Realistic fire rendering

Garoe Dorta-Perez

CM50170 - Research project

Unit Leader: Dr Darren Cosker

University of Bath

August 2015

Signature of Author . . . .

Garoe Dorta-Perez

Contents

Chapter 1

Introduction

"*A case that can be made for fire being, next to the life process, the most complex of phenomena to understand.*" - Hoyt Hottle

For centuries humans has been attracted to fire due to its attractive presence and its dangerous nature. Understanding and simulating combustion phenomena has many applications, such as in the film and computer games industries, where it is widely use in visual effects; or in the engineering community, where the modelling of combustion in engines or fire safety evaluations are frequently demanded. Computer generated examples include a particle-based technique by [Ree83] which was used in the Star Trek II film, parametric curves were used to drive the flames [LF02] in the film Shrek and, the more recent work of [HG09] based on 2D screen projection for the film Harry Potter and the Deathly Hallows. In these and in many other applications, using real flames is an expensive and hazardous endeavour.

Fire can be modelled as a fluid, however its behaviour more complex, due to its multiphase flow, chemical reactions and radiative heat transport, than other fluids, such as water or smoke, which the computer graphics community have intensively researched. As a result of the aforementioned complexity and the interdisciplinary nature of the problem, fire simulation is still an open problem in computer graphics.

A great deal of work done in the area has sacrificed complexity for interactiveness, therefore producing simplified models which hope to deceit the observer by exploiting the chaotic essence in fire motion. Nevertheless, physically-based simulations incorporate the intrinsic processes that occur in a combustion scenario:

Flame motion:

Fuel erosion: when the fuel reaches a certain temperature, it is vaporized into a gaseous state, which rises under the influence of buoyancy.

Black body radiation: the chemical species present in the fuel and the byproducts of the combustions emit energy in various wavelengths.

In order to be able to produce a realistic result, all of the preceding characteristics have to be taken into consideration. In this report a short review in the field of fire simulation and rendering is presented, a state of the art physically based rendering model by [PP06] is discussed in detail, and an implementation of the given model in Maya® is outlined.

TODO Add some fancy pictures from real, film and videogames flames and explosions
TODO Explain everything in more detail and go from slower from general to technical

Chapter 2

Previous Work

In order to display a realistic fire scene in a computer generated world two differentiated stages are needed. Firstly, the fire dynamics have to be collected, this can either be done through a data capture session or simulated using a fluid solvers. Secondly, the previously gathered data is to be visualized on the screen using some rendering technique. We refer the interested readers to the more detailed survey on the topic, which has been recently presented by [HGH14].

2.1 Simulation

Particle-based methods were the first approach to simulate the visual animation of fire. A number of particles are emitted from certain locations, each particle has a set of attributes such as shape, velocity, color or lifetime. The first model with particle systems was presented by [Ree83], the particles speed and colour were perturbed with a Gaussian noise at each time step, and the colour was subject to an additional linear perturbation on its lifetime. Two particle systems were used in a hierarchy, one would control fire spread and the other a single explosion effect. An extension was proposed by [PP94], the authors modified the particle system such that each particle shape would be defined by a series of non-overlapping coplanar triangles. The transparency of would increase towards the outer vertices, thus providing an improved visual effect.

Noise-based methods focus on synthesizing the high fluctuation present in fire procedurally. The objective is to approximate the turbulence present in fire with an appropriate statistical model. Using a variation of Perlin noise, [Per85] presented images of a corona of flames. However, the method is limited to 2D, where the color is a combination of non-linear arbitrary functions. This work was extended by [PH89] to 3D, where they use volumetric rendering to achieve improved results.

Geometry skeleton

Data driven

Physically based simulate the fire combustion processes, including flame propagation or the chemical reactions that convert fuel into gaseous products. Incompressible flow equations were used by [SF95] to drive a fire simulation. Given initial fuel conditions, the fire spread is advected on a grid using an advection-diffusion type equation. Building on the work on a semi-Lagrangian fluid solver of [Sta99], a model which includes gaseous fuel and gaseous byproducts was proposed by [NFJ02]. In order to include the characteristics of the noise-based methods, [HSF07] combined the previous model with a set of third-order equations from detonation shock dynamics presented by [YS96]. As with the noise-based methods, this addition is visually attractive, yet it is not physically based. Capitalizing on the recent advances in GPUs parallel processing power, [HG09] proposed a fixed camera model. Particle properties are computed on a three-dimensional coarse grid, which are then projected into several view dependant two-dimensional slices. The authors' model is based on the assumption that fine variations, which are perpendicular to the projection plane, are not individually visible and, they do not affect significantly the overall flow.

