Kuopio Tomography Challenge 2023 (KTC2023)

Mikko Räsänen, Petri Kuusela, Jyrki Jauhiainen,
Muhammad Arif, Kenneth Scheel, Tuomo Savolainen,
Aku Seppänen

June 2023

Contents

1 Introduction

The Finnish Inverse Problems Society (FIPS) proudly presents the Kuopio Tomography Challenge 2023 (KTC2023). We invite all scientists and research groups to test their electrical impedance tomography (EIT) reconstruction algorithms on our real-world data. The top participants of the challenge will be invited to a minisymposium at the Inverse Days Conference to be held in Lahti, Finland, in December 2023.

1.1 What is Electrical Impedance Tomography?

In electrical impedance tomography, the internal electric conductivity distribution of a physical body is reconstructed using electric current and voltage measurements taken at the boundary of the object. Mathematically, the problem is to recover a non-negative coefficient of a diffusion equation from boundary data consisting of electric current density and potential.

In real measurement setups, the object is imaged using measurement electrodes, which have a finite size and which cover only parts of the object boundary. The significance of this is that the reconstruction must be computed from an incomplete set of boundary data, a highly ill-posed and challenging task.

2 Challenge Description

The purpose of the challenge is to recover the shapes of 2D targets imaged with electrical impedance tomography, collected in the Process Tomography Laboratory at the University of Eastern Finland, Finland. The experimental setup, targets, and measurement protocol are described in the following sections.

The outcome of the challenge should be an algorithm which takes in the EIT data, and its associated metadata about the measurement geometry, and produces a reconstruction which has been segmented into three components: background (water), resistive inclusions, and conductive inclusions. The challenge dataset is divided into multiple difficulty levels, with the data becoming more limited as the difficulty level increases.

2.1 Targets

The targets consist of a circular water tank with an inner diameter of 23.0 cm, and various inclusions placed into the tank. To create different imaging targets, varying numbers and types of inclusions were placed into the water. The inclusions were either resistive compared to the water (3D printed PLA plastic), or more electrically conductive than the water (electrically conductive 3D printed plastic or metal).

The data collected for the KTC2023 challenge consists of two separate sets, with identical experimental setup and settings. One set is provided to the competitors as training set for algorithm development, and the other will be used by the organizers to test the reconstruction algorithms. The test set will be made public after the end of the competition.

2.2 Training Dataset

The training dataset can be downloaded here `https://doi.org/10.5281/zenodo.8252370`.

The training dataset consists of five phantoms. One of them is the water tank filled with only water, and its measurements can be used as a reference

data to mitigate the effects of systematic errors. The other four of the training phantoms contained inclusions.

Training data for each difficulty level can be created by subsampling these datasets to create limited data that match the active measurement electrodes in each level of the actual challenge dataset (See Table 1). We encourage subsampling the training datasets to create limited EIT data, and comparing the reconstruction results to the result obtainable from the full EIT data collected using all of the measurement electrodes.

We have also included examples of the segmented image reconstructions to demonstrate the desired format for the final competition entries. The segmented image reconstructions were obtained by the following steps:

- Compute a linear difference EIT reconstruction using the reference phantom data and data from a phantom containing inclusions
- Interpolate the resulting EIT reconstruction from the finite element mesh into a 256 by 256 pixel image
- Divide the image pixels into three classes using triclass Otsu's method

For a more detailed explanation on the example reconstruction algorithm, we refer to Appendix B.

The competitors do not need to follow the above reconstruction and segmentation procedure, and are encouraged to explore various techniques to generate the segmented EIT image reconstructions. The competitors are encouraged to generate extra training data using simulations. The organizing committee will provide a Matlab and Python code to solve the forward problem of EIT (i.e. simulate the data) for a given simulated electric conductivity phantom. It's up to the competitors to generate simulated phantoms for training data generation.

The test targets have been photographed using a digital camera. The resulting photographs have been segmented to create ground truth images, which will be used to evaluate the results obtained by the submitted algorithms using the scoring system given in Section 3.3. In the training dataset, the ground truth images corresponding to the targets are provided. Figure 1 shows the positioning of the circular water chamber in the square ground truth images. The pixel centers are given by $[x, y] \in \{-R+\frac{w}{2}, -R+\frac{3w}{2}, ..., R-\frac{w}{2}\} \times \{-R+\frac{w}{2}, -R+\frac{3w}{2}, ..., R-\frac{w}{2}\}$, where R is the radius of the inner boundary of the water chamber and $w = \frac{2R}{256}$ is the pixel width.

