2019 Thailand-Japan MicroWave (TJMW2019)

Auto-Encoder/Decoder for Planar Filter Analysis/Synthesis

Kosuke KATAYAMA† and Takaaki BABA‡

†Graduate School of Information, Production and Systems, Waseda University, Kitakyushu-shi, Fukuoka, 808-0135 Japan

‡Research Center of Information, Production and Systems, Waseda University, Kitakyushu-shi, Fukuoka, 808-0135 Japan

E-mail: †kkatayama@aoni.waseda.jp, ‡tbaba@waseda.jp

Abstract In this paper, we propose the first study which estimates the frequency response directly from the geometry of a planar filter, and which also synthesizes the planar filter geometry directly from the given frequency response using a convolutional neural network (CNN) based auto-encoder/decoder. we also explain the way to generate an accurate and massive dataset for training the auto-encoder/decoder. In our experiments, the frequency response is estimated in 1.5 msec and the filter geometry is synthesized in 2.7 msec, respectively.

Keyword Convolutional neural network, Edge model de-embedding/embedding, Electromagnetic simulation, *F*-parameter, Microwave

1. INTRODUCTION

In a microwave region, planar filters are utilized. Since the planar filter is composed of distributed elements, element values of a filter theory cannot be used. Therefore, filter designers must tune the filter geometry using an electromagnetic (EM) simulator until the frequency response meets their wants. Since this task takes long time due to the long duration of EM simulations, several neural networks have been proposed for the reduction of design time. However, these proposals need the design parameters of the filter geometries and the neural networks tune the design parameters such as line lengths, widths and so on. Extraction of the design parameters gives another task to designers. Moreover, a designer must set constraints of the design parameters and convert the obtained design parameter into an actual geometry although the neural networks are employed for the reduction of tasks. On the other hands, there is a convolutional neural network (CNN) called the auto-encoder/decoder (Fig. 1). This auto-encoder/decoder converts an image into a similar image via a compact vector containing a feature of the image. Since the planar filter has a two-dimensional geometry in its surface, the auto-encoder/decoder can deal it with the image.

In this paper, we mapped the frequency response into the feature vector of the auto-encoder/decoder. As a result, auto-encoder outputs the frequency response from the planer filter image. Conversely, auto-decoder output the planer filter image from the desired frequency response. This is a first paper which converts bidirectionally between the filter geometry and frequency response not to be detouring the design parameters. In Section 2, we consider the dataset generation for training the auto-encoder/decoder because the quantity and quality of dataset are very important for training as well as other neural networks.

2. DATASET GENERATION

High-speed calculation of the frequency response is vital to generate a huge amount of dataset. The frequency response of a planar filter can be calculated quickly by a cascade production of *F*-parameters [1, 2]. However, the accuracy of the calculation is not enough because uniformity of current distribution is corrupted at the edge where the two lines are connected as shown in Fig. 2. The accuracy of the frequency response guarantees the consistency between the filter geometry and its frequency response which is necessary for the convergence of the neural network [3]. Since the uniformity of the current distribution is corrupted at the edge of the wires, the total *F*-parameter F_{total} is not a simple product of *F*-parameters ($F_{\mathrm{left}} \cdot F_{\mathrm{right}}$) which is extracted under the conditions of uniform current distributions (Fig. 2). If we succeeded to extract the edge model F_{edge}, the accuracy of F_{total} would be improved as $F_{\mathrm{left}} \cdot F_{\mathrm{edge}} \cdot F_{\mathrm{right}}$. Following part of this section describes how to extract the F_{edge} (de-embedding) and implant the F_{edge} (embedding). Since the F_{edge} is functions of signal line widths (W_1, W_2), all combinations of W_1 and W_2 should be examined as illustrated in Fig. 3. This combination table contains information about the edge models at the part of $W_1 < W_2$ and $W_1 > W_2$ with lead lines of W_1 and W_2. This combination table also contains straight lines when W_1 equals to W_2 and they are used for de-embedding the lead lines. Since these lines ($W_1 = W_2$) are two times longer than lead lines of de-embedding targets ($W_1 < W_2$, $W_1 > W_2$), they should be halved using equations in Fig. 4. In this case, the scaling factor n is 0.5. Figure 5(a) shows the edge model de-embedding. Since the non-diagonal part of the table in Fig. 3 contains lead lines (W_1 and W_2) and the edge model F_{edge}, the lead lines are de-embedded using diagonal part of the table ($W_1 = W_2$) after halving their lengths. The edge model F_{edge} is a cascade product of the inverse function of corresponding lead line F_1, de-embedding target, and the inverse function of F_2. As a result, F_{edge} represents the non-uniformity of the current distribution as frequency response and its length is zero because total lengths of the de-embedding target ($L_1 + L_2$) is coincident with the total subtraction lengths (L_1, L_2). Figure 6 shows a random generation of a filter geometry. Since the filter geometry has a symmetrical structure over vertically and horizontally, the quarter size of the geometry is generated as an image. Random widths (W_n) and lengths (L_n) are generated and connected until the accumulated length reaches the half of the total length L_{total}. Conversely, entire geometry is a unfold image over vertically and horizontally. The frequency response of the randomly generated geometry is calculated and stored in a dataset with its quarter-sized image. Depending on the geometry the line lengths are scaled as shown in Fig. 4. Edge models are embedded depending on the line widths as shown in Fig. 5(b). Since the edge model has no length, the embedding does not affect the total length and affects only the frequency response.

