Neural Bidirectional Texture Function Compression and Rendering

Luca Quartesan
luca@traverseresearch.nl
Traverse Research BV
Breda, Netherlands

Carlos Pereira Santos
santos.c@buas.nl
Breda University of Applied Sciences
Breda, Netherlands

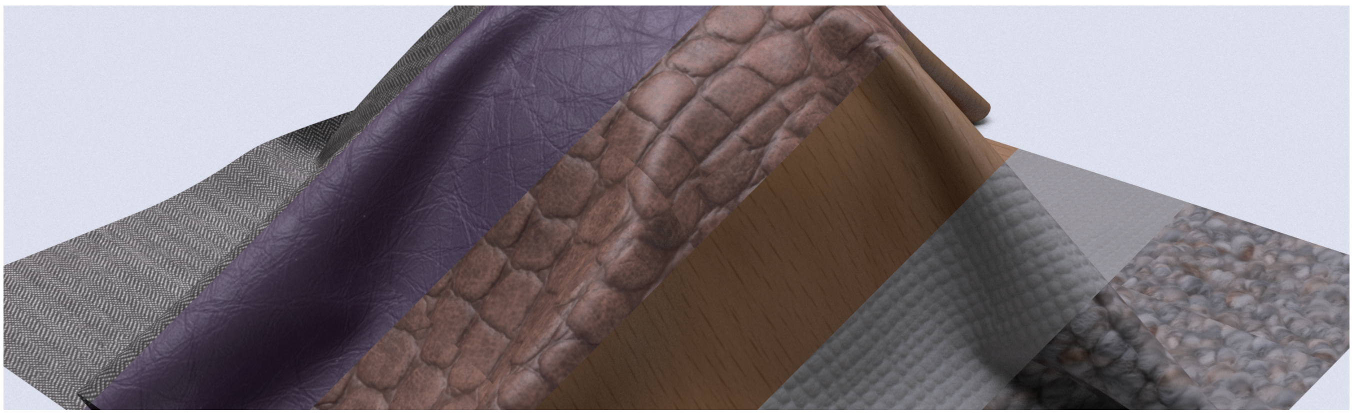

Figure 1: Examples of neural BTF materials from UBO2014 rendered using Mitusba 2, trained with configuration 4. From left to right: fabric05, leather06, leather10, wood08, wallpaper05, carpet06.

ABSTRACT

The recent success of Machine Learning encouraged research using artificial neural networks (NNs) in computer graphics. A good example is the bidirectional texture function (BTF), a data-driven representation of surface materials that can encapsulate complex behaviors that would otherwise be too expensive to calculate for real-time applications, such as self-shadowing and interreflections.

We propose two changes to the state-of-the-art using neural networks for BTFs, specifically NeuMIP. These changes, suggested by recent work in neural scene representation and rendering, aim to improve baseline quality, memory footprint, and performance. We conduct an ablation study to evaluate the impact of each change. We test both synthetic and real data, and provide a working implementation within the Mitsuba 2 rendering framework.

Our results show that our method outperforms the baseline in all these metrics and that neural BTF is part of the broader field of neural scene representation.

Project website: https://traverse-research.github.io/NeuBTF/.

CCS CONCEPTS

• **Computing methodologies → Rendering**; **Machine learning**; *Computer vision.*

SA '22 Posters, December 06-09, 2022, Daegu, Republic of Korea

ACM ISBN 978-1-4503-9462-8/22/12.
https://doi.org/10.1145/3550082.3564188

KEYWORDS

neural networks, bidirectional texture function, neural representation, neural materials

1 INTRODUCTION

The BTF [Dana et al. 1999] is a material model that models how a planar surface changes its appearance when illuminated and viewed from different directions. It is parameterized by surface position p_{uv}, light ω_i and view ω_o directions $f(p_{uv}, \omega_i, \omega_o)$. A BTF can be collected from captures of a material from a combination of light and view directions. Thus, it can capture a wide range of complex effects, including parallaxing, self-shadowing, and subsurface scattering. Recently, several studies have explored the possibility of using NNs to compress and interpolate BTF data [Kuznetsov et al. 2021; Rainer et al. 2020, 2019].

The current state-of-the-art in using NNs for BTFs is NeuMIP [Kuznetsov et al. 2021]. NeuMIP consists of two modules: offset and decoder. Both modules are composed of a neural texture pyramid, similar to a mipmap, and a small multi-layer perceptron (MLP). For each material, the neural texture pyramid is trained alongside the MLP. The system works as follows: First, the input coordinates are used to sample the offset texture, then this value is concatenated to the view direction and input into the offset MLP, which outputs a depth used to calculate an offset using a traditional parallax technique. The offset is then used to sample the texture of the decoder, which is then concatenated with view and light directions, and fed into the decoder MLP, which outputs the final color.

This work investigates whether this method can be optimized by improvements found in the field of neural scene representation.

Table 1: Overview of our ablation study in terms of quality (PSNR), memory (MB), and performance (it/s). PSNR is reported for ten materials, the first five from the dataset provided in NeuMIP and the last five from UBO2014.

