UNIVERSITY OF CALIFORNIA SAN DIEGO

Generative Neural Networks Enable Real-time EM Metastructure Designs

A dissertation submitted in partial satisfaction of the
requirements for the degree Doctor of Philosophy

in

Electrical Engineering (Applied Electromagnetics)

by

Erda Wen

Committee in charge:

Professor Daniel F. Sievenpiper, Chair
Professor Prabhakar R. Bandaru
Professor Michael M. Fogler
Professor Peter Gerstoft
Professor Zhaowei Liu
Professor George C. Papen

2023

The Dissertation of Erda Wen is approved, and it is acceptable in quality and form for publication on microfilm and electronically.

University of California San Diego

2023

DEDICATION

TO MY FAMILY

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

ACKNOWLEDGEMENTS

Not everyone has the privilege to enjoy their Ph.D. program, and I feel so lucky and proud that I do. For this, I thank Professor Sievenpiper for never being pushy, always being inspiring, and constantly encouraging me to try things, making this research possible in the first place.

I thank all colleagues from Applied EM Group along the way for always being kind and helpful. Dr. Dia'aaldin Bisharat, he never hesitates to share his insightful research ideas, providing so much inspiration in my first year. Dr. Robert Davis, he took care of everything in the lab and was always ready to lend support, meanwhile being a genius. Dr. Matt Smith, he helped me so much with my experiments with his precious experiences, without whom the project may last two times longer. I also thank Dr. Kyle Thackston, Dr. Sara Kandil, Dr. Shreya Singh, Dr. Yun Zhou, and Feng Li, for making my PhD program so colorful.

My biggest thank goes to my family. I thank my parents Qingwen Li and Quanyi Wen, for supporting me chasing my dream so far away from home, and I know it was not easy emotionally and financially. I thank my fiancee Xiaozhen Yang for always being there, completing my life. I love you.

The material in this dissertation is based on the following papers or manuscripts.

Chapter 2 is a reprint of the paper: Wen, E., Yang, X., & Sievenpiper, D. F. (2022). "Real-time 2-D beamforming with rotatable dielectric slabs enabled by generative neural network." *IEEE Transactions on Antennas and Propagation*, 70(9), 8360-8367. The dissertation author was the primary investigator and author of this paper.

Chapter 3 and Chapter 4 are based on material in a manuscript to be published: Wen, E., Yang, X., & Sievenpiper, D. F. (2023). "Real-data-driven Real-time Reconfigurable Microwave Reflective Surface". *arXiv preprint* arXiv:2305.11899. The dissertation author was the primary investigator and author of this paper.

VITA

2016	B. Eng. in Electrical Engineering, University of Electronic Science and Technology of China
2017	M. S. in Electrical Engineering, The Ohio State University
2023	Ph. D. in Electrical Engineering (Applied Electromagnetics), University of California San Diego

PUBLICATIONS

Erda Wen, Xiaozhen Yang, and Daniel F. Sievenpiper. "Real-data-driven Real-time Reconfigurable Microwave Reflective Surface." *arXiv preprint* arXiv:2305.11899 (2023).

Erda Wen, Daniel F. Sievenpiper, and Patrick P. Mercier. "Analysis of Coil Coupling in the Near-Field Far-Field Hybrid Region." *IEEE Antennas and Wireless Propagation Letters* (2023).

Erda Wen, et al. "Designing Topological Defect Lines Protected by Gauge-Dependent Symmetry Indicators." *Physical Review Applied* 17.6 (2022): 064008.

Erda Wen, Xiaozhen Yang, and Daniel F. Sievenpiper. "Real-Time 2-D Beamforming With Rotatable Dielectric Slabs Enabled by Generative Neural Network." *IEEE Transactions on Antennas and Propagation* 70.9 (2022): 8360-8367.

Erda Wen, Daniel F. Sievenpiper, and Patrick P. Mercier. "Channel characterization of magnetic human body communication." *IEEE Transactions on Biomedical Engineering* 69.2 (2021): 569-579.

Erda Wen, Xiaozhen Yang, and Daniel F. Sievenpiper. "Observing flat wavefront formation with diffusive transport in microwave graphene with topological insulator protected edges." *Applied Physics Letters* 118.16 (2021): 161102.

Xiaozhen Yang, **Erda Wen**, and Daniel F. Sievenpiper. "All-passive Microwave-Diode Nonreciprocal Metasurface." *arXiv preprint* arXiv:2305.02207 (2023).

Xiaozhen Yang, **Erda Wen**, and Daniel F. Sievenpiper. "Broadband time-modulated absorber beyond the Bode-Fano limit for short pulses by energy trapping." *Physical Review Applied* 17.4 (2022): 044003.

Miao Meng, Hossein Rahmanian Kooshkaki, Xiaoyang Wang, Shih-Kai Kuo, **Erda Wen**, and Patrick P. Mercier. "A GMSK/PAM4 Multichannel Magnetic Human Body Communication Transceiver." *IEEE Solid-State Circuits Letters* 5 (2022): 66-69.

Xiaozhen Yang, **Erda Wen**, and Daniel F. Sievenpiper. "Power-Dependent Metasurface With

Self-Induced Bandgap." *IEEE Antennas and Wireless Propagation Letters* 21.6 (2022): 1115-1119.

Shreya Singh, Robert J. Davis, J. Bisharat Dia'aaldin, Jiyeon Lee, Sara M. Kandil, **Erda Wen**, Xiaozhen Yang, Yun Zhou, Prabhakar R. Bandaru, and Daniel F. Sievenpiper. "Advances in Metasurfaces: Topology, Chirality, Patterning, and Time Modulation." *IEEE Antennas and Propagation Magazine* 64.4 (2021): 51-62.

CONFERENCES

Erda Wen, Xiaozhen Yang, and Daniel F. Sievenpiper. "Generative Neural Network Enables Reconfigurable Metasurface on Real-Time Free-Form Targets." *2023 International Applied Computational Electromagnetics Society Symposium (ACES)*. IEEE, 2023.

Erda Wen, Xiaozhen Yang, and Daniel F. Sievenpiper. "Surface Wave Wavefront Formation with Photonic Graphene and Topological Waveguide." *2022 IEEE International Symposium on Antennas and Propagation and USNC-URSI Radio Science Meeting (AP-S/URSI)*. IEEE, 2022.

Erda Wen and Chi-Chih Chen. "Top-Fed P-Band Dual Circular Polarization Patch Antenna Design." *2018 AMTA Proceedings.* IEEE, 2018.

Erda Wen and Chi-Chih Chen. "A Low-sidelobe Ka-Band Array Antenna Design." *2017 AMTA Proceedings.* IEEE, 2017.

ABSTRACT OF THE DISSERTATION

Generative Neural Networks Enable Real-time EM Metastructure Designs

by

Erda Wen

Doctor of Philosophy in Electrical Engineering (Applied Electromagnetics)

University of California San Diego, 2023

Professor Daniel F. Sievenpiper, Chair

This dissertation discusses how generative-type artificial neural network (ANNs) enables the real-time inverse design process of reconfigurable EM megastructures. Specifically, we demonstrate two designs: 1. a 2-D beamformer with rotatable dielectric slabs and 2. a conformal EM coating responding to free-form design goals on reflection pattern in a dynamic environment, the latter being an ultimate ambition of EM scattering control and nearly impossible to realize with conventional methods. These two examples demonstrate the superiority of ANN methods in dealing with high-nonlinear EM design problems requiring fast responses. Furthermore, the proposed data-driven free-form inverse-design approach can be accommodated for other science/engineering tasks.

Chapter 1

Introduction

The artificial neural network (ANN), a concept that dates back to the 1940s, finally embraces its prosperity in the early 21st century, thanks to the fast development of computing power and the easier accessibility of more big data, for good or bad. For the first time, computers can perform tasks that are conventionally considered only creatures with *real* intelligence, i.e., human beings, can do: identifying things from video clips, playing games, writing, drawing... One interesting question is, for other science and engineering topics where well-established analytical or optimization approaches have always been used and proven very effective, can ANNs still gain an edge over them in any aspect and provide features that no other methods can provide?

In this dissertation, we try to answer this question from an EM perspective and demonstrate the merit of the ANN approach in designing EM metastructure. Particularly, we exhibit two outstanding advantages of it:

1. the ability to handle highly non-linear mappings, which allows it to be used for EM structures under very complex environments, and

2. relativity fast prediction speed, which is critical to realize real-time capability.

For readers with different backgrounds, a brief and gentle introduction to some key concepts throughout this dissertation are as follows.

1.1 EM Metamaterial and Metasurfaces

The property of a normal bulk material is decided by its structure on the atomic level; likewise, the EM property of a metasurface is decided by the exquisitely engineered geometrical details in sub-wavelength scales. A metamaterial is usually made of periodic or quasi-periodic elements called "units", which interact with the EM wave with comparable wavelength through resonances, and as a whole, create a property usually not found in nature.