2.3 Data for the challenge

The actual challenge data consists of 21 phantoms, arranged into seven groups of gradually increasing difficulty, with each level containing three different phantoms, labeled A, B, and C. As the difficulty level increases, the number of inclusions increases and their shapes become increasingly complex. Additionally, the data is made more limited as the difficulty level increases by discarding the measurements from some of the electrodes (Table 1).

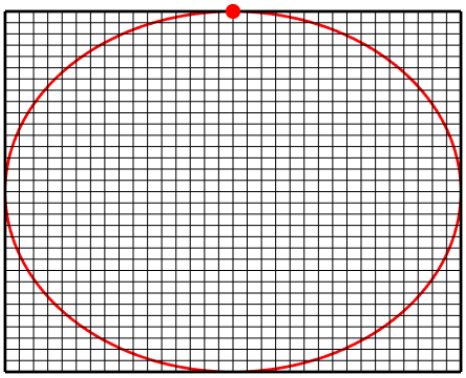


Figure 1: Positioning of the circular water chamber in the ground truth pixel images. The red circle represents the inner boundary of the water chamber. For visual readability, the figure has eight times the pixel width used in the challenge, i.e. 32 by 32 pixels. The center point of electrode 1 is marked at the top of the figure. In the ground truth images, pixels whose center is outside the circular boundary are assigned the value 0, same as the pixels whose center coincides with the water segment.

The test targets have been photographed using a digital camera. The resulting photographs have been segmented to create ground truth images, which will be used to evaluate the results obtained by the submitted algorithms using the scoring system given in Section 3.3.

2.4 Dataset file format

The dataset is shared using the MATLAB .mat files (version 7). For the training dataset, the ground truth images are included. The shared example codes demonstrate how to read .mat files in Python.

3 Rules of the Competition

3.1 How to enter the competition

Deadline for registration: Register before 23.59 EET (Eastern European Time) on 30th of September 2023, using this electronic form.

3.2 Deadlines and what needs to be submitted

Send your submission to ktc2023("at")fips.fi before 5th of November 2023, 23.59 EET (Eastern European Time).

What needs to be submitted? Briefly, the codes must be in Matlab or Python 3.X and the algorithms must be shared with us as a private GitHub

repository at latest on deadline. Check the following subsections for detailed instructions. Only submissions that fulfill the requirements listed below will be accepted. The teams can submit more than one reconstruction algorithm to the challenge, however, each algorithm must be in a separate repository. The maximum number of algorithms is the number of members of the team. Your team do not need to register multiple times in case you decide to submit more than one algorithm to the challenge. The team can send a single email with the links to all the repositories.

After the deadline, there is a brief period during which we can troubleshoot the codes together with the competing teams. This is to ensure that we are able to run the codes. The troubleshoot communication is done mainly via 'Issues' section of the submitted repository, so pay attention to any activities in the repository after the deadline.

3.2.1 How we expect your code to be

1. Your code must contain a main routine that we can run to apply your algorithm automatically to every data file in the input directory. This is the file we will run to evaluate your code. Give it a name that is easy to identify, like main.m or main.py. Important: The input directory contains only the test dataset. No training dataset is provided to your code during the assessment. Therefore, any training procedures must be performed by your team before the submission.

2. Your main routine must require three input arguments, in this order:

(a) (string) Folder where the input files are located (.mat files)

(b) (string) Folder where the output files must be stored

(c) (int) Group category number, following Table 1. (Values between 1 and 7)

You can assume we are providing the full path to the directories.

3. The main routine must produce one .mat file containing a 256 by 256 pixel array of the reconstructed target in the output folder for each dataset file in the input folder, except 'ref.mat'. In other words, your code must go through all .mat files (except 'ref.mat') in the input folder, producing one reconstruction per file. The input data files are named data1.mat, data2.mat, and so on. The .mat file with the reconstruction must be named with the number of the input data (for example 1.mat, 2.mat etc.), and be located in the given output folder. You can assume all files in the input folder belongs to the same difficulty level, provided as the third argument to your main function. The reference data 'ref.mat' will be given in each input file folder. It will be the same as in the shared training data, so note that full reference data is available even at harder difficulty levels, where the target data is limited.

