Applied Mathematics for Modern Challenges
Vol. 1, No. 2, December 2023, pp. 87-104
doi:10.3934/ammc.2023007

MODEL-BASED DEEP LEARNING APPROACHES TO THE HELSINKI TOMOGRAPHY CHALLENGE 2022

Clemens Arndt[1], Alexander Denker[1], Sören Dittmer[1,2], Johannes Leuschner[1], Judith Nickel[*1] and Maximilian Schmidt[1]

[1]Center for Industrial Mathematics, University of Bremen, Germany

[2]Department of Applied Mathematics and Theoretical Physics
University of Cambridge, United Kingdom

Abstract. The Finnish Inverse Problems Society organized the Helsinki Tomography Challenge (HTC) in 2022 to reconstruct an image with limited-angle measurements. We participated in this challenge and developed two methods: an Edge Inpainting method and a Learned Primal-Dual (LPD) network. The Edge Inpainting method involves multiple stages, including classical reconstruction using Perona-Malik, detection of visible edges, inpainting invisible edges using a U-Net, and final segmentation using a U-Net. The LPD approach adapts the classical LPD by using large U-Nets in the primal update and replacing the adjoint with the filtered back projection (FBP). Since the challenge only provided five samples, we generated synthetic data to train the networks. The Edge Inpainting Method performed well for viewing ranges above 70 degrees, while the LPD approach performed well across all viewing ranges and ranked second overall in the challenge.

1. **Introduction.** The task in computed tomography reconstruction is to recover an image x from measurements y given by the Radon transform [14]

$$y(\varphi, s) = A[x](\varphi, s) = \int_{L(\varphi,s)} x(t)\,\mathrm{d}t. \tag{1}$$

Each measurement results from an integral over a straight line L parameterized by a distance $s \in \mathbb{R}$ and an angle $\varphi \in [0, \pi]$.

When we sample s and φ sparsely, this becomes a challenging inverse problem. The goal of the HTC 2022 was to recover the shapes of 2D phantoms from limited-angle sinogram data. In limited-angle CT, the goal is to recover x while only having measurements for angles from some small interval $[\varphi_{\min}, \varphi_{\max}]$ – we denote these measurements by $A|_{[\varphi_{\min},\varphi_{\max}]}[x]$.

The challenge has two phases. In the first phase, the organizers provided a dataset of five 512×512 pixels phantoms with full-angle sinograms, filtered back projection reconstructions, segmentation masks, and information about the measurement geometry. The target phantoms are homogeneous discs with differently shaped holes. We used this data set to develop this paper's algorithms. The second phase consisted of evaluating the algorithms; thus, no changes to the submitted methods were allowed at this stage of the challenge. The evaluation was split into

2020 *Mathematics Subject Classification.* Primary: 94A08.
Key words and phrases. Limited angle tomography, deep learning, inverse problems.
*Corresponding author: junickel(at)uni-bremen.de. Alphabetical author order.

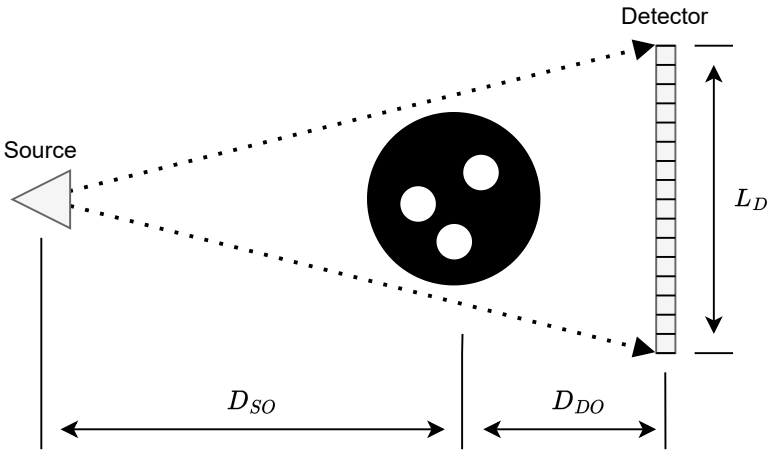

FIGURE 1. 2D Fan-beam geometry used for data collection. Adapted from https://fips.fi/HTC2022.php.

7 levels, starting with a view range of 90° in level 1 and decreasing by 10° per level down to 30° in level 7. The performance of the models is evaluated based on their ability to segment the reconstructed phantoms into material and air accurately.

We tackle this challenge with two different deep learning approaches: an adapted Learned Primal-Dual (LPD) method in Section 3.1 and an Edge Inpainting approach in Section 3.2. We trained both models using synthetic data with simulated measurements; see Section 2. The results are discussed in Section 4.

2. Dataset.

A significant difficulty of the challenge was the need for more data. The challenge organizers provided a small dataset of only 5 pairs of full view sinograms and phantoms [12]. Hence we built methods to generate many pairs of synthetic phantoms and simulated measurements similar to the five provided. We then used this synthetic data to train our two data-driven approaches.

2.1. Modeling of the forward operator.

The challenge data was collected using the University of Helsinki's in-house cone-beam computed tomography scanner. The measurement parameters, e.g., the distance of the source to origin D_{SO}, distance of detector to origin D_{D0}, number of detector pixels, and pixel size, were provided by the organizers. Using the number of detector pixels and the pixel size, one can calculate the length of the detector L_D. The data given was already background and flat-field corrected, and the attenuation data has already undergone log transformation. We used the provided measurement parameters to define a 2D fan beam ray transform operator in ODL [1] using the ASTRA [17] backend. We used this approximated forward operator to generate the new measurements from the phantom created in Section 2.2 and in the LPD model in Section 3.1.

2.2. Data generation.

All of our synthetic phantoms share the following features. They have binary values, which correspond to air and material, respectively. The material always has the same homogeneous density. The discs have a circular shape and fixed size. The center of the disc is positioned randomly around the center of the squared images.

We used four different methods to generate four different types of holes inside these discs (see Figure 2), most of them inspired by the given data.

The first method generates discs with a random number (up to 15) of circular holes. The sizes and locations of the holes are also random, and the holes are not allowed to intersect each other or the boundary.

The second method works analogously but uses randomly drawn polygons instead of circles as holes.

The third method creates holes by drawing a grid of lines inside the disc. At first, a big circular hole, almost the size of the disc itself, is set into the middle of the disc. Then, a grid of randomly drawn lines separates the big hole into smaller holes. The lines of the grid are not perfectly straight, and the orientation of the grid is random. Thus, the shapes of the resulting holes vary significantly.