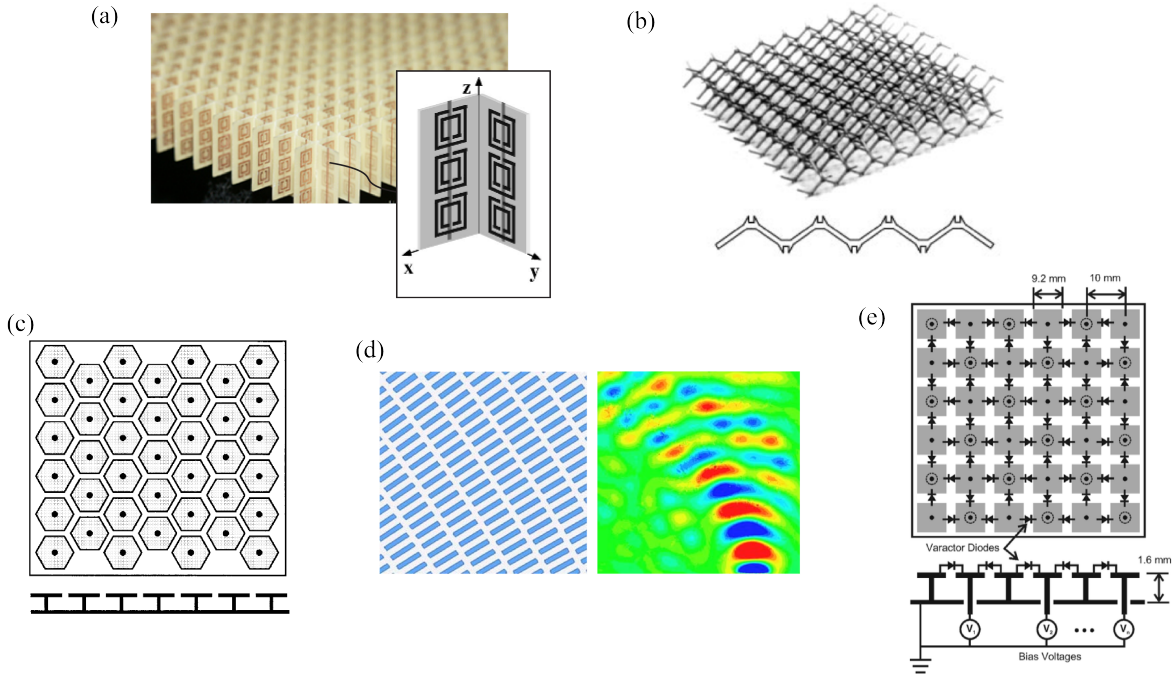

Figure 1.1. EM metamaterials and metasurfaces. (a) Negative refractive index material, from [1], inside which the electric field, magnetic field, and wave vector follow a left-hand rule instead of the ordinary right-hand rule. (b) 3D wire mesh photonic crystals with unconventional bandgap, from [2]. (c) High-impedance "mashroom" surface, from [3]. (d) Surface with varying impedance tensor, from [4]. (e) Tunable impedance surface, from [5].

Some intriguing effects of significant physical and practical interests include negative reflective index - creating effective negative permittivity ε_r and permeability μ_r [1, 6]; frequency selective plasmonic behavior [2] with band structure not seen in ordinary photonic crystals, and so on.

Metasurfaces are 2-D versions of metamaterial - sub-wavelength structures built on sheet

materials, for EM purposes, usually on printed circuit boards (PCBs). The most relevant design to what will be adopted in this dissertation is the impedance surface. Fig. 1.1(c) illustrates the earliest "mushroom structure" that exhibits high character impedance. The resonance created by patches, vias and gaps prohibits a transverse magnetic field on the surface, effectively creating a "perfect magnetic conductor", a boundary condition never seen in nature. To take a step further, arbitrary impedance can be achieved using various approaches. For example, [4] uses a gradually changed geometry to tune the local impedance tensor on the surface to guide the surface wave at will. Another way to tune the impedance is to introduce tunable active components, [5] shows a mushroom structure as in [3] integrated with varactors to tune the local impedance, which can be used to realize beam-forming. A design adopting this mechanism will be used in Chapter 3.

1.2 Artificial Neural Networks

Inspired by how brains process complicated tasks through connections between neurons, artificial neural networks aim at solving highly non-linear classification/regression with architectures consisting of interconnected "nodes". The nodes are usually grouped in layers to form multidimensional matrices (or tensors) for easier calculation with different matrix operations depending on different applications, as illustrated in Fig. 1.2, the most simple structure being a fully-connected layer:

$$\mathbf{Y} = g\left(\mathbf{WX} + \mathbf{b}\right), \tag{1.1}$$

where the output layer $\mathbf{Y}$ can be used as the output of the network or as a "hidden layer" served as the input of the next layer. Parameters in weight matrix $\mathbf{W}$ and bias vector $\mathbf{b}$ can be determined by training process with a dataset of large enough scale, usually with back-propagation and gradient descent methods.

The activation function $g(\cdot)$ here is a function that involves non-linearly that makes all the magic happen, without which cascaded products of the matrix can be simply merged into one. Two very common activation functions that will be used in this dissertation are rectified

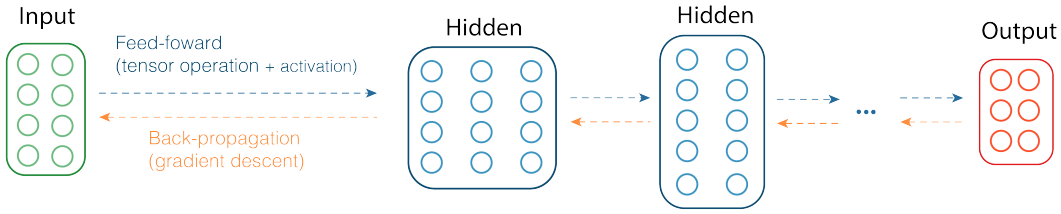

Figure 1.2. A simple feed-forward ANN. Other common ANN networks are mostly variations adopting different layer topologies and different tensor operations.

linear unit (ReLU):

$$g(x) = \max(x, 0), \tag{1.2}$$

and hyperbolic tangent (tanh) function:

$$g(x) = \tanh(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}}. \tag{1.3}$$

Other types of layer structures and layer connections that will be used in this dissertation are as follows.

1.2.1 Convolutional Neural Network

Convolutional layers are used in cases where the same local features in the matrix are supposed to have similar effects, and thus the weight can thus be shared. One example is image processing - sudden changes in colors can be considered as edges no matter where they locate in an image. Compared with fully connected layers, the convolutional layers need significantly fewer parameters for the same input. For 1-D cases, to "distribute" the weight parameter $\mathbf{W} = [w_0, w_1, .., w_{l-1}]$, sometimes referred to as filter or impulse response depending on different applications, across the whole input $\mathbf{X} = [x_0, x_1, ..]^T$, we use the convolution operation $\mathbf{Y} = \mathbf{W} \otimes \mathbf{X}$:

$$y_i = [w_{l-1}, w_{l-2}, ...w_0][x_{ik}, x_{ik+1}, ..x_{ik+l-1}]^T, \tag{1.4}$$

where k is the "stride" of the sliding filter. Also, within one layer, multiple filters can be used to form multiple columns of $\mathbf{Y}$ matrix.

1.2.2 Recurrent Neural Network

Reading is a temporal procedure involving simultaneous memorization and data process. This is how the human brain handles sequential data. Inspired by this, recurrent neural networks take each element in a data sequence and use them to update an internal state to emulate a "memory effect". Suppose at the time step i , the signal $\mathbf{X}_i$ if fed into the input, the internal hidden state H will update with:

$$\mathbf{H}_i = g\left(\mathbf{W}\mathbf{X}_i + \mathbf{U}\mathbf{H}_{i-1} + \mathbf{b}\right). \tag{1.5}$$

Meanwhile, the output at that time step is:

$$\mathbf{Y}_i = g\left(\mathbf{W}_{out}\mathbf{H}_i + \mathbf{b}_{out}\right), \tag{1.6}$$

Either all outputs can be used to create a sequential output, or the output at the last time step can be used as to "conclude" the whole sequence.

1.2.3 Generative Networks

Identifying a cat from a picture and generating a picture of cats are essentially very different types of tasks - the former *extracts* information while the latter *generates* information. For tasks requiring some "creativity" like drawings or writings, architectures of special types, usually referred to as *generative*, are supposed to be used, reasons to be discussed in the next Chapters. Some very popular generative models are:

- Generative Adversarial Network (GAN): simultaneously training a generator that generates contents and a discriminator that tries to discriminate generated content from true content.

- Variational autoencoder (VAE): cascading an encoder and decoder where the decoder serves as the generator, generating content from a given latent space parameter.
- Diffusion model: emulating a denoise process so as to generate content from random noise.
- Generative pre-trained transformer (GPT): an encoding-decoding structure that takes advantage of the multi-head attention technique.

In this dissertation, we will introduce another architecture, tandem network, in Chapter 2, which has proven to be useful for engineering problems, and we will propose a modified version of it, sequential tandem network, in Chapter 4 that empowers it to handle inverse-design with free-form design goals.

1.3 Structure of this Dissertation

Two projects will be reported in this dissertation. As an appetizer, Chapter 2 demonstrates a toy model to showcase how a generative neural network enables the inverse design evolving high non-linear EM system. We focus on a real-time beamforming structure that consists of simply 8 rotatable dielectric slabs as an example of an EM problem with great complexity. The determination of the optimal architecture and the detailed training process is discussed, and the advantage of the proposed data-driven tandem neural network method over some conventional numerical methods is shown. The system is realized and measured with a platform controlled by a low-cost system-on-a-chip (SoC) computer to verify its real-time design capability. We showcase the proposed neural network as a competitive candidate for complex real-time reconfigurable EM designs.