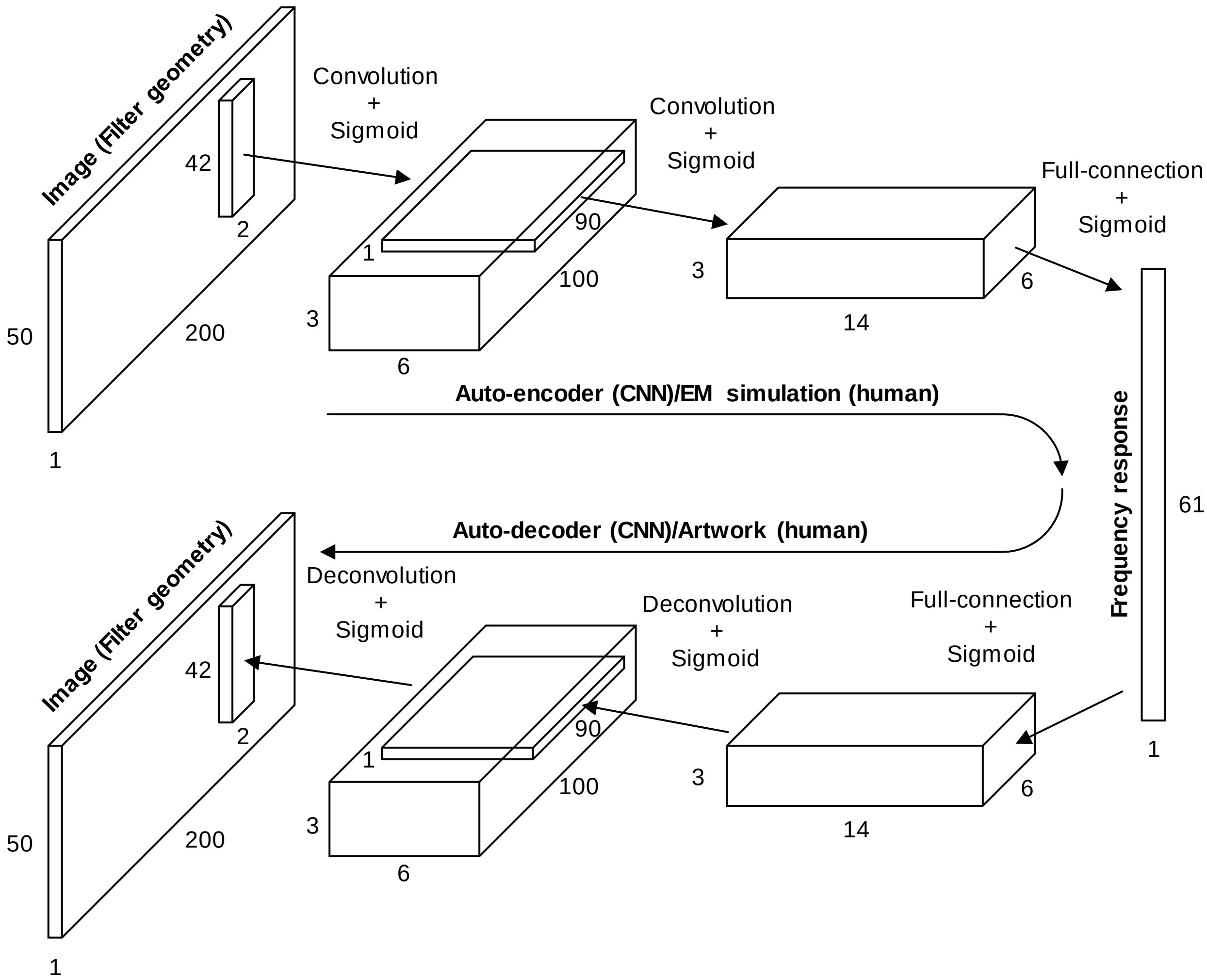


Fig. 1. Auto-encoder and decoder of a planar filter using convolutional neural networks (CNNs). Since the surface of a planar filter has a two-dimensional geometry, it can be assumed as an image. The system of an auto-encoder and decoder is a special case of a neural network which converts an image to its similar image via a compact vector which contains a feature of the images. The compact vector is mapped to a frequency response of a planar filter in this case. The path from an image to its feature vector is called auto-encoder. Auto-decoder is a path from a feature vector to its related image. Since the images have higher-order vectors than the feature vector, convolution, and de-convolution are utilized for down- and up-sampling in the auto-encoder and decoder paths. A couple of full-connected neural networks are the bridges of a convoluted image to the feature vector or the deconvoluted image from the feature vector. Non-linear sigmoidal functions are inserted properly to extract logical relationships and to normalize the values. This auto-decoder and