Other effects directly related to fire have also been explored. [FOA03] presented a model to simulate suspended particles during explosions. An incompressible fluid model drives the motion of air and hot gases, and the suspended particles follow the their movements. Sound is a important factor to increase the believability of a finished fire animation. [CJ11] proposed a method to automatically generate plausible noise given for a given fire simulation. Low frequency sound is estimated using a physical model whose inputs are the flame front and heat release. A data driven sound synthesis approach, based on the work by [WL00], is applied to generate the high frequency content.

Erosion???

2.2 Rendering

Rendering fire is more challenging than rendering other type of participating media, the main reason being that fire is a light-emitting source. The luminescence radiated by a flame is generated by black body radiation due to the high temperature of the particles present during the combustion. In order to render fire realistically, light absorption and scattering in the media, including air, has to be taken into consideration.

2.2.1 Raster-Based

Raster-based techniques sacrifice quality in the interest of interactive frame rates. Some form of texture mapping is usually practised, usually only the surface of the flame is considered, and the illumination of the scene is approximated base on some parameter which can be easily computed such as a as well, e.g. the fire height.

[Ree83] applied a linear colour assignment to each particle in their simulation based on their lifetime. [LKMD01] applied the same technique to render fires on mesh surfaces, where a particle would begin with a light yellow colour, evolve to red and finish in black at the end of their lifetime. [LF02] presented a technique were a based flame picture is mapped onto the two-dimensional flame profile with a base colour. For each particle an intermediate emitting value is computed, and the final colour is super-sampled profile from an approximation of the cross-sectional area of the flame, as it would appear from the camera. [ZCM11] proposed a method were fire particles and their attributes are first projected onto a set of slicing planes, which are orthogonal to the camera direction. The planes are then blended to the screen in back-to-front order, and a one-dimensional colour texture is used as transfer function to convert flow attributes to colours and opacities.

2.2.2 Ray-Tracing-Based

Volume ray-tracing techniques offer astonishing results, however the associated computational costs are considerable. Rays are shot from the view plane and evaluated at small increments; the total radiance at origin of the ray is computed by integrating the radiance at each step size. A further drawback for ray tracing techniques, in comparison to raster-based methods, is the lack of a standard ray-tracing pipeline.

[RHC95] presented a method to perform accurate ray casting on sparse measured data. The fire was modelled as a series of stacked cylindrical rings, where each ring has with uniform properties. The total radiance at each point is integrated using a MonteCarlo method, summing up the measured irradiances at sample locations. [NFJ02] proposed a ray marching technique to solve the Radiative Transport Equation, see Chapter 3. The emitted light is computed using Planck's formula of black body radiation, the light scattering in the media and visual adaptation to the fire spectra are modelled. [FOA03] also included black body radiation in their animation of fire with suspended particles, however the mapping to RGB was manually adjusted to match the images of real explosions. Direct illumination shadows were computed using deep shadow maps [LV00], while scattering and illumination by other objects in the scene used the technique proposed by [JB02]. An extension to [NFJ02] was presented by [PP06], the authors' model has physically-based abortion, emission and scattering properties. The spectroscopy characteristics of different fuels are achieved by modelling the electronic transitions between states in the molecules. Non-linear trajectories of light in the medium due to light refraction effects are included as well. In order to minimize the effects induced by the limitations of the RGB colour space, the visual adaptation process is presented as a post-processing effect. [HG09] proposed a rendering method whose main objective was user-friedliness for artists. Using the fixed camera slices described in Section 2.1, the authors' perform a simple volume rendering to join them in a single image. Black body radiation is used for the light, the images are motion-blurred with a filter based on the velocities in the slices, and the heat distortion is added as post-processing user defined filter.

Chapter 3

Methodology

Not only explain in more detail every equation, what it means, comparison with other papers and ideally improvements or were it fails at least. Should I explain what is and how to do ray marching? What is the spectrum? How to integrate it to RGB coefficients? Add all the values for the constants here or in implementation details??

3.1 Radiative Transport Equation

The Radiative Transport Equation (RTE) models the variation of spectral radiance in the medium $L(\lambda, \mathbf{x}, \boldsymbol{\omega})$, where λ is a given wavelength in m, $\mathbf{x}$ is the point of interest in space, and $\boldsymbol{\omega}$ is a vector that points towards the viewing direction. The RTE is defined as

$$\begin{aligned}(\omega\nabla)L(\lambda, \mathbf{x}, \boldsymbol{\omega}) = &- \sigma_a(\lambda, \mathbf{x})L(\lambda, \mathbf{x}, \boldsymbol{\omega}) + \sigma_a(\lambda, \mathbf{x})L_e(\lambda, \mathbf{x}, \boldsymbol{\omega}) \\ &- \sigma_s(\lambda, \mathbf{x})L(\lambda, \mathbf{x}, \boldsymbol{\omega}) + \sigma_s(\lambda, \mathbf{x})L_i(\lambda, \mathbf{x}, \boldsymbol{\omega}),\end{aligned} \tag{3.1}$$

