A Neural Model for High-Performance Scanning Electron Microscopy Image Simulation of Porous Materials

Tim Dahmen
Hochschule Aalen
Aalen, Germany
German Research Centre for Artificial Intelligence (DFKI)
Saarbrücken, Germany
tim.dahmen@hs-aalen.de

Markus Kronenberger
Fraunhofer ITWM
Kaiserslautern, Germany
markus.kronenberger@itwm.fraunhofer.de

Niklas Rottmayer
RPTU Kaiserslautern-Landau
Kaiserslautern, Germany
niklas.rottmayer@rptu.de

Katja Schladitz
Fraunhofer ITWM
Kaiserslautern, Germany
katja.schladitz@itwm.fraunhofer.de

Claudia Redenbach
RPTU Kaiserslautern-Landau
Kaiserslautern, Germany
claudia.redenbach@rptu.de

Abstract

We present a surrogate model in the form of a neural network that can approximately replicate the Monte-Carlo simulations conventionally performed to simulate scanning electron microscopy imaging of porous materials. These materials are of high practical relevance but scanning electron microscopic images of their microstructures feature difficult to interpret artifacts, specifically in areas where the electron beam enters pores in the sample surface and interacts inside the pores. Because of these artifacts, synthetic back-scattered electron and secondary electron images are of high interest both for verifying image interpretation and as training data in a machine learning context. However, the Monte-Carlo simulations of the physical interaction of the electron beam with the solid material are computationally very demanding. Our surrogate model accepts three-dimensional microstructure representations of porous materials in the form of lists of primitives. The system converts these lists to a specific data representation suitable for a neural network. It then uses a convolutional architecture to generate two-dimensional back-scattered electron and secondary electron images in a single forward pass, realizing 4-5 orders of magnitude performance improvement over the first order simulations. Remarkably, the model performs well on arbitrary microstructures like systems of cubes, even though it was trained on structures consisting of spheres and cylinders only.

1. Introduction

Scanning electron microscopy (SEM) is a tremendously important imaging technology in numerous fields including life sciences, material sciences, and semiconductors. The specimen is scanned by a focused electron beam, and back-scattered electrons (BSE) or secondary electrons (SE) are detected by the respective detectors. The method can cover a large range of resolutions and fields of view, from optical resolution ($> 0.2\,\mu$m) all the way down to atomic resolution (< 0.1 nm). SEM techniques can be extended to three-dimensional (3D) imaging using several approaches [13]. In focused ion beam scanning electron microscopy (FIB-SEM), a specimen is imaged by sequentially removing thin layers of material using a focused ion beam. Each resulting surface is then scanned using a focused electron beam such that 3D information is acquired layer-by-layer [2, 7, 12, 17].

Creating SEM images synthetically has been of interest for a long time for various goals, most notably as a verification tool in instrument development. The conventional approach is a physical simulation of the paths of individual electrons and their interaction with the microstructure. To this end, the emission of electrons, their trajectories through

magnetic fields, their interaction with the sample, and finally the effect on the detector are simulated using a Monte-Carlo algorithm. This approach has the advantage that it simulates imaging from first order physics and therefore can accurately capture various physical phenomena. Also, the stochastic nature of most processes at the quantum level is naturally represented by the randomness inherent to Monte-Carlo algorithms. Several software tools are available, including CASINO [6], Win X-Ray [10], SRIM/TRIM [25], and MCSEM [11].

As SEM images are increasingly processed using machine learning approaches, the demand for large quantities of synthetic SEM images as training data arises. However, the conventional Monte Carlo simulations require simulating not only the primary electrons emitted by the gun, but also the entire cascade of secondary electrons that are created by the interaction of the electron with the sample. The simulation is therefore computationally very expensive. This computational demand can in principle be met by distributing the workload to large clusters, but the involved cost and energy consumption are a concern.

Prill [19] combined physically correct interaction of the electrons with the sample exploiting the MONSEL library [16] with a variety of acceleration techniques including precalculation of electron paths and simplification of paths in depths where no SE can be emitted to reduce the computation time to an extent enabling generation of FIB-SEM stacks instead of just individual SEM images. The tool can handle geometries consisting of the geometric union of spheres, cylinders and cubes. Synthetic FIB-SEM stacks generated by Prill's method have been used successfully to compare and optimize segmentation algorithms [20, 22], train a machine learning model [8], and to consistently quantify image quality [21]. Calculation of a stack of realistic size nevertheless still takes several hours on a cluster.

In computer graphics, neural rendering refers to the idea of using artificial neural networks to replace the rendering pipeline partially or entirely. The idea had recently profound impact on the field of optical image synthesis. Hardware accelerated ray tracing [23], in combination with recurrent denoising autoencoders [3], has enabled interactive global illumination computations on consumer hardware. A neural graphics pipeline for controllable image generation from geometric models has been a research goal for a while [4]. While methods that rely on geometric scene descriptions have to use hybrid approaches [12], another branch of research focuses on image-based scene representations. In view synthesis, specifically the use of Neural Radiance Fields (NeRF) [4] and consecutive work [14, 15, 24] have received widespread attention. Neural rendering has been a prominent part of computer graphics since its invention. A relatively recent overview of the field can be found in [14].

Here, we introduce a surrogate model for the Monte-Carlo simulation of SEM imaging. The model is several orders of magnitude faster than conventional simulations and reaches interactive performance on a single high-end GPU. As a main technical contribution, we suggest using a combination of extended height-fields and normal maps as data representation for the surrogate model.