We then demonstrate how to migrate the same philosophy to a more complicated model but with more significance: real-time inverse design of an EM coating. Manipulating the EM scattering behavior from an arbitrary surface dynamically on arbitrary design goals is an ultimate

ambition for many EM stealth and communication problems, yet it is nearly impossible to accomplish with conventional analysis and optimization techniques.

In Chapter 3 we present a reconfigurable conformal metasurface prototype that can sever as a coat an arbitrary 2-D surface. Then in Chapter 4 we present a workflow that enables it to respond to multiple design targets on the reflection pattern with extremely low on-site computing power and time. The metasurface is driven by a sequential tandem neural network which is pre-trained using actual experimental data, avoiding any possible errors that may arise from calculation, simulation, or manufacturing tolerances. The platform empowers the surface to operate accurately in a complex environment, including varying incident angle and operating frequency, or even with other scatterers present close to the surface. The proposed data-driven approach requires a minimum amount of prior knowledge and human effort yet provides maximized versatility on the reflection control, stepping towards the end form of intelligent tunable EM surfaces.

Finally, in Chapter 3, we conclude how this approach can be universally used or modified to other complicated EM systems, we list several exciting examples to be explored in the future, including redistributing the energy inside a cavity and solving sensitivity issue in EM structure designs.

Chapter 2

A Beamformer Design with Rotatable Scatterers

Exploring how ANNs can benefit engineers and scientists in solving electromagnetic (EM) problems has been an active topic across the community for the past decade. Most straightforwardly, since feedforward neural networks (FNNs) or convolutional neural networks (CNNs) can approximate complicated nonlinear functions, they can be used to replace the computationally expensive EM simulations and provide a cost-efficient approach to a wide range of analytical problems, to name a few, DOA estimation with small amount of elements [7], directivity estimation for dipole antenna arrays [8], channel estimation for multiple-input multiple-output (MIMO) systems [9], inverse EM scattering problems for imaging [10, 11], and building a fast solver for the Poison equation [12]. Once the analytical model is made less resource-intensive, inverse design can also be realized with simply a fine search, like designing monopoles in [13], or with other iterative or non-iterative optimizers, like optimizing antenna multi-parameters in [14] and [15].

To take a step further, the inverse design problem can also be solved with purely neural network approaches. In some cases, an FNN or a simple modified version of an FNN can do the job. Examples include antenna array synthesis [16], determining loop antenna radius with given radiation patterns [17], and multiobjective antenna design by adding an extreme learning machine (ELM) layer after the FNN [18]. However, in most cases, the one-to-many mapping issue makes

these naive network architectures impossible to build. To address this, some more delicate networks are reported: [19] and [20] propose to design photonic metasurfaces/metamaterials with a generative adversarial network (GAN) and networks with bidirectional configuration, respectively. Other studies propose generator-predictor architectures for optimizing metasurface with a specific frequency response [21], and modulating linear arrays with a certain beampattern [22]. Though the performance of the methods described above is very well examined, two factors are rarely mentioned:

1. The necessity of using the ANN method: considering that the analytical model is easy to implement or may even have analytical solutions, the ANN method does not necessarily over-perform conventional models. For example, the beamforming problem in [22] may also be solved with sidelobe suppression algorithm reported in [23];

2. The real-time capability: one advantage of ANN methods, especially when compared to iterative algorithms, is its very little on-site computing time *after* the network is trained. If real-time reconfigurability is not required, the time in generating training samples makes ANN approaches bear the same pain as conventional methods.

In this Chapter, we introduce a 2-D structure inverse design problem to enhance radiation towards given directions. The complexity of both the analysis and synthesis aspects of the problem is discussed in detail in section 2.1, followed by our solution with the generator-predictor neural network in section 2.2. Finally, in section 2.3, we develop a platform with the commercially available SoC computer Raspberry Pi to verify the feasibility of real-time reconfigurability.

2.1 Diffractive Slab Model

The geometry of the 2-dimensional design discussed in this work is illustrated as in Fig.2.1. An out-of-plane electric point source, which excites an outgoing TM_z wave, is surrounded by eight dielectric slabs, with a separation of $a = \lambda/5$. The material of the slabs is

chosen to be Rogers TMM13i to take advantage of its high dielectric constant of around 12.85 that can produce strong scattering needed for creating directionality. The slabs are designed to rotate freely in-plane to realize reconfigurable beamforming, and their rotational angles are denoted as $\Theta = [\theta_1, ..., \theta_8]^T$. Our objective is to establish both analytic and synthetic methods for the problem, i.e., 1) predicting the directivity given the rotational angle Θ, and 2) inverse designing Θ values in a real-time scenario to enhance the radiation towards a certain direction or several certain directions. The normalized far-field pattern can be evaluated to this intent:

$$D(\phi) = \frac{|E_{\text{far}}(\phi)|}{\sqrt{\sum_{\varphi \in \Phi} |E_{\text{far}}(\varphi)|^2}}, \tag{2.1}$$

where Φ represents the vector consisting of all observation angles.

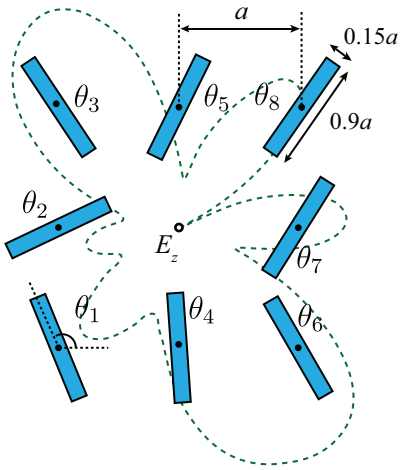

Figure 2.1. 2-D geometry of the beamforming structure.

2.1.1 Nonlinear Nature of the Problem

The solution for the scattering by a single dielectric slab from a point source, as shown in Fig. 2.2 , can be theoretically calculated with the uniform geometrical theory of diffraction

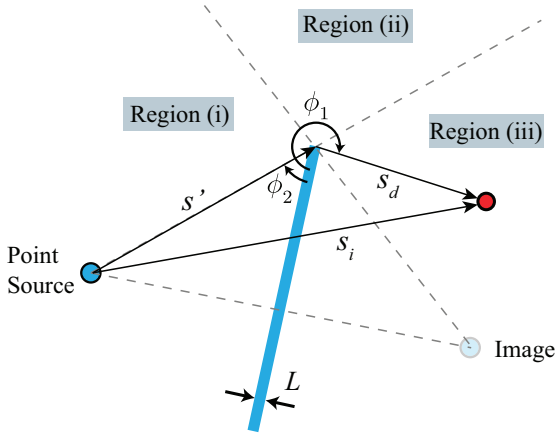

Figure 2.2. Illustration of the scattering of a point source by a single dielectric slab.

(UTD) [24], assuming the slab is electrically long and thin:

$$E_z^{(ii)} = \frac{e^{-jks_i}}{\sqrt{s_i}} + C\frac{e^{-jk(s'+s_d)}}{\sqrt{s_i s_d}}, \tag{2.2}$$

$$E_z^{(iii)} = T\frac{e^{-jks_i}}{\sqrt{s_i}} + C\frac{e^{-jk(s'+s_d)}}{\sqrt{s_i s_d}}, \tag{2.3}$$

,where transmission coefficient T can be written in terms of the reflection coefficient R_1 at the interface between the air and the dielectric [24]:

$$T = \frac{(1-R_1^2)P_d P_t}{1-R_1^2 P_d^2 P_a}, \tag{2.4}$$

$$R_1 = \frac{cos\theta_i - \sqrt{\varepsilon_r - sin^2\theta_i}}{cos\theta_i + \sqrt{\varepsilon_r - sin^2\theta_i}}, \tag{2.5}$$

in which the phase delay terms are:

$$P_t = e^{jkd\cos(\theta_i-\theta_t)/\cos(\theta_t)}, \tag{2.6}$$

$$P_a = e^{jk2d\sin\theta_i\sin\theta_t/\cos(\theta_t)}, \tag{2.7}$$

$$P_d = e^{jk_r d/\cos(\theta_t)}. \tag{2.8}$$

The diffraction term C is given by:

$$C = \frac{-e^{-j\pi/4}}{2\sqrt{2\pi k}} \cdot \left[\frac{F(kLa\phi')}{\cos(\phi'/2)} \pm \frac{F(kLa\phi'')}{\cos(\phi''/2)} \right], \tag{2.9}$$

where $\phi' = \phi_1 - \phi_2$, $\phi'' = \phi_1 + \phi_2$, and

$$F(x) = 2j\sqrt{x}e^{jx} \int_{\sqrt{x}}^{\infty} e^{-j\tau^2 d\tau}. \tag{2.10}$$

While this expression is already complicated for implementing inverse design, several extra factors make the problem even more complex: to start with, the size of the slabs in this model may not be electrically large enough to ensure the accuracy of this approximation; in addition, strong coupling and multi-reflection may exist between the slabs which sit so close together. These make it nearly impossible to find an analytical expression of its near-field distribution or far-field pattern. To illustrate this complexity, Fig.2.3 shows the simulated normalized far-field at the azimuth directions $\phi = 0^\circ, 90^\circ, 180^\circ, 270^\circ$, with respect to θ_1 and θ_2 combinations. The rest of the parameters θ_3 to θ_8 are fixed at a set of random values. It is quite obvious that the relation between the directivity and the parameters θ is highly nonlinear and is characterized by strong fluctuations.

