representation in CG, enabling more primitives and complex images to be processed than polygons, in the same computation time [43]. Depending on particle properties, the particles can be rendered using ray tracing methods. Advantages of producing clouds with particles is in their inherent representation of volumes;

less storage is required than an equivalently detailed cloud with other models. Disadvantages occur when high cloud detail is required, as the number of particles increases, so too does the computational cost.

Metaballs have been used to control the shape of cloud models in CG. Metaballs are also known as *'blobs'* and *'soft objects'*. Metaballs represent volumes of scalar field functions that are additive. Each metaball has a radius of influence, so when the surfaces of neighbouring metaballs intersect, the effect of each point on each surface is calculated, and the metaballs deform accordingly [47], turning into 'blobs' (see Figure 8). Clouds can be described as big 'blobs'. When clouds collide, they grow bigger due to coalescence of water droplets [2], intuitively, they draw similarities to metaballs. Metaball implementations for cloud models usually define each ball with a center, radius and density at the center of the ball [48, 11, 49].

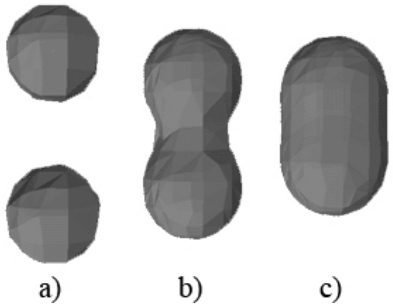


Figure 8: Deforming Metaballs [50]. a) Two metaballs. b) Deformation of points on each surface. c) The metaballs have superimposed to create one metaball.

A naïve implementation for clouds used metaballs to form the basic shape of a cloud [48]. Smaller metaballs were then generated recursively by using a fractal method [51] to form the fringe of the cloud. An image driven approach to model clouds from satellite images using density volumes at each ball center is another technique [11]. Wyvill's field function [52] defined the density function, and optimisations made to the radius and density of each ball, reduced error between the satellite and synthesized images.

Metaballs can be rendered a number of ways including ray-tracing, splatting [53] or using the marching cubes algorithm [54]. Advantages to using metaballs is their inherent ability to model the macrostructure of clouds,

as their surfaces are smooth and malleable. Disadvantages lie in rendering, where techniques to discretise the volume are computationally high for the entire volume. Complexity for the metaball arises from the calculation of the density function when interactions occur between points on a surface, as well as the constant tracking of the field of influence.

2.2 Surface Clouds

Texture mapping on the surface of ellipsoids and polygons has been used to model the visual aesthetic of clouds. Texture mapping is a technique used to paste an image onto a geometrical object like a sphere, ellipsoid or polygon (see Figure 9). Using textures reduces the computational cost needed for rendering clouds compared to methods already presented.

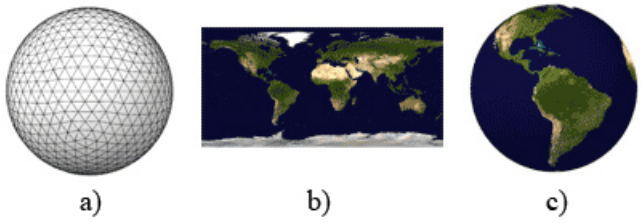


Figure 9: Texture Mapping [55]. a) Sphere with no texture. b) Texture image. c) Sphere with texture.

The visual appearance of clouds was modelled with a naïve implementation focusing on aesthetic results, using a texturing function on the surface of ellipsoids [9]. The texture function represented the spectral content of the texture pattern, similar to fractal surfaces. 2D clouds were placed as textures on a planar surface, while 3D clouds were generated with textures mapped onto the surface of ellipsoids. For added realism, the 3D ellipsoids were able to link, creating more complex cloud shapes, reminiscent of metaballs.

Texture maps are rendered using the ray tracing method. Advantages of using texture maps for cloud modelling, is the ability to economically visualise the simulation. Disadvantages lie in the inability to produce physically based cloud models. The complexity of texture maps lie in the mathematical tex-

turing function, if it is overly complex, the advantage of using texture maps to economically visualise simulations is lost.

Procedural noise functions are able to model the wavy, soft appearance of clouds. Procedural noise techniques create realistic textures with algorithms that produce graphical representations of noise (see Figure 10) and turbulence. Noise acts as a narrow bandpass filter that has statistical invariance under rotation and translation (see Figure 10). Turbulence approximates visual appearance of turbulent flow by, approximating the magnitude of deformation from swirling around isosurfaces of the noise domain [56].

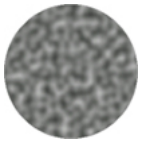


Figure 10: Perlin Noise [57].

Clouds have been successfully rendered using procedural noise techniques to fill volume densities [22, 58]. Turbulence was used in conjunction with a color spline to produce soft looking clouds. Clouds were modelled using turbulence to generate random, but continuous density data to fill the cloud volume [56].

Procedural noise can be rendered using standard ray tracing techniques, or by using Z-buffer or A-buffer scanline techniques [59, 60]. Advantages of a procedural noise cloud model is the ability to use nonlinear functional composition to model the stochastic aspects of clouds, as well as the flexibility of modifying the cloud appearance. Disadvantages are in the computational cost of applying noise and turbulence to large datasets, the computation for each point being $O(N^2)$.

2.3 Irregular Light

Irregular LTMs can be divided into two approaches, stochastic or deterministic. Stochastic approaches use a random sampling to determine solutions to the LTM. Deterministic approaches use mathematical functions that have

specific solutions for defined inputs. Monte Carlo integration is an irregular homogeneous stochastic approach to solving the LTM, whereas spherical harmonics are an irregular homogeneous deterministic approach.

Monte Carlo integration is a stochastic method used to solve integral equations. Monte Carlo methods randomly sample the integral domain, alleviating the computational burden of solving the light transport equation in its entirety. Solutions can be found by intelligent sample choices, termed *importance sampling*, with the specific method depending on the problem being solved [61]. In CG, this method is known as Monte Carlo ray tracing and applying it to multiple scattering within clouds reduces the need for complex computation. Images generated from Monte Carlo ray tracing techniques can appear noisy if not enough samples are chosen to converge on the solution.

One approach which made no restrictive assumptions about the characteristics of objects or physical phenomena in the rendering process, incorporated a unique phase function for its importance sampling [62]. The Schlick phase function was used to generate the new direction of a ray after scattering, ensuring that only the most significant scattering events determined light intensity.

Similar to pure Monte Carlo ray tracing, *photon mapping* [63] has been shown to significantly improve the efficiency of solving the LTE by using kernel density estimation [26]. This technique also introduces very little noise and aliasing, while maintaining the flexibility of Monte Carlo ray tracing.

Spherical harmonics intuitively model cloud particles, as water droplets can be approximated to being spheres [2, 3, 1]. Spherical harmonics $Y_l^m(\theta, \phi)$ are the angular portion of the solution to Laplace's equation in spherical coordinates [64]. Spherical harmonics form an orthonormal basis of the functions on the unit sphere [65], meaning that a function $f(\theta, \phi)$ may be represented by an infinite series expansion:

$$f(\theta, \phi) = \sum_{l=0}^{\infty} \sum_{m=-l}^{l} P_l^m Y_l^m(\theta, \phi) \tag{2}$$

Multiple scattering in volume densities has been solved using spherical harmonics [12]. The rendering phase introduced a two-pass method for calculating light scattering in volumetric media. The first pass required the scattering and absorption to be calculated along each ray and through each cloud, storing the result on voxels. The second pass traced rays from the eye and through the voxels, where the scattering of light to the eye was calculated. The two-pass method has become quite common as a result of this work [62, 26, 66, 67]. The spherical harmonic approximation yielded homogeneous matrices of PDEs which were solved by relaxation, however, only the first few spherical harmonics were used. The method was criticised by some [68, 65], as results did not accurately prove that the PDEs were solved for isotropic scattering, "since all the pictures were produced by the simpler method" [68].

Spherical harmonics have become known as the P_N-method in transport theory literature [65, 69], where N is the degree of the highest harmonic in the expansion. Further research into this method observed that the general N-term expression led to a diffusion type equation, because results were truncated to the P_1 expansion [65].

The *diffusion approximation* is valid when scattering events are frequent, as in clouds [65]. If photons are travelling in random directions at any point in the medium, the multiple scattering events become apparent and the light is said to be *diffuse*. The diffusion approximation represented scattering media on a voxel grid and described two methods to solve for intensity. In the first method, the diffusion approximation was discretised to generate a system of equations which were then solved using the multi-grid method [70]. The second method was a finite element method, where a Gaussian basis function was used as a distance kernel.

2.4 Regular Light

Regular LTMs are generally discretised into volume elements, with each voxel representing the sample space for a light intensity at that point in space.

The *discrete ordinates* method uses a collection of M discrete direction *bins*, chosen to give optimal Gaussian quadrature in the integrals over a solid angle [69]. The process however produces ray-effects, as a result of shooting energy from an element in narrow beams along the discrete directions, missing the regions between them [71]. Modifications to the equations were made [71], however, the methods share mathematical similarities to the spherical harmonics method [12]. The modified equations imply that M properly placed directions specify the directional intensity distribution to the same detail as M spherical harmonic coefficients.

For rendering of participating media with multiple anisotropic light scattering, a propagation approximation method for light scattering into M directions was proposed [68]. Two significant enhancements were presented: reduction of the ray-effect by spreading the shot radiosity into the entire direction bin; and treating multiple scattering within a single receiving element before shooting the ray again. These enhancements reduced the computational cost, but these reductions only apply to regular cubic grids and as such, the method will not work on more general finite elements.

The LTE can be solved using the *finite element method*. The method describes when an unknown function is approximated by slicing the domain of the function into a number of small elements, over which the function can be approximated using simple basis functions. The function can then be represented with a finite number of unknowns and solved numerically. *Radiosity* is a global illumination model in CG that simulates diffuse reflections among many surfaces, and often uses the finite element method to gather solutions. *Ray tracing* differs from radiosity in that it simulates light reflecting only once off each surface, but in radiosity, the surfaces of a scene represent the domain of the radiosity function.

The Zonal Method is a generalised LTM for encapsulating the radiative transfer in volumes of participating media [72]. The volume of a participating medium was divided into finite elements which were assumed to have constant radiosity. Form factors were then computed for all surface/surface, volume/volume and surface/volume pairs. Each form factor, F_{kj}, represented

the ratio of the energy leaving an element, E_j, and entering an element, E_k. The algorithm calculated the factors using a depth buffer to project surfaces onto a half cube, a variation of the hemi-cube algorithm [73]. When the form factors had been found, a system of equations for the radiosities of surfaces and volumes were constructed. Complication in the method arose from surface, volume and path radiosities requiring a double integral solution. The Zonal method assumed no interference between light rays, making it limited to isotropic scattering media, thus not suitable for clouds.

2.5 Summary and Motivation

Computer Graphics reasearchers have also developed hybrid cloud and lighting models from these techniques, most notably in cloud modelling. Although surface and spatial cloud techniques can be used in isolation to model clouds, hybrid methods have produced highly detailed clouds (see Table 1). A noticeable omission from Table 1 are particle systems. This can be attributed to the high number of particles required to fill cloud volumes for highly detailed models, which significantly impacts on the computational cost of rendering.

Voxel Grids	Procedural Noise	Metaballs	Texture Mapping	Authors [Citation]
⊗		⊗		Dobashi [74]
⊗		⊗		Miyazaki [75]
	⊗	⊗		Ebert [10]
	⊗	⊗		Schpok [76]
		⊗	⊗	Trembilski [77]
⊗			⊗	Liao [78]

Table 1: Hybrid Cloud Simulation Models

As with cloud models, hybrid LTMs are also possible. One model incorporated finite element techniques and Monte Carlo integration [48]. The light transport between voxels was computed using the phase function as a form factor. Two simplifying observations reduced the cost of computation in this method: not all directions contributed to the illumination of a given voxel; only the first few orders of scattering contributed strongly to the illumination

of a voxel. The final intensity of the pixel was evaluated by solving integral equations, but the method relied heavily on the construction of form factors.

In concluding our review of cloud and light models in CG, we are now able to consider our research questions (from Section 1.4) in light of these models. We review these models in order to find those that fit the criteria for representing physically accurate cloud/light simulations and have a direct correlation with our research questions.

Can we develop a physically accurate cloud model that couples the interaction between cloud and light particles?
To construct a cloud model of high physical accuracy, a particle representation is ideal, as it is analogous to the individual aerosols and associated water droplets that form clouds. Particles can be assigned attributes that govern its size and light interaction properties, and can be grouped into shapes that resemble clouds.

We require a LTM that is able to handle decoupling the solution to the LTE from the actual geometry used, as large quantities of particles will significantly impact upon computation. Although an offline approach will be undertaken, it is still reasonable to find the optimal solution to our current cloud/light problem.

We would like to use a LTM that encapsulates the lighting calculation as individual photons, thus conforming to the ideals of a complete particle system description of cloud/light interactions. This would enable us to extend the notion that each cloud particle exists only if it has had a light particle pass through it, otherwise the cloud particle would not be visible, and thus not exist (from a Computer Graphics point of view).

Can this model avoid evaluating multiple scattering events for cloud particles that don't receive any light?
The major issue when implementing a physically accurate multiple scattering solution for high albedo media, is in the high computational cost attributed

to solving the double integral equation for in-scattered effects [41]. In CG literature, models used to represent this problem have generally been solved with simplifying assumptions that allow for certain levels of physically accurate simulations. These simplifications attempt to find a balance between a physics inspired illumination model, with high computational cost, and an aesthetically pleasing, computationally inexpensive, illumination model.

We argue that when making simplifying assumptions in evaluating the LTE, we compromise the integrity of the final result by not adhering to the fundamentals of cloud and light particle interactions. We would like to be able to use importance sampling to selectively choose the cloud particles that don't receive any light, thus eliminating the need to perform calculations at that point. The calculation of the in-scattered effects (even for an offline approach), should be able to converge onto the result at a fast rate, without compromising the integrity of the simulation, thus an efficient Monte Carlo method is sought. Our scenes would consist of many cloud particles, which would need to be correctly handled by the chosen LTM.