2. Materials and Methods

2.1. Synthetic Microstructure Creation

Microstructures are simulated as stationary Boolean models which are commonly used and flexible stochastic geometry models [5]. The model can be decomposed into a point process for particle center points and a distribution of random shapes and sizes for the particles. Stationary Boolean models use a stationary Poisson point process to generate a sequence of particle locations in space. Its intensity $\lambda > 0$ reflects the mean number of points per unit volume. For each location, a random particle is drawn from a particle distribution independently of the position and the other particles. Particle distributions can range from fixed shape and size over fixed shape and random size up to random shape and size.

For the synthetic training data, we chose the particles to be either spheres with random radius or cylinders with random radius and height. For all characteristics, we used uniform distributions on suitable intervals. Furthermore, the orientation of cylinders was sampled either uniformly on the unit sphere or from a von Mises-Fisher distribution [9] with the z axis as mean direction and concentration parameter $\kappa = 20$. The number of particles was chosen such that the porosity Φ is 0.3, 0.5, or 0.7.

Miles' formulae [18] yield a one-to-one correspondence between porosity Φ and intensity λ via

$$\lambda = -\log(\Phi)/\overline{V},$$

where $\overline{V}$ denotes the mean particle volume. The use of porosity over intensity serves two purposes. First, many applications have prior knowledge of the porosity but not the intensity. Secondly, the porosity is more tangible when comparing different models and structures than the intensity.

The individual parameters for generating the spheres and cylinders are listed in Table 1. For each combination of porosity and radius interval, we sampled three sphere realizations. For cylinders, two realizations with von Mises-Fisher distributed orientation and one with uniform orientation were generated for each combination of porosity, radius interval, and height interval. This resulted in 18 microstructures consisting of spheres and 54 microstructures consisting of cylinders. Each such volume is discretized into 1,130 slices of size 850×850 pixels, see Figure 1 for volume renderings of some examples.

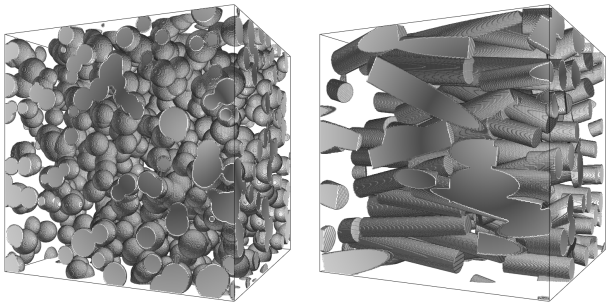

Figure 1. Renderings of geometries of spheres and cylinders. Both examples have expected porosity 70 %. Smaller porosity is obtained by using larger particles or by increasing the particle intensity.

Shape	Parameter	Values
Sphere	Porosity	0.3, 0.5, or 0.7
	Radius	[10,15] or [30,50]
Cylinder	Porosity	0.3, 0.5, or 0.7
	Radius	[5,10], [10,20] or [30,50]
	Height	[100,150] or [300,500]
	Orientation	Uniform or
		von Mises-Fisher ($\kappa = 20$)

Table 1. Parameter values and ranges for generating synthetic training data.

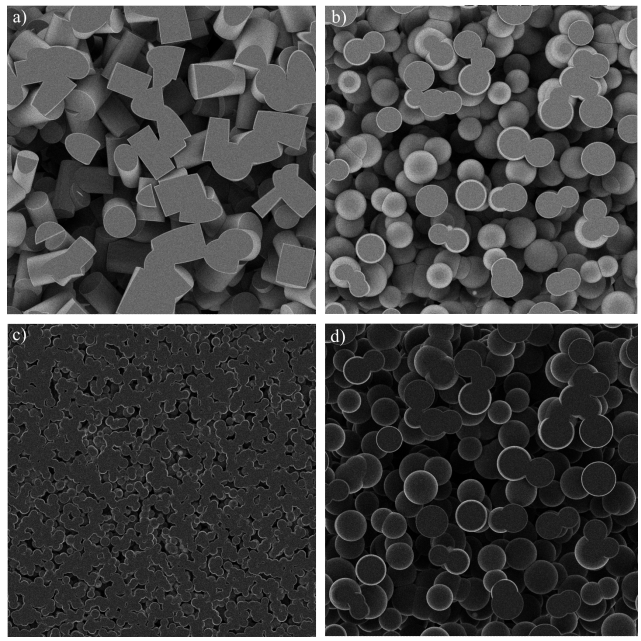

Figure 2. SEM images simulated using Prill's FIB-SEM Monte-Carlo simulation for realizations of Boolean models. a) Uniformly oriented cylinders with porosity 70 % imaged in BSE contrast. Parts of the structure which are not located in the current slice are visible through the pores, which complicates the segmentation. b) Spheres imaged in BSE contrast. c) Microstructure with smaller spheres imaged in SE contrast. d) The same microstructure as in b), imaged in SE contrast.

2.2. Monte-Carlo Simulation of SEM Imaging

FIB-SEM stacks were simulated by Prill's algorithm on a cluster with 108 nodes (Dell PowerEdge C6420) with dual Intel Xeon Gold 6240R, 384 GB RAM. Each simulation of a slice was started as one job using the Slurm job scheduler with a memory limit of 10 GB per job. The solid matter is supposed to be carbon. Slices of the simulated images are shown in Figure 2.