The fourth method uses a Gaussian mixture model to generate holes. At first, several Gaussian functions, which define a mixture model, are initialized randomly inside the disc. Then, we use a sublevel set defined by the 70% percentile of the pixel values of these combined functions to define where material is present; the remaining parts are the holes.

These methods are fast enough to be used during training to generate training data on-the-fly.

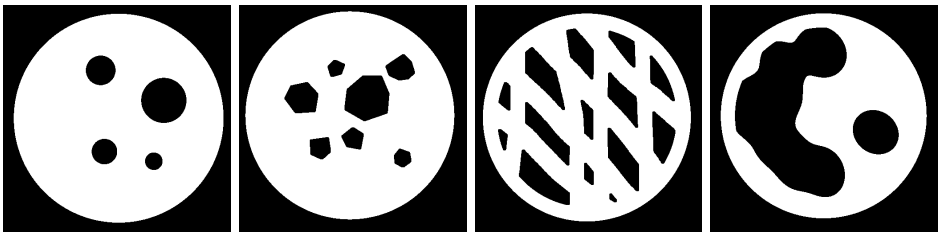

FIGURE 2. Examples of generated synthetic data samples. The different methods for data generation from left to right: circular holes, polygons, grid of lines, and Gaussian mixture.

3. **Methods.**

We develop two deep learning approaches: an adapted version of the LPD method [3] trained directly for segmentation and a sequentially trained Edge Inpainting model. In the following exposition, we drop the subscript $[\varphi_{\min}, \varphi_{\max}]$ and denote the limited-angle Radon transform by A.

3.1. **Learned Primal-Dual.**

In deep learning, selecting an appropriate architecture is crucial to any successful approach. In particular, traditional iterative algorithms, e.g., proximal gradient descent and primal-dual hybrid gradient [8], inspire the architecture class for *learned iterative methods* for inverse problems. One creates these powerful architectures by replacing parts of the iterative algorithms with neural networks – typically the proximal mappings. One can then train these architectures end-to-end, e.g., in a supervised manner [6]. We want to point out that while these learned iterative methods are motivated by classical iterative methods, they do not possess the same theoretical guarantees. A powerful example of a learned iterative method is the Learned Primal-Dual (LPD) proposed by Adler et al. [3]. As our first method, we build upon the LPD architecture by incorporating some common modifications [9].

The LPD swaps the proximal mappings in the dual and primal update with convolutional neural networks $F_{\theta_k} : Y^{N_{\text{dual}}} \times Y \times Y \to Y^{N_{\text{dual}}}$ and $G_{\theta_k} : X^{N_{\text{primal}}} \times X \to X^{N_{\text{primal}}}$. The networks in each unrolling step k have separate weights. We implemented the dual network F_{θ_k} as small residual convolutional networks and the primal networks G_{θ_k} as U-Nets. The use of U-Nets was motivated by the effectiveness of multi-scale architectures when working with images. Further, instead of using the adjoint A^* in the primal update, we use the filtered back projection (FBP) $A^\dagger$ using the Hann filter and a frequency cutoff of 0.75. This choice is motivated by Hauptmann et al. [9], where the authors show that this change can increase performance. The application of a filtered gradient is reminiscent of the Newton method, which has a higher convergence speed than traditional gradient descent. In the challenge, the performance was judged not by the reconstruction but by a downstream segmentation task. We address this by placing a U-Net $T_\theta : X^{N_{\text{primal}}} \to X$ after the last iteration of the LPD. In addition, we normalized the sinograms such that the model is invariant to intensity shifts. Similar to [2], we trained the resulting model directly for the segmentation task by minimizing the binary cross-entropy between the output and the provided segmentation masks. The pseudocode for the full model is given in Algorithm 1. We used $K = 4$ unrolling steps and memory channels $N_{\text{primal}} = 4$ and $N_{\text{dual}} = 2$. In total, the network has 2.4M parameters. The full model is denoted by R_Θ with $\Theta = (\theta, \theta_1, \dots, \theta_K)$ denoting all trainable parameters. The U-Nets used in the primal update and in the segmentation step consist of 4 scales with skip connection on all scales. The specific details of the implementation are given in Table 1.

Applying the forward operator and FBP during each unrolling step produces significant memory requirements during training, limiting our batch size to 6. To address this small batch size, we used group normalization [18] instead of batch normalization.

As with most deep learning models, most of the computational effort is spent in training the network. Once trained, a forward pass through the network requires only 4 evaluations of the forward operator and 5 evaluations of the FBP, which has a similar computational complexity as the adjoint. Thus, the evaluation of the LPD is cheaper than classical iterative methods which may require hundreds of iterations to create a suitable solution.

The evaluation phase of the HTC 2022 was split into 7 levels with decreasing angular ranges. Using the same training configuration, we trained one LPD model instance for each level. The specific angular subset $[\varphi_a, \varphi_b]$ was not fixed and only became known during testing. To robustify the model, we shift the sinograms by $-\varphi_a$ to the angular range $[0°, \varphi_b - \varphi_a]$, input this shifted sinogram to the LPD network, and rotate the output by φ_a to get the initial orientation back. Using this pre- and postprocessing, we can restrict the training of the LPD model to sinograms with a 0° starting angle.

We submitted three variants of this modified LPD model: trained only on the synthetic data[1], additional finetuning on the 5 challenge phantoms[2] and an additional equivariance constraint in the evaluation process[3].

For the first variant, we trained the network instances for 41 666 steps with a batch size of 6 on the synthetic phantoms with simulated measurements having

[1] https://github.com/alexdenker/htc2022_LPD2
[2] https://github.com/alexdenker/htc2022_LPD
[3] https://github.com/alexdenker/htc2022_LPD3

Algorithm 1 Modified Learned Primal-Dual

$x_0 \in X^{N_{\text{primal}}}, h_0 \in Y^{N_{\text{dual}}}$
for $k = 1, \dots, K$ **do**
  $h_k = F_{\theta_k}(h_{k-1}, Ax_{k-1}^{(1)}, y^\delta)$
  $x_k = G_{\theta_k}(x_{k-1}, A^\dagger h_k^{(0)})$
end for
$\hat{x} = T_\theta(x_K)$

TABLE 1. The implementation details of the primal, dual, and segmentation network in the LPD model.

	Primal U-Net G_{θ_k}	Segmentation U-Net T_θ
scales	4	4
channels	16, 32, 64, 64	16, 32, 64, 128
skip channels	16, 32, 32	8,8,8
activation function	Leaky ReLU	Leaky ReLU
downsampling	max pooling	max pooling
upsampling	nearest neighbor	nearest neighbor
kernel size	3	3
	Dual CNN F_{θ_k}	
number of layers	4	
channels	64	
activation function	LeakyReLU	
kernel size	3	

1% relative additive Gaussian noise. In total 250 000 synthetic data samples were generated. We used the Adam optimizer [11] with a batch size of 6 and an initial learning rate of 1×10^{-4}. We used a step learning rate scheduler which decayed the learning rate by 25% every 4166 gradient updates.

