# Implementation of Rea1l Time Atmospheric Scattering

Simon Wallner*

June 25, 2011

This document accompanies a small demo implementation of [Hoffman and Preetham, 2002]'s atmospheric scattering model. The Implementation tries to follow the paper closely but diverges in some aspects. Below you will find more detail on the general approach, the divergences and where the results differ from the original implementation.

## 1 Background

The atmospheric scattering model [Hoffman and Preetham, 2002] was released in 2002 and more information about this approach can also be found in [Hoffman and Preetham, 2003] and [Preetham, 2003]. Earlier relevant work for this approach is [Preetham et al., 1999].

The model uses Rayleigh and Mie scattering to model scattering in the earth's atmosphere. A few simplifications are made to make the model work in real time on current and even last generation consumer hardware.

## 2 Model and Simplifications

The model only accounts for single scattering events, and constant density mediums. Earth's curvature is only accounted for and approximated in the skylight computation. In the case of aerial perspective earth is assumed to be flat.

*me@simonwallner.at

# 3 Implementation

The demo was implemented with the information found in the above mentioned papers. It is implemented as an OpenGL 3.2 demo, with GLSL shaders. Most of the implementation is done in the fragment shader, although big parts could have been offloaded to the vertex engine for similar results (dependent on scene tessellation).

Computations are done in HDR and a simple log-luminance centred linear tone mapping operator is used as a post process to display the image.

## 3.1 General

Atmospheric scattering is highly wavelength dependent, and therefore computations are performed for 3 selected wavelengths. This solution is by no means correct but gives a good looking approximation. The three used primaries are (according to [Hoffman and Preetham, 2003]) 400nm, 530nm and 700nm.

SI units are used throughout the implementation unless otherwise noticed.

## 3.2 Divergences

The implementation diverges from the papers in a few aspects. The scale factor for the Rayleigh phase function was chosen differently to achieve better looking results.

The *concentration factor* used in the Mie scattering function was computed differently, as the formula given in [Preetham et al., 1999] did not match the numbers in [Hoffman and Preetham, 2003].

## 3.3 Skylight

the curvature of the earth's surface is approximated by the function given in [Hoffman and Preetham, 2003]

$$\frac{l(\theta_s)}{l_{zenith}} = \frac{1}{\cos\theta_s + 0.15(93.885 - \theta_s)^{-1.253}} \tag{1}$$

where $\theta_s$ is the zenith angle and $l_zenith$ is the optical length of the medium in zenith direction. In the paper these lengths are given as $8.4km$ for molecules (Rayleigh scatterers) and $1.25km$ for aerosols (Mie scatterers).

the above formula is only usefull in the range of $[0, 90]$ degrees and is therefore capped, even though zenith angles of well above 90 deg can occur in practice.

The sky is generally assumed to be black (except for a faint skymap) and sky color is only a result of in-scattering.

To approximate the earth shadow a very crude approximation is used. The sun light's intensity is attenuated as the sun approaches the horizon.

### 3.4 Aerial Perspective

For aerial perspective the incident light on the ground is first computed by calculating extinction and in-scattering in the direction of the sun.

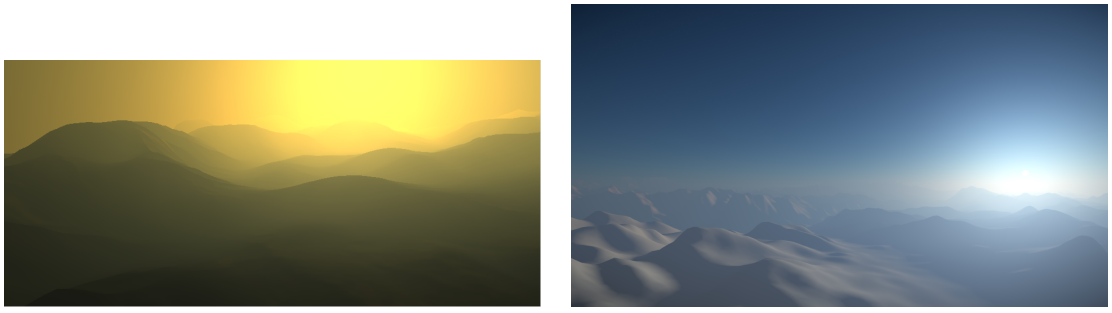
This incident light is then used to shade the ground and is then attenuated by the out-scattering as it travels towards the viewer. The atmosphere is assumed to be of constant density at ground level which simplifies computations. In-scattering is also evaluated and added to the result.

## 4 Results and Discussion

Although most parts are implemented in the fragment shader, rendering takes only about 10ms per frame on standard consumer hardware[1] at a resolution of $1280 \times 720$.

The Result looks pleasing and lifelike for most parameter choices. A big difference however is in situations where the sun is close to the horizon. The typical yellow-reddish sky of the rising and setting sun could not be achieved with the present implementation.

However, this effect can be clearly seen in the reference demo provided by Preetham and Hoffman. I assume that either my implementation is fundamentally flawed or not all implementation details are revealed in the cited papers.



(a) reference (b) result

[1] ATI Radeon 4830

# 5 Compilation and Usage

All source code can be found in a repository hosted at github[2] in the branch `RTVIS`. CMake is used as a build tool, and instructions on how to build the project can be found in the `readme` file.

Currently, the only supported platform is *Windows 7* 32bit, however it should be possible to port it to other platforms without much problems.

## 5.1 controls

The application can be configured via a config file found in the media folder (`kocmoc.properties`). It uses a self-documented syntax and there you can change resolution and window mode, among other settings.

The most important controls are as follows:

**F1** print the key mapping to the console

**WASD** move the camera

**mouse** rotate camera

**directional keys** move the sun

**I/O** decrease/increase the Rayleigh scattering coefficient, prints to console

**K/L** decrease/increase the Mie scattering coefficient, prints to console

**H/J** decrease/increase the directional constant g of the Henyey-Greenstein phase function, prints to console

**F/G** decrease/increase turbidity, prints to console

**.** save screenshot to working directory

# 6 License



The skymap image is courtesy of *ƒ NASA/Goddard Space Flight Center Scientific Visualization Studio*[3]

Although not demanded by the license, it would be nice to let me know if you are using parts of this project. If you have any questions regarding this project, please feel free to contact me any time via email.

[2] https://github.com/SimonWallner/kocmoc-demo
[3] http://svs.gsfc.nasa.gov/goto?3572

# References

[Hoffman and Preetham, 2002] Hoffman, N. and Preetham, A. (2002). Rendering outdoor light scattering in real time. *Game developers conference*.

[Hoffman and Preetham, 2003] Hoffman, N. and Preetham, A. (2003). Rendering outdoor light scattering in real time. *ATI Technologies*.

[Preetham, 2003] Preetham, A. (2003). Modeling skylight and aerial perspective. *ATI Research, ACM SIGGRAPH*.

[Preetham et al., 1999] Preetham, A., Shirley, P., and Smits, B. (1999). A practical analytic model for daylight. *SIGGRAPH '99: Proceedings of the 26th annual conference on Computer graphics and interactive techniques*.