2.3. Data Representation

The most obvious representation for this type of data would be encoding the microstructures as volume data, i.e. storing density information on a three-dimensional regular grid. However, we found that this representation works very poorly as detailed in the discussion section. We therefore suggest a different representation of the microstructures as follows.

The geometry was encoded as two-dimensional (2D) images where each pixel consists of two components. The first component was an extended heightfield (EHF) with four values, the second component was a normalmap with three values, resulting in a total of seven values per pixel. A heightfield (also called heightmap or displacement map) is a data structure widely used in computer graphics to encode 2D surfaces embedded in a 3D space, for example digital elevation data. Hereby, one assumes a flat and rectangular basis, which is subdivided by a regular grid. Deviations from the basis shape are modeled by storing the distance in normal direction ("elevation") of each grid point. If the coordinate system is selected such that the x and y components span the basis, the z component is exactly this elevation. This means, the surface information is described by storing the z component of each point in a two-dimensional array, while the x and y components and the connectivity are implicitly given by the position in the array (Figure 3). An advantage of the representation is memory-efficiency: only one scalar value is stored per surface point. More importantly, the representation allows to represent 3D surfaces as 2D images, facilitating the use of a pre-trained convolutional neural network. A disadvantage of a heightfield representation is that it has a fixed resolution in the xy plane and can only describe surfaces that are approximately parallel to this plane. Surfaces approximately orthogonal to the xy plane are poorly sampled and overhangs cannot be modelled at all (Figure 3(b)).

To mitigate this limitation, one can use extended heightfields (EHF). The representation is a generalization of

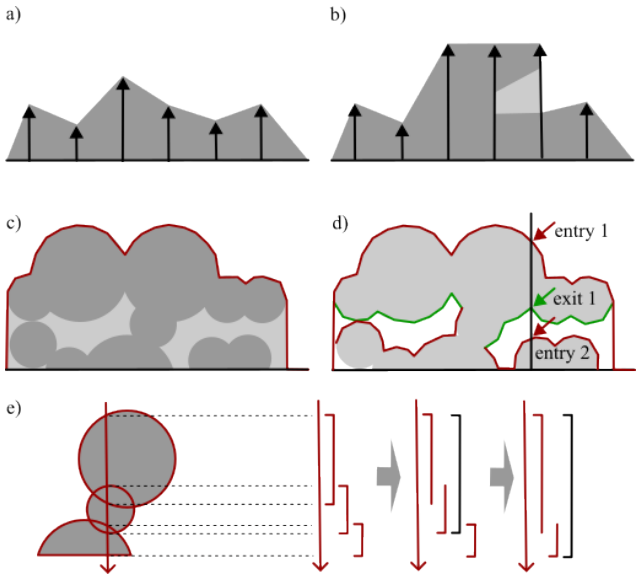

Figure 3. a) The surface is modeled by storing elevation at fixed distances. b) Overhangs cannot be modelled this way, as any horizontal position stores only one elevation value. c) The heightfield representation of a porous material captures the surface of the topmost pore, but cannot capture interior pores. d) In an extended heightfield, upper and lower surfaces of pores are stored, up to the n-th layer. This way, a limited number of inner pores can be represented. e) Our algorithm computes the heightfields directly from constructive solid geometry (union) of convex primitives. The entry and exit point for the primitive form an interval in z-direction. When the intervals of two primitives overlap, they can be merged to a single interval.

heightfields which is best understood by considering a line through a grid position xy, perpendicular to the basis shape. In a conventional heightfield, the array contains exactly one value which stores the z position of the first foreground (solid) voxel (entry 1) of the volume. In an EHF, additional values store the z position of the next background (pore) voxel below (exit 1), the next foreground voxel after this (entry 2) and so on up to a fixed number. We used a 4-extended heightfield, storing two entry and two exit points for each xy coordinate. This representation allowed us to represent a reasonable degree of porosity close to the surface of the sample while still achieving a compact data representation and discarding details of irrelevant, deeper pores. The principle is depicted in Figure 3(d). An example for the representation is shown in Figure 4.

2.4. Efficient Extraction of Extended Heightfields

The shape of the synthetic microstructure is defined by an implicit constructive solid geometry (CSG), i.e. the microstructure is the geometric union of multiple basic shapes. The conventional way to render EHFs from such geometry is to first tessellate the higher order surfaces such as cylinders and spheres, then resolve the CSG in the mesh representation, and finally use ray tracing with a spatial subdivision structure to extract the EHF. However, the approach is inefficient and unreliable for large numbers of primitives, as tessellating the curved surfaces leads to a drastic increase in the number of primitives, and resolving CSG on meshes is both slow and unreliable in all freely or commercially available rendering or modeling software we are aware of.

We therefore used a custom algorithm that resolves CSG and extracts the EHF in a single step. We exploit that the projection direction of the EHF is always the negative z direction, and the basic shapes (prior to resolving CSG) are convex. We first sort the primitives according to the lower z coordinate of their axis aligned bounding box. The computation of the EHF is then performed independently for each pixel, which allows to resolve the CSG as a one-dimensional problem. For each pixel, we loop over the primitives, compute the interval consisting of entry and exit point, and maintain a sorted list of all such intersections. When a new interval is inserted in the list, all entries between entry and exit can be discarded, which inherently resolves the CSG (see Figure 3e). The fact that the primitives are sorted by their lowest z coordinate allows for early termination once the required n entries of the EHF are determined. The core algorithm was implemented in CUDA, parallelizing over all pixels of the EHF. The approach achieved satisfactory performance as detailed in the result section.