where L_i is defined by

$$L_i(\lambda, \mathbf{x}, \boldsymbol{\omega}) = \int_{4\pi} L(\lambda, \mathbf{x}, \boldsymbol{\omega}_i)\Phi(\lambda, \boldsymbol{\omega}, \boldsymbol{\omega}_i)d\boldsymbol{\omega}_i, \tag{3.2}$$

where σ_a is an absorption coefficient, σ_s is a scattering coefficient, L_e is the emitted spectral radiance at the point, L_i is the in-scattering radiance, Φ is a scattering phase function and ω_i is a scattering sampling direction. In order to get an analytical solution to the aforementioned equation, the properties of the medium are assumed to be homogeneous over a small segment $||\Delta x||$ in space,

$$L(\lambda, \mathbf{x} + \Delta\mathbf{x}, \boldsymbol{\omega}) = e^{-\sigma_t(\lambda,\mathbf{x})\|\Delta x\|} L(\lambda, \mathbf{x}, \boldsymbol{\omega}) + \\ (1 - e^{-\sigma_t(\lambda,\mathbf{x})\|\Delta x\|}) \frac{\sigma_a(\lambda, \mathbf{x}) L_e(\lambda, \mathbf{x}, \boldsymbol{\omega}) + \sigma_s(\lambda, \mathbf{x}) L_i(\lambda, \mathbf{x}, \boldsymbol{\omega})}{\sigma_t(\lambda, \mathbf{x})}, \tag{3.3}$$

where $\sigma_t = \sigma_a + \sigma_s$ is the extinction coefficient.

3.2 Scattering

The scattering phase function is defined by

$$\Phi(\lambda, \boldsymbol{\omega}, \boldsymbol{\omega}_i) = \frac{1 - g(\lambda)^2}{4\pi(1 + g(\lambda)^2 - 2g(\lambda)\boldsymbol{\omega}\boldsymbol{\omega}_i)^{\frac{3}{2}}}, \tag{3.4}$$

where g can be a function of the wavelength, although in most cases is chosen to be constant. The value of g must be in the range $(-1, 1)$, where $g < 0$ corresponds to backwards scattering, $g = 0$ to isotropic scattering, and $g > 0$ to forwards scattering.

3.3 Soot Absorption

The spectral absorption coefficient of soot is defined as

$$\sigma_a(\lambda, \mathbf{x}) = \frac{48N(\mathbf{x})\pi R^3 nm}{\lambda^{\alpha(\lambda)}((n^2 - m^2 + 2)^2 + 4n^2m^2)}, \tag{3.5}$$

where $N(\mathbf{x})$ is the number density, density per unit volume, R is the radius of a soot particle, n, m and $\alpha(\lambda) = 1.39$ are optical constants for different types of soot. In Table 3.1, values for the optical constants n and m are provided for several materials, the data was obtained from [DS69]. The radius of soot particles was determined in the range $R \in \{50\ \text{Å} \ldots 800\ \text{Å}\}$ by [DS69]. Since our data is defined with soot densities, we have chosen the radius to be the mean value $R = 425 \times 10^{-10}$ metres.

3.4 Black Body Radiation

Planck's equation for black body radiation characterizes the electromagnetic radiation emitted by a black body in thermal equilibrium at a given temperature T. The equation is defined as

Table 3.1: Absorption constants for propane and acetylene, wavelengths are in nanometres.

		Wavelengths			
		435.8	450	550	650
Propane	n	1.57	1.56	1.57	1.56
	nm	0.46	0.5	0.53	0.52
Acetylene	n	1.56	1.56	1.56	1.57
	nm	0.46	0.48	0.46	0.44

$$B_\lambda(T, \lambda, \eta) = \frac{2hc^2}{\lambda^5(e^{\frac{hc}{\lambda kT}} - 1)}, \tag{3.6}$$

where $h = 6.62606957 \times 10^{-34}$ J/s is Planck's constant, $k = 1.3806488 \times 10^{-23}$ J/K is Boltzmann constant and $c = c_0/\eta$ is the speed of light in the current medium, where $c_0 = 299792458$ m/s is the speed of light in a vacuum and η is the refraction index of the medium. The refraction index η varies across the medium, the procedure to compute will be described in Section 3.6.

3.5 Emission From Chemical Species

The emission and absorption coefficients associated with a given spectral frequency can be computed as

$$\sigma_a = \frac{\phi(\lambda) N_2 A_{21} \lambda^4 (e^{\frac{hc}{\lambda kT}} - 1)}{8\pi c}, \tag{3.7}$$

$$j_\lambda = \sigma_a B_\lambda(T, \lambda, n), \tag{3.8}$$

where $\phi(\lambda)$ is the normalized spectral line, N_2 is the number density of elements, A_{21} is an Einstein coefficient measuring the transition probabilities of an spontaneous emission, and j_λ is the emitted radiance for the current chemical component.