We would like to use a LTM that does not introduce any noticeable artifacts into the scene geometry, or produce any ray effects. Using finite element methods, slicing the domain into a number of small elements would produce mesh artifacts within the cloud particle structure, resulting in a non-realistic representation of the lighting at a point in the cloud. While using discrete ordinates or spherical harmonics could lead to ray effects due to bin or voxel distributions. Using Monte Carlo approaches, it is possible to use arbitrary geometry without mesh artifacts, allowing for complex scenes using relatively low memory consumption. The variance (noise) that appears in a Monte Carlo rendered scene can be alleviated by taking more samples, or by making 'smarter' intelligent samples.

We would like a LTM that decouples scene geometry from the lighting model, efficiently samples the integral domain, and has a reasonable computational cost associated with it.

Can this model be used to simulate sunbeams?
Using our cloud/light particle model, we would like to achieve correct cloud self-shadowing. This would enable us to produce sunbeams; with light egressing our cloud model in areas that are optically thin providing the contrast to areas of the model that are optically thick, thus creating sunbeams.

By reviewing previous research of cloud and lighting models in CG, we have been able to answer our research questions from Chapter 1. By using a particle system approach to cloud and light particles, we can alleviate the computational burden of evaluating the costly multiple scattering term in the LTE. Using a modified Monte Carlo method for evaluating the LTE, allows us to create complex scene geometry that is free from mesh artifacts found in finite element methods. By using intelligently designed importance sampling, we will be able to reduce the variance in the final rendered image, and converge on our result a lot sooner. The next chapter describes in detail, the design decisions made in order to implement our approach.

3 Design

The previous chapter thoroughly discussed previous CG research in cloud and light modelling and answered research questions posed in Chapter 1. Our motivations have led to the design of a particle system approach to modelling cloud and light particle interactions. This chapter describes how we methodically constructed a design implementation based on our motivations.

The design consists of three parts: requirements, representation and implementation. The *requirements* form the basis of knowledge required to model the interactions of light with cloud particles using a modified Monte Carlo method. It consists of a break-down of the mathematical descriptions of the light transport equation (LTE) and Monte Carlo integration. The *representation* provides an overview of the implementation design, detailing the modified Monte Carlo method used and how it suits our cloud/light particle duality. Finally, the *implementation* provides details of how our motivations from the previous chapter combined with design requirements and representations enable use to achieve a cloud/light particle duality model.

3.1 Requirements

A solid mathematical framework is required for the light transport equation (LTE) and Monte Carlo integration in order to clearly define our physical and mathematical representations for cloud/light interactions. These mathematical constructs will need to be implemented, thus they require a clear understanding of their intricate details, preventing possible misrepresentations of their fundamental applications to rendering in our model.

3.1.1 The Light Transport Equation

In Chapter 1, we presented the basic physical interactions between light and clouds. The interaction of light through a cloud was discussed in terms of absorption and scattering, and it was found that as light travels through a cloud, it encounters multiple scattering events due to its optical thickness.

The radiative light transport equation (LTE) for participatory media describes the intensity of light distribution exiting a medium. A mathematical framework of the LTE must be established in order to understand the approaches made by the light transport methods described in Chapter 2, so that any implementation of these methods has a grounding in the physical concepts that drive it. This section examines the individual components of this equation, by creating a mathematical framework that will elaborate on optical depth, the phase function and effects of attenuation on light intensity.

Optical Depth

Optical depth is a dimensionless quantity describing the *opacity* of a medium as light passes through it [47]. The total optical depth $\tau(s, s')$, of a line segment between s and s' in an inhomogeneous medium is related to the extinction coefficient σ_t by

$$\tau(s, s') = \int_s^{s'} \sigma_t(\vec{x} + t\vec{\omega})dt \tag{3}$$

where $\vec{\omega}$ is the direction of propagating light [79, 62]. An infinite optical depth for a medium means it is opaque.

The *transparency* of a medium is derived from optical depth, and is the percentage of light that leaves a point at s and reaches a point on s' along the line segment.

$$T(s, s') = e^{-\tau(s,s')} \tag{4}$$

Transparency is a more useful concept for clouds, therefore the optical depth of a segment can be described as the inverse of transparency [69]

$$\alpha(s, s') = 1 - T(s, s')$$

Phase Function

Upon entering the cloud, incoming light undergoes a series of scattering and

absorption events that modify both the directional structure of the incoming light field and its intensity. As a result of multiple scattering events, the original intensity distribution undergoes angular, spatial and temporal spreading which results in a different intensity distribution (see Figure 11).

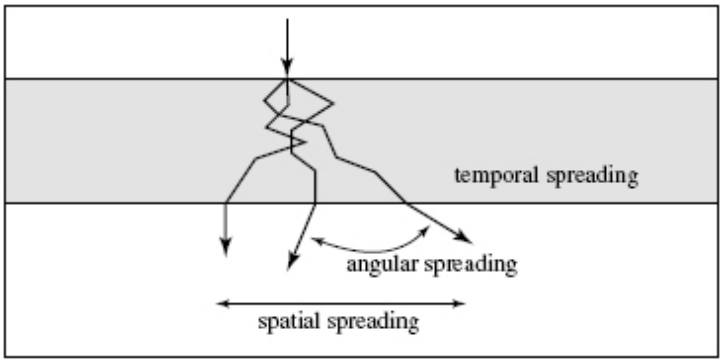


Figure 11: Forms of scattering in a cloud [25]. The original intensity undergoes a series of scattering events that result in angular, spatial and temporal spreading of the original intensity distribution.

The phase function describes the probability density of light coming from direction $\vec{\omega}$ and scattering into direction $\vec{\omega}'$. The phase function is normalised so that

$$\int_{4\pi} P(x, \vec{\omega}', \vec{\omega}) d\omega' = 1$$

and it depends only on the phase angle [80],

$$cos(\theta) = \vec{\omega} \cdot \vec{\omega}'$$

The following phase functions are representative of spherical particles, which suit their application to clouds, as water droplets are spherical.

Isotropic Phase Function

An isotropic phase function scatters light equally in all directions [62],

$$P_I(\phi) = 1 \tag{5}$$

Rayleigh-Gans Phase Function

When particles are small compared to the wavelength of incident light, the phase function is given by [62],

$$P_{RG}(\phi) = \frac{3}{4}\frac{(1 + cos^2\phi)}{\lambda^4} \tag{6}$$

Mie Phase Function

When particles are large compared to the wavelength of incident light, the phase function is defined by Mie theory. A common formula for approximating the Mie scattering for spherical particles is the Henyey-Greenstein phase function [62, 81, 82, 83],

$$P_{HG}(\phi) = \frac{1}{4\pi}\frac{1 - g^2}{(1 + g^2 - 2gcos\phi)^{3/2}} \tag{7}$$

Although these phase functions can be used for clouds, not all can be applied for a physically accurate model. The anisotropic nature of clouds requires either the Rayleigh-Gans or the Henyey-Greenstein phase functions. Taking into account that the nature of particles inside clouds are aerosols, where particles are larger than the wavelength of incident light, the scattering anisotropy will be modelled using the Henyey-Greenstein phase function [82].

Effects of Attenuation on Light Intensity

Remember that the extinction coefficient is the sum of all absorption and scattering, $\sigma_t = \sigma_a + \sigma_s$, it describes the total amount of attenuation to a photon's intensity per unit length travelled through a medium. We are then able to calculate the effects of the attenuation of light intensity in a cloud.

Absorption

The absorption coefficient, σ_a is the probability of absorption per unit length, therefore the change in intensity dL due to absorption over a distance ds in direction $\vec{\omega}$ is [26],

$$\frac{dL(x, \vec{\omega})}{ds} = -\sigma_a(\vec{x})L(x, \vec{\omega}) \tag{8}$$

Out-Scattering

The scattering coefficient, σ_s is used to calculate the out-scattering, and is similar to absorption [26]. The change in intensity dL due to scattering σ_s, over a distance ds in direction $\vec{\omega}$ is,

$$\frac{dL(x,\vec{\omega})}{ds} = -\sigma_s(\vec{x})L(x,\vec{\omega}) \tag{9}$$

Extinction

Because $\sigma_t = \sigma_a + \sigma_s$, equations (8) and (9) are effectively combined into a single equation [26],

$$\frac{dL(x,\vec{\omega})}{ds} = -\sigma_t(\vec{x})L(x,\vec{\omega}) \tag{10}$$

In-Scattering

The in-scattered intensity of a particle depends on the amount of scattering and the direction of scattering. The phase function is used to determine how much of the scattered light at x, is scattered in the direction $\vec{\omega}'$. As light from any direction may be scattered into direction $\vec{\omega}$, it is important to calculate the intensity of incoming directions over the entire particle. Therefore, the change in intensity dL, due to in-scattering over a distance ds, in direction $\vec{\omega}$ is,

$$\frac{dL(x,\vec{\omega})}{ds} = \sigma_s(\vec{x}) \int_{4\pi} P(x,\vec{\omega}',\vec{\omega})L(x,\vec{\omega})d\vec{\omega}' \tag{11}$$

where $\vec{\omega}$ is the incoming direction of in-scattered light [26].

The Light Transport Equation

The direction $\vec{\omega}$, and intensity of light $L(x,\vec{\omega})$, change as a result of a ray of light passing through a cloud. Intensity of light is attenuated due to absorption and out-scattering (Figure 3a,b), while in-scattering intensifies light (Figure 3c) [5, 84].

$$\frac{dL(x,\vec{\omega})}{ds} = -\sigma_t(\vec{x})L(x,\vec{\omega}) + \sigma_s(\vec{x}) \int_{4\pi} P(x,\vec{\omega}',\vec{\omega})L(x,\vec{\omega})d\omega \tag{12}$$

This equation can be solved by bringing the extinction term to the left hand side and multiplying by the integrating factor [69]

$$\exp\left(\int_0^s \sigma_t(t)dt\right)$$

If a ray is parameterised in terms of $t \in [0, D]$, where $t = 0$ is the point on the medium where the light is incident, and $t = D$ where the light exits the medium, equation 12 can be integrated to find the exitant intensity at $t = D$ [69].

$$L(D, \vec{\omega}) = T(0, D)L(0, \vec{\omega}) + \int_0^D T(s, D)g(s)ds \tag{13}$$

where $L(0, \vec{\omega})$ is the incident light intensity, $T(s, D)$ is the transparency defined in equation 4 and $g(s)$ is

$$g(s) = \sigma_s(\vec{x}(s)) \int_{4\pi} P(x, \vec{\omega}', \vec{\omega})L(\vec{x}(s), \vec{\omega}')d\omega \tag{14}$$

Equation 13 is similar to the classic rendering equation [85], where the $g(s)$ term is

$$g(s) = R(x(s))f_s(x(s))L_d \tag{15}$$

where $R(x(s))$ is the surface reflectivity color, $f_s(x(s))$ is the Blinn-Phong surface shading model, and L_d is the intensity of a point light source.

Equation 13 is the *volume rendering equation*, it is the equation that must be solved in order to render participating media. The first term represents the *extinction term* and the second term represents the *in-scattering term*. The extinction term is the light intensity coming from the background and reaching the viewer. The in-scattering term represents light scattered into the view direction from all particles within the cloud. Using this model, an accurate multiple scattering solution is computationally expensive.

Equation 13 describes the light intensity in a *five-dimensional space* (three for position and two for direction) compared to the *four-dimensional space* (two

for surface and two for direction) of the classic rendering equation [86], as each point in space is influenced by the light it receives from every other point, not just points on the surface. Utilising Monte Carlo integration techniques, it is possible to approximate Equation 13 in a more efficient manner than if using standard deterministic quadrature techniques. In order to understand how this is achieved, the next section describes our next requirement, Monte Carlo integration.

3.1.2 Monte Carlo Integration

Monte Carlo integration uses random numbers to approximate integrals. It is a useful technique to use when regular deterministic methods are unsuitable, or impractical. We first review key concepts of probability theory, such as continuous random variables, probability density functions, expected value and variance. Once we establish the definitions of these terms, we describe the technique known as Monte Carlo integration and provide clarification of the term *importance sampling*. We then show how standard Monte Carlo integration can be applied to approximate higher order integrals.

Probability Background

A *continuous random variable* is a scalar or vector quantity that randomly takes on some value on the real domain

$$\{x : x \ \mathbb{R} \in -\infty < x < \infty\}$$

The behaviour of x is entirely described by the distribution of values it takes. Quantitatively, this distribution is described by the *probability density function* (pdf) p, associated with x, denoted by $x \sim p$ [30].

The probability that x will take on a value in the interval $[a, b]$ is given by the integral

$$\text{Probability}(x \in [a, b]) = \int_a^b p(x)dx$$

The density $p(x)$, has two characteristics:

$$\begin{aligned} p(x) \geq 0 \qquad & \text{Probability is non-negative} \\ \int_{-\infty}^{\infty} p(x)dx = 1 \qquad & \text{Probability } (x \in \mathbb{R}) = 1 \end{aligned}$$

The canononical random variable ξ takes on values

$$\{\xi : \xi \ \mathbb{R} \in 0 \leq \xi < 1\}$$

with uniform probability. This implies that the pdf for $\xi \sim q$ is

$$q(\xi) = \begin{cases} 1 & \text{if } 0 \leq \xi \leq 1 \\ & \\ 0 & \text{otherwise} \end{cases}$$

The probability that ξ takes on a value in an interval $[a, b] \in [0, 1)$ is:

$$\text{Probability}(a \leq \xi \leq b) = \int_a^b 1 \ dx = b - a$$

The average value that a real function of a one-dimensional random variable (with underlying pdf, p), will take on is called its *expected value*, $E(f(x))$.

$$E(f(x)) = \int f(x)p(x)dx$$

The expected value of a one-dimensional random variable can be calculated by setting $f(x) = x$. The expected value is linear [30]:

$$E(x + y) = E(x) + E(y)$$

$$E(f(x) + g(y)) = E(f(x)) + E(g(y))$$

The *variance* $V(x)$ of a one-dimensional random variable is by definition the expected value of the square of the difference between x and $E(x)$:

$$V(x) \equiv E([x - E(x)]^2)$$

Algebraic manipulation yields,

$$V(x) = E(x^2) - [E(x)]^2)$$

which is convenient for calculations. The variance of a sum of random variable is the sum of the variances, *only if the variables are independent* [61].