2.1.2 Discussion on Possible Approaches

Common conventional optimizers, such as gradient descent algorithms or genetic algorithms, are based on an iterative process, which essentially prohibits them from operating in real time. However, if the forward model could be expressed in a relatively simple form (which can be numerical), these algorithms would not have to perform computationally expensive calculations such as full-wave simulation for every sample during the iterations, allowing them to achieve inverse design fast enough to be considered as real-time. Here we examine two approaches for that purpose targeting at approximating the directivity with rotational angles: interpolation and

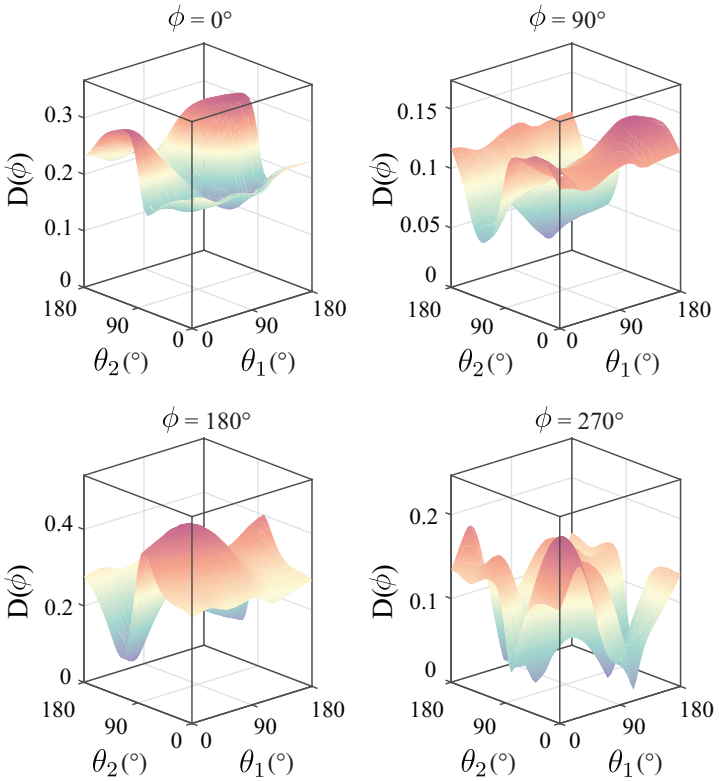

Figure 2.3. Directivity with respect to the combination of θ_1 and θ_2. The random parameters $[\theta_3, ..., \theta_8] = [44.5°, 25.2°, 112.9°, 0.2.5°, 150.5°, 82.8°]$.

multivariate regression.

8-D Interpolation

In order to establish a complete map of Θ and $D(\phi)$, one needs to perform a grid search on the 8-dimensional input parameter space as the sample points. Table 2.1 lists the total number of simulations needed for different sweep resolutions, with the simple formula $(180°/\Delta\theta)^8$. From Fig. 2.3, it is safe to claim that even for a very coarse estimation, $\Delta\theta$ should be smaller than 30°, which brings the sample number to the order of 10^6. Additionally, obtaining more accurate results would be exponentially more difficult. At a 10° resolution, the sample size of around 10^{10} renders this approach quite infeasible even when each full-wave simulation takes only seconds.

Table 2.1. # of Simulations needed vs. sweep resolution, using interpolation methods.

$\Delta\theta$	60°	45°	30°	10°
# of Samples	6561	65536	1.68×10^6	1.10×10^{10}

Multivariate Regression

The multivariate polynomial regression method can be used with the form of:

$$D(\phi) = \sum_{p_1+...+p_8 \leq K, i=1..M} \beta_i \theta_1^{p_1} \theta_2^{p_2} ... \theta_8^{p_8}. \tag{2.11}$$

The parameter can be calculated with $\hat{\beta} = (\underline{\underline{\Theta}}^T \underline{\underline{\Theta}})^{-1} \underline{\underline{\Theta}}^T \underline{D}$, where the matrix $\underline{\underline{\Theta}}$ has a dimension of $N_s \times M$, N_s being the number of samples and M being input dimension of the variables, which is listed in Table 2.2. The same issue occurs here that for higher accuracy, the scale of the problem expands exponentially, but it is relatively acceptable compared to the interpolation method. Nevertheless, in Section 2.2.2, we will further show that this method significantly underperforms the proposed neural network predictor even when the order is prohibitively high.

Table 2.2. Input dimension vs. order of polynomial regression

Order K	2	3	4	5	6	7	8
Input Dim M	45	165	495	1287	3003	6435	12870

It is worth noting that even if the function $D(\phi) = f(\Theta)$ can be determined, it is still challenging to implement the inverse design using the aforementioned iterative methods. In order to achieve fast convergence, they may still require high on-site computing power.

2.2 Generative Deep Neural Network Methods

Simple FNNs, even the single-layered ones, can theoretically approximate any function in any precision if the scale is large enough. In our case, it is tempting to think that if an underlying function $\Theta = f^{-1}(D(\phi))$ exists, one may assign $D(\phi)$ and Θ as the input and output of an FNN

and expect it to approximate $f^{-1}(\cdot)$, so that Θ can be determined with a target $D(\phi)$. This is however not true since there exist multiple sets of input Θ that map to the same (or very similar) output $D(\phi)$, making it impossible to properly train the network. This will be further discussed in Section 2.2.2. Intuitively for most inverse design problems, the design parameters, in this case the structure's geometry, contain more information than the targets, in this case the amplitude of the pattern in certain directions. As a result, a generative network is often needed instead of a naive FNN.

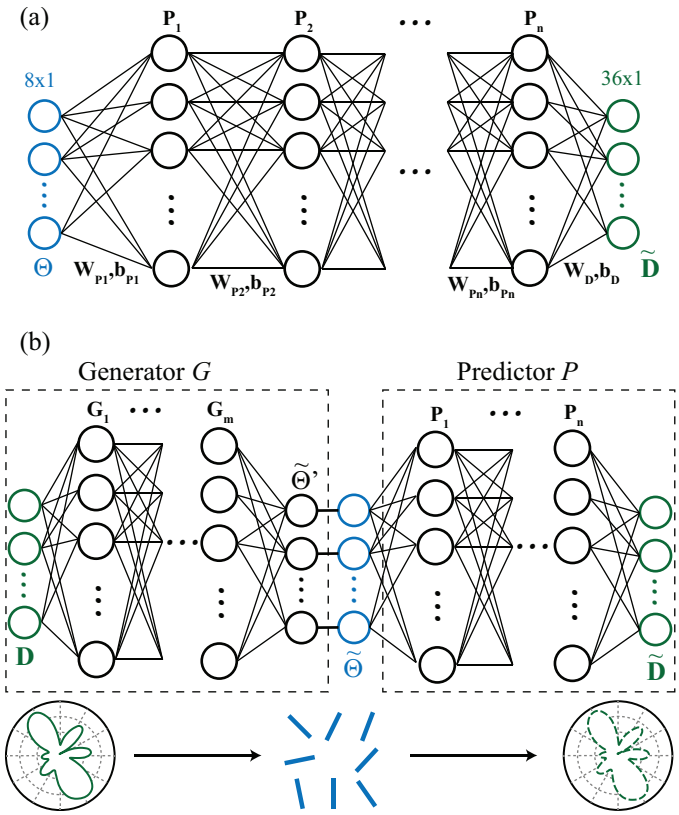

Figure 2.4. Tandem network architecture. The architecture of (a) the prediction network P, and (b) the whole network including the both generation network G and the prediction network P.

2.2.1 Network Architecture

In this study, a generative network, tandem neural network, based on a pre-trained prediction network is adopted, similar architectures proved useful for other physical design

problems such as photonic nano-particles [25]. Fig. 2.4 (a) illustrates the predictor P, which is a simple $n+1$ layered fully-connected neural network with the rotational angles as input parameters and directivity $D(\Phi)$ as the output, where the set of azimuth angles $\Phi = [0^\circ, 10^\circ, ..., 350^\circ]^T$. This can be expressed as:

$$\mathbf{P}_1 = g(\mathbf{W}_{P_1}\Theta + \mathbf{b}_{P_1}), \tag{2.12}$$

$$\mathbf{P}_k = g(\mathbf{W}_{P_k}\mathbf{P}_{k-1} + \mathbf{b}_{P_k}), k = 2...n, \tag{2.13}$$

$$\tilde{\mathbf{D}} = g(\mathbf{W}_D\mathbf{P}_n + \mathbf{b}_D), \tag{2.14}$$

in which $\mathbf{W}$ and $\mathbf{b}$ are the weight matrix and bias vector for each layer, and the function $g(\cdot)$ is the activation function, which we here choose to be the most common ReLU function. For a stable learning process, the vector Θ and $\mathbf{D}$ are normalized to $[0,1]$ with a scalar $\mathbf{X} \leftarrow \frac{\mathbf{X}-\min(\mathbf{X})}{\max(\mathbf{X})-\min(\mathbf{X})}$.

Note that the form of output depends on the form of the design target, and in this Chapter, we choose to describe the required directivity in a 10° resolution. Since the $D(\phi)$ is a continuous and smooth function with respect to Θ, this network can be easily trained.