In the second variant, we additionally fine-tuned for 2000 steps on random angular subsets of the 5 challenge phantoms. Again, we used the Adam optimizer with a fixed learning rate of 5×10^{-6} and a batch size of 5. In the last variant, we experimented with an equivariance constraint during evaluation. This variant used the same weights as the fine-tuned LPD. We first compute the reconstruction given the limited-angle sinogram, i.e., $\hat{x} = R_\Theta(y)$. We fix a number of angles $0 \leq \alpha_1, \dots, \alpha_T \leq \pi$, rotate the output of the network and simulate new measurements $y_{\alpha_i} = AT_{\alpha_i}\hat{x}$ with T_{α_i} denoting a rotation matrix by the angle α_i. We then compute reconstructions from these rotated measurements $\hat{x}_{\alpha_i} = T_{-\alpha_i} R_\Theta(y_{\alpha_i})$. We then compute the final reconstruction as the mean over the rotations:

$$x = \frac{1}{T} \sum_{i=1}^{T} \hat{x}_{\alpha_i}. \tag{2}$$

In our implementation, we choose $T = 50$. We observed a slight performance increase on our synthetic data and the 5 challenge phantoms.

3.2. **Edge inpainting.** We base our second approach on the work by Bubba et. al. [7], which also deals with limited angle computed tomography. Their reconstruction

method aims to be consistent with the measurement data and reliable in that they use neural networks only for subtasks that model-based methods fail to solve. They use Quinto's fundamental visibility analysis of limited angle CT [15] to realize these objectives. The visibility analysis shows that the reliable recovery of a generalized function f's edges is only possible if the edges are tangent to a line contained in the measured data. An edge not tangent to any measured line is impossible to reconstruct.

Following [7], we call – according to the visibility analysis – recoverable edges 'visible' and all others 'invisible.' Note that we know the measurement geometry prior to reconstruction, and it fully determines edges' (in-)visibility.

In order to calculate the visible and invisible edges of a generalized function f, one can use the wavefront set of f, which is the set of positions of the singular support of f and their unsmooth directions. We refer the reader to [10] for a detailed discussion. Bubba et. al. [7] use the shearlet transform to determine the wavefront set and, in particular, the (in-)visible singularities of a function. To be more precise, they apply a variational approach with an ℓ^1-penalty in the shearlet domain to solve a sparse regularization problem and determine the visible singularities of the target function. They then use the visible coefficients of the shearlet transform as input to a neural network, which they train to estimate the invisible shearlet coefficients. In the final step, they combine the visible and invisible coefficients predicted by the classical method and the neural network, respectively, to obtain a reconstruction of the target via the inverse shearlet transform.

The setting in [7] is rather general as the algorithm can reconstruct any objective function f. In contrast, the objective functions in the challenge have a specific structure, and the goal was not to reconstruct the objective function but to create a segmentation of it. Therefore, we adapted the approach of [7] to the setting in the challenge. The main difference is that we do not use the shearlet transform but estimate the wavefront set using gradients. Similarly, we start with a variational approach using Perona-Malik and a W-shaped functional for regularization. We then use the resulting reconstruction to estimate the visible edges by calculating the gradient field and discarding all gradients corresponding to invisible edges. The visible edges are used as input to a neural network with the task of inpainting the invisible edges. In the final step of our pipeline, we use a second neural network to create a segmentation mask from this inpainted output.

Once the full pipeline is trained, both the inpainting and segmentation network can be evaluated quite fast. The main computational complexity lies in obtaining the visible edges via a variational regularization as this is done with an iterative algorithm with many evaluations of the forward operator and its adjoint. Thus, the complexity of our method is comparable to classical reconstruction methods.

In the following paragraphs, we discuss the pipeline's individual steps; see also Figure 3. The complete reconstruction scheme and all the weights of the networks are available on GitHub[4].

Classical reconstruction. First, we normalize the input sinogram data to make the reconstruction method invariant to changes in the intensity of the phantoms. We then define our variational approach via

$$\min_{x \in X} \frac{1}{2}\|Ax - y\|^2 + R(x), \tag{3}$$

[4] https://github.com/arndt-c/htc2022_edge_inpainting

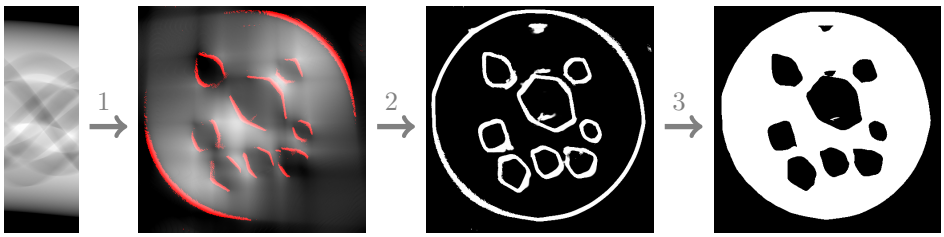

FIGURE 3. Visualization of the steps of the Edge Inpainting method: (1) variational reconstruction and extraction of visible edges, (2) inpainting of invisible edges, (3) segmentation.

where the penalty term R is a combination of the Perona-Malik function [5]

$$G(s) = \frac{T^2}{2}\left(1 - \exp\left(-\frac{s^2}{T^2}\right)\right) \quad \text{for } s = \|\nabla x_{i,j}\|_2 \tag{4}$$

and a W-shaped functional

$$W(s) = \frac{(s-a)^2(s-b)^2}{(b-a)^3} \quad \text{for } s = x_{i,j}. \tag{5}$$

Here, a and b are the air and material intensity levels, respectively, estimated according to the 5 given challenge phantoms.

We use Perona-Malik to support the subsequent edge extraction, as it sharpens the edges and smooth areas between them. The binary nature of the phantoms motivates the W-shaped functional. The choice $R(x) = \sum_{i,j} 2000\, G(\|\nabla x_{i,j}\|_2) + W(x_{i,j})$ is a suitable weighting of the penalty terms.

We solve the minimization problem (3) via alternating gradient steps w.r.t. the data discrepancy (the gradient is $A^*(Ax - y)$) and the regularization terms implemented using automatic differentiation in PyTorch. We also include optimization steps w.r.t. the values a and b to make the penalty term adaptive. Due to the high computational cost of the Radon transformation and its adjoint, we restrict the number of iteration steps to 40.