encoder paths are also human mimic tasks. When designing a planar filter, the human creates the filter geometry from the desired frequency. Afterword, the frequency response is confirmed using an electromagnetic (EM) simulator to ensure its response meets the desired frequency.

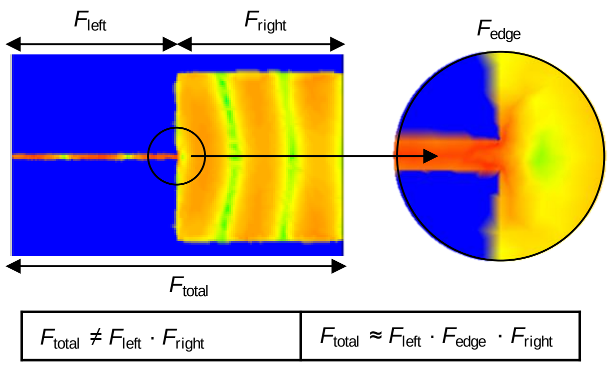


Fig. 2. The current density of connected transmission lines (TLs) whose widths are different. At the edge where the two TLs are connected, the current distribution is non-uniform. Therefore, the simple product of left- and right-side F-parameters (F_{left}, F_{right}) is different from the total one (F_{total}). For the accurate calculation of the F_{total}, the non-uniformity of the connection (F_{edge}) should be considered.

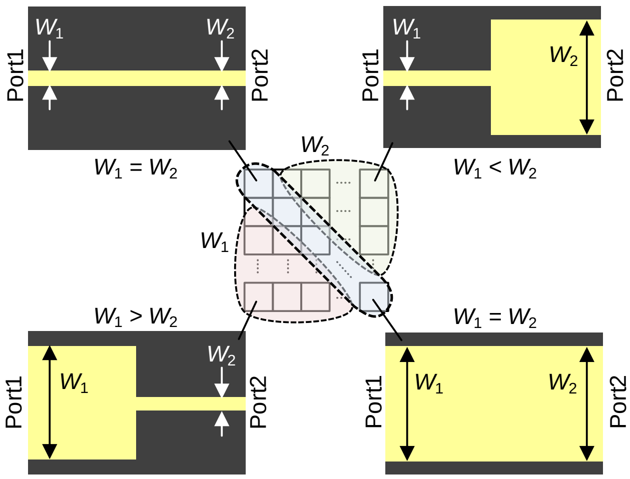


Fig. 3. A combination table of transmission lines. This table is generated by changing the line width W_1 (port 1 side) and W_2. It contains all variations of a straight line ($W_1 = W_2$) and all combinations of line width ($W_1 < W_2$). Note that the $W_1 > W_2$ is generated by the port swapping of $W_1 < W_2$.