Input file format: .mat file containing:

- injected currents 'Inj' (or 'Injref' in ref.mat), a 32 x 76 matrix, containing in each column the current injected through each of the 32 electrodes

- measured potentials 'Uel' (or 'Uelref' in ref.mat), a vector of length 2356, a vectorization of a 31 x 76 matrix containing on each column the 31 potential measurements as described in Section 4

- the measurement pattern 'Mpat', a 32 x 31 matrix, encoding the information how the potentials are measured (which is explained in Section 4). This is the same in all of the data files.

Note: At higher difficulty levels some of the measurement data is left out as show in in Table 1. These data entries will be replaced by NaN in the input data, so make sure your codes can handle that!

Output file format: .mat file containing a 256 x 256 array named 'reconstruction' with the following possible values for the elements: 0 (water), 1 (resistive inclusions) or 2 (conductive inclusions).

4. The teams are allowed to use freely available Python modules or Matlab toolboxes. Toolboxes, libraries and modules with paid licenses can also be used if the organizing committee also has the license. For example, the most usual Matlab toolboxes for image processing and deconvolution can be used (Image processing toolbox, wavelet toolbox, PDE toolbox, computer vision toolbox, deep learning toolbox, optimization toolbox). The teams can contact us to check if other toolboxes are available.

For getting started, we recommend taking a look at the simple reconstruction algorithm and EIT forward problem solver provided by the organizers. This code package is provided for both Matlab and Python.

Note that using the provided codes (incl. the provided forward solver) for working with the data is by no means compulsory. The competitors are free to use any and all computational tools they want to in computing the reconstructions and segmentations, as long as the organizing committee also has access to them.

3.2.2 How we expect your Github repository to be

1. Algorithms must be shared with us as a private GitHub repository at latest on deadline.
2. Competitors can update the contents of the repository as many times as needed before the deadline, even if they have already sent the link to the organizing committee. We will consider only the latest RELEASE of your repository on Github up to the deadline. Attention: Simple commits to the main branch will not be considered. You MUST create a release with your final version for the competition. Please see Github's documentation on how to create releases. We will only consider the latest release before the deadline for the competition. If the latest release does not work, we will not try with older releases.
3. If your algorithm requires uploading large files to Github (e.g. with trained coefficients of a neural network), you can use Git Large File Storage (preferable way) or store them in another server and add the link to the Github installation instructions.

4. Your repository must contain a README.md file with at least the following sections:

- Authors, institutions, addresses.
- Brief description of your algorithm and a mention of the competition. Here you can also refer to published works of the team related to the algorithm you implemented.
- Installation instructions, including any requirements. Matlab users: Please specify any toolboxes you use. Python users: Please specify the modules you use. If you use anaconda, please add to the repository an environment.yml file capable of creating an environment than can run your code. Otherwise, please add a requirements.txt file generated with pip freeze.
- Usage instructions.
- Present a few examples of the reconstructions from the training set.

Finally, the competitors must make their GitHub repositories public at latest on 30th of November, 2023. In the spirit of open science, only a public code can win KTC2023.

3.3 Scores and Leaderboard

In short, the final score each algorithm gets is the sum of individual scores given for each target, provided the algorithm passes every level. The detailed procedure of constructing the scores and leaderboard is as follows:

1. All teams start with difficulty level 1. The reconstructions of the three samples (A, B, and C) of this level will be assessed quantitatively following the criteria described in Section 3.3.1, and their scores will be summed, forming the total score of the first level S_1

$$S_1 = S_1^A + S_1^B + S_1^C. \tag{1}$$

   The team with the highest score S_1 will be used as reference for the cut-off score of this level: any team with score S_1 at least 50% of the highest score will pass to the next level.

2. The same procedure is repeated for all difficulty levels, up to level 7, but considering only the teams that passed the cut-off score of the previous level:

$$S_N = S_N^A + S_N^B + S_N^C. \tag{2}$$

3. Denote by N_{max} the hardest level that at least one team could enter. If there is only one team, they win.

4. In case of several algorithms reaching level N_{max}, the competitors are arranged on the leaderboard based on the sum of all the scores from levels 1 ... N_{max}. After the last algorithm reaching level N_{max}, the leaderboard will be populated by algorithms that reached only level $N_{max}-1$, arranged with the same principle. The same procedure will be repeated for all levels to get the full leaderboard. In the very unlikely case of a tie, the organizing committee will make the final decision on the winner.