Alternatively, one could use the filtered back projection (FBP) instead of the variational method for the reconstruction. Despite its computational complexity, we chose the variational approach over the FBP as it demonstrates significantly higher robustness to noise (see Figure 4). Similar observations were also made in [7].

Estimation of visible edges. Since the overall goal is a segmented image, we only need information about the edges of the reconstruction. To obtain these, we compute the gradient field of the reconstruction from the previous step using a convolution with the Laplace filter. We then use a threshold to reject gradients with a small magnitude; all remaining gradients correspond to edges in the image. That is why Perona-Malik regularization, which prefers sharp edges and smooth areas in between, is beneficial.

The gradients also contain information about the corresponding edges' orientation (an angle between zero and 360°). If the angular range of the CT measurements is $[\varphi_{\min}, \varphi_{\max}]$, all gradients whose orientation is in one of the intervals $[\varphi_{\min} - 90, \varphi_{\max} - 90]$ or $[\varphi_{\min} + 90, \varphi_{\max} + 90]$ correspond to visible edges in a

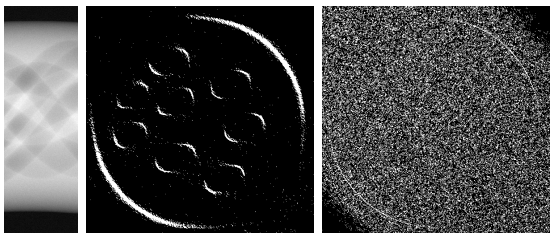

FIGURE 4. Here we compare the extracted visible edges via the variational method (middle) and the FBP (right). Both methods use the noisy sinogram on the left with 3% relative additive Gaussian noise.

parallel beam geometry. For the given fan beam geometry, we shrink the intervals of visible angles symmetrically by 10° to avoid unwanted invisible edges.

Inpainting of invisible edges with a U-Net. After extracting visible edges, we use a neural network to predict the invisible edges. The network's inputs consist of binary images, where we encode the visible edges via '1's. The output is also a binary image with visible and invisible edges encoded by '1's. The network consists of a U-Net [16] of depth 6 with skip connections on all scales. For an overview of the training and architecture parameters, see Table 2. As discussed above, we apply group normalization after each convolution due to the small batch size. The activation function is set to LeakyReLU with a negative slope of 0.2 at all layers except the last one, where we use sigmoid to enforce the output to be in the interval $[0, 1]$. In the decoder part of the U-Net, we use nearest neighbor interpolation to upsample the input of the respective blocks as it produces non-smooth images, which is beneficial given that the output should be binary.

We train the network for 16 000 steps using the Adam optimizer [11] with a learning rate of 2×10^{-5} and a batch size of 4 due to memory constraints. In total we used 64 000 samples for training. As in the LPD case, we train separate instances for each angular range, resulting in 7 different sets of network weights. We chose the weighted binary cross entropy (BCE) as our loss function with a weight on edges, i.e., areas with value '1', cf. [19].

The training data consists of on-the-fly generated synthetic phantoms, see Section 2.2, where we calculate visible and invisible edges (input and target, respectively) using the approach described in the preceding step. We want to stress that we do not use the variational approach to calculate the visible edges as it would slow down the training tremendously. We expect the overall results to be slightly better when training with visible edges extracted from reconstructions of the variational approach. However, this would require a fixed dataset, which we do not consider beneficial in this setting.

Further, we want to point out that the network can alter the extracted visible edges; hence our method, unlike the one presented in [7], does not provide any real guarantees. We opted for this approach as it simplifies the pipeline, and we observed that the inpainting network did not change the input edges significantly.

Segmentation with a U-Net. The last step of our pipeline consists of segmenting the output of the edge inpainting network. For this, we use the same U-Net architecture

TABLE 2. Details of the training setup and U-Net's architecture for the inpainting and segmentation task.

Kernel size:	9×9	Channels:	16, 32, 64, 128, 256, 256
Scales:	6	Skip channels:	16, 32, 64, 128, 256
Parameters:	≈ 34M	Downsampling:	max pooling
Optimizer:	Adam	Upsampling:	nearest neighbor
Batch size:	4	Activation:	LeakyReLU
Loss function:	weighted BCE	Overall gradient steps:	16000

as for the inpainting network and train the network using the synthetic data of Section 2.2 with the same training parameters except for the learning rate of 1×10^{-5} (see Table 2). The segmentation task does not depend on the angular range of the sinogram. However, we train the network for each angular range separately as we use the output of the previous edge inpainting network as input, which depends on the angular range.

4. **Results.** To evaluate our models, we show results on the test data of the challenge and out-of-distribution data. We detail in which cases the models show a good performance and the characteristics of the reconstruction errors – occurring, especially in more difficult levels. An overview of all results (scores and reconstructions from all methods of all participating teams) can be found on the challenge website[5].

4.1. **Challenge data.** The test set of the challenge consists of three different phantoms per angular range. The complexity of the phantoms increases as the viewing angle decreases, i.e., the phantoms contain more holes of different shapes (see Figures 7 - 10). For evaluating the different methods, the challenge organizers use the Matthews correlation coefficient (MCC) between ground truth segmentation masks I_t and segmentation calculated from the reconstruction I_r. One defines the MCC as

$$S = \frac{TP \cdot TN - FP \cdot FN}{\sqrt{(TP + FP)(TP + FN)(TN + FP)(TN + FN)}}$$

where TP denote the number of true positives, TN the number of true negatives, FP the number of false positives and FN the number of false negatives. The score S is a number between -1 and 1 where 1 indicates a perfect match and -1 indicates a total mismatch between I_r and I_t. For comparing the performance of the different methods on each level, the organizers use the overall score defined as the sum of the scores of the reconstructions on three different phantoms A, B, C for each level, i.e.,

$$S_N = S_N^A + S_N^B + S_N^C \quad \text{with } N \in \{1, \dots, 7\}.$$

For more details on the challenge's scoring system, we refer the reader to the challenge website[6].

Our best performing method on the test data was the fine-tuning LPD method without equivariance postprocessing. We depict the overall scores of the best LPD variant, the Edge Inpainting method, and the method of the challenge's winning team (provided by the challenge organizers) in Figure 5. LPD results in the best

[5] https://www.fips.fi/HTCresults.php
[6] https://www.fips.fi/HTC2022.php

scores up to level 5 and is merely slightly worse on the last two levels than the winning team's method. The Edge Inpainting method scores well on the first three levels, but the performance drops significantly from level 4 onwards.