3.6 Refraction

Assuming a negligible reflection index, the refraction angles for the rays can be easily computed using Snell's law

$$\frac{\sin\theta_1}{\sin\theta_2} = \frac{\eta_2}{\eta_1}, \tag{3.9}$$

where θ_1 is the incident angle, θ_2 is the refracted angle, η_1 is the index of refraction of the media which the ray is coming from and η_2 is the index of refraction of the media which the ray is going to.

[Cid96] proposed a method to compute the refractive indices of air

$$\eta_{air} = 1 + \frac{\rho_a \eta_{axs}}{\rho_{axs}} + \frac{\rho_w \eta_{ws}}{\rho_{ws}}, \tag{3.10}$$

where ρ_{axs} is the density of dry air, $\rho_{ws} = 0.00985938$ is the density of pure water vapour, ρ_a and ρ_w are the equivalent quantities for dry air, η_{axs} is the refractive index for CO_2 and η_{ws} is the refractive index for water vapour.

$$\eta_{axs} = \eta_{as}\left(1 + 0.534 \times 10^{-6}\left(x_c - 450\right)\right), \tag{3.11}$$

$$\eta_{as} = 10^{-8}\left(\frac{k_1}{k_0 - \lambda} + \frac{k_3}{k_2 - \lambda}\right), \tag{3.12}$$

where $x_c \in \{300 \ldots 450\}$ is the proportion in ppm of CO_2, in our case $x_c = 450$, $k_0 = 238.0185\,\mu m^{-2}$, $k_1 = 5792105\,\mu m^{-2}$, $k_2 = 57.362\,\mu m^{-2}$, $k_3 = 167917\,\mu m^{-2}$ and λ is the wave number (reciprocal of the vacuum wavelength).

$$\eta_{ws} = 1.022 \times 10^{-8}\left(w_0 + w_1\lambda + w_1\lambda^2 + w_3\lambda^3\right), \tag{3.13}$$

where $w_0 = 295.235\,\mu m^{-2}$, $w_1 = 2.6422\,\mu m^{-2}$, $w_2 = -0.03238\,\mu m^{-2}$ and $w_3 = 0.004028\,\mu m^{-2}$

The ρ densities are computed as follows

$$\rho = \frac{p m_a}{z r T}\left(1 - x_w\left(1 - \frac{m_w}{m_a}\right)\right), \tag{3.14}$$

where p is the pressure in pascals, $1\,atm = 101325\,Pa$ in our case, $r = 8.31451\ Jmol^{-1}K^{-1}$, $m_w = 0.018015\ kg/mol$,

$$m_a = 0.0289635 + 12.011 \times 10^{-8}(x_c - 400)), \tag{3.15}$$

$$z = 1 - \frac{p}{T}\left(a_0 + a_1 t + a_2 t^2 + (b_0 + b_1 t)\,x_w + (c_0 + c_1 t)\,x_w^2\right) + \left(\frac{p}{T}\right)^2\left(d + e x_w^2\right), \tag{3.16}$$

where $a_0 = 1.58123\ \times 10^{-6} KPa^{-1}$, $a_1 = -2.9331\ \times 10^{-8} Pa^{-1}$, $b_0 = 5.707\ \times 10^{-6} KPa^{-1}$, $b_1 = -2.051\ \times 10^{-8} Pa^{-1}$, $c_0 = 1.9898\ \times 10^{-4} KPa^{-1}$, $c_1 = -2.376\ \times 10^{-6} KPa^{-1}$, $d = 1.83\ \times 10^{-11} K^2 Pa^{-2}$ and $e = -0.765\ \times 10^{-8} K^2 Pa^{-2}$.

$$T = t + 273.15, \tag{3.17}$$

$$x_w = \frac{fhv}{p}, \tag{3.18}$$

$$f = \alpha + \beta p + \gamma t^2, \tag{3.19}$$

where $\alpha = 1.00062, \beta = 3.14 \times 10^{-8} Pa^{-1}$ and $\gamma = 5.6 \times 10^{-7}\,°\text{C}^{-2}$, and $h \in \{0 \dots 1\}$ is the air relative fractional humidity, 0 for dry air and given by the user for moist air.

The pressure p can also be computed using the ideal gas law

$$pV = \frac{NRT}{V} = nT, \tag{3.20}$$

where N is the number of molecules, V is the total volume of the gas, T is the temperature of the gas, n is the number density, $R = kN_a$, where $k = 1.3806488 \times 10^{-23}\ J/K$ is the Boltzmann constant and $N_a = 6.02214129 \times 10^{23}\ mol^{-1}$ is Avogadro constant.