5. If one team submits more than one algorithm to the competition, then each submission will be temporarily assumed to belong to different 'virtual' teams when computing the scores and cut-offs. However, this team cannot be in more than one position in the leaderboard. In this situation, the organizing committee will consider only the highest performance algorithm when ranking the winners. For example, say team X ends with one algorithm at first place, and another at third place. In this case, the team wins the first place but the third place will be given to another team.

Special situations: The spirit of the competition is that the solution is a general-purpose algorithm, capable of reconstructing electrical impedance tomography images of the targets. The organizing committee has the right to disqualify an algorithm trying to violate that spirit.

Conflict of interest: researchers affiliated with the Department of Technical Physics of University of Eastern Finland are welcome to participate in the competition, but will not be added to the leaderboard and cannot win the competition.

3.3.1 Reconstruction assessment method

The reconstructions will be assessed quantitatively, comparing the reconstructed 'trinary' image I_r (i.e. array of values belonging to {0, 1, 2}) with the ground truth trinary image I_t, assigning a numeric score. I_r is assumed to have a dimension of 256 x 256 pixels, otherwise a score 0 will be given to the reconstruction I_r.

The score of the reconstruction I_r is based on the structural similarity index measure (SSIM) [3], taken separately of the conductive and non-conductive inclusions. SSIM is calculated for Gaussian ($\sigma = 80$ pixels) weighed neighbourhoods centered at each pixel. For two neighbourhoods r and t, the SSIM value is defined by the formula

$$\mathrm{SSIM}(r,t) = \frac{(2\mu_r\mu_t + c_1)(2\sigma_{rt} + c_2)}{(\mu_r^2 + \mu_t^2 + c_1)(\sigma_r^2 + \sigma_t^2 + c_2)}, \tag{3}$$

where μ_i and σ_i are the weighed sample mean and variance of the neighbourhood $i \in \{r,t\}$, σ_{rt} is the cross-correlation of the neighbourhoods, and $c_1 = 1 \cdot 10^{-4}$ and $c_2 = 9 \cdot 10^{-4}$ are constants related to the dynamical range of the images.

The final scoring will be done with the MATLAB version of SSIM ('KTC-ssim.m') shared with the training data and example codes. There are minor

differences to the other available SSIM implementations, but for training or internal validation, you may consider using the other more efficient implementations.

4 Data Collection

The challenge data was collected at the Process Tomography Laboratory at the University of Eastern Finland. The measurement device is a current-injection voltage-measurement electrical impedance tomography device constructed in-house ([1]). The measurement setup consists of the EIT device, a circular plastic imaging chamber, and 32 stainless steel electrodes at the boundary of the imaging chamber. Sixteen of these electrodes were used for current injections - the 16 current injection channels of the EIT device were connected to electrodes 1,3,5,...,31. All of the electrodes were used to measure the voltages between adjacent pairs of electrodes during the current injections.

For the correct alignment of the reconstructed images, the center of electrode 1 is considered to be at the 12 o'clock position. The electrodes were numbered in the counterclockwise direction. Each electrode covered an angle of 5.625 degrees, which was also the angular spacing between adjacent electrodes.

A total of 76 pairwise current injections were applied as follows:

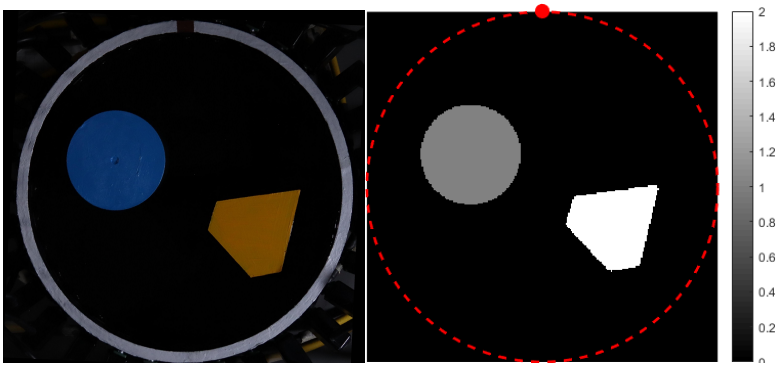
- **Set 1: "Adjacent" injections.** Current injections between electrodes 1-3,3-5,5-7,...,31-1. **This set had 16 injections.**
- **Set 2: All against 1.** Current injections between electrodes 3-1,5-1,7-1,...,31-1. **This set had 15 injections.**
- **Set 3: All against 9.** Current injections between electrodes 1-9,3-9,5-9,...,31-9. **This set had 15 injections.**
- **Set 4: All against 17.** Current injections between electrodes 1-17,3-17,5-17,...,31-17. **This set had 15 injections.**
- **Set 5: All against 25.** Current injections between electrodes 1-25,3-25,5-25,...,31-25. **This set had 15 injections.**