Many problems involve the sums of independent random variables x_i, where the variables share a common density, p. When the sum is divided by the number of variables, we get an estimate of $E(x)$ [87].

$$E(x) \approx \frac{1}{N} \sum_{i=1}^{N} x_i$$

As N increases, the variance of this estimate decreases. By the *Law of Large Numbers*, we gain a statistical confidence as N approaches ∞:

$$\text{Probability} \left[E(x) = \lim_{N \to \infty} \frac{1}{N} \sum_{i=1}^{N} x_i \right] = 1$$

Monte Carlo Integration

Given a one-dimensional integral $\int_a^b f(x)dx$ and a random variable $x \sim p$, we can approximate the expected value of $f(x)$ as [88]:

$$\begin{aligned} E\left[\int_a^b f(x)dx\right] &= E\left[\frac{1}{N}\sum_{i=1}^{N}\frac{f(x_i)}{p(x_i)}\right] \\ &= \frac{1}{N}\sum_{i=1}^{N}\int_a^b \frac{f(x)}{p(x)}p(x)dx \\ &= \frac{1}{N}\sum_{i=1}^{N}\int_a^b f(x)dx \\ &= \int_a^b f(x)dx \end{aligned} \tag{16}$$

For Equation 16 to be valid, p must be positive where f is non-zero.

We want to reduce the variance of f/p, such that f and p have a similar

shape, in order to get a good estimate. Choosing p intelligently is called *importance sampling*, because when p is large where f is large, more samples will exist in important regions.

Given a multi-dimensional integral,

$$\int_{x_0}^{x_1}\int_{y_0}^{y_1}\int_{z_0}^{z_1} f(x,y,z)dxdydz$$

if N samples are chosen uniformly from multi-dimensional pdf such that $\kappa \sim p$, and $\kappa_i = (x_i, y_i, z_i)$ then the estimator is,

$$\frac{(x_1 - x_0)(y_1 - y_0)(z_1 - z_0)}{N}\sum_i f(\kappa_i)$$

The pdf p is a constant value

$$p(\kappa) = \frac{1}{(x_1 - x_0)}\frac{1}{(y_1 - y_0)}\frac{1}{(z_1 - z_0)}$$

The volume rendering equation (see Equation 13) can be approximated by applying multi-dimensional Monte Carlo integration, using uniformly chosen samples from the volume rendering equation's pdf. As we increase the number of samples used in the estimate, we approach the solution to the equation.

The mathematical framework developed in this chapter, assists in our ability to make an informed choice on the particular Monte Carlo ray tracing technique we will use in our model. In the next chapter, we discuss and detail the Monte Carlo ray tracing techniques that best suit our motivation for a cloud/light particle duality.

3.2 Representation

The previous chapter showed how the volume rendering equation can be solved using multi-dimensional Monte Carlo integration. As we increase the number of samples, the estimator converges to the correct result. Classic Monte Carlo ray tracing [89, 90, 21] or path tracing [91, 88, 92] methods use a distributed ray tracing approach [33] for evaluating the volume rendering equation [93]. Such approaches require large numbers of samples for correct convergence, reducing the efficiency of solving the integral. They also treat light as straight paths, whereas we require a technique that uses a particle model.

Photon Mapping is a Monte Carlo ray tracing technique [94] that reduces the complexity of solving the volume rendering equation and, treats light as particles [95, 63]. Photon mapping uses biasing to reduce variance and a kernel density estimate for solving the in-scattering term of the volume rendering equation [26, 88]. It also has the advantage of using a point sampling data structure that is independant of scene geometry, thus making it suitable for complex scenes.

In this chapter, we describe the photon mapping method; how solutions to the volume rendering equation can be approximated using kernel density estimation and, how the decoupled representation of illumination from geometry fulfills our motivations for a cloud/light particle duality.

3.2.1 Photon mapping

Photon mapping is the name given for an algorithm that generates, stores, and uses illumination as points, and the *photon map* is the data structure used to process these points [86]. Photon mapping is a two-pass method in which the first pass is building the photon map and the second pass is rendering. The photon map is built using *photon tracing*, where photons are emitted from the lights and traced through the scene. The second pass uses the information stored in the photon map to make the rendering more efficient [63].

In this section we describe: how each pass of the photon mapping method is applied to participating media; how it reduces the complexity of solving the in-scattering term of the volume rendering equation; and how photon mapping suits our motivation for a cloud/light particle duality.

Photon tracing for participating media

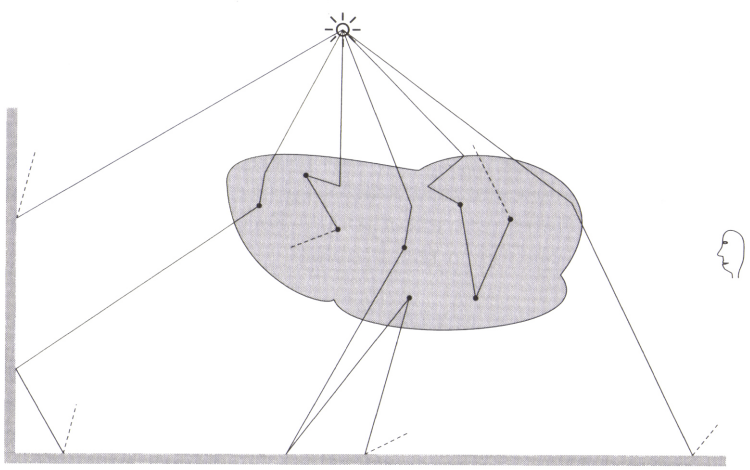


Figure 12: Photon tracing a participating medium [86]. Photons entering a medium pass through it until an interaction event takes place; scattering or absorption. These photons are stored at the location of interaction in the volume photon map.

Photons are created at the light sources in a model. A large number of photons are emitted from each light source (similar to *backwards ray tracing* [96]), and the power, Φ, of the light source is divided amongst all the emitted photons, thus each photon transports a fraction of the light source power, Φ_i. When a photon is emitted from a light source, it is traced through the scene using photon tracing; which is similar to ray tracing except that photons propagate flux, and rays gather radiance.

When a photon enters a participating medium, it moves through the medium until it is either scattered or absorbed. The probability of an interaction transpiring is determined from the exctinction coefficent. The average distance d,

that a photon moves through a medium before an interaction is given by [86],

$$d = \frac{1}{\sigma_t}$$

When a ray of light passes through the medium, its power is attenuated by $e^{-\sigma_t s}$, where s is the distance through the medium. When tracing photons, we may importance sample according to this formula, resulting in the following expression for d:

$$d = -\frac{\log \xi}{\sigma_t} \tag{17}$$

Using this formula to determine the distance to the next interaction, eliminates the need to reduce the power of the photon as it moves through the medium [86].

At the point of interaction, the photon is either scattered or absorbed. The probability of scattering is given by the albedo of the medium,

$$\Lambda = \frac{\sigma_s}{\sigma_t}$$

To decide whether a photon is scattered or absorbed, *Russian roulette* is used [26]. Russian roulette is a Monte Carlo technique used to remove unimportant photons, so that effort can be concentrated on the more important photons (those that actually scatter, rather than absorb) [97]. This is achieved by comparing ξ to Λ,

$$\{\xi : \xi \in [0,1]\} \rightarrow \begin{cases} \xi \leq \Lambda & \text{Photon is } \textit{scattered} \\ \xi > \Lambda & \text{Photon is } \textit{absorbed} \end{cases} \tag{18}$$

If the photon is scattered, the new direction of propagation is found by importance sampling the phase function. We recall the Henyey-Greenstein phase function from Section 3.1.1:

$$P_{HG}(\phi) = \frac{1}{4\pi} \frac{1 - g^2}{(1 + g^2 - 2gcos\phi)^{3/2}}$$

Given the incoming direction of a photon $\vec{\omega}$, the angle of the new scattering direction ϕ, is given by [87],

$$\cos(\phi) = \frac{1}{2g}\left(1 + g^2 - \left(\frac{1 - g^2}{1 - g + 2g\xi}\right)^2\right) \tag{19}$$

When a photon interacts with the medium, it is stored in the volume photon map. By storing only the photons that have scattered at least once before, we discard the contribution of direct illumination for each photon path; concentrating the volume photon map with multiply scattered photons only (indirect illumination) [86]. The storage of these points of interacaction makes it possible to use a kernel density estimation for approximating the out-scattered radiance at a given point in space.

Kernel density estimate for participating media

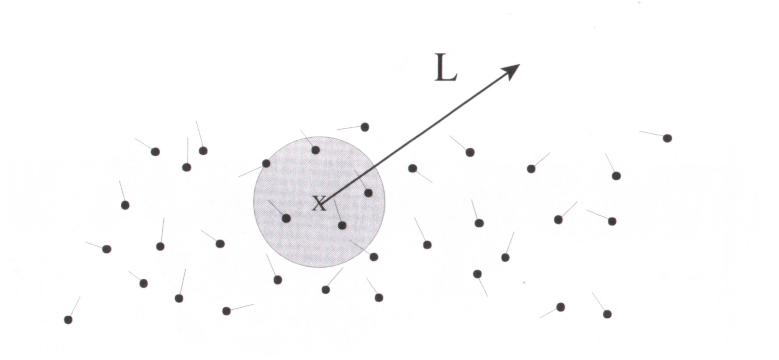


Figure 13: Kernel density estimate for participating media [86].

The stored photons in the photon map represent a fraction of the light source power, Φ_i, received indirectly by a point within the volume. The density of photons in a given region $d\Phi/dA$, estimates light intensity in that region. Kernel density estimatation does not require that surfaces be broken into patches, as the technique operates directly on individual hit points (the photons in the photon map) [92]. This decoupling of the illumination from actual scene geometry makes it possible for a reasonable estimate of illumination to be calculated at any point in a scene, regardless of its complexity.

To estimate the change in out-scattered radiance $L_o(x, \vec{\omega})$, at a point, we evaluate the in-scattered radiance at x, (see Equation 11) [26]

$$\frac{dL_o(x, \vec{\omega})}{dx} = \sigma_s(\vec{x}) \int_{4\pi} P(x, \vec{\omega}', \vec{\omega}) L(x, \vec{\omega}) d\vec{\omega}' \tag{20}$$

As the stored photons represent incoming flux, Equation 20 must be converted to an integral over incoming flux [26]:

$$L(x, \vec{\omega}) = \frac{d^2\Phi(x, \vec{\omega})}{\sigma_s(x) d\vec{\omega} dV} \tag{21}$$

Where dV relates to the density of the photons in a volume.

By combining Equations 20 and 21 we get [26]:

$$\begin{aligned} \frac{dL_o(x, \vec{\omega})}{dx} &= \sigma_s(\vec{x}) \int_{4\pi} P(x, \vec{\omega}', \vec{\omega}) \frac{d^2\Phi(x, \vec{\omega})}{\sigma_s(x) d\vec{\omega} dV} d\vec{\omega} \\ &= \int_{4\pi} P(x, \vec{\omega}', \vec{\omega}) \frac{d^2\Phi(x, \vec{\omega})}{dV} \end{aligned} \tag{22}$$

By locating the n nearest photons to position x, the incoming flux Φ_i can be approximated. Equation 22 can be seen as expanding a sphere around the point x, until it contains enough photons for the estimate (see Figure 13). The volume of the sphere determines the density of the photons, therefore, the resulting out-scattered light intensity can be estimated as:

$$\begin{aligned} \frac{dL_o(x, \vec{\omega})}{dx} &= \int_{4\pi} P(x, \vec{\omega}', \vec{\omega}) \frac{d^2\Phi(x, \vec{\omega})}{dV} \\ &= \sum_{i=1}^{N} P(x, \vec{\omega}_i', \vec{\omega}) \frac{\Delta\Phi_i(x, \vec{\omega_i}')}{\Delta V} \\ &= \sum_{i=1}^{N} P(x, \vec{\omega}_i', \vec{\omega}) \frac{\Delta\Phi_i(x, \vec{\omega_i}')}{\frac{4}{3}\pi r^3} \end{aligned} \tag{23}$$

Equation 23 is the *volume radiance estimate*, it is biased, but consistent [88]. Bias occurs because of the assumption that the nearest photons represent the illumination received at x, while the nearest neighbour density estimate also contributes error. These approximations can introduce artificats into the rendered image, however, the estimate is also consistent and locally defined. Consequently, as more photons are used in the photon map and in the radiance estimate, it will converge to the correct result [88]. The ability to compute a volume radiance estimate using the photon map makes it possible to render participating media more efficiently.

Rendering participating media

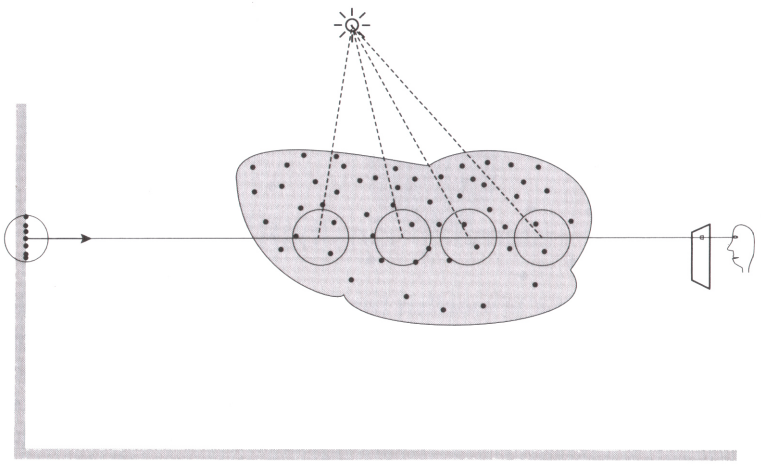


Figure 14: Rendering participating media [86].

The second pass of photon mapping uses a modified Monte Carlo technique for rendering [26]. Participating media is rendered by integrating along the ray that intersects with the medium, a technique known as *ray marching* [87, 22]. In ray marching, the ray from the eye is subdivided into little segments Δx, where local assumptions about the incoming light and the properties of the medium are constant.

At each segment, Δx, a sample point is chosen where the direct illumination component is recursively calculated by moving backwards through the medium until the exit point. The direct illumination for each segment, must be attenuated by the distance the shadow ray moves through the medium (see Figure 14). To calculate the multiple scattering component for each segment, Δx, the sphere of directions around Δx needs to be sampled [98, 87]. We can avoid this by using the volume radiance estimate.