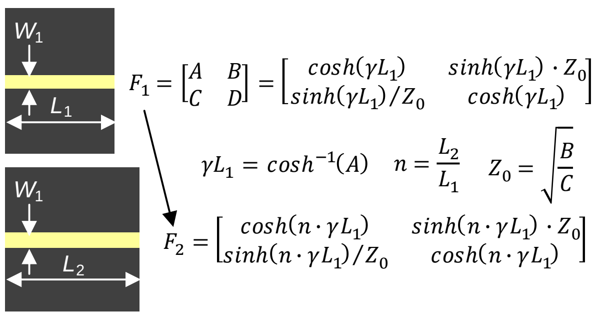
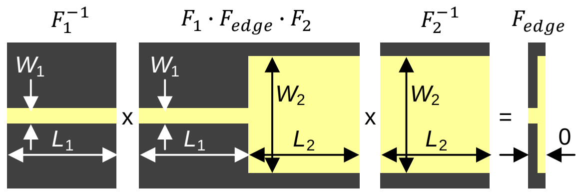
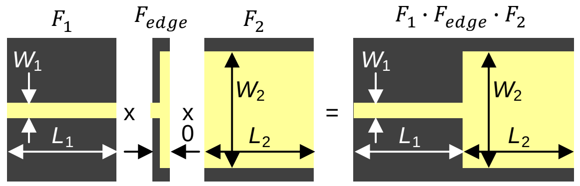


Fig. 4. Length scaling of a transmission line (TL). Since the propagation constant γ and characteristic impedance Z_0 are not depending on the length of the TL, scaled F-parameter (F_2) is calculated from F_1 using the scaling factor n.



(a) Edge model (F_{edge}) de-embedding.



(b) Edge model (F_{edge}) embedding.

Fig. 5. Edge model F_{edge} de-embedding and embedding. Edge model de-embedding is calculated from the inverse functions of TLs whose length and width are coincided with the those of target ones. Conversely, F_{edge} is embedded between TLs when the line widths are different.

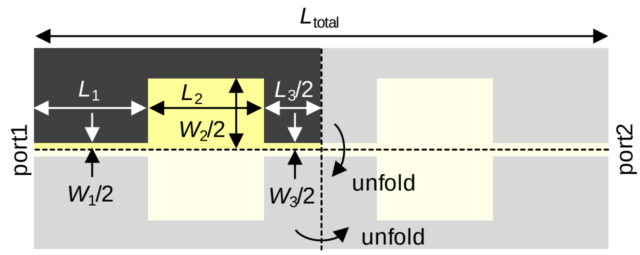


Fig. 6. Random geometry generation of a filter. TL length (L_1, L_2 and $L_3/2$) and width ($W_1/2$, $W_2/2$ and $W_3/2$) are generated randomly until the sum of the length ($L_1 + L_2 + L_3/2$) reaches the half of the total length ($L_{total}/2$). The geometry is mirrored vertically and horizontally.