The source code for the conversion mechanism is made available at github. The implementation uses C++ as host language for the CUDA code and provides a Python language binding for convenient integration. If our surrogate model is used with general geometry (not consisting of an implicit CSG of a large number of primitives), we suggest extracting the EHF conventionally using ray tracing.

2.5. Training Data Preparation

We sliced each volume iteratively with a distance of 10 voxels, then Monte-Carlo simulated BSE and SE images using Prill's tool and created the data representations (EHF and normalmap) described above. This process was repeated until 25% of the volume was used, such that 27 datapoints consisting of 7 images each (4-EHF, normalmap, BSE, SE) were created from each volume. The lower 75% of the volume served as guard region only, to prevent electron paths leaving the generated porous structure at the bottom.

We also conducted experiments with different data representations of the microstructures. For comparison, we trained networks that contained standard heightfields only, standard heightfields plus normalmaps, EHF only, and the 3D volume data directly.

Conventionally, images are normalized to match statistics of the dataset used for pretraining (most often ImageNet) during encoding. However, the statistics for the ex-

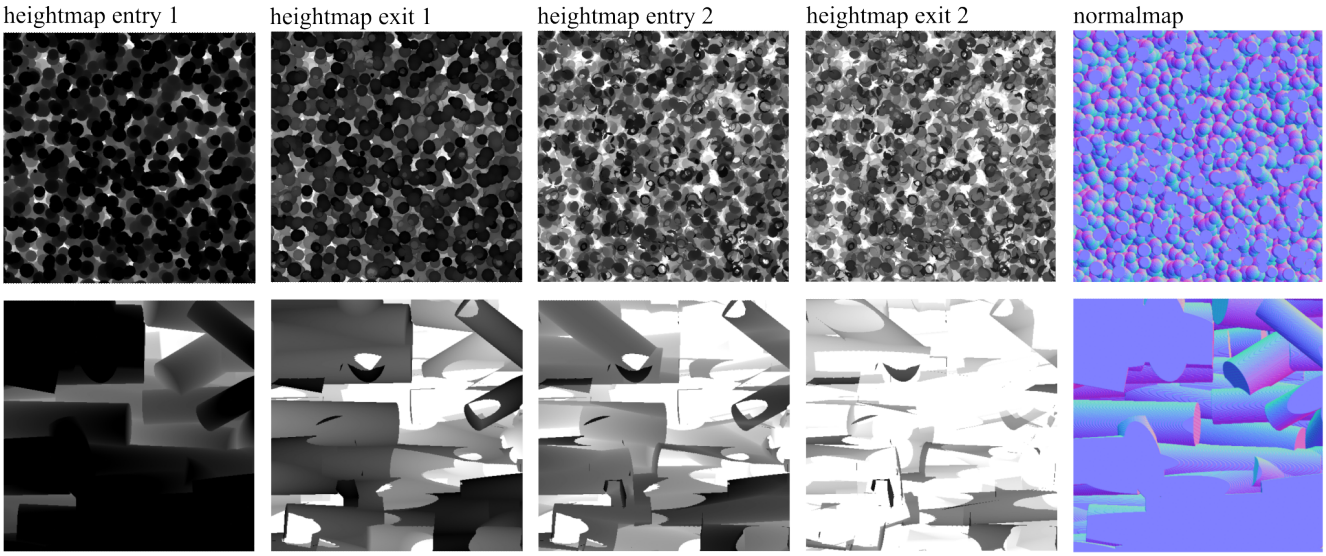

Figure 4. The representation of the volume data is changed to a 4-EHF (entry 1, exit 1, entry 2, exit 2) and a normal map, represented as red, green, and blue channels of an image. The top row of the figure shows a Boolean sphere model, the bottom row shows a Boolean cylinder model. The deeper layers of the EHF (exit 1, entry 2, exit 2) appear visually unusually, featuring 'holes' that can be mistaken as rendering artifacts, but really result from overlapping of several pores in the simulated microstructure.

tended heightfields and normal maps were so different from conventional rgb images, that we found that normalization has a negative impact.

2.6. Network Model

As a network model, we used conventional ResNet models with 34, 50, 101, and 152 blocks. The output was always a two-channel image, where one channel contained the BSE contrast image and the other channel contained the SE contrast image. The input was a 2D image with one to seven channels, depending on the data representation.

We used weights pretrained on the ImageNet dataset. As the pretrained models feature only three input channels (red, green, blue) but we required up to seven channels depending on the data representation, the missing entries on the features of the first layer were filled with zeroes. As a consequence, the order of the input channels was relevant. We tried putting the normal map in the first channels (such that the pretrained features were used in the normal map) and putting the normal map in the last channels (such that the pretrained features were used in the EHF). Other than the adaption of the input and output channels, no additional modifications were performed in the network architecture.

We also investigated using a 3D ResNet architecture directly on the volume data. As PyTorch does not provide a predefined 3D version of ResNet and no pretrained weights are available for this case, we implemented a custom 3D ResNet with random initialization. Our network used 7 residual blocks with one or two convolutions each, kernel size of $3 \times 3 \times 3$, $3 \times 3 \times 5$ or $3 \times 3 \times 7$ and 8, 16 or 32 kernels per convolution layer. When it became clear that this approach does not lead to satisfactory results, we canceled the systematic investigation. A prediction generated using the 3D ResNet, is provided as supplementary S3.