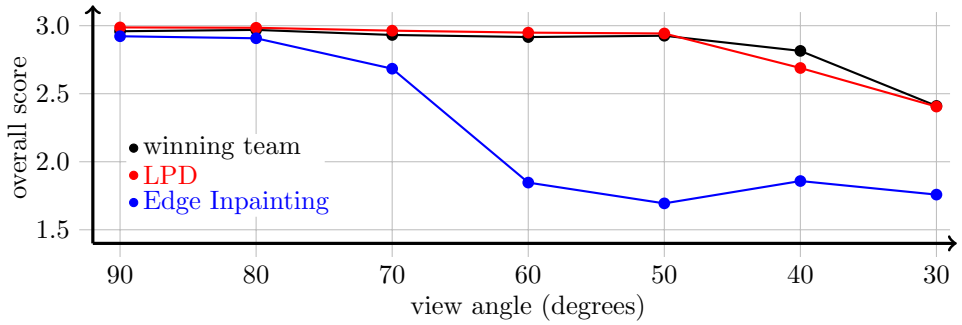

FIGURE 5. The official overall scores of our main methods in the challenge on the different levels in comparison with the method of the winning team. Accessed at https://www.fips.fi/HTCresults.php.

Figure 6 shows the overall scores of the different variants of the LPD method. One can observe that LPD's fine-tuning on the provided training data significantly boosted performance. Focusing on the fine-tuned model with equivariance post-processing, one can see that the post-processing step decreases the model's performance. With the equivariance modification, we aimed to increase the robustness of the model. However, as this modification only affects the inference of the model, it was not clear at the beginning whether this post-processing step is beneficial.

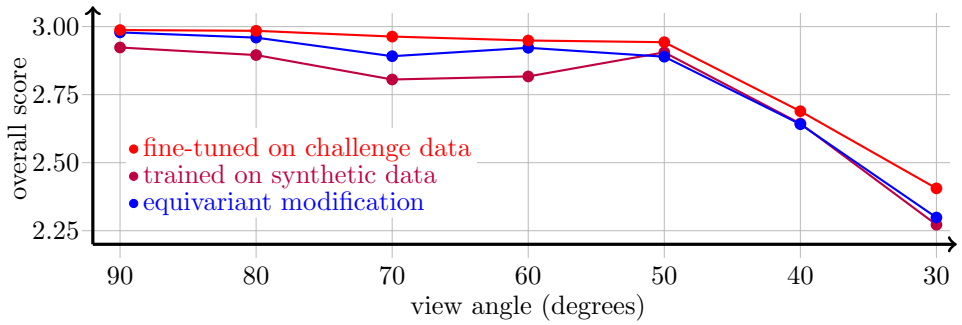

FIGURE 6. The official scores of the different variants of the LPD in the challenge.

We depict the reconstructions of the Edge Inpainting method and the fine-tuned LPD method for the three different phantoms of level 1, 3, 5, and 7 in Figures 7, 8, 9 and 10. Focusing on the reconstructions of the Edge Inpainting method, one can observe that in levels 1 and 3, the method can approximate most of the holes correctly. Nevertheless, some reconstruction errors are visible; still, they have a relatively small impact on the score. In levels 5 and 7, the inpainting method cannot correctly reconstruct the holes and shapes, which is consistent with the

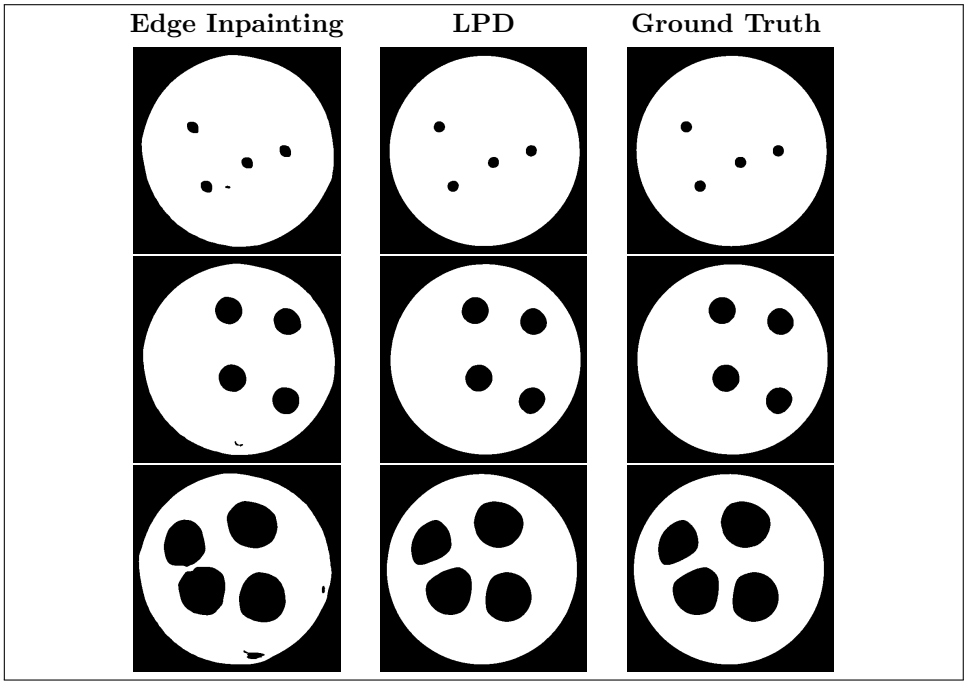

FIGURE 7. Reconstructions of the Edge Inpainting method and of LPD (fine-tuned variant) in **level 1** (90°).