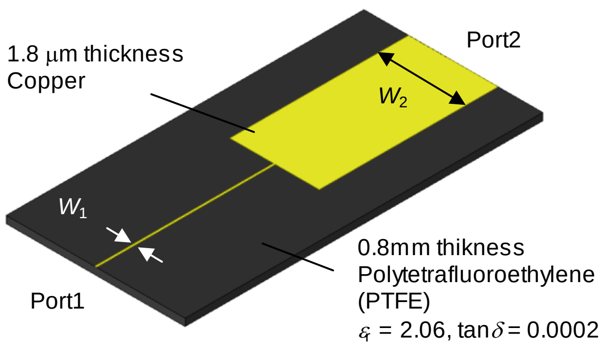
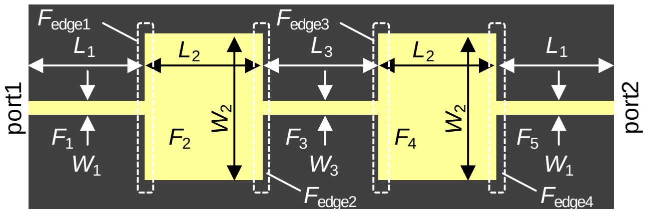
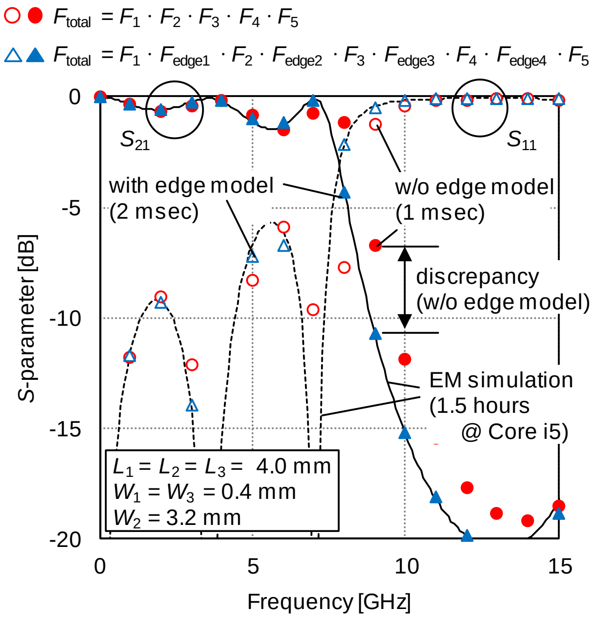


Fig. 7. The appearance of the EMPro, which is an electromagnetic (EM) simulator provided by Agilent technology Inc. Signal line of a 1.8 μm copper foil is on a 0.8 mm polytetrafluoroethylene (PTFE) bulk which is shield with a perfect electrical conductor (PEC). Signal line widths of W_1 and W_2 are swept to generate the combination table of Fig. 3. The width sweeping and S-parameter extraction are performed automatically by a built-in Python script.



(a) Geometry.



(b) Frequency responses.

Fig. 8. Geometry of a planar filter and its frequency response. Dash and solid-lines are S_{11} and S_{21} when the geometry is analyzed with an electromagnetic simulator. Symbols denotes the reconstructed responses calculated from F-parameters. Circle (triangle) symbols are case of ignoring (considering) the edge model.

3. EXPERIMENTS

Figure 1 also shows a structure of the auto-encoder/decoder. The size of the target planer filter is 1 x 4 cm. Metal and bulk parts are converted into white and black pixels. One pixel corresponds to 0.1 x 0.1 mm and the quarter-sized image becomes 50 x 100. Since the frequency response is a magnitude of S_{21} from DC to 30 GHz (500 MHz step), the number of points is 61. The pairs of image and frequency response are stored in the dataset. The images are fed to the auto-encoder and output of the auto-encoder is compared with the frequency response and residue are fed back to train the network. The frequency responses are fed to the auto-decoder and output of the auto-decoder is compared with the image and residue are fed back to train the network. In the auto-encoder path, an image is converted into 6 x 3 x 100 matrix using 42 x 2 convolution kernels (stride is 4 x 2). The 6 x 3 x 100 matrix is converted into 14 x 3 x 6 using 1 x 90 kernels (stride is 1 x 2). The 14 x 3 x 6 matrix are fully connected to the frequency response.

In the auto-decoder path, frequency response is fully connected to 14 x 3 x 6 matrix. The 14 x 3 x 6 matrix is converted into 6 x 3 x 100 using 1 x 90 deconvolution kernels (stride is 4 x 2). The 6 x 3 x 100 matrix is converted into 50 x 100 image using 42 x 2 kernel (stride is 1 x 2). The calculation costs of the auto-encoder/decoder are 302,652 and 7,323,372 in terms of the product-sum.