2.7. Hyperparameters and Loss Function

We used flattened L1 and L2 loss as standard loss functions. We used an Adam optimizer with an initial learning rate of 0.001, which was experimentally determined to be sufficiently small. We trained for 50 epochs, which showed satisfactory convergence, then determined the optimal hyperparameters and data representation.

Visualizations of the losses of all experiments are provided as supplementary S1.

2.8. Image Quality and Noise

BSE and SE images exhibit a natural level of shot noise, which is a consequence of electron statistics. This noise is automatically synthesized by Monte-Carlo simulations, if the number of simulated electrons corresponds to the real number of electrons and detector quantum efficiency is considered correctly. In contrast, neural networks do not create noisy images per se, and the output of the neural network is the expected value of the detector signal without any shot noise. To determine the amount of noise to add, we measured the mean and standard deviation of pixel values in homogeneous areas of the image, which gives more reliable results than other estimates for the signal-to-noise ratio [1].

We measured an SNR of 8.33, which is in line with our expectations from physics.

2.9. Analysis Method

For the purpose of analysis only, we scaled the dataset to a data range of $(0, 100)$, such that all measurements can be understood as percentage of the data range. As a measure of image quality of the predictions, we computed the mean pixel error (neural network prediction minus simulation ground truth), also called L1-norm. The residual is partially caused by systematic errors in our model, and partially a result of noise in the ground truth. The impact of noise can be measured by calculating the mean pixel error on homogeneous parts of the model. The noise influence was 0.89 % of the data range for BSE images and 2.07 % of the data range for SE images. We therefore provide two values, total error (measured from the data) and systematic error, where the influence of the noise was subtracted. We additionally provide structural similarity index (SSIM) and peak signal to noise ratio (PSRN).

3. Results

The presented surrogate model can create BSE images of general geometry with a mean pixel error of 0.98% of the data range. The SE images had a mean pixel error of 3.09%. In order to test the generalization capability of the model, we used training data consisting of spheres and cylinders, and test data consisting of cubes. For simulation of the test data, Prill's simulator was extended to also handle cube structures. Generalization results are shown in Figure 5.

So our model introduces a clearly measurable systematic error in the same order of magnitude compared to the error introduced by the shot noise.

3.1. Computational Cost

In terms of computational demands, the neural network part of our model requires 173 ms on average for a single image prediction on an Nvidia V100. Additionally, the conversion of the data to the 4-EHF plus normal map representation required 26 ms on average per slice on the same hardware. This data preprocessing has linear performance in the number of primitives, so the actual time required strongly depends on the intensity with fine granular microstructures (with many small primitives) leading to slower runtime and coarse-granular structures (with fewer, larger primitives) leading to faster runtimes. This timing excludes storage operations (loading and parsing the list of primitives), which is only required once per volume, or potentially storing the results to disc.

For comparison, the original Monto-Carlo simulation requires approx. $100 - 200$ min, see Table 2, so we achieve a speedup of 4-5 orders of magnitude. However, the original simulation code is not fully optimized and currently uses a single thread of an Intel Xeon Gold 6240R CPU, so our results do not allow for any quantitative performance comparison of the algorithms.

	Porosity		
Geometric Primitive	0.3	0.5	0.7
Sphere	128.4	123.3	120.4
Cylinder	211.1	168.5	144.4
Cube	224.4	229.5	220.5

Table 2. Mean runtimes in minutes for generating synthetic data using Prill's method based on various base geometries. Averaged over several samples in each case. One sample has 1 130 slices of size 850×850 pixel, which are simulated in parallel on a cluster. 1 000 electrons per pixel are simulated.

3.2. Influence of the Data Representation

We tested several data representations including 4-EHF plus normalmap, normalmap plus 4-EHF, standard heightfields plus normalmaps, standard heightfields only, and EHF only. All representations were tested by training a ResNet152 using an L1-loss for 50 epochs. It was found that normalmap plus 4-EHF give the best results (Table 3) and this data representation was used for all following experiments.

3.3. Influence of the Network Architecture

We tested conventional ResNet architectures with 34, 50, 101, and 152 residual blocks. For all network architectures, we used the standard weights deployed with PyTorch, which are pretrained on the ImageNet dataset. Results are displayed in Table 4. We generally found that less deep networks result in better model performance and best results are obtained by the ResNet34 architecture. Strikingly, the 3D volume representation did not result in satisfactory predictions, independent of the details of the used architecture. Once this became obvious, we canceled the full exploration of the parameter space and focused on the 2.5D data representations.

3.4. Influence of the Hyperparameters

We trained the models for a total of 50 epochs, storing the weights after epoch 1, 20 and 50. We unfroze all layers for training during all epochs. Results after 50 epochs are displayed in Table 3. For the smaller models (ResNet34 and ResNet50), we found that 50 epochs are sufficient and improvements after this are below statistical threshold. For the deeper models, results were somewhat inconclusive in this regard, and performance might potentially improve by further training. Training curves for all experiments are provided as supplementary material S2. We did not observe any overfitting behavior independent of the training duration.