In order to use the volume radiance estimate, we split the volume rendering equation into components for single L_s, and multiple L_m, scattering [26]:

$$L(x, \vec{\omega}) = L_s(x, \vec{\omega}) + L_m(x, \vec{\omega}) \tag{24}$$

For each ray, the single scattering term is evaluated using ray tracing, while the multiple scattering term is evaluated using the volume radiance estimate (see Figure 14). By using the volume radiance estimate, we eliminate the need to sample the sphere of directions around the segment of interest, Δx, thus increasing the efficiency of approximating the multiple scattering component of the volume rendering equation.

Having detailed the steps involved in using the photon mapping method for rendering participating media, we discuss its applicability in motivating our cloud/light particle duality model. We discuss how photon mapping techniques in the first pass can be used to model cloud particles, and how multiple scattering calculations can be minimised during the rendering phase.

The important distinction that photon mapping [26] has over other Monte Carlo techniques [89, 90, 21, 88, 92, 93, 91] is its inherent nature as a light particle model [95]. Light particles can be coupled with cloud particles as a result. We can achieve this by visualising the correlation between the point of next interaction and, the existence of a cloud particle at that point. By importance sampling the distance (see Equation 17) and probability of scattering (see Equation 18), only the most statistically relevant areas of the cloud will contain cloud/light particles that contribute to multiple scattering calculations. From Chapter 1, the unique lighting of clouds is determined

by multiple scattering events that take place when light particles interact with cloud particles. We propose, that creating a cloud/light particle duality using photon mapping will provide: good approximations of realistic cloud models; and, good approximations of the light interactions that take place within them.

During the rendering phase, evaluating the multiple scattering term can be inherently optimised with our model. We can achieve this using a concept of visual importance from the perspective of cloud/light interactions, as opposed to the observer [99]. Our first pass determines where photons interact with cloud particles (where cloud particles exist) in a cloud region. We then re-shape the initial cloud region to correlate with areas consisting of cloud/light particles. The new cloud regions are then rendered using standard ray marching techniques for participating media [98]. Optimisation occurs in restructuring the scene; areas not contributing will be omitted (consequently, they won't be rendered), thus eliminating wasteful photon map queries to approximate multiple scattering during rendering.

These observations of the photon mapping method form the basis of our motivation for a cloud/light particle model. Previous particle system approaches to cloud modelling in computer graphics decouple the light calculations from the cloud representation. In our approach, we have found a correlation between photons interacting within a participating medium and the position of cloud particles. This *new* approach in describing cloud and light particle interactions, combines their existence thus creating a particle duality. Our method is biased due to the fundamental description of the photon mapping method, however, it is consistent, as irregularities are defined locally [88]. By increasing the number of photons used in the photon map and in the nearest neighbour search, we are able to get a better approximation of cloud/light particle positions, thus increasing our convergence on the correct result. In the next section, we discuss how we implement our representation of the cloud/light particle model to test our theory.

3.3 Implementation

In the previous section, we detailed the photon mapping method for approximating the integral of the volume rendering equation (see Equation 13). We associated the photon representation in photon mapping with our motivation for creating a cloud/light duality model to represent clouds. We discussed in terms of placing visual importance on light/cloud interactions, how we can minimise the cost of evaluating the multiple scattering component of the volume radiance estimate (see Equation 23) using our new model. In this section, we describe how we implement our new cloud/light model, and describe the assumptions made in developing this new technique.

Our implementation relies on many forms of sampling, photon emission, phase functions, Lambertian reflectance distributions, as such, we would like to use pseudo-random number generator that does not repeat often and has a good distribution between the uniform numbers generated. The Mersenne Twister is a specifically designed pseudo-random number generator for Monte Carlo simulations, it has a period of $2^{19937} - 1$ and a 623-dimensional equidistribution with up to 32-bit accuracy [100]. The Mersenne Twister is used in our implementation, as it generates very unique uniform random numbers with negligible serial correlation between successive values; making our implementation as random as possible.

We make several design assumptions when implementing the concept model for our cloud/light particle representation. These assumptions are:

- There is only one light source in the scene representing the Sun;
- Cloud regions are defined as simple homogeneous spherical volume regions;
- Diffuse Lambertian surfaces are used for visualising ground shadows only; and
- There is no atmosphere surrounding the homogeneous cloud representation, ensuring homogeneity throughout the scene.

Having outlined our design assumptions, we detail our implementation method. Our new method consists of two passes. The first pass is similar to photon mapping, where we emit photons from the light source towards the scene. We then locate the photon interactions within the volume regions, and create new volume regions that have multiple scattering importance. The second pass is a modified Monte Carlo ray tracer, that uses ray marching to render the volume regions of the scene. We organise this section by specifying the photon/cloud particle emission pass and the rendering pass of our model.

3.3.1 Photon/Cloud Particle Emission

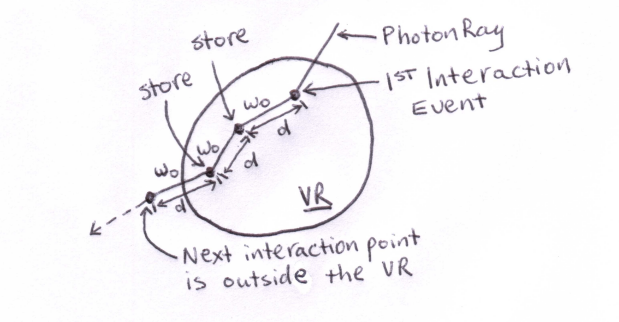


Figure 15: Photon/Cloud Particle Emission.

The first pass of our model is similar to the photon mapping method: we begin by tracing photons into the scene. When a photon enters a volume region (VR), it passes through it until the first interaction even occurs (see Figure 15). If after Russian roulette the photon survives, we importance sample a new photon direction (using Equation 19), and a new distance to the next interaction event (using Equation 17). If the new point, d, along direction, $\vec{w_o}$, is within the VR, we store the photon's flux, position and direction of travel. We repeat this process until one of two conditions are met:

- the photon is absorbed; or
- the next interaction point is outside the VR.

If the photon is absorbed, we first store the photon's flux, position and direction of travel. Next, we terminate the photon's path and emit the next photon from the light source. If the next interaction point is outside the volume region (VR), the photon continues to propagate until the next intersection or interaction event occurs.

As each interaction event occurs within the medium, the photon's intensity is attenuated by the distance it has travelled through the medium:

$$I_{new} = e^{\sigma_t d} \tag{25}$$

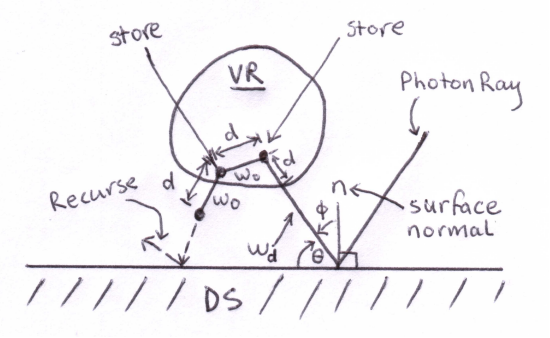


Figure 16: Photon interactions with a diffuse surface.

If the photon intersects a Lambertian diffuse surface (DS), it is reflected in a perfectly random direction $\vec{w_d}$ [87]. Given two uniformly distributed random numbers, $\xi_1 \in [0, 1]$ and $\xi_2 \in [0, 1]$, the direction in spherical coordinates (θ, ϕ) is:

$$\vec{w_d} = (\arccos(\sqrt{\xi_1}), 2\pi\xi_2) \tag{26}$$

If the reflected photon enters a VR, then all interaction events that take place within the VR are stored in the photon map (see Figure 16). This is because all indirect illumination must be considered as contributing to multiple scattering in clouds. The propagation of a photon in a VR after reflecting from a DS, is the same process as above. When the volume photon map is full, we use the stored points of multiply scattered photons to generate our cloud/light particle model.

Cloud/light particle correlation

We analyse the density distribution of the multiply scattered photons for each volume region (VR) individually. We do not add any further bias into the analysis, as we uniformly sample points in the sphere defined by the VR. At each sample point, we find the n nearest photons within a search distance d, of the sample point. The distance d is calculated using Equation 17. If we don't find n, we select another sample point and we repeat the process. If we do find n, a new VR is created; centerred at the sample point, with radius equal to the search distance. We would like the aggregate of volumes for the new VRs to be similar to the original VR. We achieve this using a process similar to Russian roulette:

$$V_{current} = V_{aggregate} + V_{newVR} \rightarrow \begin{cases} V_{current} \leq V_{original} & \textit{Keep} \text{ new VR} \\ V_{current} > V_{original} & \textit{Discard} \text{ new VR} \end{cases} \quad (27)$$

We continue this process until all VRs are analysed in the scene. The next step creates a new scene description based on the newly created VRs and initiates the Monte Carlo ray tracer.

3.3.2 Monte Carlo Ray Tracing

Our implementation of a modified Monte Carlo ray tracer uses a distributed ray tracing approach to gathering radiance received by the observer's eye [33]. The algorithm supports two possible cases of interactions with volume regions (VRs) in the scene:

1. A viewing ray passes through a single VR without obstruction;
2. A shadow ray originating at a point on a Lambertian diffuse surface intersects a VR.

Case 1

When a ray cast from the eye (with origin r_{oe} and direction r_{de}) intersects a VR at position x_M, we cast a secondary ray (with origin x_M and direction r_{de}) to find the exit point of the medium, x_0 (see Figure 17). To prepare

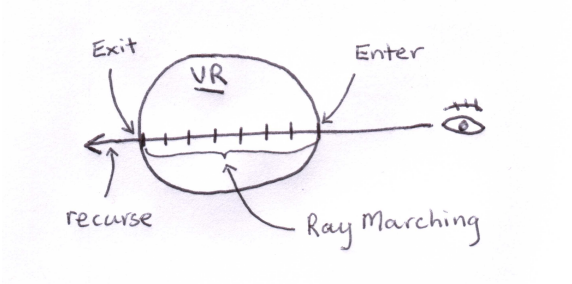


Figure 17: Ray Tracing: Case 1.

for ray marching the VR, we construct a ray from $x_0 \rightarrow r_{oe}$ in direction $(r_{oe} - x_0)$ [98]. We then step through the medium using ray marching. Once the radiance has been accrued from the ray marching process, we recurse the ray tracing step from position x_0, in direction r_{de}. This recursion allows us to step back through the scene, gathering the radiance from any VRs that might be either behind or inside the current VR being evaluated (see Figure 18).

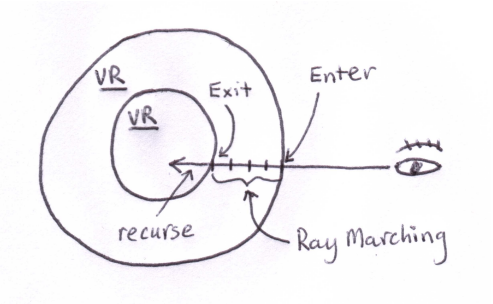


Figure 18: Example of recursion.

Case 2

When a ray cast from the eye (with origin r_{oe} and direction r_{de}) intersects a Lambertian diffuse surface at position x_{ds}, we cast a shadow ray (originating at x_{ds} in the direction towards the light source ω_{ls}), to check for any occluding objects [30, 31, 87].

If the shadow ray intersects a VR at a point x_{si}, the radiance at x_{ds} is a function of the radiance at x_{si} and the local shading model of the diffuse surface at x_{ds}. The recursive ray tracer is used to evaluate the radiance at

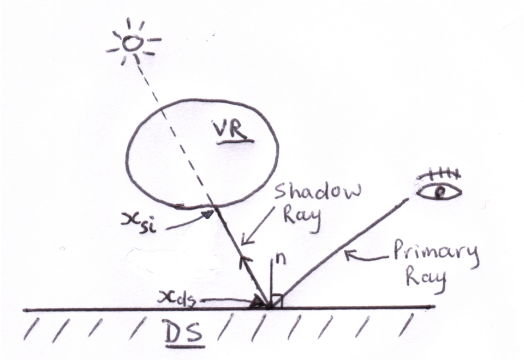


Figure 19: Ray Tracing: Case 3.

x_{si}, so that the radiance at x_{ds} is evaluated correctly (see Figure 19). The radiance at x_{si} is evaluated by using ray marching in the volume region.

Ray marching volume regions

Ray marching numerically integrates the radiance at the point of intersection of the medium, *Enter*, by recursively stepping back through the medium in small steps, and evaluating each step independently. We use Equation 24 to split the calculation into components for single scattering and multiple scattering.

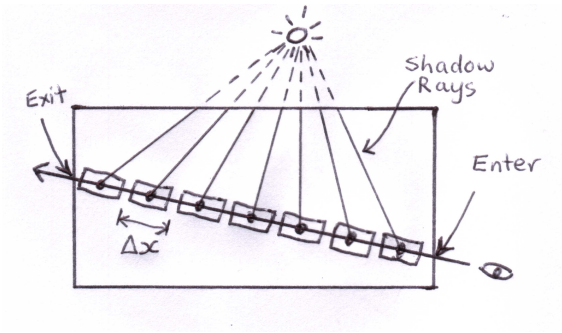


Figure 20: Ray Marching: Single Scattering. The single scattering component is computed using ray marching. The contribution from the light source on each segment is attenuated by the distance travelled by the shadow ray in the medium.