The trained predictor is then used as the discriminator of a designer network G, which takes the target $D(\Phi)$ as the input and yields a predicted set of rotational angles $\tilde{\Theta}$, forming a network structure illustrated in Fig. 2.4 (b). The generator G has a similar form to the predictor P, with an additional layer $\tilde{\Theta}'$ to limit the rotational angle to $0^\circ - 180^\circ$:

$$\mathbf{G}_1 = g(\mathbf{W}_{G_1}\mathbf{D} + \mathbf{b}_{G_1}), \tag{2.15}$$

$$\mathbf{G}_k = g(\mathbf{W}_{G_k}\mathbf{G}_{k-1} + \mathbf{b}_{G_k}), k = 2...m, \tag{2.16}$$

$$\tilde{\Theta}' = g(\mathbf{W}_\Theta\mathbf{G}_m + \mathbf{b}_\Theta) \tag{2.17}$$

$$\tilde{\Theta} = \min(\tilde{\Theta}', 1). \tag{2.18}$$

For this network, the goal is to decrease the discrepancy of the target directivity $D(\Phi)$ and $\tilde{D}(\Phi)$, which is predicted by P with $\tilde{\Theta}$. The difference between predicted $\tilde{\Theta}$ and the actual

rotational angle Θ in the training set is out of the scope in this setup, thus the one-to-many mapping issue aforementioned is avoided, and the weights of the network can converge during the training process.

2.2.2 Training Process and Results

The dataset used to train the network is obtained in the commercial software COMSOL Multiphysics using the 2-D frequency domain FEM method. For each setup, the rotational angles are set to uniformly distributed random values in $[0^\circ, 180^\circ)$ and a circular radiation boundary with a radius of $4a$ is used to collect the complex far-field information, which is then transformed into the desired form, in this case, a normalized 36x1 vector of the magnitude of directivity. 95% of the data are used for training while the remaining 5% serve as the validation set to evaluate the performance of the network. We pick the size of the validation to be less than common practice 20-30% based on two reasons: 1) the simulation results contain much smaller statistical errors than those in other common NN applications, such as image pattern recognition, so less validation data are needed to cover the variations, and 2) the training set size has a larger impact on the performance compared to the architecture of the network itself, as will be seen later in Fig. 2.5 and Fig. 2.6, as a results, a larger training set is preferred to reduce the computationally expensive simulations needed.

The construction and the training of the network are done with the open-source machine learning platform Tensorflow and the high-level API Keras for Python 3. For a feed-forward network in general, the selection of the cost function is based on the distribution of the output variables. In our case where there is no prior knowledge on the distributions for D, we set the cost function to be the Gaussian cross-entropy, i.e., the mean square error (MSE) between the actual value $\mathbf{D}$ in the dataset and the predicted value $\tilde{\mathbf{D}}$:

$$L_P = \frac{1}{N_D} \sum_{i=1}^{N_D} (\mathbf{D}_i - \tilde{\mathbf{D}}_i)^2 \tag{2.19}$$

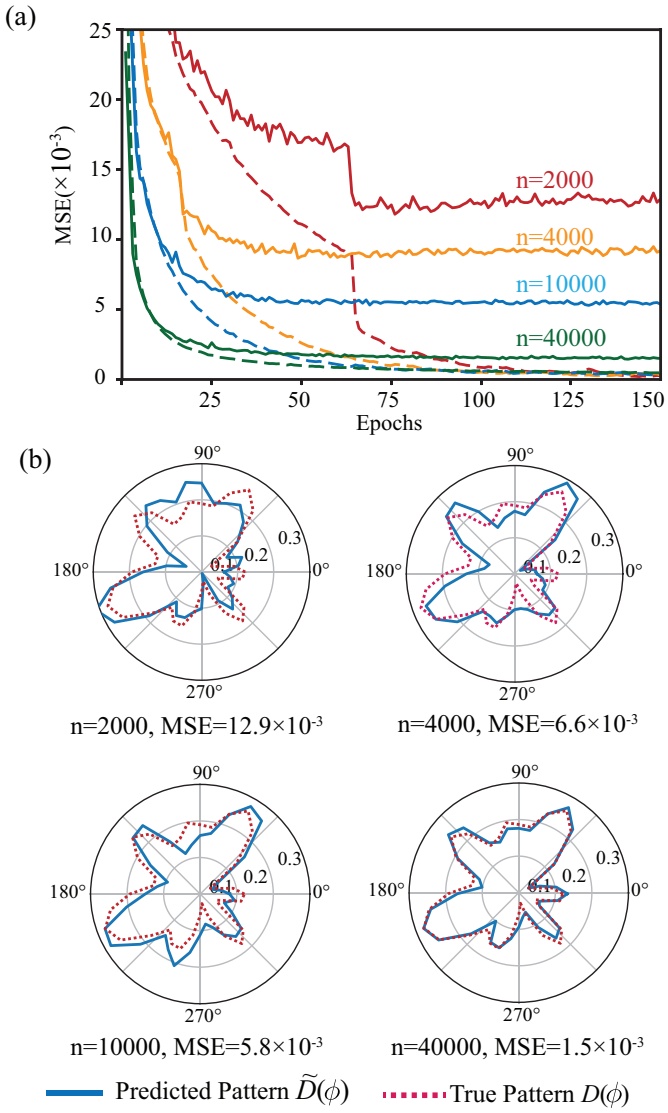

Figure 2.5. Training process of the predictor. (a) The training process of the predictor P with various dataset sizes. The solid curve and dashed curves represent the cost functions for the validation set and the training set, respectively. The mini-batch size in this setup is 100 and the learning rate is 5×10^{-4}. (b) A visual example of prediction from the validation set.

We here adopt the backward propagation optimization method Adam [26], the update of the parameters is described as:

$$\mathbf{W}_t, \mathbf{b}_t = \mathbf{W}_{t-1}, \mathbf{b}_{t-1} - \alpha \frac{\hat{\mathbf{m}}_t}{\sqrt{\hat{\mathbf{v}}_t} + \varepsilon} \tag{2.20}$$

This involves an adaptive learning rate combining a user-defined hyperparameter α as

well as the first and the second moments estimation of the gradients $\hat{\mathbf{m}}_t$ and $\hat{\mathbf{v}}_t$. The training set is divided into mini-batches and for each single iteration, one mini-batch is used, and a complete pass over the entire training set makes one epoch. The details of the algorithm can be found in [26]. The stochastic character of this method makes it more likely to jump out of a local minimum compared to simple gradient descent, as can be found in the $n = 2000$ case in Fig.2.5(a) at around epoch 70.

Fig. 2.5(a) illustrates the evolution of the loss function of the predictor along the learning process with different dataset sizes. The structure of the network, i.e., the number of layers n and the nodes number per layer are determined by a grid search method, as shown in Fig. 2.6 (a). From the two plots, we can observe the impact of both input size and the network scale on the network performance: 1) larger numbers of layers and nodes generally provide more nonlinearity thus a better fit, but beyond a certain point, the cost function converges, and overfitting starts to emerge, and 2) while the network shows extremely small error for the training set regardless of the size of the input dataset, the loss for the validation set is largely dependent on it: the increased dataset size reduces the overfitting significantly and allows the network to approach the "true" underlying expression of $f(\cdot)$.

Fig. 2.5(b) shows the patterns of the prediction for one particular case in the validation set, which is representative as its MSE is rather close to the average. It can be found that $40,000$ samples are enough to create a fairly accurate prediction with an average MSE of 1.5×10^{-3}. The size of the sample needed is remarkably smaller than that in Table 2.1 to allow a complete mapping. Table 2.3 lists the performance of the polynomial regression introduced in Section 2.1.2 with the same $40,000$ samples. The error only converges to around 7×10^{-3} before the order K becomes too large to calculate.