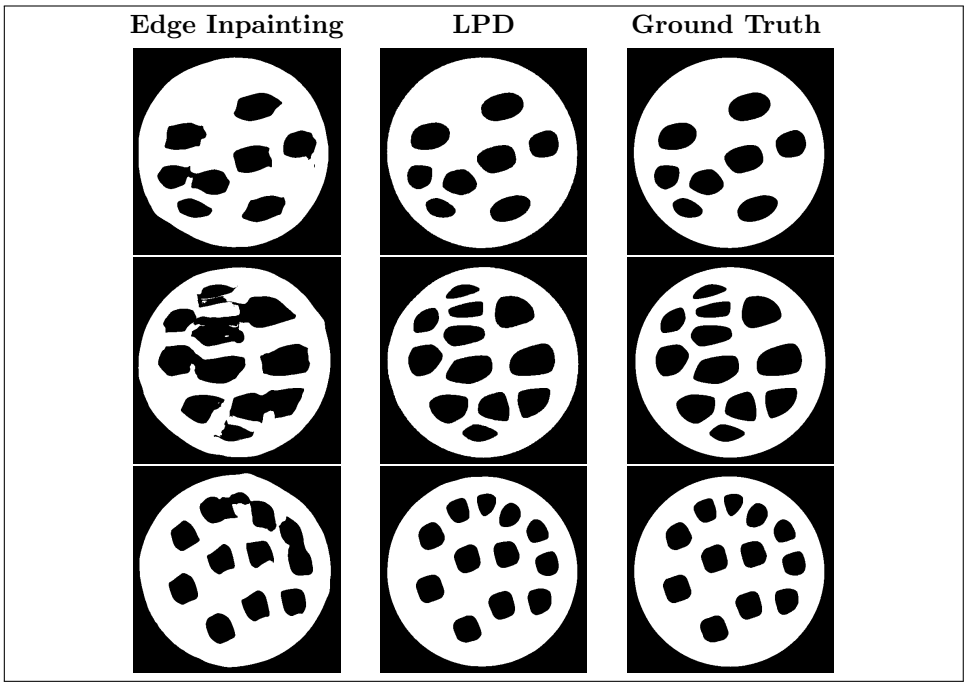

FIGURE 8. Reconstructions of the Edge Inpainting method and of LPD (fine-tuned variant) in **level 3** (70°).

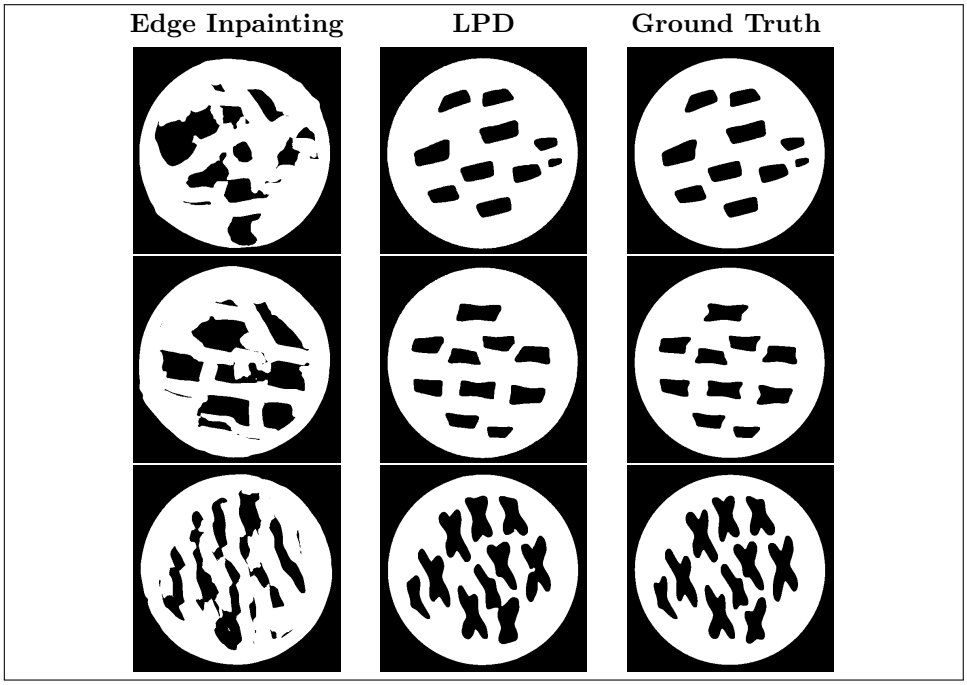

FIGURE 9. Reconstructions of the Edge Inpainting method and of LPD (fine-tuned variant) in **level 5** (50°).

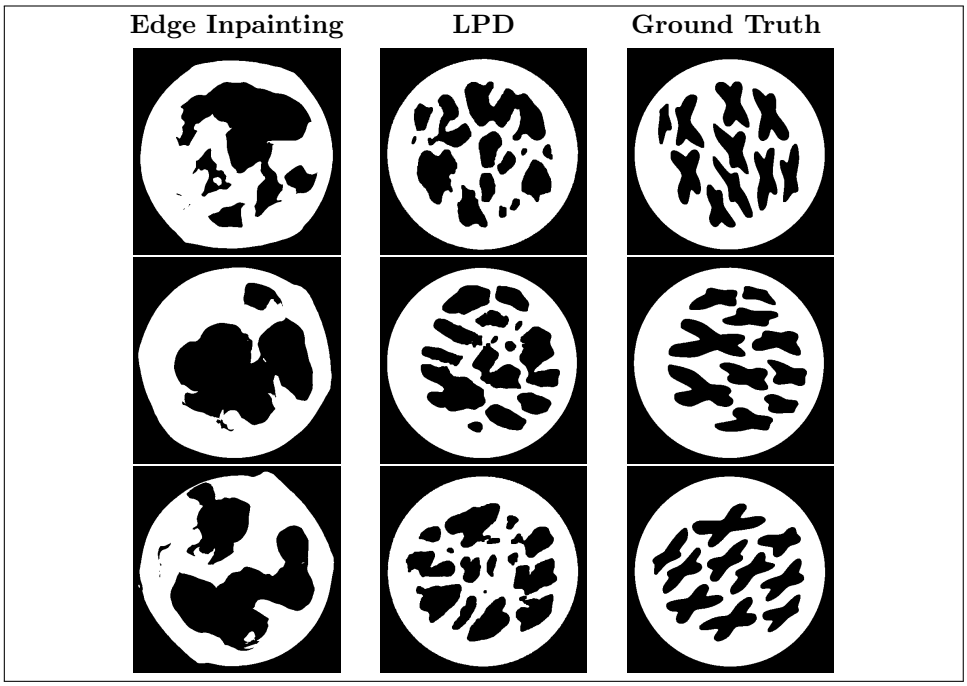

FIGURE 10. Reconstructions of the Edge Inpainting method and of LPD (fine-tuned variant) in **level 7** (30°).