The single scattering component, L_s, is computed using [86]:

$$\begin{aligned} L_{s_{n+1}}(x,\vec{\omega}) = &\sum_{l}^{N} L_l(x,\vec{\omega}')P(x,\vec{\omega}',\vec{\omega})\sigma_s(x)\Delta x \\ &+ e^{-\sigma_t(x)\Delta x}L_n(x+\Delta x,\vec{\omega}) \end{aligned} \tag{28}$$

As we only consider one light source in the scene, Equation 28 simplifies to:

$$\begin{aligned} L_{s_{n+1}}(x,\vec{\omega}) = &L(x,\vec{\omega}')P(x,\vec{\omega}',\vec{\omega})\sigma_s(x)\Delta x \\ &+ e^{-\sigma_t(x)\Delta x}L_n(x+\Delta x,\vec{\omega}) \end{aligned}$$

Computing the transparency of the medium is also simplified due to our assumptions of homogeneous volume regions. In homogeneous volumes, σ_t is constant, thus not requiring us to integrate Equation 3, to find the transmittance of the medium. Consequently, our final ray marching algorithm for the single scattering component has the form:

$$\begin{aligned} L_{s_{n+1}}(x,\vec{\omega}) = &L(x,\vec{\omega}')P(x,\vec{\omega}',\vec{\omega})\sigma_s(x)\Delta x \\ &+ e^{-\sigma_t\Delta x}L_n(x+\Delta x,\vec{\omega}) \end{aligned} \tag{29}$$

The contribution from the light source on each segment, Δx, must be attenuated by the distance the shadow ray moves through the medium (see Figure 20). The homogeneity assumption permits the computation of the attenuation to be expressed as:

$$e^{-\sigma_t\Delta x}L(x,\vec{\omega})$$

At each segment point, Δx, we must also account for the multiple scattering component of the illumination (see Figure 21). The multiple scattering component, L_m, for volume regions is computed using the volume radiance estimate from the photon map [86]:

$$L_{m_{n+1}}(x,\vec{\omega}) = \left\{ \sum_{i=1}^{N} P(x,\vec{\omega}_i',\vec{\omega}) \frac{\Delta\Phi_i(x,\vec{\omega_i}')}{\frac{4}{3}\pi r^3} \right\} \Delta x \tag{30}$$

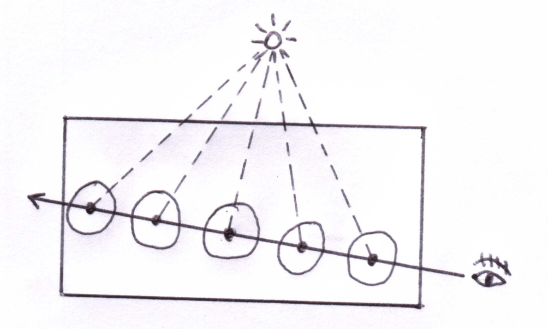


Figure 21: Ray Marching: Multiple Scattering. The multiple scattering component of the volume rendering equation can be seen as sampling the sphere of directions around the sample point. We calculate this component using the radiance estimate from the photon map.

Our final ray marching algorithm for computing the radiance at the point where the primary ray enters the medium is given by:

$$\begin{aligned} L_{n+1}(x,\vec{\omega}) = & L(x,\vec{\omega}')P(x,\vec{\omega}',\vec{\omega})\sigma_s(x)\Delta x \\ & + e^{-\sigma_t \Delta x} L_n(x+\Delta x,\vec{\omega}) \\ & + \left\{ \sum_{i=1}^{N} P(x,\vec{\omega}_i',\vec{\omega}) \frac{\Delta\Phi_i(x,\vec{\omega_i}')}{\frac{4}{3}\pi r^3} \right\} \Delta x \end{aligned} \tag{31}$$

The implementation of our cloud/light particle model has a significant impact on Equation 31. Our new technique analyses the density distributions of multiply scattering events in cloud regions and removes areas of low importance. Subsequently, it strengthens the multiple scattering evaluation in rendering pass, by concentrating effort in important areas.

In the next chapter, we analyse the results of our cloud/light particle duality implementation. We evaluate our design based algorithm complexity and compare our rendered images with photographs of real clouds.

4 Evaluation

The previous chapter detailed the design of our cloud/light particle model. We made assumptions about our implementation and described our two pass method. In this chapter, we evaluate the performance of those algorithms based on computational cost, and compare our rendered images to photo-realistic clouds. The performance tests and rendered images were made on a single 2.64GHz AMD Athlon 64 4000+ processor with 2 gigabytes of memory.

4.1 Complexity

Scene complexity was evaluated based on memory usage and render times. Three scene files were generated (see Table 2) to test the complexity the design. Each test scene comprised of either one, two or three cloud regions. Each cloud region had a $\sigma_s = 0.80$, $\sigma_a = 0.20$ and the Henyey-Greenstein phase function $g = 0.90$. The images are 100x100 in size, with four image samples per pixel and four shadow rays.

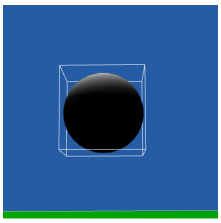
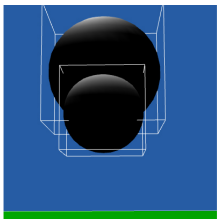
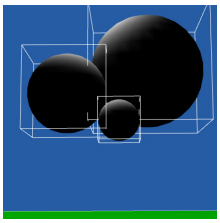
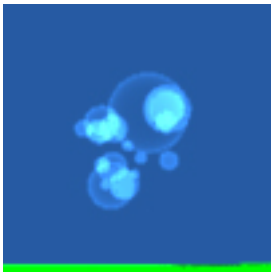
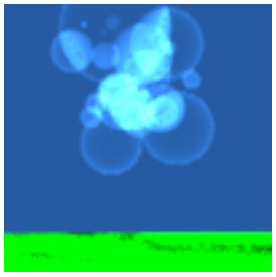
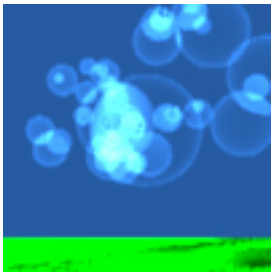
	1	2	3
Scene files			
Images			

Table 2: Cloud Region Complexity Image Results

It can be seen in the images of Table 2, that cloud shadows on the diffuse plane at the bottom of each image are of poor quality, however, they are not hard - having smooth transitions to areas receiving full light. Increasing the number of shadow rays would result in better quality shadows. Initial tests took longer than two hours for sixty-four shadow rays used in these simple test scenes. The rendered images, although not realistic clouds, share many similarities. The brighter areas within each spherical cloud region correspond to areas where there is a higher density of cloud particles, resulting in more multiple scattering effects. These areas resemble the brightness attributed to cumulus clouds (see Figure 1). The areas that appear transparent can be thought of as having less density, resulting in the observer seeing through them, just like high altitude cirrus or altrostratus clouds (see Figure 1).

Table 3 shows the statistics gathered for each of the nine scene tests. The memory usage is attributed to the size of the photon map and the amount of objects that replaced the initial cloud regions. Rendering time is also affected by the traversal of the photon map data structure in approximating the multiple scattering component of the volume rendering equation.

Test No.	Cloud Regions	Photon Map Size	Photons in Estimate	Render Time	Memory Usage
1	1	10,000	100	48sec	15Mb
2	2	10,000	100	1min 32sec	22Mb
3	3	10,000	100	10min 17sec	25Mb
4	1	100,000	500	2min 9sec	17Mb
5	2	100,000	500	5min 7sec	20Mb
6	3	100,000	500	13min 8sec	29Mb
7	1	1,000,000	1000	1h 32min 3sec	40Mb
8	2	1,000,000	1000	3h 14min 42sec	60Mb
9	3	1,000,000	1000	5h 37min 33sec	65Mb

Table 3: Computational Complexity

As the number of volume regions increased, so did the computational cost associated with storing the photon interactions, the number of newly created objects and the final rendering of the image. Table 4 displays on average, how many volume regions were created for each test. The new volume regions are

algorithmicly only placed, in areas that have a concentration of photons equal to the specified photon estimate used in the volume radiance estimate.

Test No.	Cloud Regions	Average Number Volume Regions
1	1	20
2	2	36
3	3	55
4	1	35
5	2	45
6	3	87
7	1	35
8	2	63
9	3	93

Table 4: Volume regions created in test scenes

4.2 Results

Having reviewed the computational cost of our algorithms, we verified our model was able to produce clouds of photo-realistic quality. Having viewed the results from our complexity computations, we required the model to be designed with more cloud regions and higher density areas.

We developed a cloud model using twenty-one volume regions of varying sizes to match our reference photograph of a cumulus cloud. Table 5 shows our reference photograph and our initial model. After the duality process, 256 volume regions were constructed. Ten million photons were used in the construction of this scene, with 5000 in the volume radiance estimate. The memory consumption of our model was approximately 320Mb, fluctuations occured during the searching of the photon map. The resulting image of this test is in Appendix C.

The analysis of our results show that our images require greater depth in the number of shadow rays to evaluate the volume rendering equation. The cost of each ray sent into the scene has the associated cost of the ray marching

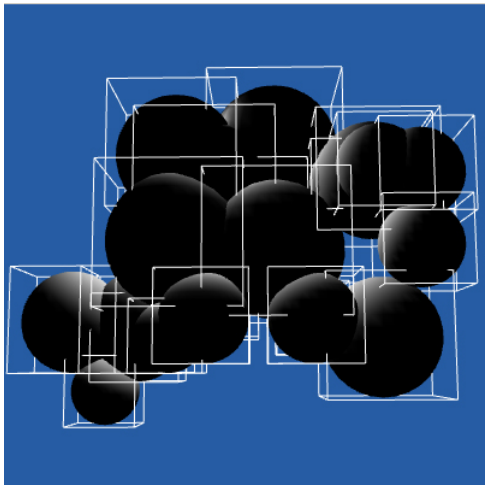
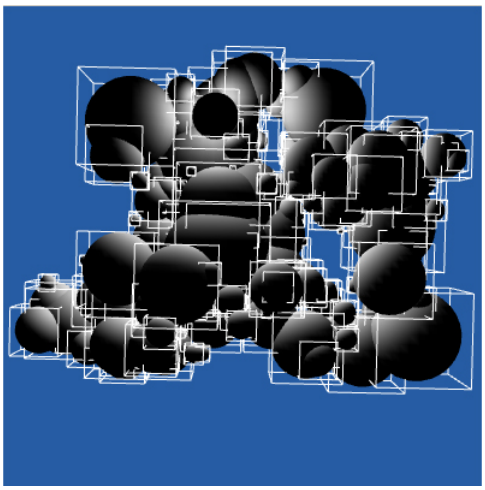
Description	Cumulus Clouds
Reference Image	
Initial Model	
After Duality	

Table 5: Photo-realistic Cloud Comparison: Pre-Rendered Models

traversal through the volume regions. The cost increases as each primary ray interacts with more than one volume region.

Our implementation assumption to maintain homogeneity throughout the scene has impacted on our results, not being able to generate sunbeams. Shadows attributed to clouds can be seen on the diffuse surfaces in Table 2. However, volumetric shadows that provide the contrast between light and cloud shadows can not be visualised as no atmosphere exists, thus no particles for light to reflect off. This assumption has made it possible for us to evaluate our lighting calculations separately.

Our lighting calculations are valid as a result of cloud regions of higher density receiving more in-scattered illumination than areas of lower density. Our results show that as the number of volume regions that are spawned within other volume regions after the duality process increases, we begin to see the affect of multiple scattering and some self-shadowing (image 2 of Table 2).

5 Conclusion

"Luminous beings are we, not this crude matter!"
– Master Yoda

When photons interact with cloud particles in nature, the salient effect is multiple scattering; where all cloud particles receive and scatter light according to the physics of the light transport equation. Computer graphics researchers have used many models to simulate the interactions that take place, by separating the two components that create the visual appearance of clouds. Many computer graphics models for clouds use spatial, surface or hybrid techniques. Computer graphics models pertaining to the interaction of light with clouds, have used regular and irregular methods for solving the complex five-dimensional integral of the light transport equation. Methods to minimise the computational cost of the volume rendering equation often use simplifying assumptions that do not adhere to the laws that govern the visual appearance of clouds; they often only calculate the single scattering term or limit the multiple scattering term to the first few orders of scattering.

In this thesis, we explored the notion of a duality existing between the light scattering events that take place within a cloud volume and the cloud particles themselves. It is a natural relationship, but one that has not yet been explored by computer graphics researchers. The cost of modelling highly detailed physically accurate clouds with particles systems is considered too costly; when the eye can forgive textures placed on surfaces to visualise clouds. However, we believe that there are many applications in scientific visualisation that require the level of detail that our model can provide and aspire to.

In our new cloud generation method, we generated random photon-cloud particles located at the scattering events in a cloud region using kernel density estimation à la photon mapping. Using the density at a given point within the cloud region, and the probability of scattering events taking place, we generated new cloud regions that had a higher probability of contributing to multiple scattering. When a cloud particle doesn't receive any indirect

illumination from other cloud particles, it is not rendered, thus unnecessary radiance calculations are eliminated.

Our concept implementation uses spheres of different sizes to represent cloud regions. The visualised spheres can also be thought of as looking at cloud particles through a microscope; the models represent the aerosols and associated water droplets within clouds. Our model has been implemented in a way that further extensions to the model are possible. These extensions are described next.

5.1 Future work

We have currently developed a new and natural way of modelling clouds and light - by creating a duality between their respective interactions. There are areas of this model that can be improved, as such, future work in this area has a number of opportunities to expand and grow.

Our current method for evaluating new cloud regions uses a uniform random sampling strategy to pick locations of high multiple scattering densities. The rendered complexity test images from our results show, that volume areas of equal density are treated in the same way regardless of the radius used to calculate the desnity estimate. Future work in optimising this algorithm could be with a density importance estimation rather than next scattering interaction in defining the new volume regions.

Our results showed that the computational cost is quite high for rendering any detailed images of a reasonable quality and size. Our concept is unique, and could be developed further on cluster-based-multi-node simulation machines. Our implementation did not utilise any optimisation processes in the rendering phase. Optimisations such as irradiance caching, using the photon map for importance sampling and the efficient stratification of photons should be investigated to validate using such high-performance computing power.

Further work can be done to remove our assumptions of a homogeneous environment. By applying the ability to use heterogeneous cloud volumes,

we are able to explore more permutations within a generated scene. We would be able to apply an atmosphere to our model, thus being able to visualise volumetric shadows, thus sunbeams would be visible. It would also be beneficial in modelling the deformation of interacting clouds from different regions in the atmosphere, such as when cumulonimbus clouds interact with altostratus.

The current way in which clouds are modelled is non-interactive, and requires careful placement of objects within a scene using a text editor. Future work in providing a GUI for placing the initial cloud model would significantly increase the throughput of the end user. By providing a GUI, it would also be possible to move away from spheres to model the cloud macrostructure. Metaballs are an intuitive in the way clouds are shaped, as such, interacting metaballs for model placement in a GUI would allow for greater control and freedom, and be very useful in commercial avenues of computer generated movies and scientific visualisations.

