A Compact Representation for Multiple Scattering in Participating Media using Neural Networks

Liangsheng Ge, Beibei Wang, Lu Wang, Nicolas Holzschuch

A Compact Representation for Multiple Scattering in Participating Media using Neural Networks

Liangsheng Ge
Shandong University

Beibei Wang*
Nanjing University of Science and Tech.

Lu Wang
Shandong University

Nicolas Holzschuch
Univ. Grenoble Alpes, Inria, CNRS, Grenoble INP, LJK

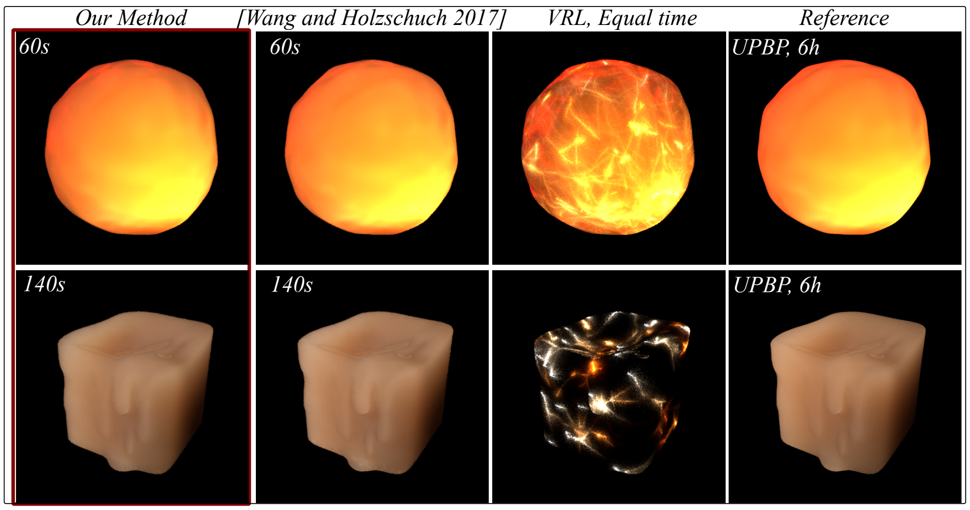

Figure 1: Equal time comparison for global illumination in participating media: our method based on neural networks (left), storing the full table [Wang and Holzschuch 2017], and Virtual Ray Lights (VRL) [Novák et al. 2012]. Right: reference solution.

CCS CONCEPTS

•Computing methodologies → Rendering;

KEYWORDS

Participating Media, Multiple Scattering, Neural Networks

ACM Reference format:
Liangsheng Ge, Beibei Wang, Lu Wang, and Nicolas Holzschuch. 2018. A Compact Representation for Multiple Scattering in Participating Media using Neural Networks. In *Proceedings of SIGGRAPH '18 Talks, Vancouver, BC, Canada, August 12-16, 2018,* 3 pages.
DOI: 10.1145/3214745.3214758

1 OVERVIEW

Many materials, such as milk or wax, exhibit scattering effects; incoming light enters the material and is scattered inside, giving a translucent aspect. These effects are computationally intensive as they require simulating a large number of events. Full computations are expensive, even with accelerating methods such as Virtual Ray Lights. The dipole approximation [Jensen et al. 2001] is fast, but a strong approximation. Precomputing the material response for multiple scattering [Moon et al. 2007; Wang and Holzschuch 2017] integrates well with existing rendering algorithms, allowing separate computation for single- and double- scattering, and fast computation for multiple scattering. Their main issue is efficient storage for the precomputed multiple scattering data.

We present a method to encode multiple scattering effects using a neural network. We replace the precomputed multiple scattering table of [Wang and Holzschuch 2017] (40 MB) with a trained neural network, with a cost of 6490 bytes (1623 floats). At runtime, the neural network is used to generate multiple scattering. We demonstrate the effects combined with Virtual Ray Lights (VRL), but our approach can be integrated with other rendering algorithms.

2 PREVIOUS WORKS

Moon et al. [2007] precompute multiple scattering effects and store the result on a set of concentric spheres. Müller et al. [2016] extend the approach with varying sphere size, and use it to convert granular materials to heterogeneous media.

Wang et al. [2016] precompute multiple scattering in a 2D table, using the symmetry of revolution, and use it for point-based global

*Joint first author

SIGGRAPH '18 Talks, Vancouver, BC, Canada

DOI: 10.1145/3214745.3214758

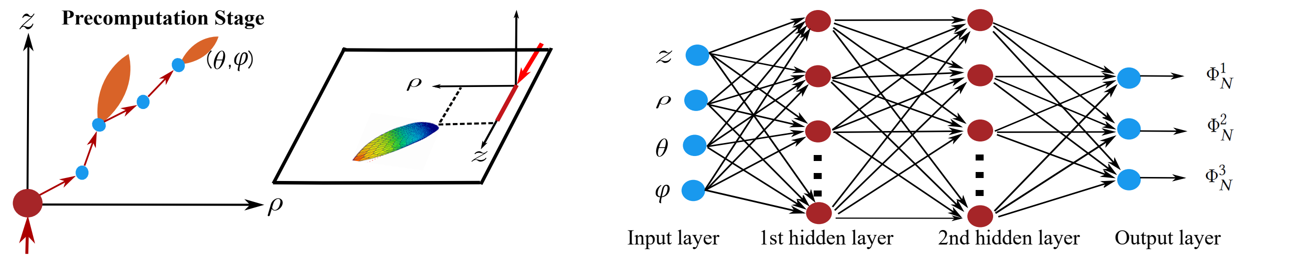

Figure 2: We precompute multiple scattering effects assuming an infinite medium, and compress the result using a two-layer neural network; we store the network coefficients.

illumination with participating media. [Wang and Holzschuch 2017] combine this with many illumination simulation algorithms.