3.6.1 Davis' Saturation Vapour Pressure

In [Cid96] paper, the author used the method proposed by [Dav92] to compute the saturation of vapour pressure

$$v = e^{AT^2 + BT + C + D/T}, \tag{3.21}$$

where $A = 1.2378847 \times 10^{-5} K^{-2}$, $B = -1.9121316 \times 10^{-2} K^{-1}$, $C = 33.93711047$ and $D = -6.3431645 \times 10^{3} K$. This equation is relatively easy to compute, however incorrect results will be obtained for temperatures below 0 °C. As we are concerned with fire rendering, which entails high temperatures, this method was chosen for its simplicity.

3.6.2 Huang's Saturation Vapour Pressure

The International Association for the Properties of Water and Steam (IAPWS) adopted an alternative technique to [Dav92], which was proposed by [Hua98]. The more recent method will address the drawbacks of the previous one, giving reasonable results for temperatures below 0 °C, nevertheless, the generality comes with increased complexity in the equations.

$$v = 10^6 \left(\frac{2C}{X}\right)^4, \tag{3.22}$$

$$X = -B + \left(B^2 - 4AC\right)^{\frac{1}{2}}, \tag{3.23}$$

$$\begin{bmatrix} A \\ B \\ C \end{bmatrix} = \begin{bmatrix} 1 & K_1 & K_2 \\ K_3 & K_4 & K_5 \\ K_6 & K_7 & K_8 \end{bmatrix} \begin{bmatrix} \Omega^2 \\ \Omega \\ 1 \end{bmatrix}, \tag{3.24}$$

$$\Omega = T + \frac{K_9}{T - K_{10}}, \tag{3.25}$$

where $K_1 = 1.16705214528 \times 10^3$, $K_2 = -7.24213167032 \times 10^5$, $K_3 = -1.70738469401 \times 10$, $K_4 = 1.20208247025 \times 10^4$, $K_5 = -3.23255503223 \times 10^6$, $K_6 = 1.49151086135 \times 10$, $K_7 = -4.82326573616 \times 10^3$, $K_8 = 4.05113405421 \times 10^5$, $K_9 = -2.38555575678 \times 10^{-1}$, and $K_{10} = 6.50175348448 \times 10^2$.

3.6.3 Ciddor's Method Summary

The inputs of Ciddor's technique are a wavelength λ_0, a temperature t, a pressure p, a relative humidity value h and a CO_2 concentration x_c. The steps to compute the index of refraction are listed below

1. Precompute $z_a = 0.9995922115$, from Equation 3.16 with $t = 20$ °C, $p = 101.325$ Pa and $x_w = 0$.
2. Find $\lambda = 1/\lambda_0^2$, and T with Equation 3.17.
3. Find v with the method described in Section 3.6.1 or in Section 3.6.2.
4. Find x_w using Equations 3.19 and 3.18.
5. Find η_{as} using Equation 3.12.
6. Find η_{ws} using Equation 3.13.
7. Find m_a using Equation 3.15.
8. Find η_{axs} using Equation 3.11.
9. Find z_m using Equation 3.16.
10. Find $\rho_{axs} = (p_0 m_a)/(z_a r T_0)$, where $p_0 = 101325$, and $T_0 = 288.15$, using Equation 3.14.
11. Find $\rho_w = (x_w p m_w)/(z_m r T)$ using Equation 3.14.
12. Find $\rho_a = ((1 - x_w) p m_a)/(z_m r T)$ using Equation 3.14.
13. Find the index of refraction η_{air} using Equation 3.10.

3.7 Visual Adaptation

The human eye presents a non-linear response to incident radiance L, the reaction has been modelled with certain success with the simple function

$$R(L, \tau) = \frac{L}{L + \tau}, \tag{3.26}$$

where τ is a non-linear adaptation state. This state is determined by the visual system to maximize the perception of features for a given scene.

Chapter 4

Implementation details

4.1 Application Overview

Explain how to use the shaders and the commands, add images of the scenes in Maya

The model chosen for our data is that of a three-dimensional voxel dataset, a cube in space is be uniformly divided and we have a data value for each voxel. This data is either a soot density estimate for the *fire volume shader* or a temperature estimate for the *fire light shader*. Two file formats are supported, dense float voxel data is ASCII or sparse binary RGBA values, as shown in Figure 4-1. In the ASCII format, the file first line are three integers, separated by spaces, declaring the width, height and depth of the data, and *whd* lines with a single floating point value. The binary sparse format starts with a 4 byte integer declaring how many data points are in the file, each voxel has its coordinates x, y, z as 4 bytes integers and a rgb alpha colour, where each channel is an 8 bytes floating point double value.