On reflecting on Master Yoda's phrase, the significance of the duality model can be conceptualised. Where there is light intersect matter, the duality principle is enforced. In computer graphics, surfaces or volumes not receiving any light do not need to be rendered, thus light becomes the master of the environment. Possible extensions that have not been discussed in this thesis are that light/particle dualities exist for many other mediums. Fire can be modelled as both a particle system and a light source, and would present an interesting challenge in capturing the duality concept. Sub-surface scattering that occurs in skin or in milk could be visualised using the duality principle.

APPENDIX A - Publications

"Design of a simulation of atmospheric sunbeams" WSEAS Transactions on Computers, vol. 5, no. 10. pp. 2466-2471 October, 2006 ISSN 1109–2750

Design of a simulation of atmospheric sunbeams

JOSEPH HURA[1] RICHARD HALL[2]

Dept. Computer Science & Engineering,
La Trobe University, Melbourne,
3086, Victoria, Australia

j.hura@latrobe.edu.au[1] richard.hall@latrobe.edu.au[2]

Abstract: Computer graphics researchers have made significant progress in modelling clouds and light. However, for a model of atmospheric sunbeams to have any pseudo-physical basis, it is necessary for these two classes of models to interact. In this paper we review each of these classes and design a model for offline rendering of sunbeams. Applications could include photorealistic rendering of outdoor scenes for computer generated movies using high-performance computing.

Key-Words: sunbeams; lighting; clouds; atmospheric scattering;

Figure 22: Crepuscular rays (above cloud).

Introduction

In atmospheric science, *sunbeams* (aka crepuscular rays) are defined to be light rays produced by sunlight penetrating cloud gaps interleaving with shadows cast by clouds or mountains [101, 27](see Figure 22). In the context of this paper, we adopt this definition but exclude the mountain-generated shadow source as our focus is on clouds and light. Within atmospheric science, the physics of sunbeams are described by radiative transport theory [5], which represent light and cloud interactions as a complex mathematical system. However, it is possible to give a pseudo-physical explanation of sunbeams in terms of the interaction of clouds and light.

Clouds are an atmospheric body (that exist at one of several altitudes)

consisting of particles of aerosols and associated water droplets [1]. When *light* particles intersect with cloud particles, the salient effect is multiple scattering (photons diverge from their straight path ala Mie theory [8]). Different colours and brightness of cloud particles is produced by light redistribution and wavelength filtering via scattering by different types of particle [2]. Clouds cast shadows; these regions are illuminated by airlight (diffuse sunlight scattered by air molecules [27]). Direct and scattered sunlight exiting clouds thus create a contrast between sunlight and airlight volumes, producing *sunbeams* [101] which converge at the solar point due to perspective [27].

The organisation of this paper is as follows. In Section 5.1 we discuss computer graphics models for clouds, and comment on their potential applicability for rendering sunbeams. Having discussed how clouds are modelled, we are able to discuss how lighting is modelled for clouds, to which we devote Section 5.1. Section 5.1 presents our final design and discusses future work.

Cloud models

Cloud models in computer graphics can be categorised as surface and spatial models. *Surface* models wallpaper a cloud skin onto computationally-cheap shapes to produce the appearance of clouds, rather than attempting to represent the actual shape of a cloud body. For example, these models apply some ad-hoc but aesthetically acceptable texturing function (eg. poor-man's Fourier series [9] or procedural noise [57]) to the surface of simple objects like meshes, spheres or ellipsoids. Such models can be rendered in a variety of ways, including the forward rendering pipeline, raytracing, and Z or A buffer scanline techniques, in software and hardware [10]. Entire cloud volumes can be represented by single objects compared to spatial models.

In contrast to surface models, *spatial* models attempt to represent the 3D volume of clouds by partitioning space into many objects. The appearance of clouds is produced only by the interactions of light between these object volumes. These volumes can have rigid boundaries eg. voxels and particle systems or deformable boundaries eg. metaballs.

Voxels (volumetric pixels) are a discretisation of 3D objects into a large group of small adjacent cubes, commonly used for physically based simulation. Each cube contains data such as pressure, temperature, and velocity, derived from partial different equations based on the Navier-Stokes equations for simulating fluids [12]. Due to their structure, voxels are particularly suitable for ray marching methods for rendering (forward or backward) and traditional ray tracing clouds. Despite their popularity, voxels provide a less intuitive model of clouds than particle systems because cloud particles are generally different sizes.

Particle systems simply consist of a set of particles, each which represents a point in 3D space with attributes such as size, density and colour; a cloud is by nature a particle system. The model was originally introduced to model composite objects exactly like clouds [43] and has been expanded to approximate shading, cater for self-shadowing, external shadows and external light sources [44]. Each particle is rendered by blending it into a frame buffer, with the transparency of a particle governing the final pixel color [46].

Metaballs (aka blobs and soft objects) are a spatial model of clouds with deformable boundaries, represented by volumes of additive scalar field functions. Metaball implementations for cloud models usually define each ball with a centre, radius and density at the centre of the ball [48, 11]. When the surfaces of neighbouring metaballs intersect, the effect on each point on each surface is calculated with respect to the radius of influence of each intersecting metaball, and these surfaces deform accordingly [47]. Metaballs provide a natural analogue for clouds, since when clouds collide they grow bigger due to coalescence of water droplets [2]. There are several approaches to choosing how many metaballs and what arrangement will produce an aesthetically acceptable cloud: user-defined [48], fractal [51], and cloud image-fitting [11]. Metaballs have been rendered via ray-tracing and splatting [53] and also via the marching cubes algorithm [54].

Several researchers have created hybrid models (see Table **??**) out of these surface and spatial models of clouds. A noticeable omission from the table is particle systems, which have probably been avoided largely

due to the high-performance computing costs associated with large numbers of particles in clouds.

Having discussed cloud models, we now consider these models with respect to our physical explanation of sunbeams. Generally speaking, in computer graphics it is the *shape* of objects which determine lighting and shadows further into a scene as opposed to the *appearance* of objects (surface model). Nonetheless, a computationally cheap approach to sunbeams could use these surface models as a filter for sunlight. Light egressing the cloud surface from any point would be a function of the texture at that point on the incident sunlight (ala [102]) as opposed to light scattered by the shape of the cloud, thus surface models may produce sunbeams of questionable physical accuracy.

To construct a model of clouds with high physical accuracy, we argue that the closest intuitive candidate is a hybrid of particle systems and metaballs. Particles with varying sizes and distributions (in contrast to voxels) can be used to model the microstructure of clouds in terms of aerosols and associated water droplets. Light egressing clouds will thus consist both of direct sunlight penetrating gaps in the clouds and light bouncing off these particles (which effectively become a weak secondary light source) as required by our physical model. Metaballs will be used to provide an invisible bounding shape macrostructure for these particles, on the grounds that clouds do coalesce into larger physical bodies and smaller clouds can sometimes break away from these large bodies. Such an approach to modelling clouds, while computationally expensive, is acceptable for offline rendering.

Lighting Models for Clouds

There are several models of lighting in computer graphics; arguably the most appropriate for rendering light and cloud particle interactions is *radiosity* as it models the interaction of light between diffuse surfaces and is directly derived from physics models of radiative heat transfer [19]. Similar to the rendering equation [18], at any position x and direction $\boldsymbol{\omega}$ in space, the change in outgoing light intensity $L(\boldsymbol{x}, \boldsymbol{\omega})$ is equal to the sum of all incoming light from all direc-

tions (32) minus any light which is absorbed (33). This equation is referred to henceforth as the light transport equation (LTE).

$$\frac{dL(\boldsymbol{x},\boldsymbol{\omega})}{ds} = K_s(\boldsymbol{x}) \int_{4\pi} P(\boldsymbol{x},\boldsymbol{\omega},\boldsymbol{\omega}')$$

$$L(\boldsymbol{x},\boldsymbol{\omega}')d\boldsymbol{\omega}' \tag{32}$$

$$- K(\boldsymbol{x})L(\boldsymbol{x},\boldsymbol{\omega}) \tag{33}$$

We now elaborate on the components of these equations. Equation 32 introduced two new terms. Firstly, the out-scattering coefficient K_s is the scattering cross section per unit volume. Secondly, the phase function $P(\boldsymbol{x},\boldsymbol{\omega},\boldsymbol{\omega}')$ is the probability density of light coming from direction $\boldsymbol{\omega}$ and scattering into direction $\boldsymbol{\omega}'$. Several phase functions have been used in computer graphics: constant, anisotropic, Lambertian, Rayleigh scattering, and Henyey-Greenstein (see Equation 34); the latter being most popular [6]. Note that g is the eccentricity of an ellipse.

$$P_{HG}(\phi) = \frac{1}{4\pi} \frac{1-g^2}{(1+g^2-2gcos\phi)^{3/2}} \tag{34}$$

Equation 33 introduced one new term: the extinction coefficient ($K(\boldsymbol{x})$) is the sum of both absorption K_a and scattering coefficients K_s, describing the total amount of attenuation of a photon per unit volume. $K(\boldsymbol{x})$ is related to various optical properties of a medium including optical depth and transparency [69].

We intend to normalise LTE values in the range [0..1] and relate this range to physical light intensity in the following way. At one extreme, a 0 indicates no light intensity which is equivalent to airlight volumes in the cloud's umbra. At the other extreme, a 1 indicates areas which receive maximum sunlight. In the middle are values representing varying degrees of penumbra receiving scattered light. These regions should frequently border direct sunlit volumes, since light scatters off cloud particles bordering the gaps through which sunlight permeates.

There are four main methods for solving the LTE in cloud modelling: monte carlo; discrete ordi-

nates; spherical harmonics; and the finite element [20]; all which have been successfully applied to voxels. With the *monte carlo* method, large numbers of photons (with the same energy) are sent into the scene from the light source in random directions, essentially performing a random sample of the integral domain of the LTE [103]. Without enough samples, images produced by monte carlo methods may contain significant aliasing. There are several improvements to the basic method that ensure that only the most significant scattering events determine light intensity (eg. the Schlick phase function [62] and photon mapping [63]).

The *discrete ordinates* method uses a collection of M discrete direction *bins*, chosen to give optimal Gaussian quadrature in the integrals over a solid angle [69]. The process however produces ray-effects, as a result of shooting energy from an element in narrow beams along the discrete directions, missing the regions between them [71] that requires compensation [68]. In addition, discrete ordinates is somewhat crude, because with outscattering we need to finely sample a particular solid angle of a sphere rather than have a uniform sampling across the whole sphere [12]. Consequently these equations have been modified, but these new equations share mathematical similarities to the spherical harmonics method [12].

The *spherical harmonic* method is based on representing the directional variation of the radiance at each position as an expansion in spherical harmonics, up to a certain order, and solving a system of partial differential equations for the spatial variation in the spherical harmonics coefficients [12]. The spherical harmonics $Y_l^m(\theta, \phi)$ form an orthonormal basis of the functions on the unit sphere [65], meaning that a function of a solid angle $f(\theta, \phi)$ may be represented by an infinite series expansion where P^m are the associated Legendre polynomials.

$$f(\theta, \phi) = \sum_{l=0}^{\infty} \sum_{m=-l}^{l} P_l^m Y_l^m(\theta, \phi) \quad (35)$$

Rendering LTE through participating media using spherical harmonics has become a popular technique [67]. Typically, scattering and absorption through each cloud are initially calculated using spherical harmonics with

the results stored in voxels, followed by ray tracing. Spherical harmonics have become known as the P_N-method, where N is the degree of the highest harmonic in the expansion [69]. Using only the first order N-term approximates diffuse scattering [70].

The *finite element* method is based on a geometrically similar mesh of discrete regions to which equations are applied. A system of simultaneous equations is thus constructed that is solved using various mathematical techniques. One method for representing radiative heat transfer used finite elements with constant radiosity [72]. Form factors were computed for all surface/surface, volume/volume and surface/volume pairs. Each form factor, F_{kj}, represented the ratio of the energy leaving an element, E_j, and entering an element, E_k. The algorithm calculated the factors using a depth buffer to project surfaces onto a half cube (ala the hemi-cube algorithm). When the form factors had been found, a system of equations for the radiosities of surfaces and volumes were constructed.

Having discussed lighting models for clouds in computer graphics; specifically, the four main methods for solving the light transport equation, we now consider these methods with respect to our physical explanation of sunbeams. Fundamentally, all of these methods sample the integral domain of the LTE in some way: monte carlo methods (in their native form), sample it randomly; discrete ordinates and finite element methods partition the space of the domain; and spherical harmonics make use a simplified approximation of the integral.

We believe that the method for solving the LTE that is the most physically faithful is a modified monte carlo method, because enough rays cast into the scene can be genuinely representative of the integral domain of the LTE, as opposed to approximative. We intend to address the issue of the wastefulness of monte carlo (it fires off many rays in many directions that will not intersect any cloud particles) such that it takes a long time to produce an image of acceptable aesthetics, at very high associated computational cost. We are currently investigating methods for constraining forward ray direction by using directions derived from backward ray tracing from the eye source to the

light source.

Conclusion

In this paper we investigated current methods for representing clouds and light in computer graphics. We designed a simulation of atmospheric sunbeams that will hybridise two models of clouds: particle systems and metaballs; and will utilise modified monte carlo methods to solve the light transport equation. This simulation was designed with the requirement of maintaining pseudo-physical faithfulness, aiming to produce high quality images of different types of clouds with good physical correspondence. Due to the high performance needs of our simulation, we will implement our design on a machine that is suitable for our needs. We intend to trial our algorithms on a 97 node, 194 CPU Linux cluster based on Xeon 2.8 GHz CPUs, with between 1 and 2 Gigs of RAM per node and 1 Terabyte of disk space, and design our algorithms to use MPI. We will need to experiment with distributing groups of particles between processors without overloading inter-node communication bandwidth.

Like most computer graphics research, the primary evaluation criteria of our simulation results will be its visual aesthetic. Initially, we will experiment with a number of simple homogenous particles bounded by simple shapes in order to determine that our implementation of the light transport equation is adequately effective in order to produce sunbeams with simple particles. Then, we will systematically scale up the complexity of the bounding shapes from simple shapes to interacting metaballs, and create different pockets of heterogenous particles within the cloud participating media. In following such an approach we will be able to note computational resource usage, thus be able to compare compute times and scene complexity with the aesthetics of the result.