Ren et al. [2013] used a multilayer acyclic feed-forward neural network to map scene data to indirect illumination.

3 NEURAL NETWORK MODEL FOR PRECOMPUTED MULTIPLE SCATTERING

3.1 Precomputing multiple scattering

First we compute multiple scattering effects assuming a light source with a dirac in position and direction in an infinite participating medium. The problem has symmetry of revolution: we store multiple scattering using cylindrical coordinates for position $r(\rho, z)$ and spherical coordinates at each point for direction (θ, φ) (see Figure 2). This step takes 8mn for a given material. [Wang and Holzschuch 2017] used this data to render participating media, combined with other algorithms for low-order scattering.

3.2 Neural Network Model

We can see the precomputed multiple scattering as a complex mapping from a 4 dimension domain (input coordinates, $(\rho, z, \theta, \varphi)$) to a 3 dimension domain (the RGB channel), where the output function has an exponential falloff with the first two coordinates, and can be highly anisotropic for the last two coordinates (assuming anisotropic material).

We treat it as a regression problem and train a neural network to learn the multiple scattering function Φ, approximating it with $\Phi_N(\rho, z, \theta, \varphi, \mathbf{w})$, where $\mathbf{w}$ is the weights and biases of Φ_N, found by minimizing:

$$E = \sum_i \|r_i - \Phi_N(\rho, z, \theta, \varphi, \mathbf{w})\|^2 . \quad (1)$$

3.3 Neural Network Structure & Training

For our neural network, we used two fully connected hidden layers with 20 nodes each (see Figure 2). The networks were optimized using the ADAM optimizer in TensorFlow with a learning rate of 0.01. We compute the loss for the network as the difference between predicted radiance and computed radiance. The network is trained using the *L2* error metric. We split the precomputed table data from section 3.1 and use 70 % of the data for training, and the rest for validation. We normalize the input parameters to $[-1.0, 1.0]^4$ and shuffle them. We switch the output parameters to the *log* domain to make them well distributed. We train the models with 10,000 iterations. It takes 25 mn to train the network on a given material.

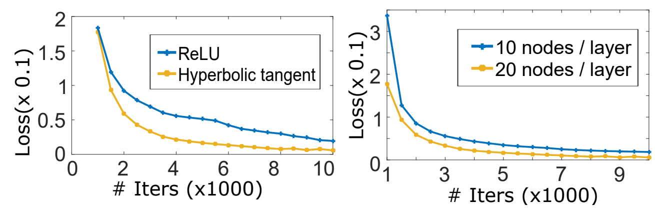

Figure 3: Loss as a function of the number of iterations, for different network settings.

Figure 3 shows the impact of the neural network parameters: using 10 or 20 nodes, and using tanh or Rectified Linear Unit (ReLU) as the activation function. In practice, we used 20 nodes and tanh. Computing multiple scattering response and training the neural network is done in a precomputation step.

To render, we extract the precomputed table data from the network. We used the Mitsuba Renderer [Jakob 2010], and VRL for low-order scattering.

4 CONCLUSION AND FUTURE WORK

We have presented a neural network model to represent multiple scattering events in participating media. The model provides a very compact representation for precomputed multiple scattering, and can be combined with many existing rendering algorithms, providing similar results for a fraction of the memory cost. In future work, we want to extend the range of parameters for learning (albedo, phase function anisotropy), so the entire space of materials can be represented with a single neural network model.

REFERENCES

Wenzel Jakob. 2010. Mitsuba Renderer. http://www.mitsuba-renderer.org/. (2010).

Henrik Wann Jensen, Stephen R. Marschner, Marc Levoy, and Pat Hanrahan. 2001. A Practical Model for Subsurface Light Transport. In *SIGGRAPH*. ACM, 511–518.

Thomas Müller, Marios Papas, Markus Gross, Wojciech Jarosz, and Jan Novák. 2016. Efficient Rendering of Heterogeneous Polydisperse Granular Media. *ACM Trans. Graph. (proc. SIGGRAPH Asia)* 35, 6 (December 2016), 168:1–168:14.

Jonathan T. Moon, Bruce Walter, and Stephen R. Marschner. 2007. Rendering Discrete Random Media Using Precomputed Scattering Solutions. In *Rendering Techniques (proc. EGSR)*. 231–242.

Jan Novák, Derek Nowrouzezahrai, Carsten Dachsbacher, and Wojciech Jarosz. 2012. Virtual Ray Lights for Rendering Scenes with Participating Media. *ACM Trans. Graph. (proc. SIGGRAPH)* 31, 4, Article 60 (July 2012), 11 pages.

Peiran Ren, Jiaping Wang, Minmin Gong, Stephen Lin, Xin Tong, and Baining Guo. 2013. Global Illumination with Radiance Regression Functions. *ACM Trans. Graph.* 32, 4, Article 130 (July 2013), 12 pages.

Beibei Wang, Jean-Dominique Gascuel, and Nicolas Holzschuch. 2016. Point-Based Light Transport for Participating Media with Refractive Boundaries. In *EGSR (EI&I)*.

Beibei Wang and Nicolas Holzschuch. 2017. Precomputed Multiple Scattering for Light Simulation in Participating Medium. In *ACM SIGGRAPH 2017 Talks (SIGGRAPH '17)*. Article 35, 2 pages.