Table 2.3. Performance of Multivariate Polynomial Regression

Order K	2	3	4	5	6	7
MSE($\times 10^{-3}$)	16.9	14.0	10.1	8.82	7.19	6.88

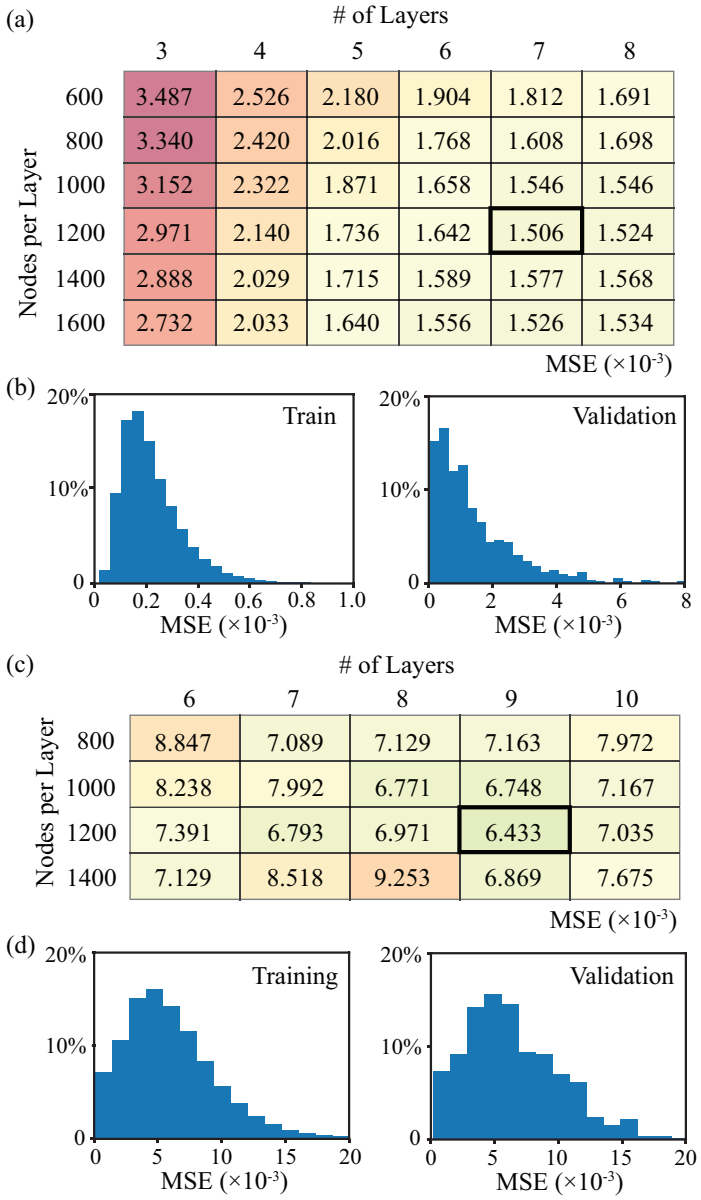

Figure 2.6. Hyper-parameter grid search to determine the number of layers and the number of nodes per layer. Results for (a) the predictor P and (c) the generator G. The networks are trained for 200 epochs, and the best-performed network is selected, equivalently an early stopping procedure to reduce over-training. The mini-batch size and the learning rate for the generators are 100 and 5×10^{-5} respectively. The final architectures used are highlighted with thick borders. (b) and (d) illustrate the error distribution for the corresponding predictor P and the generator G, respectively.

The generator is then trained in a similar manner, with the weight and bias matrix fixed for the predictor. The metric of the network is the MSE between the input directivity **D** and the

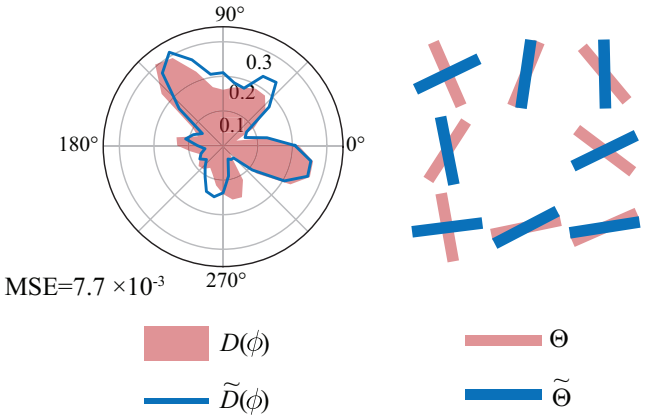

Figure 2.7. An exemplary case of the design with the target pattern from the validation set.

predicted directivity $\tilde{\mathbf{D}}$ with the predicted angle $\tilde{\Theta}$:

$$L_G = \frac{1}{N_D}\sum_{i=1}^{N_D}(\mathbf{D}_i - \tilde{\mathbf{D}}_i)^2 \tag{2.21}$$

Fig. 2.6 (b) lists the gird search results of the generator G, and the architecture of the whole network is thus determined. Fig. 2.7 shows an exemplary result of the predicted angle $\tilde{\Theta}$ and their corresponding pattern $\tilde{\mathbf{D}}$ from one sample in the validation set, in comparison to the actual angle Θ and pattern $\mathbf{D}$. It can be seen that even though similar patterns are generated, the designed rotational angles are quite different from those in the dataset, which confirms the one-to-many mapping issue in the naive architecture: consider both cases appear in the training set, with (2.20), the gradients of the two are likely to cancel out and leave the weight and bias matrix trapped at a local minimum.

Now we may validate the network's performance for requests on arbitrary patterning. Our main focus is to maximize the radiation towards one or more directions with various beamwidths. The elements in target $\mathbf{D}$ are set to binary values that cover the desired angles for enhanced radiation. We can evaluate this enhancement effect by examining the overall power gain in the

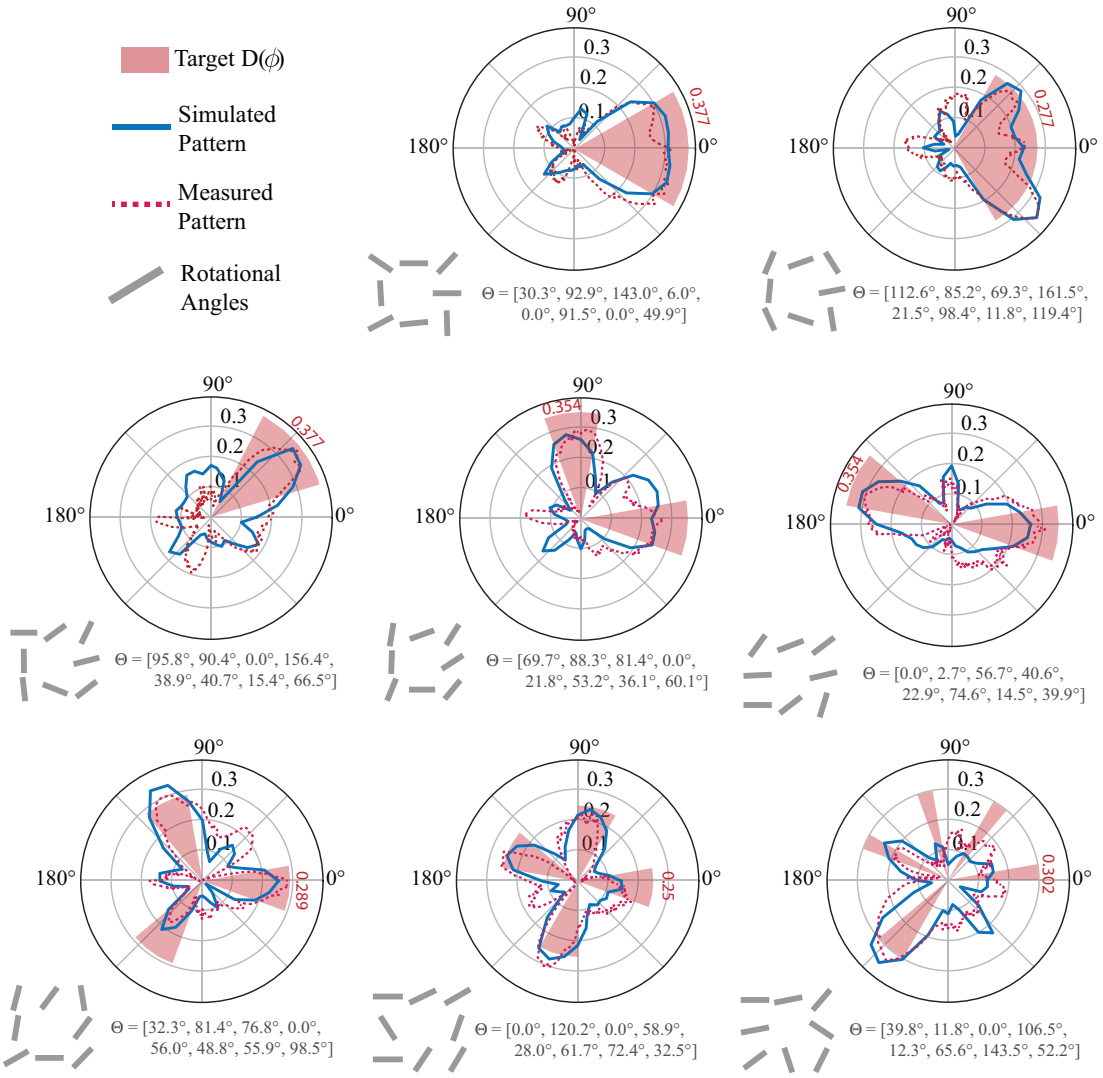

Figure 2.8. Visual examples of the inverse design for arbitrary request on the directivity. The binary value of the input features and outputs angles are marked with the plot of patterns and slabs, respectively.

target directions compared with an isotropic radiator:

$$Enh \equiv \frac{1}{N_{D,\text{target}}} \sum_{\phi \in \Phi_{target}} D^2_{\text{iso}}(\phi), \tag{2.22}$$

in which the isotropic directivity relates to the normalized directivity by

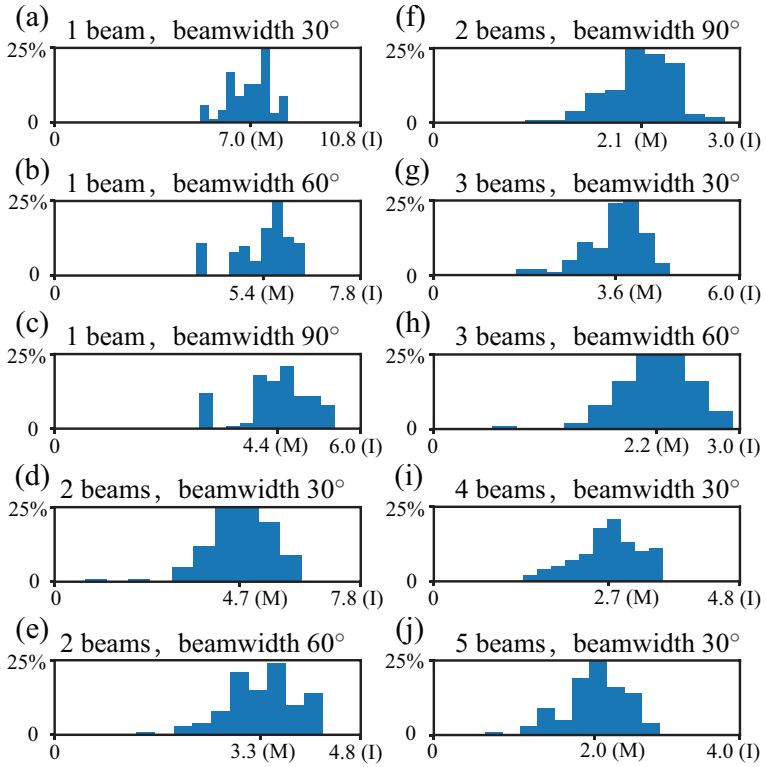

Figure 2.9. Distribution of enhancement *Enh* in dB on samples with random beam angles. For multi-beam scenarios, the beams are separated at least 10° from each other. Each scenario contains 100 sets. The mark (M) represents the Mean value and (I) represents the maximum possible value assuming an Ideal case where all energy radiate within target directions.

$$D_{\text{iso}}(\phi) = |E_{\text{far}}(\phi)| \,/\, |E_{\text{far,iso}}(\phi)| = \sqrt{N_D} D(\phi) = 6D(\phi). \tag{2.23}$$

Fig. 2.9 plots the distributions of *Enh* for randomly generated requirements. Some visual examples are given in Fig. 2.8.