Acknowledgement We would like to thank the Department of Computer Science and Computer Engineering at La Trobe University for supporting this research.

APPENDIX B - Perspective

From Section 1.3, emitted rays of light from the Sun are near parallel. With visual perspective, the eyes refocus near-parallel rays arriving from distant objects toward a focal point within the eye, the cornea. Our brain can only process the angular size of an object by analysing its angular measurement within our field of view [104].

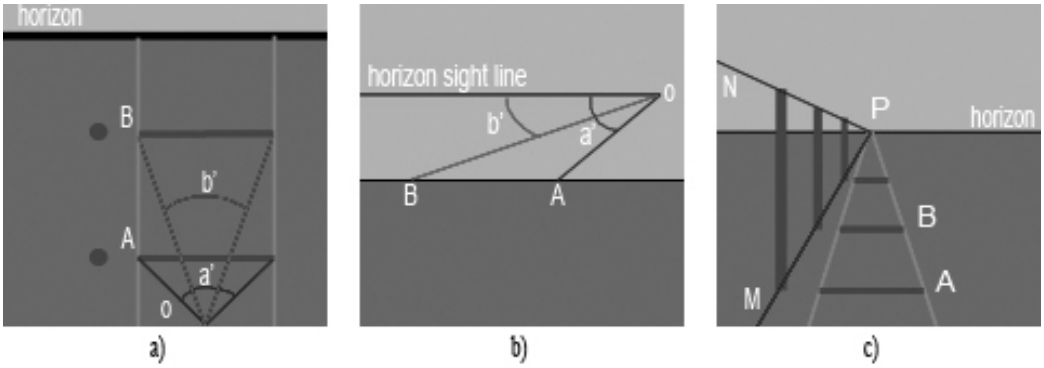


Figure 23: Perspective Traintracks. a) Top View: The angle created by b' is smaller than that of a', so B appears smaller than A. b) Side View. The angle created by b' is smaller than that of a' to the horizon sight line. c) Observer's View: All parallel lines extending to the horizon converge to P, the vanishing point.

With reference to Figure 23, a classic train track scenario is illustrated to explain perspective [104]. The scenario consists of straight railroad tracks stretching to the horizon, as well as telephone poles on one side of the tracks. The train tracks, tie points and telephone poles are equidistant and the telephone poles are all of the same height. As these objects approach the horizon, the tracks converge to a single point, while the distance between tie points decreases and the telephone poles diminish in size.

Figure 23a shows the observer's position O, in the middle of a train track, with tie points A and B being equidistant. From the observer's position, the angle created by b' is smaller than that of a' therefore B will appear smaller than A.

Figure 23b helps determine the relative placement of objects within an observer's field of view. The angle of declination of point B is smaller than

that of A with respect to the angle of declination to the horizon ($0°$). Point A will then appear below point B, with both being below the horizon line.

Figure 23c is the observer's view of the railway tracks as a combination of the previous two images. The tops and bottoms of the perpendicular telephone poles are joined by invisible orthogonal lines, M and N, that extend to point P. All parallel lines within the scenario extend to the horizon, converging at point P, the vanishing point. This is called perspective.

APPENDIX C - Color Plate

Our resulting rendered image (see Table 7) highlights the areas of multiple scattering significance. Although the rendered image differs from the reference image in surface appearance, our aim is to correctly visualise areas of multiple scattering significance. Our results show that cloud regions of higher density receiving more in-scattered illumination than areas of lower density; that as the number of volume regions spawned within other volume regions after the duality process increases, we begin to see the affect of multiple scattering and some self-shadowing.

To approach a photo-realistic representation given our accurate lighting model, we would use turbulent noise and our lighting calculations with respect to optical depth (see Section 3.1.1) to create the surface appearance of the cloud. This has been shown in previous approaches to increase the photo-realism of lighting calculations [26].

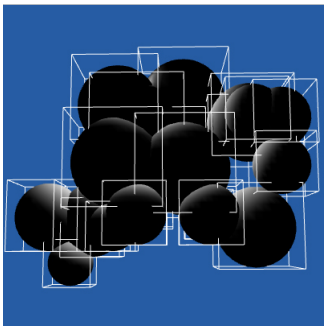
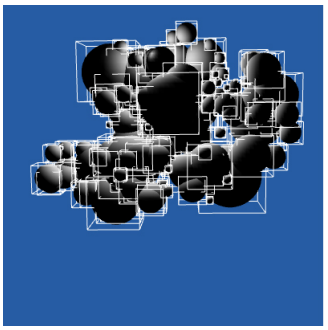
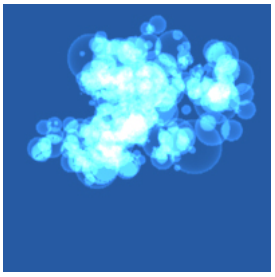
Description	Cumulus Clouds
Reference Image	
Initial Model	
After Duality	
Rendered Image	

Table 7: Photon-realistic Cloud Comparison: Results

References

[1] M. L. Salby, *Fundamentals of Atmospheric Physics*. Academic Press, 1996.

[2] R. Goody, *Principles of Atmospheric Physics and Chemistry*. Oxford University Press, 1995.

[3] R. G. Fleagle and J. A. Businger, *An Introduction to Atmospheric Physics*. Academic Press, second ed., 1980.

[4] J. V. Iribarne and H.-R. Cho, *Atmospheric Physics*. D. Reidel Publishing Company, 1980.

[5] S. Chandrasekhar, *Radiative Transfer*. New York, USA: Dover Publications, Inc., 1960.

[6] J. F. Blinn, "Light reflection functions for simulation of clouds and dusty surfaces," in *SIGGRAPH '82: Proceedings of the 9th annual conference on Computer graphics and interactive techniques*, (New York, USA), pp. 21–29, ACM Press, 1982.

[7] J. Sloup, "A survey of the modelling and rendering of the earth's atmosphere," in *SCCG '02: Proceedings of the 18th spring conference on Computer graphics*, (New York, USA), pp. 141–150, ACM Press, 2002.

[8] H. van de Hulst, *Light Scattering by Small Particles*. New York, USA: Dover Publications, Inc., 1981.

[9] G. Y. Gardner, "Visual simulation of clouds," in *SIGGRAPH '85: Proceedings of the 12th annual conference on Computer graphics and interactive techniques*, (New York, USA), pp. 297–304, ACM Press, 1985.

[10] O. Deusen, D. S. Ebert, R. Fedkiw, F. K. Musgrave, P. Prusinkiewicz, D. Roble, J. Stam, and J. Tessendorf, "The elements of nature: interactive and realistic techniques," in *GRAPH '04: Proceedings of the conference on SIGGRAPH 2004 course notes*, (New York, USA), p. 32, ACM Press, 2004.

[11] Y. Dobashi, T. Nishita, H. Yamashita, and T. Okita, "Modeling of clouds from satellite images using metaballs," in *PG '98: Proceedings of the 6th Pacific Conference on Computer Graphics and Applications*, (Washington DC, USA), pp. 53–60, IEEE Computer Society, 1998.

[12] J. T. Kajiya and B. P. V. Herzen, "Ray tracing volume densities," in *SIGGRAPH '84: Proceedings of the 11th annual conference on Computer graphics and interactive techniques*, (New York, USA), pp. 165–174, ACM Press, 1984.

[13] G. V. G. Baranoski, J. Wan, J. G. Rokne, and I. Bell, "Simulating the dynamics of auroral phenomena," *ACM Trans. Graph.*, vol. 24, no. 1, pp. 37–59, 2005.

[14] Y. Dobashi, T. Yamamoto, and T. Nishita, "Interactive rendering of atmospheric scattering effects using graphics hardware," in *HWWS '02: Proceedings of the ACM SIGGRAPH/EUROGRAPHICS conference on Graphics hardware*, (Aire-la-Ville, Switzerland), pp. 99–107, Eurographics Association, 2002.

[15] K. Hegeman, M. Ashikhmin, and S. Premože, "A lighting model for general participating media," in *SI3D '05: Proceedings of the 2005 symposium on Interactive 3D graphics and games*, (New York, USA), pp. 117–124, ACM Press, 2005.

[16] X. D. He, K. E. Torrance, F. X. Sillion, and D. P. Greenberg, "A comprehensive physical model for light reflection," *SIGGRAPH Comput. Graph.*, vol. 25, no. 4, pp. 175–186, 1991.

[17] J. T. Kajiya, "Anisotropic reflection models," in *SIGGRAPH '85: Proceedings of the 12th annual conference on Computer graphics and interactive techniques*, (New York, USA), pp. 15–21, ACM Press, 1985.

[18] J. T. Kajiya, "The rendering equation," in *SIGGRAPH '86: Proceedings of the 13th annual conference on Computer graphics and interactive techniques*, (New York, NY, USA), pp. 143–150, ACM Press, 1986.

[19] C. M. Goral, K. E. Torrance, D. P. Greenberg, and B. Battaile, "Modeling the interaction of light between diffuse surfaces," in *SIGGRAPH '84: Proceedings of the 11th annual conference on Computer graphics and interactive techniques*, (New York, USA), pp. 213–222, ACM Press, 1984.

[20] R. Siegel, *Thermal Radiation Heat Transfer*. Hemisphere Publishing Corp, 1981.

[21] S. Collins, "Monte carlo methods and the challenge of photo-realism in computer graphics," in *Proceedings of the 1998 Hitachi Symposium*, 1998.

[22] D. S. Ebert, D. P. F. Kenton Musgrave, K. Perlin, and S. Worley, *Texturing and Modelling: A Procedural Approach*. Morgan Kaufmann, third ed., 2003.

[23] R. D. Thompson, *Atmospheric Processes and Systems*. Routledge, 1998.

[24] A. Boudet, P. Pitot, D. Pratmarty, and M. Paulin, "Photon splatting for participating media," in *GRAPHITE '05: Proceedings of the 3rd international conference on Computer graphics and interactive techniques in Australasia and South East Asia*, (New York, USA), pp. 197–204, ACM Press, 2005.

[25] S. Premože, M. Ashikhmin, and P. Shirley, "Path integration for light transport in volumes," in *EGRW '03: Proceedings of the 14th Eurographics workshop on Rendering*, (Aire-la-Ville, Switzerland), pp. 52–63, Eurographics Association, 2003.

[26] H. W. Jensen and P. H. Christensen, "Efficient simulation of light transport in scences with participating media using photon maps," in *SIGGRAPH '98: Proceedings of the 25th annual conference on Computer graphics and interactive techniques*, (New York, USA), pp. 311–320, ACM Press, 1998.

[27] D. K. Lynch and W. Livingston, *Color and Night in Nature*. Cambridge University Press, second ed., 2001.

[28] A. Woo, P. Poulin, and A. Fournier, "A survey of shadow algorithms," *IEEE Comput. Graph. Appl.*, vol. 10, no. 6, pp. 13–32, 1990.

[29] J. Hura and R. Hall, "Design of a simulation of atmospheric sunbeams," *WSEAS Transactions on Computers*, vol. 5, no. 10, pp. 2466–2471, 2006.

[30] P. Shirley, *Fundamentals of Computer Graphics*. AK Peters, second ed., 2005.

[31] S. R. Buss, *3-D Computer Graphics: A Mathematical Introduction with OpenGL*. Cambridge University Press, 2003.

[32] T. Whitted, "An improved illumination model for shaded display," *Commun. ACM*, vol. 23, no. 6, pp. 343–349, 1980.

[33] R. L. Cook, T. Porter, and L. Carpenter, "Distributed ray tracing," in *SIGGRAPH '84: Proceedings of the 11th annual conference on Computer graphics and interactive techniques*, (New York, USA), pp. 137–145, ACM Press, 1984.

[34] J. Stam, "Stable fluids," in *SIGGRAPH '99: Proceedings of the 26th annual conference on Computer graphics and interactive techniques*, (New York, USA), pp. 121–128, ACM Press/Addison-Wesley Publishing Co., 1999.

[35] F. Losasso, F. Gibou, and R. Fedkiw, "Simulating water and smoke with an octree data structure," *ACM Trans. Graph.*, vol. 23, no. 3, pp. 457–462, 2004.

[36] R. A. Drebin, L. Carpenter, and P. Hanrahan, "Volume rendering," in *SIGGRAPH '88: Proceedings of the 15th annual conference on Computer graphics and interactive techniques*, (New York, USA), pp. 65–74, ACM Press, 1988.

[37] L. M. Sobierajski and A. E. Kaufman, "Volumetric ray tracing," in *VVS '94: Proceedings of the 1994 symposium on Volume visualization*, (New York, USA), pp. 11–18, ACM Press, 1994.

[38] S. Roettger, S. Guthe, D. Weiskopf, T. Ertl, and W. Strasser, "Smart hardware-accelerated volume rendering," in *VISSYM '03: Proceedings of the symposium on Data visualisation 2003*, (Aire-la-Ville, Switzerland), pp. 231–238, Eurographics Association, 2003.

[39] R. Fedkiw, J. Stam, and H. W. Jensen, "Visual simulation of smoke," in *SIGGRAPH '01: Proceedings of the 28th annual conference on Computer graphics and interactive techniques*, (New York, USA), pp. 15–22, ACM Press, 2001.

[40] D. Overby, Z. Melek, and J. Keyser, "Interactive physically-based cloud simulation," in *PG '02: Proceedings of the 10th Pacific Conference on Computer Graphics and Applications*, (Washington DC, USA), p. 469, IEEE Computer Society, 2002.

[41] M. J. Harris, W. V. Baxter, T. Scheuermann, and A. Lastra, "Simulation of cloud dynamics on graphics hardware," in *HWWS '03: Proceedings of the ACM SIGGRAPH/EUROGRAPHICS conference*

on Graphics hardware, (Aire-la-Ville, Switzerland), pp. 92–101, Eurographics Association, 2003.

[42] N. Foster and D. Metaxas, "Modeling the motion of a hot, turbulent gas," in *SIGGRAPH '97: Proceedings of the 24th annual conference on Computer graphics and interactive techniques*, (New York, USA), pp. 181–188, ACM Press/Addison-Wesley Publishing Co., 1997.