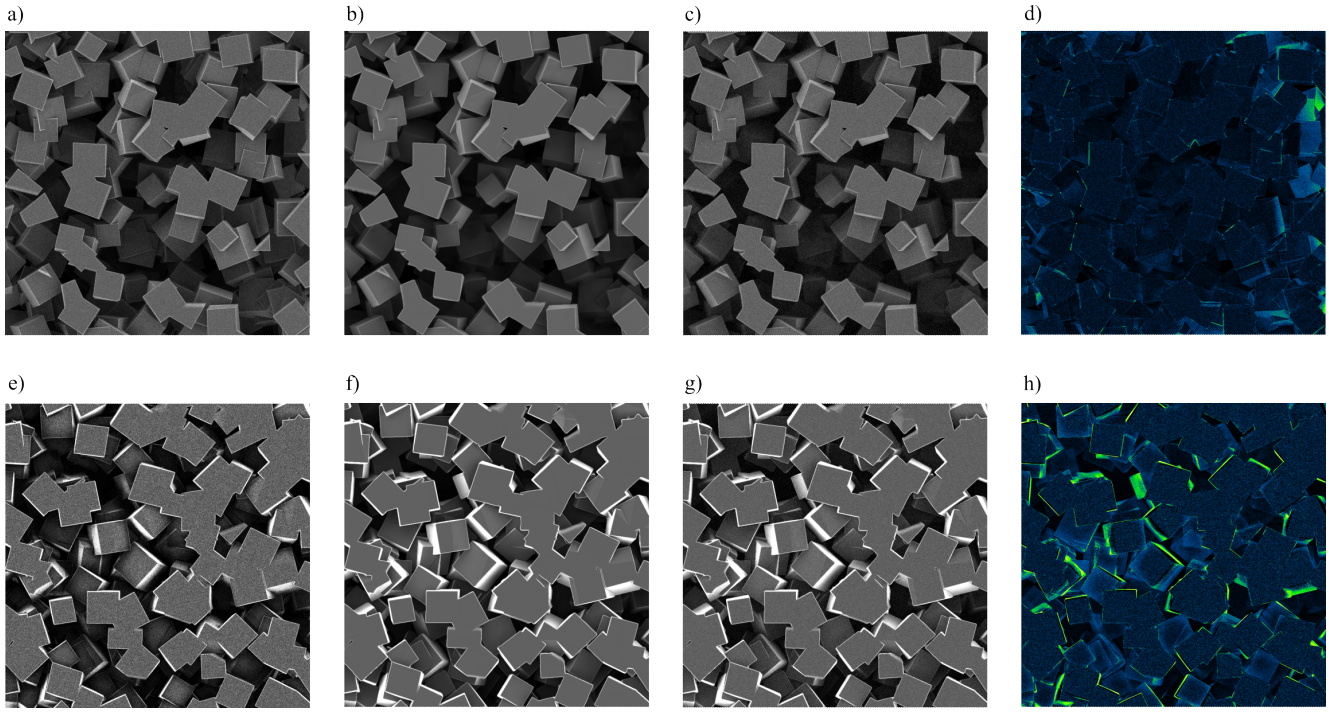

Figure 5. Results of the surrogate model. The generalization was tested by training the model on data consisting of spheres and cylinders, then using cubes as test data. The top row (a-d) shows the image with the lowest average error or the test set (a BSE image). a) Reference image (rendered with Prill's method). b) Prediction of our surrogate model. c) Prediction of our surrogate model with added synthetic noise to match the visual impression of the reference image. d) Residual image in false colors. The bottom row (e-h) shows the image with the highest average error of the test set (an SE image). e) Reference image (rendered with Prill's method). f) Prediction of our surrogate model. g) Prediction of our surrogate model with added synthetic noise. h) Residual image in false colors.

	mean pixel error (total)	mean pixel error (systematic)	SSIM	PSNR
data representation	(BSE / SE)	(BSE / SE)	(BSE / SE)	(BSE / SE)
normal map plus 4-EHF	**0.98** / 3.09	0.09 / 1.02	0.89 / 0.62	37.59 / 27.24
4-EHF plus normal map	1.02 / 3.13	0.13 / 1.06	0.89 / 0.61	37.13 / 27.12
standard heightfield plus normal map	1.05 / **3.02**	1.16 / 0.95	0.88 / 0.62	36.57 / 27.20
standard heightfield only	1.15 / 3.17	0.26 / 1.10	0.86 / 0.58	35.82 / 26.97
4-EHF	1.17 / 3.11	0.28 / 1.04	0.86 / 0.59	35.66 / 27.03
normal map only	1.60 / 3.35	0.71 / 1.28	0.81 / 0.56	32.97 / 26.61

Table 3. Influence of the data representation on the model performance. Pixel errors are given as percentage of the data range. The systematic error is the total error minus the influence of the noise in the ground truth data (0.89 % for BSE images and 2.07 % for SE images). Best results were obtained from the representation normal map plus 4-EHF. All results were measured using ResNet152, L1-loss and training for 50 epochs.

4. Discussion

We are presenting a surrogate model that can replace Prill's accelerated Monte-Carlo simulation for SE and BSE imaging. The main advantage of the model is performance: the model creates an image of 512×512 pixels in 173 ms on a V100 GPU, which is 4-5 orders of magnitude faster than Prill's Monte-Carlo simulations single threaded on a CPU.