Configurations			↑ PSNR										↓ MB	↑ it/s
	Sin	Cat	wool2	wool1	shell	rock	foam	carpet07	carpet11	fabric01	leather08	leather11		
1	✗	✗	30.752	39.734	35.025	32.811	30.320	34.473	33.636	31.278	30.808	32.648	19.6	118.6
2	✓	✗	33.022	**43.113**	**37.679**	**34.524**	**32.430**	35.459	34.588	32.353	32.598	33.892	19.6	111.6
3	✗	✓	30.956	39.953	34.807	32.868	29.774	35.433	35.038	31.945	32.139	32.951	**11.2**	**152.3**
4	✓	✓	**33.178**	42.549	37.237	34.073	32.158	**36.704**	**36.283**	**33.234**	**33.030**	**34.600**	**11.2**	140.7

2 METHOD

We propose two changes, supported by recent progress in the field of neural scene representation. First, we replace the ReLU activations with Sine as presented in Siren [Sitzmann et al. 2020] to improve the quality of the output (**Sin**). Then we change the sampling strategy. Rather than interpolating between the two closest mip levels, we concatenate the samples at each mip level higher than the query, as suggested in Instant-NGP [Müller et al. 2022], which we refer to as **Cat**. This allows for neural textures with fewer channels since the values are concatenated, resulting in larger input.

To estimate the impact of each proposed change, we performed an ablation study and reported the results in terms of compression quality using PSNR, memory in megabytes (MB) and performance in $\frac{\underline{iterations}}{second}$ (it/s). Performance is collected in Python using a PyTorch benchmarking tool on an NVIDIA GeForce RTX 3080 Ti, with an input of $512x512$ pixels. This is provided as an approximation of the impact of our changes on an optimized implementation that should achieve a real-time frame rate on a larger input, as reported in NeuMIP [Kuznetsov et al. 2021]. The quantitative results of the ablation study can be found in Table 1.

We compared the results with two types of dataset: synthetic and real captures. The synthetic dataset was provided privately by the authors of NeuMIP [Kuznetsov et al. 2021], while the real dataset is from UBO2014 [Weinmann et al. 2014]. A crucial characteristic of the synthetic data from NeuMIP is its multi-scale nature. In addition to different light and view combinations, this dataset is also captured from a variety of distances. In Table 1 we report results on a selection of five materials per dataset.

We use standard MLPs with two hidden layers, where each intermediate layer has 32 output features. For the initialization schemes, we follow the example of NeuMIP [Kuznetsov et al. 2021], while when using **Sin**, we implement the strategy of Siren [Sitzmann et al. 2020]. The neural texture pyramid uses seven feature channels, but when using **Cat**, we use only four channels.

We train up to 15,000 steps, which takes 15 to 30 minutes per experiment. Each step is carried out with a batch of four; each sample is 512×512 pixels, with different parameters of UV, light, view and distance per pixel. The loss used is $MAE + 10 \times MSE$, and we use the Adam optimizer. Our implementation is in PyTorch, while the renderings shown in Figure 1, are done in Mitsuba 2 [Nimier-David et al. 2019] with a custom BSDF plugin.

3 DISCUSSION

Although it is capable of reproducing the look of specular materials, one limitation of this system is that it can behave only like diffuse materials. This is because the dataset and the model do not address the probability distribution of the possible directions of a ray after hitting the material. Instead, a cosine-distributed hemisphere is used, as suggested by NeuMIP [Kuznetsov et al. 2021]. Therefore, it would be interesting to explore the possibilities of including support for other types of probability distribution.

From our data, we can observe that the inclusion of **Sin** positively increases quality, outperforming the baseline in all cases, while it slightly decreases performance, because Sine is more expensive than ReLU. The other change, **Cat**, significantly reduces memory ($\times$ 0.57) and improves performance ($\times$ 1.25), due to fewer parameters and fewer branching, respectively. In terms of quality, **Cat** performs close to the baseline, sometimes exceeding it by a small margin, in particular on the simpler data from UBO2014 [Weinmann et al. 2014], where all mip levels can be shared to represent a single scale. Combining the proposed changes results in a model that outperforms the baseline in all metrics evaluated.

ACKNOWLEDGMENTS

We thank Alexandr Kuznetsov for providing the code and dataset from NeuMIP [Kuznetsov et al. 2021].

REFERENCES

Kristin J Dana, Bram van Ginneken, Shree K Nayar, and Jan J Koenderink. 1999. Reflectance and texture of real-world surfaces. *ACM Trans. Graph.* 18, 1 (Jan. 1999), 1–34.

Alexandr Kuznetsov, Krishna Mullia, Zexiang Xu, Miloš Hašan, and Ravi Ramamoorthi. 2021. NeuMIP: multi-resolution neural materials. *ACM Trans. Graph.* 40, 4 (July 2021), 1–13.

Thomas Müller, Alex Evans, Christoph Schied, and Alexander Keller. 2022. Instant neural graphics primitives with a multiresolution hash encoding. *ACM Trans. Graph.* 41, 4 (July 2022), 1–15.

Merlin Nimier-David, Delio Vicini, Tizian Zeltner, and Wenzel Jakob. 2019. Mitsuba 2: a retargetable forward and inverse renderer. *ACM Trans. Graph.* 38, 6 (Nov. 2019), 1–17.

Gilles Rainer, Abhijeet Ghosh, Wenzel Jakob, and Tim Weyrich. 2020. Unified neural encoding of BTFs. *Comput. Graph. Forum* 39, 2 (May 2020), 167–178.

Gilles Rainer, Wenzel Jakob, Abhijeet Ghosh, and Tim Weyrich. 2019. Neural BTF compression and interpolation. *Comput. Graph. Forum* 38, 2 (May 2019), 235–244.

Vincent Sitzmann, Julien N P Martel, Alexander W Bergman, David B Lindell, and Gordon Wetzstein. 2020. Implicit neural representations with periodic activation functions. In *Proceedings of the 34th International Conference on Neural Information Processing Systems* (Vancouver, BC, Canada) *(NIPS'20, Article 626)*. Curran Associates Inc., Red Hook, NY, USA, 7462–7473.

Michael Weinmann, Juergen Gall, and Reinhard Klein. 2014. Material classification based on training data synthesized using a BTF database. In *Computer Vision – ECCV 2014*. Springer International Publishing, Cham, 156–171.