During each current injection, the voltages were measured between adjacent electrode pairs: 1-2,2-3,3-4,...,31-32. Therefore, the number of voltage measurements for one current injection is 31, and a full set of data contains $76 \times 31 = 2356$ voltage measurements.

As the challenge difficulty level increases, the data measured on some electrodes is discarded according to Table 1. For example, on difficulty level 2, the data collected using electrodes 1 and 2 is removed. This means that all of the voltage measurements corresponding to current injections in which electrode 1 is injecting are removed, and from the remaining measurements the voltages between pairs 1-2 and 2-3 are removed. Note that while the data corresponding to some electrodes is removed as the difficulty level increases, these electrodes were not physically removed from the imaging chamber.

Table 1: Electrode data removed in each difficulty group. Here, N_{inj} is the number of current injections used in the difficulty level, and N_{meas} is the total number of voltage measurements used in the difficulty level.

Difficulty group	Removed electrodes	N_{inj}	N_{meas}
1	-	76	$76 \cdot 31 = 2356$
2	1,2	56	$56 \cdot 29 = 1624$
3	1,2,3,4	52	$52 \cdot 27 = 1404$
4	1,...,6	48	$48 \cdot 25 = 1200$
5	1,...,8	44	$44 \cdot 23 = 1012$
6	1,...,10	30	$30 \cdot 21 = 630$
7	1,...,12	27	$27 \cdot 19 = 513$



(a) (b)

Figure 2: (a) The water chamber used to collect the data, with a single resistive plastic inclusion (blue) and a single conductive plastic inclusion (orange) present. (b) The 256 by 256 ground truth image with the inner chamber boundary and center point of electrode 1 highlighted in red.

Appendix A: The forward problem of EIT

This section briefly describes the forward model of EIT and its numerical solution using the Finite Element Method (FEM). A finite element solver for the forward problem is provided to the competitors, but using it is not compulsory.

The forward problem of EIT is given by the complete electrode model

$$\begin{aligned}
\nabla \cdot (\sigma \nabla u(\vec{r})) &= 0, && \vec{r} \in \Omega \\
u(\vec{r}) + z_l \sigma \frac{\partial u(\vec{r})}{\partial \vec{n}} &= U_l, && \vec{r} \in e_l, l = 1, ..., L \\
\int_{e_l} \sigma \frac{\partial u(\vec{r})}{\partial \vec{n}} dS &= I_l, && \vec{r} \in e_l, l = 1, ..., L \\
\frac{\partial u(\vec{r})}{\partial \vec{n}} &= 0 && \vec{r} \in \partial\Omega \setminus \bigcup_{l=1}^{L} e_l,
\end{aligned}$$

where $\vec{r}$ is the spatial coordinate vector, σ is the electric conductivity, u is the electric potential, z_l is the contact impedance between electrode e_l and the target domain Ω, and $\vec{n}$ is the outward unit normal of the boundary $\partial\Omega$. Moreover, U_l and I_l are the potential of electrode e_l and current through electrode e_l, respectively. In addition, charge conservation and zero level of potential are imposed by requiring that

$$\sum_{l=1}^{L} I_l = 0, \qquad \sum_{l=1}^{L} U_l = 0.$$

Solving the forward problem involves computing the electric potential u and the electrode potentials U for a given conductivity σ, contact impedances z and electrode currents I.

Finite Element Method solution

Our finite element method implementation is similar to the one presented in ([2]), but two-dimensional. We note that the targets used to measure the datasets were three-dimensional, but non-varying in the vertical direction. Triangular elements are used, and the electric potential is discretized using 2nd order polynomial basis functions, while the conductivity is discretized using linear basis functions.