Rather than creating a surrogate model, one could also try to further optimize the Monte-Carlo simulation itself. As the simulation of SEM imaging has not received the same amount of research interest as optical 3D rendering, one can speculate that there is also a comparatively large potential for algorithmic improvements and more efficient implementations. However, bringing SEM simulations to

data representation	loss function	mean pixel error (total) (BSE / SE)	mean pixel error (systematic) (BSE / SE)	SSIM (BSE / SE)	PSNR (BSE / SE)
ResNet34	L-1	**1.01 / 2.97**	0.12 / 0.90	0.89 / 0.63	37.12 / 27.57
ResNet50	L-1	1.03 / 3.19	0.14 / 1.12	0.88 / 0.61	36.87 / 26.88
ResNet101	L-1	1.04 / 3.19	0.15 / 1.12	0.88 / 0.61	36.79 / 26.90
ResNet152	L-1	1.02 / 3.13	0.13 / 1.06	0.89 / 0.61	37.13 / 27.12
ResNet152	L-2	1.05 / 3.39	0.16 / 1.32	0.89 / 0.60	37.04 / 26.51

Table 4. Influence of the network architecture and loss function on the model performance. Pixel errors are given as percentage of the data range. The systematic error is the total error minus the influence of the noise in the ground truth data (0.89 % for BSE images and 2.07 % for SE images). Best results were obtained with the smallest network: ResNet34. L-2 loss was tested in the ResNet152 architecture, and resulted in slightly worse results compared to L-1. All results were measured with the 4-EHF plus normal map layout, and training for 50 epochs.

the same level of fidelity as rendering systems for optical image synthesis (which are backed by a billion dollar industry in entertainment) is currently economically not feasible.

Technically, our main finding is that, counter intuitively, simulating electron-matter interaction directly on the 3D volume data maps poorly to convolutional neural networks with 3D kernels. We hypothesize that this is the case because different parts of the volume do not have identical influence on the image. Rather, the surface of the sample, which corresponds to the first filled voxel in view direction, dominates appearance. The concept that the surface voxel needs to be treated differently from all other voxels is poorly reflected by the properties of convolutional filters, which, in the 3D case, exhibit translational invariance in the z direction. As a solution, we propose changing data representation to a heightfield representation. This data structure captures the concept of a surface much better and allows using 2D convolutional networks. The heightfield representation can further be improved by providing 4-extended heightfields to model the topmost pores, and by providing partial derivatives of the surface in the form of normalmaps. As is often the case with training neural networks, a good data representation is the main factor for success, and good results can be achieved with a larger number of network architectures and hyperparameter settings.

Out model generates noise free images. Depending on the situation, this can either be beneficial, or noise can easily be added. If a realistic level of noise was required on the synthetic output images, we added the noise in a post-processing step by algorithm.

5. Conclusions

We present a surrogate model that can approximately replace conventional first order simulations using Prill's method for the simulation of scanning electron imaging of porous materials. The model creates backscatter electron (BSE) and secondary electron (SE) contrast and consists of two stages. The first stage is an algorithm, implemented to CUDA, to convert microstructure descriptions as geometric union of convex primitives to a data representation consisting of extended heightfields and normal maps. The second stage is a feed-forward convolutional network that converts this data representation to SE and BSE contrast images.

On the algorithmic level, our main contribution is the observation that convolutional neural networks perform very poorly if trained directly on a three-dimensional volume representation of the microstructures. Instead, a 2.5 dimensional representation of extended heightfields and normal maps should be used.

Our model is 4-5 orders of magnitude faster compared to first order Monte-Carlo simulations. As it is an approximate method, it introduces some systematic error. The average per pixel error (prediction-ground truth) is 0.98% of the data range for BSE images and 3.09% of the data range for SE images. For realistic imaging settings, this is in a similar range compared to the shot noise expected from electron statistics. These properties suggest that, for electron imaging, the model can eventually play the role that real-time rendering plays for optical imaging.

As there is no possibility to create real-time simulations of scanning electron imaging today, the space of applications is currently unexplored. Our work was primarily motivated by the need of large quantities of training data of scanning electron images in a machine learning context. In these situations, the cost and energy consumption of using large CPU clusters to run Monte-Carlo simulations is prohibitive. Our model solves this problem. However, we speculate that the availability of the equivalent of a real-time rendering system for electron optics will likely facilitate completely new applications over time, and the applicability of the technology might be much broader.

6. Acknowledgement

This work was supported by the German Federal Ministry of Education and Research (BMBF) [grant number 01IS21054 (poSt)] and [grant number FKZ 01IS21106 (ENGAGE)].

References

[1] Eyal Ben-Dor and Gila Notesco. A simple indicator for estimating the noise level of a hyperspectral data cube for earth observation missions. *Acta Astronautica*, 128:304–312, 2016. 5

[2] Andrew J Bushby, Kenneth M Y P'ng, Robert D Young, Christian Pinali, Carlo Knupp, and Andrew J Quantock. Imaging three-dimensional tissue architectures by focused ion beam scanning electron microscopy. *Nature Protocols*, 6(6):845–858, 2011. 1

[3] Chakravarty R. Alla Chaitanya, Anton S. Kaplanyan, Christoph Schied, Marco Salvi, Aaron Lefohn, Derek Nowrouzezahrai, and Timo Aila. Interactive reconstruction of Monte Carlo image sequences using a recurrent denoising autoencoder. *ACM Transactions on Graphics*, 36(4):1–12, 2017. 2

[4] Xuelin Chen, Daniel Cohen-Or, Baoquan Chen, and Niloy J. Mitra. Towards a Neural Graphics Pipeline for Controllable Image Generation. *Computer Graphics Forum*, 40(2):127–140, 2021. 2