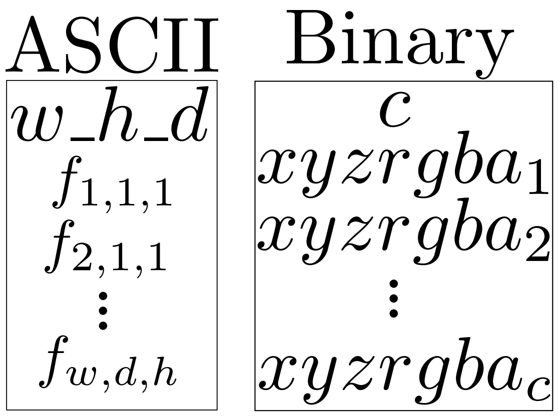

Figure 4-1: Suported file formats.

4.2 Mental Ray® Shading Approach

How mental ray shoots rays around and what is its paradigm.

Mental Ray® approach to solving the rendering equation is based on path tracing, as shown in Figure 4-2, for each pixel in the camera view, an eye ray will be shot in the scene. On an intersection with an object in the scene, its material shader will be called, this shader will shoot a light ray for each light in the scene, which in effects calls the light shader of the given light. In order to compute the irradiance at the intersection point, the light shader will probably trace a shadow ray from the light to the intersection point. Eventually, the material shader will compute the final colour with the information received from the light shader.

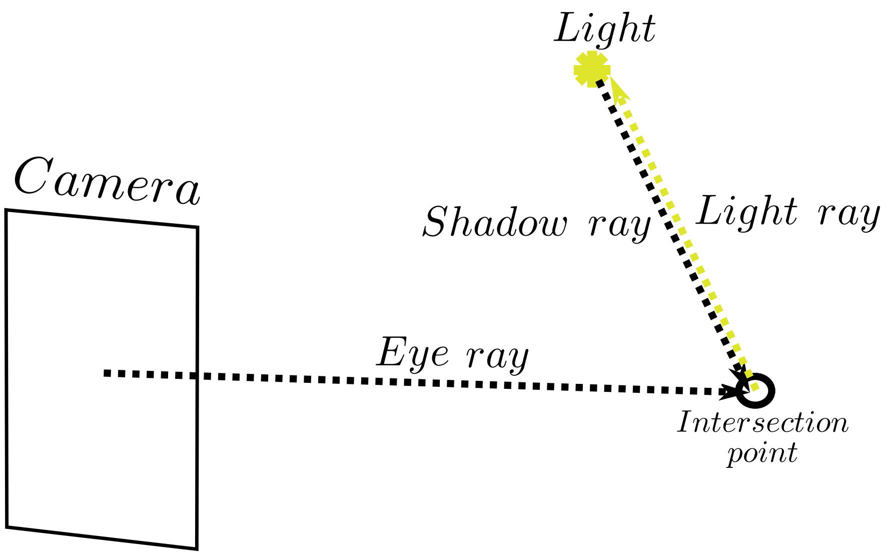

Figure 4-2: Mental Ray® simple ray casting example.

Equation 3.3 provides a radiance value for the next march increment, however we want to compute the value at the ray intersection with the volume, i.e. at the ray origin. So we need to rewrite the equation as

$$\begin{aligned} L(\lambda, \mathbf{x}, \omega) = {} & e^{\sigma_t(\lambda,\mathbf{x})\|\Delta x\|} L(\lambda, x + \Delta\mathbf{x}, \omega) + \\ & (1 - e^{\sigma_t(\lambda,\mathbf{x})\|\Delta x\|}) \frac{\sigma_a(\lambda, \mathbf{x}) L_e(\lambda, \mathbf{x}, \omega) + \sigma_s(\lambda, \mathbf{x}) L_i(\lambda, \mathbf{x}, \omega)}{\sigma_t(\lambda, \mathbf{x})}. \end{aligned} \tag{4.1}$$

4.3 Shaders Internals

Talk about things like which parts are written in parallel code, instance support, shader internal memory, sparse data, how the software escalates, memory consumption, Maya integration, spectrum to rgb integration, maybe more details about units for black body radiation and everything that was not explained before

