Technical Report: Simulating Turbid Mediums

Sebastian Negulescu

April 30, 2024

1 Introduction

The goal of this experiment is to measure the performance of different phase functions in simulating scattering behaviour of light in turbid water. To do so, we will set up a physical experiment to measure the actual behaviour of light within turbid water, and compare the results to replicated virtual experiments with the different phase functions. The turbid water will be a Sodium Chloride and distilled water solution, and the light being shone through will be that from a green laser. The Henyey-Greenstein and Exponentiated Cosine phase functions will be used for comparison with the physical results. In addition, the Physically Based Ray Tracing renderer (PBRT-v4) will be used as the framework to render the simulated experiment in. This is due to PBRT-v4's ability to efficiently render mediums of varying densities using a null-scattering integration algorithm.

2 Needed Data

To effectively simulate the interaction of light in a turbid media, all variables outside of the phase function used should be known. Meaning, scattering and absorption coefficients should be already researched and deduced values. We need the specific absorption coefficient of water and sodium chloride to calculate the absorption coefficient of our solution. Furthermore, we need the scattering coefficient of our solution which may be harder to obtain. Apart from this, we need the wavelength and intensity of our laser pointer. Ideally, the absorption and scattering coefficients for our solution will match the wavelength emitted by our laser pointer. Though this is not a reality since researched data often varies from its target usecase. Instead, we will assume that the researched data accumulated is valid for our laser's wavelength.

3 Obtained Data and Difficulties

The availability of data surrounding the absorption and scattering coefficients of Sodium Chloride is sparratic. Absorption coefficient data exists for the ultraviolet and infrared ends of the colour spectrum, but not in the visible range. However, other research suggests that the absorption for Sodium Chloride within the visible spectrum is negligible. For the scattering coefficient, research has been published on a model that estimates the scattering coefficient of a Sodium Chloride solution with any density. This model could be useful when determining a scattering coefficient to use, however the researchers are those who determined that the absorption of a Sodium Chloride solution within the visible spectrum is negligible. This is a rather large assumption to make, and could yield innacurate results. In the absence of other data, this scattering coefficient model and lack of an absorption coefficient component for Sodium Chloride could be used, but results would be needed to be taken with a grain of salt.

4 Simulation

Creating a virtual experiment requires two components: a scene and a renderer.

4.1 Scene

The scene is modelled to reflect the main physical geometry of the experiment. There are components such as a table, walls, and stands for the laser pointer and cuvette that are represented by diffuse materials. The cuvette itself is represented by a tesseract with the inner cube containing the solution and the walls around having a glass material. The solution is modeled by a homogeneous medium, meaning the density of the solution is uniform throughout its volume. It is contained within the cuvette's inner cube. Finally, the laser pointer light is modeled by a spotlight with a small beam spread. These properties used for modeling are defined under the PBRT documentation.

4.2 Renderer

The PBRT renderer, while it contains much of the required features, does not support the ability to choose the phase function out of the box. The only phase function to be used with PBRT is the Henyey-Greenstein phase function. For other phase functions to be chosen, PBRT will need to be extended.

4.2.1 Abstraction of the Phase Function Class

Originally, all medium types in PBRT contain an instance of the Henyey-GreensteinPhaseFunction class. To allow for multiple phase functions, medium types will contain an instance of the parent class to the HenyeyGreensteinPhaseFunction class. This parent class will be called PhaseFunctionBase, and will hold virtual methods for sampling the phase function. This way, phase function implementations such as the HenyeyGreensteinPhaseFunction class can override these methods with their own sampling technique for the phase function. Medium types are responsible for construction of their phase function with parameters provided by the user. To determine which phase function to create and destroy, the type will be stored in an enum field under PhaseFunctionBase. The appropriate constructor/destructor will be called depending on the type of phase function.

4.2.2 Implementation of the Exponentiated Cosine Phase Function

To implement the exponentiated cosine phase function, a new class is created. This class is a child of the `PhaseFunctionBase` class, as it is for a phase function. The constructor takes in a parameter, `n`, representing the exponent for the cosine part of the function. Virtual functions `p` and `Sample_p` are overridden from the base class to properly use the exponentiated cosine function. Function `p` returns the value of the exponentiated cosine function given the incoming and outgoing direction. Function `Sample_p` takes in the outgoing direction and a random point and using the random point, calculates the incoming direction using the warping functions for the exponentiated cosine phase function. It then returns an object containing the value of `p` with the calculated incoming and outgoing directions, and the incoming direction calculated using the random point.

One issue with the exponentiated cosine phase function is that it is only defined over the region $[0, \frac{\pi}{2}]$. This means that if the incoming and outgoing direction vectors form an angle greater than $\frac{\pi}{2}$, the function presents undefined behaviour. Fortunately, at $\frac{\pi}{2}$, the function becomes 0. We can define the function in the way that if the angle between vectors is greater than $\frac{\pi}{2}$, the function will return 0 regardless. This maintains the forward-scattering property of the exponentiated cosine phase function, while keeping continuity of the function for all angles within the interval $[0, \pi]$. An interesting experiment would be to mirror the function around $\frac{pi}{2}$ and compare the resulting function to a superimposed dual-scattering version of the Henyey-Greenstein phase function, as the exponentiated cosine phase function would then both forward scatter and backward scatter.

5 Future Work

While progress has been made in breaking ground on this project, there is still lots to be done.

5.1 Scattering

Data on the scattering coefficient of Sodium Chloride is sparse at best. Until better data can be obtained for the scattering coefficient, it might be best to use a simpler approach with the simulation. When a ray enters the medium, we can scatter it with the phase function to get a new direction. Then, we compute the mean free path length of the ray, and see if the distance to the other side of the medium is shorter than that. If it is, the ray then exits the medium on the other side. Otherwise, the ray gets absorbed. This approach disregards the need for a scattering coefficient, but requires a refactor of PBRT. Another `Integrator` class will need to be created to simplify the interaction with mediums while preserving MIS techniques due to the point lights in the scene.

5.2 Physical Experiment

To check virtual results, a physical experiment needs to be conducted. The experiment will have a setup as such:

- A table in a dark room.

- A cuvette placed on an edge of the table.
- The cuvette is filled with a pure Sodium Chloride solution mixed with a concentration used for the virtual experiment.
- A laser pointer placed on a stand on the table as well, $2m$ away from the cuvette and pointing towards it.
- A camera off the table, behind the cuvette and pointing at it.

The camera and laser pointer should end up facing each other, with the cuvette in the middle. While the laser is being shone through the cuvette, the camera on the other side will capture intensities of the light. These intensities will be compared to those obtained in the virtual image for each phase function. The difference in each virtual image to the physical experiment can then be computed and compared to determine which phase function is more accurate.

5.3 Thumbnail Scene

Ideally, this project will have an interesting image to display on the cover. While I have created a sample scene, it is quite basic and lacks materials. To improve this image, I would like to change the camera position and angle to place more emphasis on the cuvette. This means moving the camera closer to the cuvette and angling it so the laser is in the background. I would also like to fill the air with another medium akin to air so the laser beam itself is visible. Finally, I would texture the table to have a wood-like coating, and make the laser pointer black.