[5] S.N. Chiu, D. Stoyan, W. S. Kendall, and J. Mecke. *Stochastic Geometry and Its Applications*. Wiley, Chichester, 2013. 2

[6] Hendrix Demers, Nicolas Poirier-Demers, Alexandre Réal Couture, Dany Joly, Marc Guilmain, Niels de Jonge, and Dominique Drouin. Three-dimensional electron microscopy simulation with the CASINO Monte Carlo software. *Scanning*, 33(3):135–146, 2011. 2

[7] Winfried Denk and Heinz Horstmann. Serial block-face scanning electron microscopy to reconstruct three-dimensional tissue nanostructure. *PLoS biology*, 2(11): e329–e329, 2004. 1

[8] C. Fend, A. Moghiseh, C. Redenbach, and K. Schladitz. Reconstruction of highly porous structures from FIB-SEM using a deep neural network trained on synthetic images. *Journal of Microscopy*, 281(1):16–27, 2021. 2

[9] N.I. Fisher, T. Lewis, and B.J.J. Embleton. *Statistical Analysis of Spherical Data*. Cambridge University Press, Cambridge, UK, 1987. 2

[10] Raynald Gauvin, Eric Lifshin, Hendrix Demers, Paula Horny, and Helen Campbell. Win X-ray, The Monte Carlo Program for X-ray Microanalysis in the Scanning Electron Microscope. *Microscopy and Microanalysis*, 9(S02):32–33, 2003. 2

[11] D. Gnieser, C. G. Frase, H. Bosse, and R. Tutsch. Mcsem-a modular monte carlo simulation program for various applications in sem metrology and sem photogrammetry. In *EMC 2008 14th European Microscopy Congress 1–5 September 2008, Aachen, Germany*, pages 549–550, Berlin, Heidelberg, 2008. Springer Berlin Heidelberg. 2

[12] Jurgen A W Heymann, Mike Hayles, Ingo Gestmann, Lucille A Giannuzzi, Ben Lich, and Sriram Subramaniam. Site-specific 3D imaging of cells and tissues with a dual beam microscope. *J. Struct. Biol.*, 155(1):63–73, 2006. 1

[13] P Jornsanoh, G Thollet, J Ferreira, K Masenelli-Varlot, C Gauthier, and a Bogner. Electron tomography combining ESEM and STEM: a new 3D imaging technique. *Ultramicroscopy*, 111(8):1247–1254, 2011. 1

[14] Bernhard Kerbl, Georgios Kopanas, Thomas Leimkuehler, and George Drettakis. 3D Gaussian Splatting for Real-Time Radiance Field Rendering. *ACM Transactions on Graphics*, 42(4):1–14, 2023. 2

[15] Tobias Kirschstein, Shenhan Qian, Simon Giebenhain, Tim Walter, and Matthias Nießner. NeRSemble: Multi-view Radiance Field Reconstruction of Human Heads. *ACM Transactions on Graphics*, 42(4):1–14, 2023. 2

[16] Jeremiah R. Lowney. Monte Carlo simulation of scanning electron microscope signals for lithographic metrology. *Scanning*, 18(4):301–306, 1996. 2

[17] Gavin E Murphy, Kedar Narayan, Bradley C Lowekamp, Lisa M Hartnell, Jurgen A W Heymann, Jing Fu, and Sriram Subramaniam. Correlative 3D imaging of whole mammalian cells with light and electron microscopy. *Journal of structural biology*, 176(3):268–278, 2011. 1

[18] J. Ohser, W. Nagel, and K. Schladitz. Miles formulae for Boolean models observed on lattices. *Image Anal. Stereol.*, 28(2):77–92, 2009. 2

[19] T. Prill and K. Schladitz. Simulation of FIB-SEM images for analysis of porous microstructures. *Scanning*, 35:189–195, 2013. 2

[20] Diego Roldán, Claudia Redenbach, Katja Schladitz, Matthias Klingele, and Michael Godehardt. Reconstructing porous structures from FIB-SEM image data: Optimizing sampling scheme and image processing. *Ultramicroscopy*, 226:113291, 2021. 2

[21] Diego Roldán, Claudia Redenbach, Katja Schladitz, Christian Kübel, and Sabine Schlabach. Image quality evaluation for FIB-SEM images. *Journal of Microscopy*, 293(2):98–117, 2024. 2

[22] M. Salzer, T. Prill, A. Spettl, D. Jeulin, K. Schladitz, and V. Schmidt. Quantitative comparison of segmentation algorithms for FIB-SEM images of porous media. *Journal of Microscopy*, 257(1):23–30, 2015. 2

[23] V. V. Sanzharov, V. A. Frolov, and V. A. Galaktionov. Survey of Nvidia RTX Technology. *Programming and Computer Software*, 46(4):297–304, 2020. 2

[24] Philippe Weier, Tobias Zirr, Anton Kaplanyan, Ling-Qi Yan, and Philipp Slusallek. Neural Prefiltering for Correlation-Aware Levels of Detail. *ACM Transactions on Graphics*, 42 (4):1–16, 2023. 2

[25] James F. Ziegler, M.D. Ziegler, and J.P. Biersack. SRIM – The stopping and range of ions in matter. *Nuclear Instruments and Methods in Physics Research Section B: Beam Interactions with Materials and Atoms*, 268(11-12):1818–1823, 2010. 2