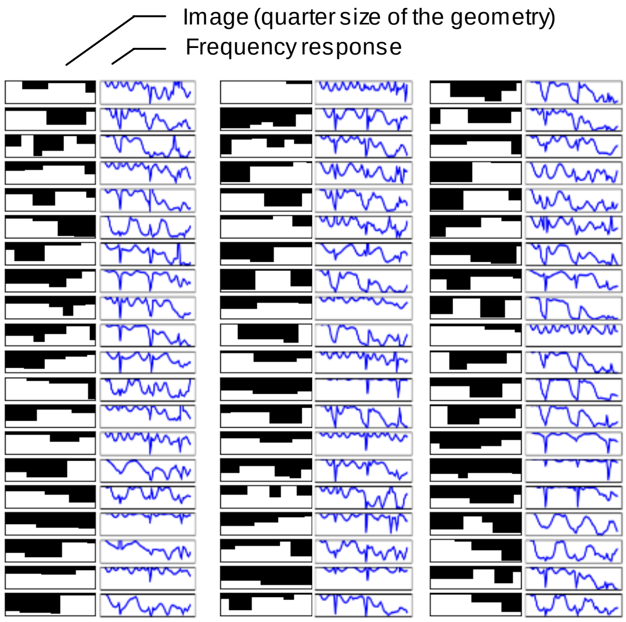


Fig. 9. Part of a dataset. The dataset contains pairs of an image and its frequency response. The image is a randomly generated quarter-sized planar filter (Fig. 6). Its frequency response is calculated quickly and accurately taking account of the edge model (Fig. 5). Quantity and quality of a dataset are important for the auto-encoder/decoder training (Fig. 1).

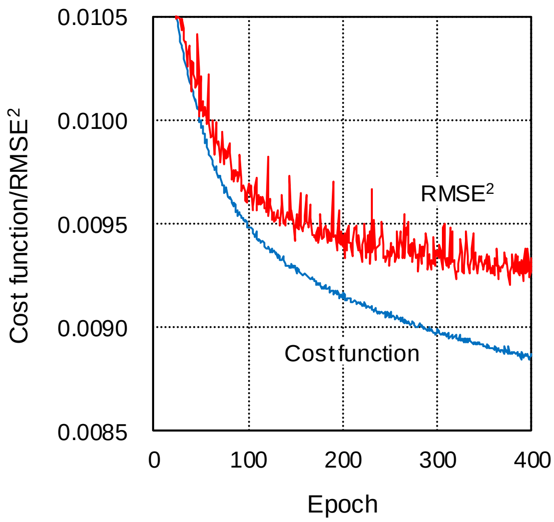


Fig. 10. Cost function and the square of RMS error (RMSE^2) as a function of Epoch while input and output of auto-encoder (Fig. 1) are images and their frequency responses (Fig. 9) , respectively.

To generate the table shown in Fig. 3, an EM simulator called EMPro is utilized as shown in Fig. 7. Figure 8(a) shows an example of a filter geometry. Figure 8(b) shows the frequency responses of the geometry. The cascaded production of *F*-parameters with edge models (triangles) exactly traces the frequency response which is the EM simulation result of the entire geometry (lines). On the other hands, the one without edge models has a discrepancy. This result shows the accuracy using edge models is comparable to that of the EM simulation even though the calculation is done in 2 msec while the EM simulation takes 1.5 hours.

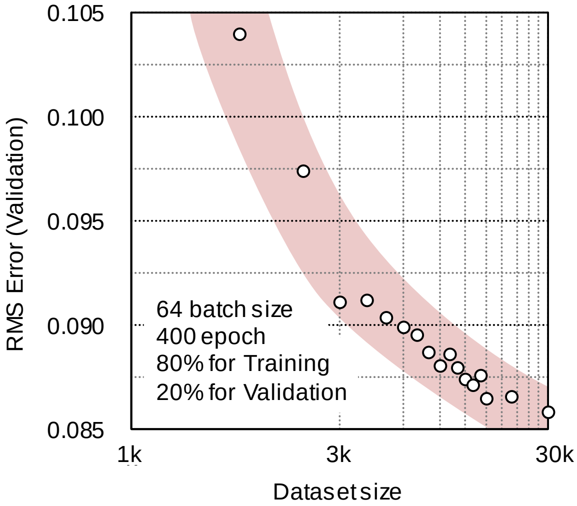
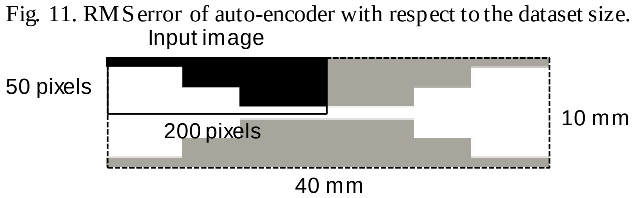
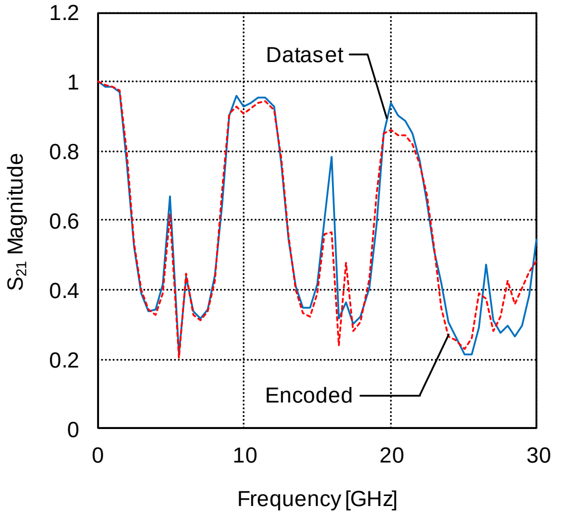