The forward solver is provided as both Matlab and Python versions. A mesh corresponding to the measurement geometry is also included. Providing the forward solver with the nodal values of conductivity, the electrode contact impedances, and the current injection pattern results in the simulated electrode voltage data.

Appendix B: Example reconstruction with segmentation

A simple algorithm is provided here as an example of what a solution to the challenge should use as input (the EIT data), and what it should output (segmented images).

We denote the EIT measurement model briefly as

$$V = U(\sigma, z) + e, \tag{4}$$

where V is a vector of measured voltages between adjacent electrode pairs, U is the FEM-based forward solver, and e is a random measurement error vector. Consider two sets of measured voltage data V_1 and V_2, corresponding to conductivities σ_1 and σ_2 at different time instants. We linearize the forward model as

$$V_i = U(\sigma_i, z) + J(\sigma_0)(\sigma_i - \sigma_0) + e_i, \qquad i = 1, 2, \tag{5}$$

where $J(\sigma_0)$ is the Jacobian of the forward model evaluated at a chosen linearization point σ_0. Subtracting the two linearized models, we get a linear observation model for the difference data ΔV,

$$\Delta V = J(\sigma_0)\Delta\sigma + \Delta e. \tag{6}$$

An estimate for the conductivity change is computed as the solution of a regularized least squares problem

$$\arg\min_{\Delta\sigma} ||L_{\Delta e}(\Delta V - J\Delta\sigma)||^2 + ||L_{\mathrm{pr}}\Delta\sigma||^2, \tag{7}$$

where $L_{\Delta e}$ is a weighting matrix defined by $L_{\Delta e}^T L_{\Delta e} = \Gamma_{\Delta e}^{-1}$, where $\Gamma_{\Delta e}$ is the covariance of the measurement noise term Δe. Similarly, $L_{\mathrm{pr}}^T L_{\mathrm{pr}} = \Gamma_{\mathrm{pr}}^{-1}$, where Γ_{pr} is the prior covariance for $\Delta\sigma$. We model the noise covariance $\Gamma_{\Delta e}$ as $\Gamma_{\Delta e} = \mathrm{diag}(a\Delta V + b\max(\Delta V))$, where a and b are constants chosen as $a = 0.05$ and $b = 0.01$. Also, a smoothness promoting prior covariance Γ_{pr} is used, given by

$$\Gamma_{\mathrm{pr}}(i, j) = c \exp \frac{||\vec{r_i} - \vec{r_j}||^2}{d},$$

where $c = 0.01$ and $d = 0.03$.

The solution to the regularized LS problem is computed as

$$\widehat{\Delta\sigma} = (J^T\Gamma_e^{-1}J + \Gamma_{\mathrm{pr}}^{-1})^{-1}J^T\Gamma_e^{-1}\Delta V. \tag{8}$$

This reconstruction is then interpolated from the finite element mesh into a 256 by 256 pixel image. Next, the resulting grayscale image is segmented into three classes (background, resistive inclusions, and conductive inclusions) using triclass Otsu's method. The triclass Otsu's method splits the image pixels into three classes by defining two tresholds such that intra-class variance

$$\sigma_w^2(t) = \sum_{i=1}^{3} \omega_i(t)\sigma_i^2(t) \tag{9}$$

is minimized. Here, t is a vector containing the two treshold values, $\omega_i(t)$ is the number of pixels in the class i, and $\sigma_i^2(t)$ is the variance of the pixel values in the class i.

Finally, the class with the most pixels is assumed to be the background (water). If this class contains either the lowest or the highest values of $\Delta\sigma$, the other two classes are combined into a single class and assumed to represent conductive or resistive inclusions, respectively. This is done because some of the targets contained only a single type of inclusions (resistive or conductive). We note that the targets may contain either resistive inclusions, conductive inclusions, or both. Also, conductive inclusions made up of the two different conductive materials (metal or conductive plastic) may be present in the same target.

References

[1] J. Kourunen et al. "Suitability of a PXI platform for an electrical impedance tomography system". In: *Measurement Science and Technology* 20 (2009), p. 015503.

[2] P. J. Vauhkonen et al. "Three-dimensional electrical impedance tomography based on the complete electrode model". In: *IEEE Transactions on Biomedical Engineering* 46 (1999), pp. 1150–1160.

[3] Z. Wang et al. "Image Quality Assessment: From Error Measurement to Structural Similarity". In: *IEEE Transactions on Image Processing* 13 (2004).