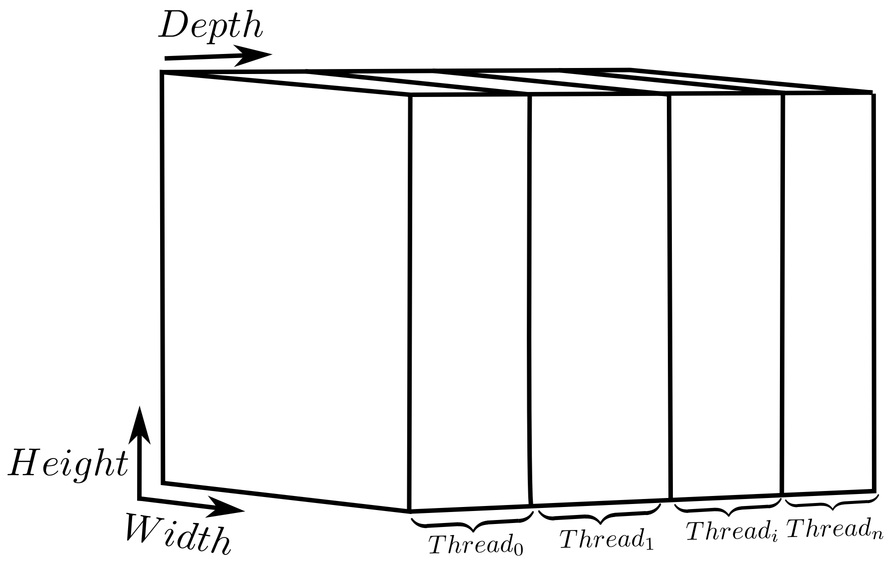

Figure 4-3: Voxel Dataset division of coefficients computation in threads.

In order to speed up the render time, all spectrum related computations are integrated to RGB coefficients. Our spectrum class has a number of samples that can be set a compile time, in general it is set to 30 samples. However, in certain cases a lower number is used, e.g. when computing the soot absorption coefficients, since we have values for the constants for only four wavelengths, using more than four samples in the spectrum would not improve the results. The integration is performed first to the XYZ colour space using a Rienmann sum. Given a Spectral Power Distribution (SPD) $S(\lambda)$, the x, y, z coefficients for $S(\lambda)$ in the XYZ space are defined with respect to spectral matching curves $X(\lambda)$, $Y(\lambda)$ and $Z(\lambda)$, such that

$$\begin{aligned} x &= \frac{1}{\int Y(\lambda)d\lambda} \int_\lambda S(\lambda)X(\lambda)d\lambda, \\ y &= \frac{1}{\int Y(\lambda)d\lambda} \int_\lambda S(\lambda)Y(\lambda)d\lambda, \\ z &= \frac{1}{\int Y(\lambda)d\lambda} \int_\lambda S(\lambda)Z(\lambda)d\lambda. \end{aligned} \tag{4.2}$$

The Rienmann sum for each coefficient is computed as

$$
\begin{aligned}
x &\approx \frac{1}{\int Y(\lambda)d\lambda} \sum_i X_i c_i, \\
y &\approx \frac{1}{\int Y(\lambda)d\lambda} \sum_i Y_i c_i, \\
z &\approx \frac{1}{\int Y(\lambda)d\lambda} \sum_i Z_i c_i,
\end{aligned}
\tag{4.3}
$$

where c_i is the i^{th} coefficient in the SPD, X_i is area for the the corresponding sample range the $X(\lambda)$ curve and equivalently for Y_i and Z_i. Note that $1/\left(\int Y(\lambda)d\lambda\right)$, X_i, Y_i and Z_i are constants and they can be precomputed for efficiency. For the Rienmann sum, the area under the samples is approximated using piecewise linear interpolation. The conversion from XYZ to RGB space is performed using

$$
\begin{bmatrix} r \\ g \\ b \end{bmatrix} = \begin{bmatrix} \int R(\lambda)X(\lambda)d\lambda & \int R(\lambda)Y(\lambda)d\lambda & \int R(\lambda)Z(\lambda)d\lambda \\ \int G(\lambda)X(\lambda)d\lambda & \int G(\lambda)Y(\lambda)d\lambda & \int G(\lambda)Z(\lambda)d\lambda \\ \int B(\lambda)X(\lambda)d\lambda & \int B(\lambda)Y(\lambda)d\lambda & \int B(\lambda)Z(\lambda)d\lambda \end{bmatrix} \begin{bmatrix} x \\ y \\ z \end{bmatrix},
\tag{4.4}
$$

where $R(\lambda)$, $G(\lambda)$, $B(\lambda)$ are the spectral curves for the red, green and blue colours respectively. Note that all the factors in the transformation matrix are constants and they can be precomputed for efficiency.

4.3.1 Fire Volume Shader

In order to achieve smoother rendering results, a trilinear interpolation is computed for each step in the ray-marching algorithm.

4.3.2 Fire Light Shader

4.3.3 Voxel Dataset Shader

Chapter 5

Results

Images everywhere and proper analysis of what the user is seeing, what is failing and why. Include default Maya fire effect, pictures of fire from the papers mentioned in the previous work section and a historical overview on the improvements in the shaders.

Chapter 6

Conclusions and Future work

Bibliography

[Cid96] Ciddor P. E.: Refractive index of air: new equations for the visible and near infrared. *Appl. Opt. 35*, 9 (Mar 1996), 1566–1573.

[CJ11] Chadwick J. N., James D. L.: Animating fire with sound. *ACM Trans. Graph. 30*, 4 (July 2011), 84:1–84:8.

[Dav92] Davis R. S.: Equation for the determination of the density of moist air (1981/91). *Metrologia 29*, 1 (1992), 67.