Fig. 11. RMS error of auto-encoder with respect to the dataset size.



(a) Input image and unfolded geometry.



(b) Frequency responses.

Fig. 12. Input image and frequency responses stored in a dataset and that encoded from the image. This image and this frequency response are not used for the auto-encoder training (they are used only for validation).

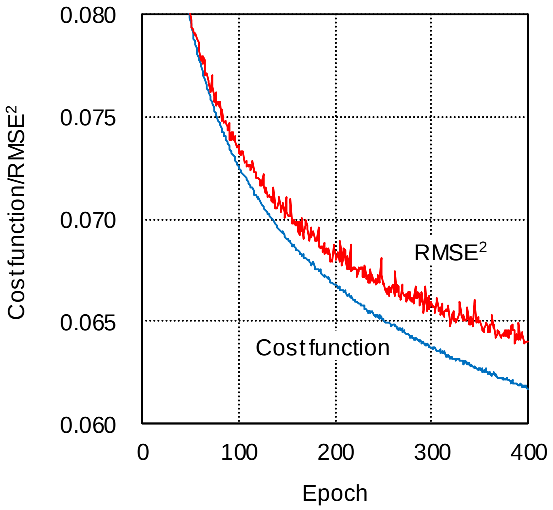


Fig. 13. Cost function and the square of RMS error (RMSE^2) as a function of Epoch while input and output of auto-decoder (Fig. 1) are frequency responses and their images (Fig. 9), respectively.

Figure 9 shows a part of the dataset. The dataset contains pars of randomly generated image and its frequency response, which is calculated accurately considering the edge models. Taking advantage of high-speed calculation, 30,000 pairs are stored in the dataset.

Figure 10 shows a learning curve of the auto-encoder. Figure 11 shows the root-mean-square (RMS) error with respect to the dataset size. For each size, 80% (20%) of the dataset is used for training (validation). The quantity of dataset improves the neural network. Figure 12 shows the input image and frequency response of the dataset and output of the auto-encoder when this image is applied. This pair of image and frequency response are not used for training and used only for validation. This result shows the auto-encoder can estimate the frequency response from the input image.

Figure 13 shows a learning curve of the auto-decoder. Figure 14 shows an example of filter synthesis. Brick-wall filter response is applied to the auto-decoder as a desired frequency response. The auto-decoder synthesis the most possible filter geometry. After the binarization of the image, the auto-encoder estimates the frequency response from the image. The sigmoidal function of the auto-decoder can be replaced with the binary-sigmoidal function to obtain the binary image, but the binary-sigmoidal function conceals the derivatives which used in the training process. Since the planar filter cannot always realize the arbitral frequency, some iterative optimizations seem to be required in a practical usage. The auto-decoder and encoder take 2.7 and 1.5 msec using nVidia RTX 2080, respectively.