From both figures we can find that impressively, with a small dimension of $3/5\lambda \times 3/5\lambda$ and only 8 adjustable parameters, this setup is able to form beam patterns that effectively enhance the radiation towards desired directions. The model works very nicely for requirements with beamwidth larger than 60°, in some cases even apporaching the maximum possible number. For narrower beamwidths, strong directivities can be achieved even though they do not approach the ideal number, which is as expected considering the small electrical size of the model. Slight

degradation occurs when very complicated patterns are requested, such as four or five separated beams where the model still provides an obvious overall advantage over isotropic radiation. Improvements are expected with larger physical size and more parameters (slab numbers), within the limitations on size and cost for different applications.

2.3 Real-time System Realization and Measurement

One of the main objectives of this study is to validate that the proposed neural network can be deployed on a system with very limited computing power to perform real-time inverse design. We choose the commercially available SoC Raspberry Pi 4 (RPi for short) with quad-core Cortex-A72 CPU and 4GB RAM. Tensorflow Lite is used to load the network and execute the inverse design. We find that this miniaturized portable computer provides more than enough computing power to allow a negligible design time (around 199 ms per design).

The whole device under test (DUT) is supported by an assembled 3-D printed frame, shown in Fig. 2.10(b). The slabs are made with 3.81 mm (150 mil) Rogers TMM13i laminate cutting to 22.86 mm (0.9 inch) by 1.5 mm, sitting on a 21 cm by 21 cm ground plane. The corresponding operating frequency is 4.725 GHz. The slabs are mounted on eight LJWRC 35kg servo motors, whose rotational angles are controlled by pulse-width modulation (PMW) signals. A 16-Channel 12-bit PWM servo driver is used as the interface between the RPi and the motors. The motors take approximately 300 ms to 500 ms to rotate depending on the rotational angles. The response time of the network and the motor makes this particular setup most fits the scenarios where a response time on the order of second is needed. A circular top cover with a rounded edge to avoid the corner and edge currents is then put on top, so that the slabs are in between two parallel plates, emulating a 2-D environment.

A vector network analyzer (VNA) Keysight E5071C and a modified Creality Ender3D printer, serving as an XYZ stage, are employed for the field measurement. One vertical probe made with a rigid SMA cable is fixed at the center as the E_z excitation and another E_z probe is

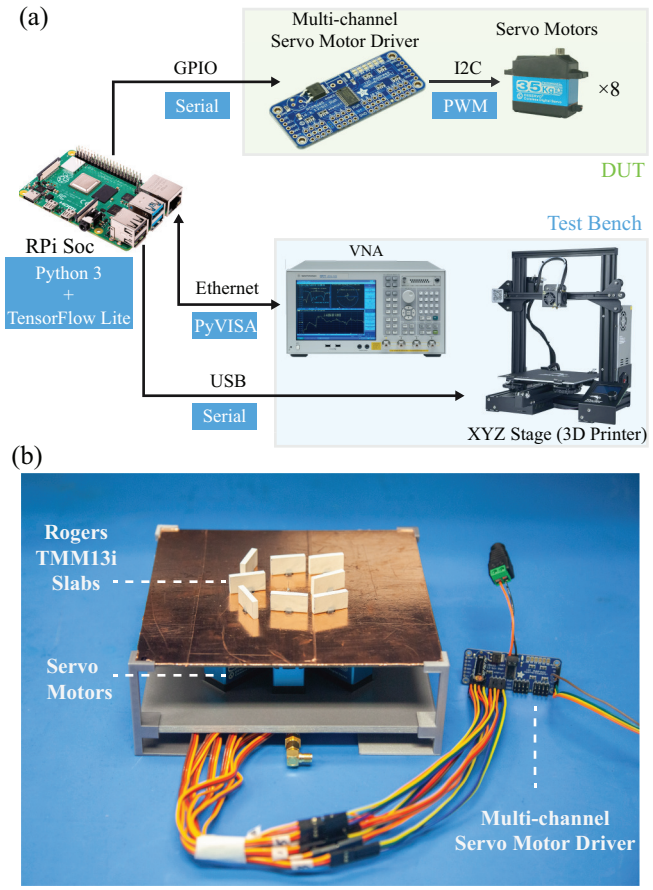

Figure 2.10. (a) The whole system controlled by the RPi, including both the DUT and the test bench. The employed environment, interfaces and protocols between the devices are listed. (b) A picture of the DUT, the top cover not included for a better view of the parts.

attached on the extruder of the 3D printer as the receiver to perform the scan. Sheet absorbers are also applied on the frame of the 3D printer to reduce their impact on the field distribution. The RPi controls both the VNA and the XYZ stage to realize automatic measurements. A whole diagram of the system, including the DUT and the test bench, is shown in Fig. 2.10 (a). The measured pattern is plotted in Fig. 2.8. Despite operating in a non-ideal 2-D scenario, the measured directivity shows reasonable agreement with the simulated results featuring enhanced radiation towards the required directions.

2.4 Conclusion

In this Chapter, a real-time beamforming system is presented with a generative network method, which is otherwise very difficult, if not impossible, to implement with conventional methods using very limited computing power. Detailed discussions are made to justify and find the optimal architecture of the network. The device and the test bench are built to verify the feasibility and the performance of this method. Further studies based on this approach may include constructing latent space to gain better knowledge of the capability of the model and adapt the initial setup according to projects' needs, and achieving 3-D pattern forming by introducing more parameters like slab tilting.

In a more general aspect, this work showcases the superiority of neural network methods as a highly nonlinear regressor and optimizer for EM problems. Notably, this tandem architecture can be easily accommodated to other nonlinear EM inverse design problems, with little additional analysis needed, such as electrically tuned beamformers or metasurfaces integrated with highly non-linear components, and conformal arrays in very irregular shapes. We believe this approach is a superior platform for future quick projects on real-time reconfigurable designs involving complex mechanisms, as is presented in the following chapters.

This chapter is a reprint of the paper: Wen, E., Yang, X., & Sievenpiper, D. F. (2022). "Real-time 2-D beamforming with rotatable dielectric slabs enabled by generative neural network". *IEEE Transactions on Antennas and Propagation*, 70(9), 8360-8367. The dissertation author was the primary investigator and author of this paper.

Chapter 3

Flexible Reconfigurable Metasurface

As is introduced in Chapter 1.1, it has been decades since the idea was first proposed to control EM field behavior with metamaterials - structures with subwavelength geometrical details [2, 27, 28]. In particular, its 2-D version, i.e. metasurface, draws broad attention and is intensively investigated due to its advantage in engineering aspects - being able to be manufactured relatively easily on thin sheet materials [29]. The reported application space of metasurfaces is vast, ranging from directing surface waves in the near-field [4], beam-forming in the far-field [5], to cloaking [30] and holography[31, 32], etc. Beyond the planer regime, efforts have been made to implement flexible metasurfaces in hope of bringing these intriguing wave manipulation capabilities to surfaces with arbitrary shapes. However, the mechanism of wave interaction with curved surfaces is significantly more complex than its flat counterpart [33], and as a result, research has mainly focused on optimizing for specific tasks such as wave-front control [34], radar-cross-section (RCS) reduction [35] or polarization conversion [36]. To realize a reconfigurable version is even more challenging, not only because of the difficulty in balancing mechanical properties and EM performance but also because of the lack of accurate and efficient algorithms to support the inverse design.

Meanwhile, recent years have witnessed the emergence of exploiting ANNs in complicated EM/photonic systems. On the one hand, as a good regressor of highly non-linear functions, ANNs provide a cost-efficient solution to many analysis problems, from solving Poisson's equa-

tions [12], to handling inverse EM scattering problems [37]. On the other hand, recent research demonstrates the strong design capability of ANNs, including optimizing linear phased arrays [22], or designing photonic devices and nanoparticles [21, 19, 20, 38, 39]. One special advantage of a pure NN-driven scheme, compared to conventional optimization methods, is that no iterative process is involved in the prediction phase, which is crucial for an on-site system in need of fast response. This feature has been exploited recently to facilitate an emerging concept - the intelligent metasurface, which refers to metasurfaces that tune themselves in an adaptive manner, with little human intervention [40], from beam-forming [41], to sensing purposes [42, 43]. In theory, a similar strategy can be used for curved surfaces, and preliminary studies on NN-driven non-planer surface has been reported for cloaking or "illusion" applications [44, 45, 46], yet a more universal and versatile scheme is still needed for the surface to dynamically operate under different types of tasks.