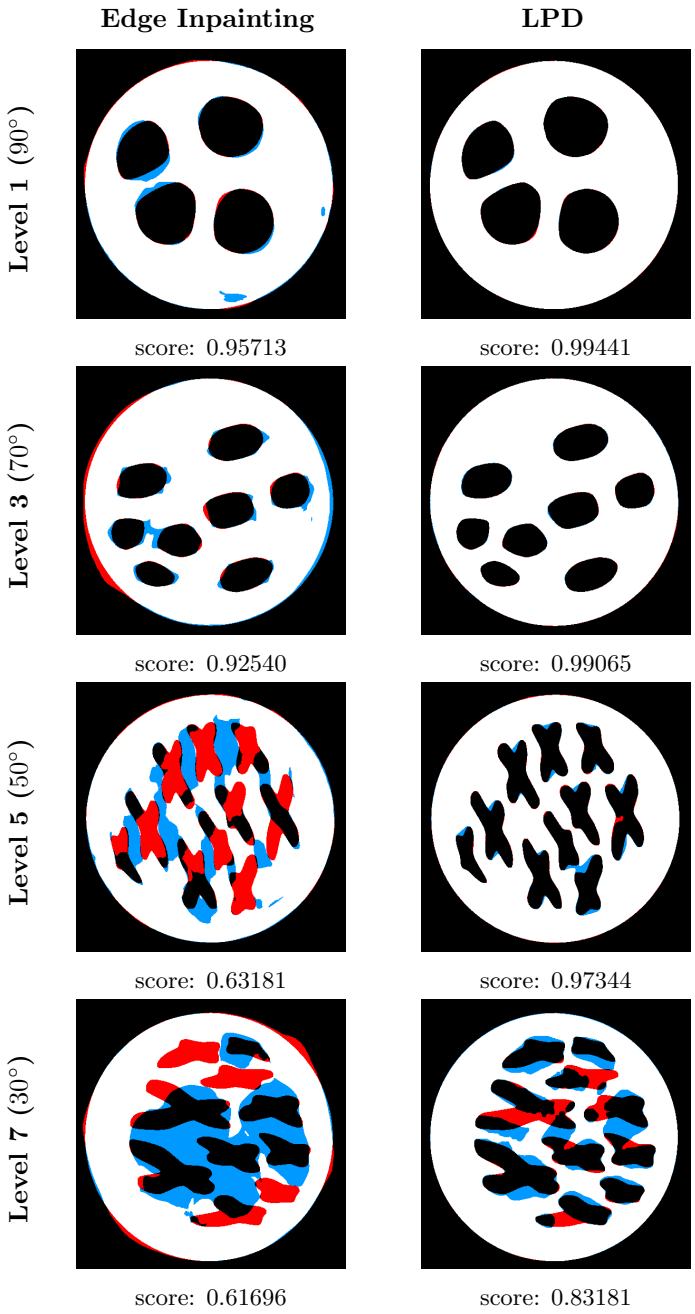

FIGURE 11. Visualization of the reconstruction errors for the Edge Inpainting method and LPD (fine-tuned variant) in **level 1** (90°), **3** (70°), **5** (50°), **7** (30°): white = true positive (material), black = true negative (air), red = false positive, blue = false negative.

results of Figure 5. Moreover, we can see that at all levels, the reconstructed outer discs are often not perfectly circular. With a suitable fine-tuning of the method, this error should, in principle, be avoidable since the data contains only circular disks. However, as the challenge's organizers did not officially specify the disk, we opted for a less restrictive method.

In contrast to the Edge Inpainting method, the fine-tuned LPD method results in nearly perfect reconstructions up to level 5 with very few visible reconstruction errors. In level 7, LPD can roughly estimate the holes' location but can no longer correctly reconstruct the shapes and the borders between them. Moreover, comparing the reconstructions of the different shapes, one can observe that LPD tends to fail to reconstruct non-convex holes. This is probably due to the underrepresentation of similar holes in the synthetic data.

We further illustrate the results and observations of the previous sections in Figures 11 and 12. The figure exemplifies our methods' true positives, true negatives, false positives, and false negatives at levels 1, 3, 5, and 7, respectively.

(A) score: 0.89855 (B) score: 0.84846 (C) score: 0.88876

FIGURE 12. Visualization of the reconstruction errors for the LPD variants (A) fine-tuned, (B) pre-trained, (C) fine-tuned and equivariance) in level 6 (40°): white = true positive (material), black = true negative (air), red = false positive, blue = false negative.

4.2. **Out-of-distribution data.** To test the generalization ability of the networks, we created three different out-of-distribution phantoms depicted in Figure 13.

We visualize the reconstructions, and the results of the intermediate steps of the Edge Inpainting method on level 1 (90°) in Figure 14. One can observe that

FIGURE 13. Example phantoms, which look significantly different from the challenge phantoms for testing the models on out-of-distribution data.

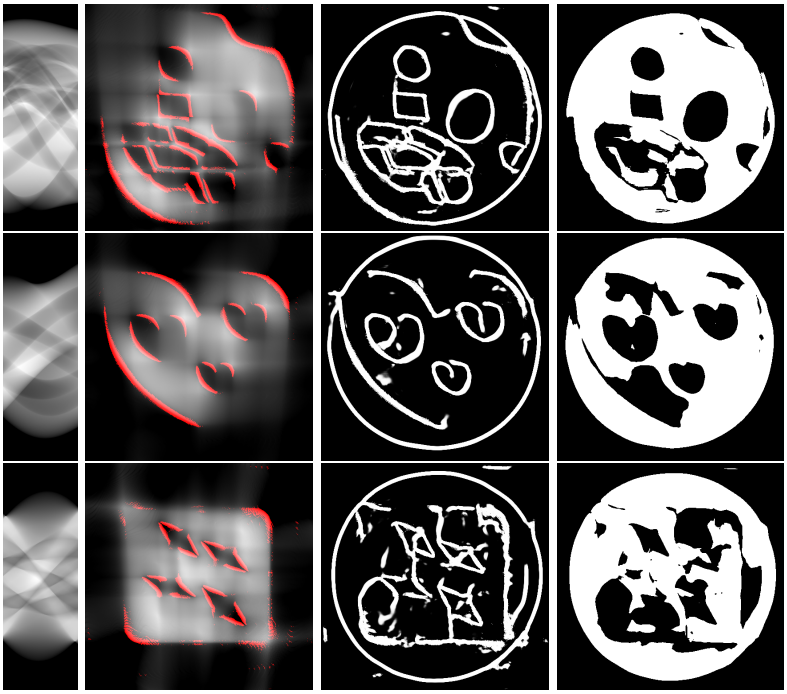

FIGURE 14. Reconstructions and intermediate steps of the Edge Inpainting method on out-of-distribution data with angular range of 90°.

the visible edges are at least visually correctly estimated, and the Edge Inpainting method does not significantly alter them. However, the Edge Inpainting method fails to connect the different shapes correctly and draws a circle in every image. This makes image segmentation exceptionally difficult, which is visible in the segmented reconstructions. These results show that the networks unsurprisingly completely adapted to the training data. Moreover, the combination of two networks might not be beneficial in this setting as the errors made by the edge inpainting network seem to get amplified in the segmentation network resulting in a loss of visible edges. Therefore, joint training of the full segmentation pipeline might lead to better results.