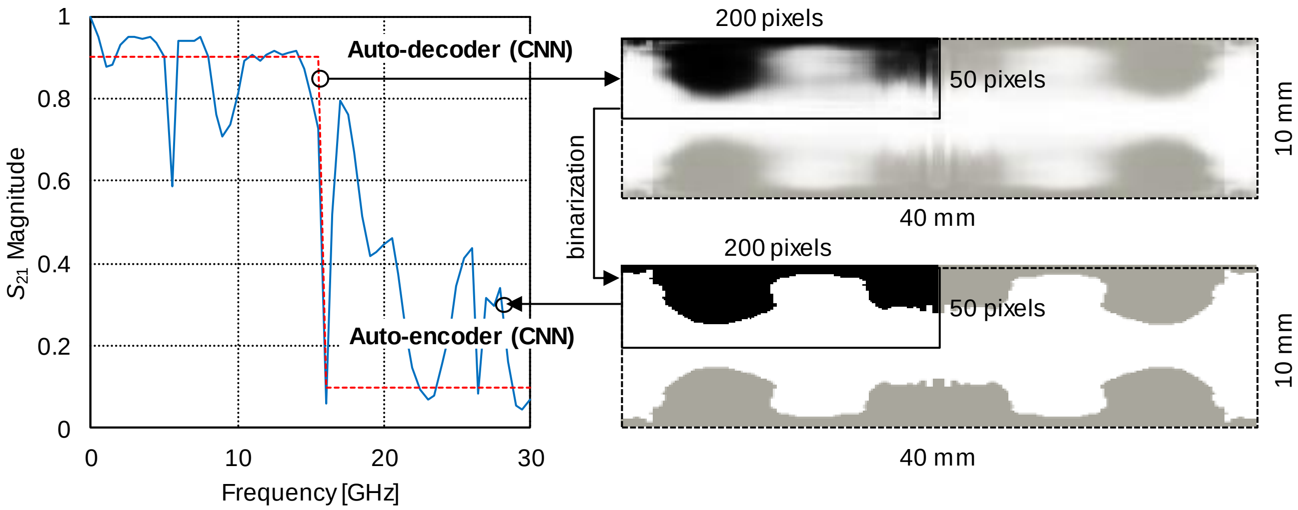


Fig. 14. An example of synthesized filter image (upper right) decoded from the desired frequency response (dash line in the graph) and a frequency response (solid line in the graph) encoded from a binarized image (bottom right). Desired frequency response is fed to the auto-decoder and a quarter-sized filter image is synthesized. Since the auto-decoder outputs gray-scale image, it is binarized using a certain image library such as OpenCV. The binarized image is fed to the auto-encoder and its frequency response is estimated. The estimated frequency response tends to be different from the desired frequency if it is physically difficult to be realized as shown in this example of a brick-wall filter. Auto-decoder and encoder take 2.7 and 1.5 msec, respectively.

4. Conclusion

In this paper, we propose the auto-encoder/decoder which is used for a planar filter analysis/synthesis. The geometry of the planar filter is regarded as an image and the feature vector of the auto-encoder/decoder is mapped to the frequency response. We also describe the way to generate a massive and accurate dataset to train the auto-encoder/decoder. We demonstrate the auto-encoder can estimate the frequency response from the input image and the auto-decoder can synthesize the filter geometry from the given frequency response. This is a first study which converts directly and bidirectionally between filter geometry and frequency response.

Acknowledgements

This work was supported by Japan Society for the Promotion of Science (JSPS) through a Grant-in-Aid for Scientific Research (No. 18K04155). This work was also supported by VLSI Design and Education Center (VDEC), The University of Tokyo with the collaboration with Agilent Technologies.

References

[1] J. S. Hong and M. J. Lancaster, *Microstrip Filters for RF/Microwave Applications*, John Wiley & Sons, New York, 2001.

[2] K. Katayama, K. Takahata, T. Ohsawa and T. Baba, "Training Dataset Synthesis for Planar Filter Circuits Learning," *The 71th Joint Conf. of Electrical, Electronics and Information Engineers in Kyushu*, pp. 264, Sept. 9, 2018.

[3] Kosuke Katayama, Kiyoto Takahata, Takashi Ohsawa, Takaaki Baba, "Edge Model Embedding Method for Planar Filter Synthesis," *RISP International Workshop on Nonlinear Circuits, Communications and Signal Processing (NCSP)*, pp. 502-505, March 5-7, 2019.