[43] W. T. Reeves, "Particle systems - a technique for modeling a class of fuzzy objects," *ACM Trans. Graph.*, vol. 2, no. 2, pp. 91–108, 1983.

[44] W. T. Reeves and R. Blau, "Approximate and probabilistic algorithms for shading and rendering structured particle systems," in *SIGGRAPH '85: Proceedings of the 12th annual conference on Computer graphics and interactive techniques*, (New York, USA), pp. 313–322, ACM Press, 1985.

[45] D. K. McAllister, "Particle systems: Image–fountain.gif." `http://www.cs.unc.edu/~davemc/Particle/`.

[46] M. J. Harris and A. Lastra, "Real-time cloud rendering," in *EG 2001 Proceedings* (A. Chalmers and T.-M. Rhyne, eds.), vol. 20 of *3*, pp. 76–84, Blackwell Publishing, 2001.

[47] J. F. Blinn, "A generalization of algebraic surface drawing," *ACM Trans. Graph.*, vol. 1, no. 3, pp. 235–256, 1982.

[48] T. Nishita, Y. Dobashi, and E. Nakamae, "Display of clouds taking into account multiple anisotropic scattering and sky light," in *SIGGRAPH '96: Proceedings of the 23rd annual conference on Computer graphics and interactive techniques*, (New York, USA), pp. 379–386, ACM Press, 1996.

[49] T. Nishita and Y. Dobashi, "Modeling and rendering of various natural phenomena consisting of particles," in *CGI '01: Proceedings of the International Conference on Computer Graphics*, (Washington DC, USA), p. 149, IEEE Computer Society, 2001.

[50] W. T. F. Encyclopedia, "Image:metaballs.gif." `http://en.wikipedia.org/wiki/Image:Metaballs.gif`.

[51] A. Fournier, D. Fussell, and L. Carpenter, "Computer rendering of stochastic models," *Commun. ACM*, vol. 25, no. 6, pp. 371–384, 1982.

[52] G. Wyvill and A. Trotman, "Ray-tracing soft objects," in *CG International '90: Proceedings of the eighth international conference of the Computer Graphics Society on CG International '90: computer graphics around the world*, (New York, USA), pp. 469–476, Springer-Verlag New York, Inc., 1990.

[53] L. Westover, "Footprint evaluation for volume rendering," in *SIGGRAPH '90: Proceedings of the 17th annual conference on Computer graphics and interactive techniques*, (New York, USA), pp. 367–376, ACM Press, 1990.

[54] W. E. Lorensen and H. E. Cline, "Marching cubes: A high resolution 3d surface construction algorithm," in *SIGGRAPH '87: Proceedings of the 14th annual conference on Computer graphics and interactive techniques*, (New York, USA), pp. 163–169, ACM Press, 1987.

[55] W. T. F. Encyclopedia, "Image:texturemapping.png." `http://en.wikipedia.org/wiki/Image:TextureMapping.png`.

[56] K. Perlin, "An image synthesizer," in *SIGGRAPH '85: Proceedings of the 12th annual conference on Computer graphics and interactive techniques*, (New York, USA), pp. 287–296, ACM Press, 1985.

[57] K. Perlin, "Making noise. based on a talk presented at gdchardcore (dec 9, 1999).." `http://www.noisemachine.com/talk1/6.html`.

[58] K. Perlin and E. M. Hoffert, "Hypertexture," in *SIGGRAPH '89: Proceedings of the 16th annual conference on Computer graphics and interactive techniques*, (New York, USA), pp. 253–262, ACM Press, 1989.

[59] L. Carpenter, "The a–buffer, an antialiased hidden surface method," in *SIGGRAPH '84: Proceedings of the 11th annual conference on Computer graphics and interactive techniques*, (New York, USA), pp. 103–108, ACM Press, 1984.

[60] D. S. Ebert and R. E. Parent, "Rendering and animation of gaseous phenomena by combining fast volume and scanline a-buffer techniques," in *SIGGRAPH '90: Proceedings of the 17th annual conference on Computer graphics and interactive techniques*, (New York, USA), pp. 357–366, ACM Press, 1990.

[61] C. P. Robert and G. Casella, *Monte Carlo Statistical Methods*. New York, USA: Springer-Verlag, 1999.

[62] P. Blasi, B. L. Saẽc, and C. Schlick, "A rendering algorithm for discrete volume density objects," *Computer Graphics Forum (Eurographics '93)*, vol. 12, no. 3, pp. 201–210, 1993.

[63] H. W. Jensen, "Global illumination using photon maps," in *Proceedings of the eurographics workshop on Rendering techniques '96*, (London, UK), pp. 21–30, Springer-Verlag, 1996.

[64] G. B. Arfken, *Mathematical Methods for Physicists*. Academic Press, third ed., 1985.

[65] J. Stam, "Multiple scattering as a diffusion process," in *Rendering Techniques '95 (Proceedings of the Sixth Eurographics Workshop on Rendering)* (P. M. Hanrahan and W. Purgathofer, eds.), (New York, USA), pp. 41–50, Springer-Verlag, 1995.

[66] J. Kniss, S. Premože, C. Hansen, and D. Ebert, "Interactive translucent volume rendering and procedural modeling," in *VIS '02: Proceedings of the conference on Visualization '02*, (Washington DC, USA), pp. 109–116, IEEE Computer Society, 2002.

[67] G. Schussman and K.-L. Ma, "Anisotropic volume rendering for extremely dense, thin line data," in *VIS '04: Proceedings of the conference on Visualization '04*, (Washington DC, USA), pp. 107–114, IEEE Computer Society, 2004.

[68] N. L. Max, "Efficient light propagation for multiple anisotropic volume scattering," in *Fifth Eurographics Workshop on Rendering*, (Darmstadt, Germany), pp. 87–104, 1994.

[69] N. Max, "Optical models for direct volume rendering," *IEEE Transactions on Visualization and Computer Graphics*, vol. 1, no. 2, pp. 99–108, 1995.

[70] W. Hackbush, *Multi-grid Methods and Applications*. Berlin, Germany: Springer-Verlag, second ed., 2003.

[71] K. D. Lathrop, "Ray effects in discrete ordinates equations," in *Nuclear Science and Engineering*, vol. 32, pp. 357–369, 1968.

[72] H. E. Rushmeier and K. E. Torrance, "The zonal method for calculating light intensities in the presence of a participating medium," in *SIGGRAPH '87: Proceedings of the 14th annual conference on Computer graphics and interactive techniques*, (New York, USA), pp. 293–302, ACM Press, 1987.

[73] M. F. Cohen and D. P. Greenberg, "The hemi-cube: a radiosity solution for complex environments," in *SIGGRAPH '85: Proceedings of the 12th annual conference on Computer graphics and interactive techniques*, (New York, NY, USA), pp. 31–40, ACM Press, 1985.

[74] Y. Dobashi, K. Kaneda, H. Yamashita, T. Okita, and T. Nishita, "A simple, efficient method for realistic animation of clouds," in *SIGGRAPH '00: Proceedings of the 27th annual conference on Computer graphics and interactive techniques*, (New York, USA), pp. 19–28, ACM Press/Addison-Wesley Publishing Co., 2000.

[75] R. Miyazaki, S. Yoshida, T. Nishita, and Y. Dobashi, "A method for modeling clouds based on atmospheric fluid dynamics," in *PG '01: Proceedings of the 9th Pacific Conference on Computer Graphics and Applications*, (Washington DC, USA), p. 363, IEEE Computer Society, 2001.

[76] J. Schpok, J. Simons, D. S. Ebert, and C. Hansen, "A real-time cloud modeling, rendering, and animation system," in *SCA '03: Proceedings of the 2003 ACM SIGGRAPH/Eurographics symposium on Computer animation*, (Aire-la-Ville, Switzerland), pp. 160–166, Eurographics Association, 2003.

[77] A. Trembilski and A. Broßler, "Transparency for polygon based cloud rendering," in *SAC '02: Proceedings of the 2002 ACM symposium on Applied computing*, (New York, USA), pp. 785–790, ACM Press, 2002.

[78] H.-S. Liao, J.-H. Chuang, and C.-C. Lin, "Efficient rendering of dynamic clouds," in *VRCAI '04: Proceedings of the 2004 ACM SIGGRAPH international conference on Virtual Reality continuum and its applications in industry*, (New York, USA), pp. 19–25, ACM Press, 2004.

[79] K. Riley, D. Ebert, C. Hansen, and J. Levit, "Visually accurate multi-field weather visualization," in *VIS '03: Proceedings of the 14th IEEE Visualization 2003 (VIS'03)*, (Washington DC, USA), p. 37, IEEE Computer Society, 2003.

[80] S. Premože, "Analytic light transport approximations for volumetric materials," in *PG '02: Proceedings of the 10th Pacific Conference on Computer Graphics and Applications*, (Washington DC, USA), p. 48, IEEE Computer Society, 2002.

[81] N. L. Max, G. Schussman, R. Miyazaki, K. Iwasaki, and T. Nishita, "Diffusion and multiple anisotropic scattering for global illumination in clouds.," in *Journal of WSCG*, vol. 12 of *3*, pp. 277–284, 2004.

[82] J. Haber, M. Magnor, and H.-P. Seidel, "Physically-based simulation of twilight phenomena," *ACM Trans. Graph.*, vol. 24, no. 4, pp. 1353–1373, 2005.

[83] D. Q. Nguyen, R. Fedkiw, and H. W. Jensen, "Physically based modeling and animation of fire," in *SIGGRAPH '02: Proceedings of the 29th annual conference on Computer graphics and interactive techniques*, (New York, USA), pp. 721–728, ACM Press, 2002.

[84] J. Arvo, "Transfer functions in global illumination," in *ACM SIGGRAPH '93 Course Notes - Global Illumination*, pp. 1–28, ACM Press, 1993.

[85] K. Engel, M. Hadwiger, J. M. Kniss, A. E. Lefohn, C. R. Salama, and D. Weiskopf, "Real-time volume graphics," in *GRAPH '04: Proceedings of the conference on SIGGRAPH 2004 course notes*, (New York, USA), p. 266, ACM Press, 2004.

[86] H. W. Jensen, *Realistic Image Synthesis Using Photon Mapping*. AK Peters, 2001.

[87] M. Phar and G. Humphreys, *Physically Based Rendering: From theory to implementation*. Morgan Kaufmann, 2004.

[88] J. Arvo, P. Dutré, A. Keller, H. W. Jensen, A. Owen, M. Pharr, and P. Shirley, "Monte carlo ray tracing," in *SIGGRAPH '03: ACM SIGGRAPH 2003 Course Notes*, (New York, USA), p. 171, ACM Press, 2003.

[89] P. Shirley and C. Wang, "Distribution ray tracing: Theory and practice," in *Third Eurographics Workshop on Rendering*, (Bristol, UK), pp. 33–43, 1992.

[90] P. Shirley, C. Wang, and K. Zimmerman, "Monte carlo techniques for direct lighting calculations," *ACM Trans. Graph.*, vol. 15, no. 1, pp. 1–36, 1996.

[91] E. P. Lafortune and Y. D. Willems, "Rendering participating media with bidirectional path tracing," in *Proceedings of the eurographics workshop on Rendering techniques '96*, (London, UK), pp. 91–100, Springer-Verlag, 1996.

[92] P. Dutré, H. W. Jensen, J. Arvo, K. Bala, P. Bekaert, S. Marschner, and M. Pharr, "State of the art in monte carlo global illumination," in *SIGGRAPH '04: ACM SIGGRAPH 2004 Course Notes*, (New York, USA), p. 5, ACM Press, 2004.

[93] B. Csébfalvi and L. Szirmay-Kalos, "Monte carlo volume rendering," in *VIS '03: Proceedings of the 14th IEEE Visualization 2003 (VIS'03)*, (Washington DC, USA), p. 59, IEEE Computer Society, 2003.

[94] H. W. Jensen, "Importance driven path tracing using the photon map," in *Rendering Techniques '95 (Proceedings of the Sixth Eurographics Workshop on Rendering)* (P. M. Hanrahan and W. Purgathofer, eds.), (New York, USA), pp. 326–335, Springer-Verlag, 1995.

[95] H. W. Jensen and N. J. Christensen, "Efficiently rendering shadows using the photon map," in *Edugraphics + Compugraphics Proceedings* (H. P. Santo, ed.), (Massama, Portugal), pp. 285–291, GRASP – Graphic Science Promotions & Publications, December 1995.

[96] J. R. Arvo, "Backward ray tracing," in *ACM SIGGRAPH '86 Course Notes - Developments in Ray Tracing*, vol. 12, 1986.

[97] J. Arvo and D. Kirk, "Particle transport and image synthesis," in *SIGGRAPH '90: Proceedings of the 17th annual conference on Computer graphics and interactive techniques*, (New York, USA), pp. 63–66, ACM Press, 1990.

[98] M. B. Nielsen, "Course notes for simulating smoke and water in cg: Ray marching," April 2006.

[99] I. Peter and G. Pietrek, "Importance driven construction of photon maps," in *Rendering Techniques '98 (Proceedings of Eurographics Rendering Workshop '98)* (G. Drettakis and N. Max, eds.), (New York, USA), pp. 269–280, Springer Wien, 1998.

[100] M. Matsumoto and T. Nishimura, "Mersenne twister: a 623-dimensionally equidistributed uniform pseudo-random number generator," *ACM Trans. Model. Comput. Simul.*, vol. 8, no. 1, pp. 3–30, 1998.

[101] J. Naylor, *Out of the Blue: A 24-hour Skywatcher's Guide*. Cambridge University Press, 2002.

[102] T. Malzbender, D. Gelb, and H. Wolters, "Polynomial texture maps," in *SIGGRAPH '01: Proc. 28th annual conference on Computer graphics and interactive techniques*, (New York, USA), pp. 519–528, ACM Press, 2001.

[103] G. I. Marchuk, G. A. Mikhailov, M. A. Nazaraliev, R. A. Dacbinjan, B. A. Kargin, and B. S. Elepov, "Monte carlo methods in atmospheric optics.," *Applied Optics*, vol. 20, March 1981.

[104] M. A. Penna and R. R. Patterson, *Projective Geometry And Its Applications To Computer Graphics*. Prentice-Hall, 1986.