The LPD methods result in nearly perfect reconstructions at level 1 (90°) but deteriorate at level 2 (80°), which is why we only show the reconstructions at level 2 in Figure 15. The LPD methods can approximate the phantom's outer shape but attempt to form circles in regions with invisible edges. In addition, the network trained on synthetic data only (not fine-tuned) leads to the best-looking reconstructions except for the stars in the third phantom. Thus, there seems to be a trade-off between a high fitting on the challenge data and a good generalization ability.

5. **Discussion.** The results and observations from Section 4 motivate some ideas for further improvements of the methods.

Two phenomenons can explain many reconstruction errors of the Edge Inpainting method. First, the inpainting network connects the wrong visible edges or fails to

LPD pre-trained on synthetic data

LPD with additional fine-tuning on challenge data

LPD with additional equivariant postprocessing

FIGURE 15. Reconstructions of the LPD variants on out-of-distribution data with angular range of 80°.

connect them. Second, the segmentation network labels the wrong areas as holes. One possibility to avoid this is to provide the networks with information about the orientation of the edges [4].

Besides, the sequential training of the two U-Nets in the Edge Inpainting method might not be beneficial. Instead, one could test an end-to-end approach or joint training of both networks; see, e.g., [2]. One could also test the approach of [7] (explained at the beginning of Section 3.2) on the challenge data and compare it with our simplified method.

For the LPD approach, the results show that this method performs very well. Nevertheless, it does not provide any reconstruction guarantees, which can be in real-world scenarios. One could combine the LPD method and the concept of visible edges to overcome this shortcoming. One could estimate the uncertainty as a first step by comparing the edges in the LPD reconstruction with the visible edges in a classical reconstruction.

If the visible edges between the classical reconstruction and the LPD approach do not align, one could correct the LPD reconstruction using the correct visible edges

of the classical reconstruction. For further insights into reconstruction guarantees in Deep Learning, we refer the reader to [13].

6. **Conclusion.** We show that it is possible to apply data-driven methods in a small data setting using a suitable chosen simulated dataset. However, the choice of simulated data is crucial, and this is only possible if a reliable approximation of the data distribution is possible in advance. Further, the generalization to new data is still an open question, as could be observed in the OOD experiment in Section 4.2. Combining model-based reconstruction with data-driven components is a promising research direction, as it ensures consistency with measured data while still providing great flexibility.

Acknowledgments. Alexander Denker and Johannes Leuschner acknowledge the support by the Deutsche Forschungsgemeinschaft (DFG) within the framework of GRK 2224/1 "π^3: Parameter Identification – Analysis, Algorithms, Applications". Maximilian Schmidt acknowledges the financial support by the German Federal Ministry for Economic Affairs and Climate Action (BMWK) and the European Social Fund (ESF) within the EXIST Transfer of Research project "aisencia".

REFERENCES

[1] J. Adler, H. Kohr and O. Öktem, Operator discretization library (ODL), *Zenodo*, (2017).
[2] J. Adler, S. Lunz, O. Verdier, C.-B. Schönlieb and O. Öktem, Task adapted reconstruction for inverse problems, *Inverse Problems*, **38**, (2022), Paper No. 075006, 21 pp.
[3] J. Adler and O. Öktem, Learned primal-dual reconstruction, *IEEE Transactions on Medical Imaging*, **37**, (2018).
[4] H. Andrade-Loarca, G. Kutyniok, O. Öktem and P. Petersen, Deep microlocal reconstruction for limited-angle tomography, *Applied and Computational Harmonic Analysis*, **59** (2022), 155-197.
[5] S. R. Arridge, M. M. Betcke and L. Harhanen, Iterated preconditioned LSQR method for inverse problems on unstructured grids, *Inverse Problems*, **30** (2014), 075009, 27 pp.
[6] S. Arridge, P. Maass, O. Öktem and C.-B. Schönlieb, Solving inverse problems using data-driven models, *Acta Numerica*, **28** (2019), 1-174.
[7] T. A. Bubba, G Kutyniok, M. Lassa, M. März, W. Samek, S. Siltanen and V. Srinivasan, Learning the invisible: A hybrid deep learning-shearlet framework for limited angle computed tomography, *Inverse Problems*, **35** (2019), 064002, 38 pp.
[8] A. Chambolle and T. Pock, An introduction to continuous optimization for imaging, *Actra Numerica*, Cambridge University Press, **25** (2016), 161-319.
[9] A. Hauptmann, J. Adler, S. Arrdige and O. Öktem, Multi-scale learned iterative reconstruction, *IEEE Transactions on Computational Imaging*, **6**, (2020).
[10] L. Hörmander, *The Analysis of Linear Partial Differential Operators. I. Distribution Theory and Fourier Analysis (reprint of the 2nd edn 1990)*, Springer, Berlin, 2003.
[11] D. P. Kingma and J. Ba, Adam: A method for stochastic optimization, arXiv preprint, `arXiv:1412.6980`, (2014).
[12] A. Meaney, F. Silva de Moura and S. Siltanen, Helsinki Tomography Challenge 2022 open tomographic dataset (HTC 2022), *Zenodo*, 2022.
[13] S. Mukherjee, A. Hauptmann, O. Öktem, M. Pereyra and C.-B. Schönlieb, Learned reconstruction methods with convergence guarantees: A survey of concepts and applications, *IEEE Signal Processing Magazine*, **40** (2023), 164-182.
[14] F. Natterer, *The Mathematics of Computerized Tomography*, SIAM, 2001.
[15] E. T. Quinto, Singularities of the X-ray transform and limited data tomography in $\mathbb{R}^2$ and $\mathbb{R}^3$, *SIAM J. Math. Anal.*, **24**, (1993), 1215-1225.
[16] O. Ronneberger, P. Fischer and T. Brox, U-net: Convolutional networks for biomedical image segmentation, *Medical Image Computing and Computer-Assisted Intervention – MICCAI 2015*, Springer, **9351** (2015), 234-241.

[17] W. van Aarle, W. J. Palenstijn, J. De Beenhouwer, T. Altantzis, S. Bals, K. Joost Batenburg and J. Sijbers, The ASTRA toolbox: A platform for advanced algorithm development in electron tomography, *Ultramicroscopy*, **157** (2015), 35-47. .

[18] Y. Wu and K. He, Group normalization, *Proceedings of the European Conference on Computer Vision (ECCV)*, (2018), 3-19.

[19] S. Xie and Z. Tu, Holistically-nested edge detection, *IEEE International Conference on Computer Vision (ICCV), Santiago, Chile*, (2015), 1395-1403.

Received June 2023; revised October 2023; early access October 2023.