[DS69] Dalzell W. H., Sarofim A. F.: Optical constants of soot and their application to heat-flux calculations. *ASME Journal of Heat Transfer 91*, 1 (1969), 100–104.

[FOA03] Feldman B. E., O'Brien J. F., Arikan O.: Animating suspended particle explosions. *ACM Trans. Graph. 22*, 3 (July 2003), 708–715.

[HG09] Horvath C., Geiger W.: Directable, high-resolution simulation of fire on the gpu. *ACM Trans. Graph. 28*, 3 (July 2009), 41:1–41:8.

[HGH14] Huang Z., Gong G., Han L.: Physically-based modeling, simulation and rendering of fire for computer animation. *Multimedia Tools and Applications 71*, 3 (2014), 1283–1309.

[HSF07] Hong J.-M., Shinar T., Fedkiw R.: Wrinkled flames and cellular patterns. *ACM Trans. Graph. 26*, 3 (July 2007).

[Hua98] Huang P. H.: New equations for water vapor pressure in the temperature range-100 c to 100 c for use with the 1997 nist/asme steam tables. In *Proc, of the 3rd International Symposium on Humidity and Moisture (Teddington: NPL)* (1998), vol. 1, pp. 69–76.

[JB02] Jensen H. W., Buhler J.: A rapid hierarchical rendering technique for translucent materials. *ACM Trans. Graph. 21*, 3 (July 2002), 576–581.

[LF02] Lamorlette A., Foster N.: Structural modeling of flames for a production environment. *ACM Trans. Graph. 21*, 3 (July 2002), 729–735.

[LKMD01] Lee H., Kim L., Meyer M., Desbrun M.: Meshes on fire. In *Computer Animation and Simulation 2001*, Magnenat-Thalmann N., Thalmann D., (Eds.), Eurographics. Springer Vienna, 2001, pp. 75–84.

[LV00] LOKOVIC T., VEACH E.: Deep shadow maps. In *Proceedings of the 27th Annual Conference on Computer Graphics and Interactive Techniques* (New York, NY, USA, 2000), SIGGRAPH '00, ACM Press/Addison-Wesley Publishing Co., pp. 385–392.

[NFJ02] NGUYEN D. Q., FEDKIW R., JENSEN H. W.: Physically based modeling and animation of fire. *ACM Trans. Graph. 21*, 3 (July 2002), 721–728.

[Per85] PERLIN K.: An image synthesizer. *ACM Siggraph Computer Graphics 19*, 3 (1985), 287–296.

[PH89] PERLIN K., HOFFERT E. M.: Hypertexture. *SIGGRAPH Comput. Graph. 23*, 3 (July 1989), 253–262.

[PP94] PERRY C. H., PICARD R. W.: Synthesizing flames and their spreading. In *Proceedings of 5th Eurographics workshop on animation and simulation* (1994), pp. 105–117.

[PP06] PEGORARO V., PARKER S. G.: Physically-based realistic fire rendering. *Natural Phenomena* (2006), 51–59.

[Ree83] REEVES W. T.: Particle Systems - a Technique for Modeling a Class of Fuzzy Objects. *ACM Trans. Graph. 2*, 2 (Apr. 1983), 91–108.

[RHC95] RUSHMEIER H., HAMINS A., CHOI M. Y.: Volume rendering of pool fire data. *Computer Graphics and Applications, IEEE 15*, 4 (Jul 1995), 62–67.

[SF95] STAM J., FIUME E.: Depicting fire and other gaseous phenomena using diffusion processes. In *Proceedings of the 22Nd Annual Conference on Computer Graphics and Interactive Techniques* (New York, NY, USA, 1995), SIGGRAPH '95, ACM, pp. 129–136.

[Sta99] STAM J.: Stable fluids. In *Proceedings of the 26th Annual Conference on Computer Graphics and Interactive Techniques* (New York, NY, USA, 1999), SIGGRAPH '99, ACM Press/Addison-Wesley Publishing Co., pp. 121–128.

[WL00] WEI L.-Y., LEVOY M.: Fast texture synthesis using tree-structured vector quantization. In *Proceedings of the 27th Annual Conference on Computer Graphics and Interactive Techniques* (New York, NY, USA, 2000), SIGGRAPH '00, ACM Press/Addison-Wesley Publishing Co., pp. 479–488.

[YS96] YAO J., STEWART D. S.: On the dynamics of multi-dimensional detonation. *Journal of Fluid Mechanics 309* (2 1996), 225–275.

[ZCM11] ZHANG Y., CORREA C. D., MA K.-L.: Graph-based fire synthesis. In *Proceedings of the 2011 ACM SIGGRAPH/Eurographics Symposium on Computer Animation* (New York, NY, USA, 2011), SCA '11, ACM, pp. 187–194.