This chapter, along with Chapter 4, aims to demonstrate one possible universal ANN-driven scheme to realize an intelligent surface that can respond to arbitrary design goals. We start by demonstrating a practical realization of a tunable conformal coating operating at microwave frequencies, whose reflection pattern can be controlled with multi-channel bias voltages. We demonstrate the difficulty in both the analysis aspect and synthesis aspects of this system. In the next chapter, a sequential neural network architecture will then be proposed to address this, which can take free-form design targets on the pattern and environmental factors as the input. Instead of using data obtained with full-wave simulations to train the network, as is a common practice in most ANN-assisted EM research, we propose using actual measurement data, which turns out to be fast, accurate, and very adaptive to different environments.

3.1 Metasurface Geometry Design

The conformal metasurface design is based on a classical tunable reflective metasurface topology in [47]. The surface is tiled with sub-wavelength metallic patches, with varactor diodes

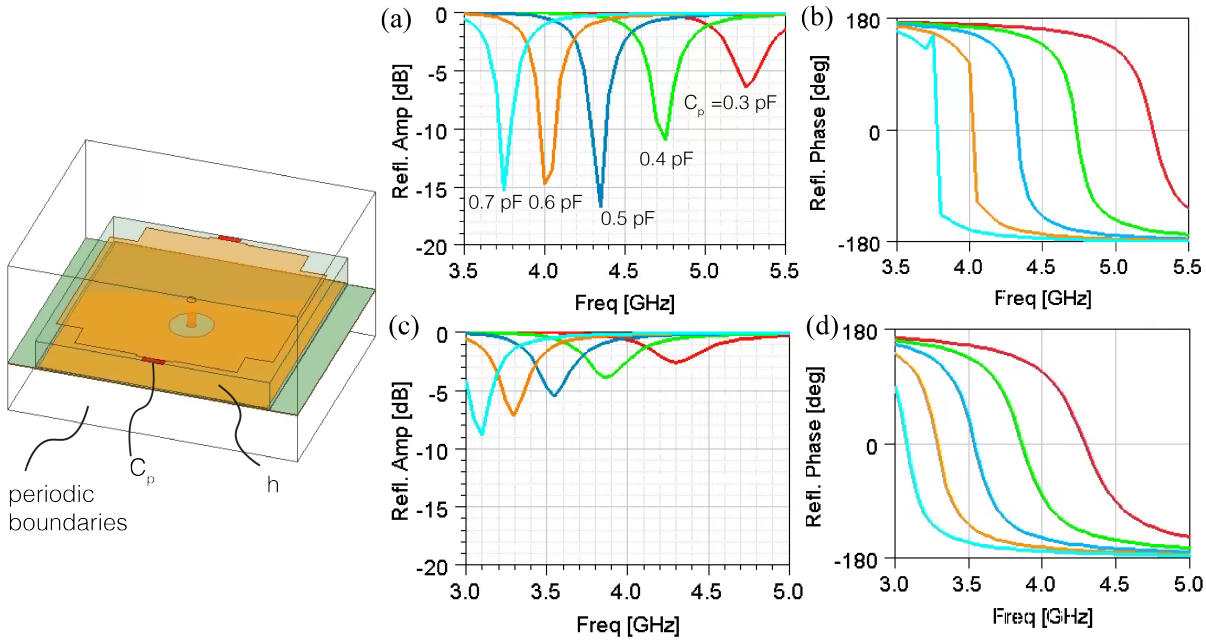

Figure 3.1. Simulated reflection amplitude and phase of a unit-cell. (a), (b) are results for $h = 30mil$ and (c), (d) are results for $h = 60mil$.

placed between the neighboring unit cells. By changing the reverse bias voltage across a varactor diode, the reflection spectrum of a unit can be shifted. This results in tuning the local reflection phase in a frequency range close to the resonance, and collectively all units form a reflection pattern in the far-field region.

While this continuous-phase tuning approach generally provides greater degrees of freedom than discrete-state tuning methods like binary phase states with pin-diodes (also known as phase coding) [48], it is more sensitive to dissipation loss which leads to a significant decrease in reflection amplitude near the resonant frequency. Fig. 3.1 shows simulation results of a unit-cell with different substrate thicknesses with different capacitance values for the varactor, using ANYSY HFSS. The results show that:

- thinner substrate decreases the efficiency of the reflection near resonance, this is as expected since thicker substrate allows better radiation for the patches;
- thinner substrate increases the quality factor of the resonance, causing a steeper slop in

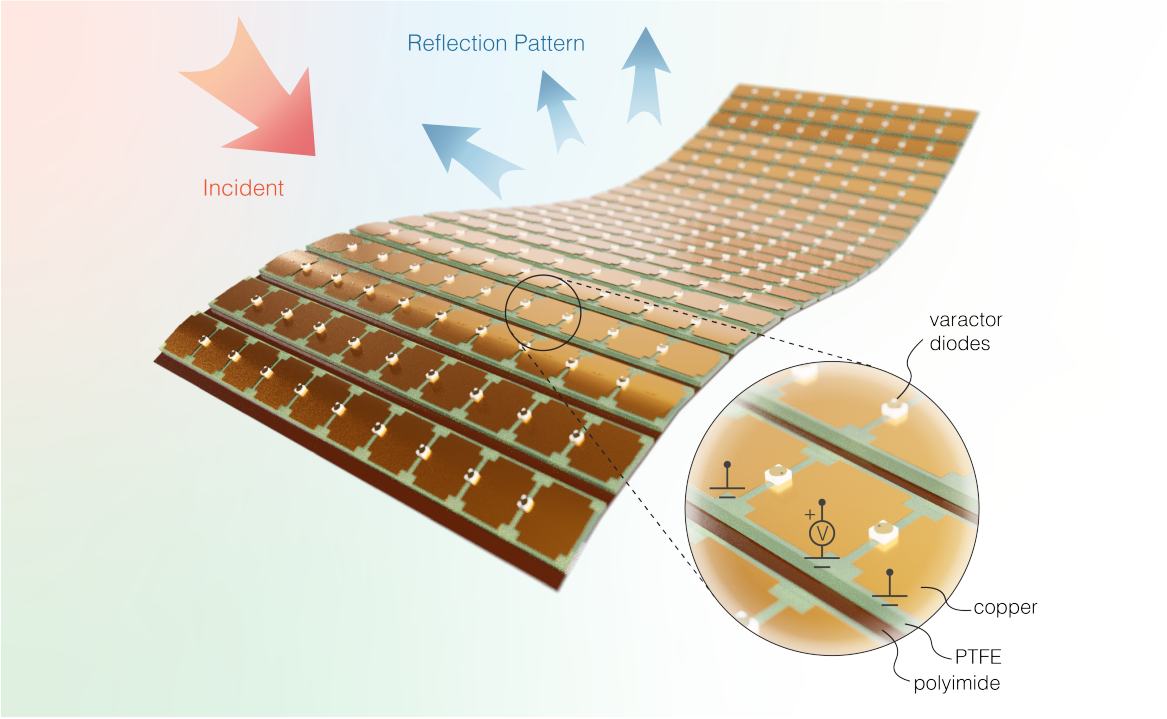

Figure 3.2. Illustration of the conformable rigid-flex printed circuit design. The substrate of each column is made with 1.57mm semi-rigid reinforced PTFE material designed for microwave applications. They are then bonded to a single flexible sheet made of 0.18 mm thick polyimide, with a feeding network on the bottom side to provide the varactors with d.c. bias voltages.

the reflection phase, increasing the sensitivity of the varactor, and

- thinner substrate shifts the operating frequency to the higher end, making the unit electrically larger, potentially causing grating lobe problems.

For these considerations, in this design thicker substrate is always preferred, and low-loss materials designed for radio frequency (RF) use are preferred to further reduce dielectric loss.

Thick RF materials are not typically flexible, so here we propose a double-layered rigid-flex stacking structure to provide the overall flexibility of the coating: unit cells are implemented using relatively thick microwave materials, which are separately attached to a single ultra-thin flexible layer that also contains circuits for bias feeding. In this demonstration, we built 24 separate columns, with 10 patches in each column, making a 38.51 cm $\times$ 13.64 cm $\times$ 1.97 mm surface. Units on each column share the same reversed bias voltage, forming a 24-dimensional

vector **V**, realizing pattern control in the azimuth plane, with intensity noted as $D(\theta)$.

The photo of the prototype is shown in Fig. 3.4 (a). It is a rigid-flex design is a 4-layer rigid-flex design shown in Fig. 3.3. The rigid layer is made with 62 mil Rogers RT/Duroid 5880 material and the flexible layer is made with 2 mil polyimide substrate, bonded together with 6 mil prepreg, shown as Fig. 3.4 (b). The unit periodicity is 13 mm along the column and 16.05 mm across the column with a 3.05 mm separation in between columns. The top copper layer is protected with ENIG finish and the solder mask is only around varactor footprints for soldering purposes. Ground plane copper layer and d.c. bias feeding network copper layer are protected with flexible coverlay, as shown in Fig. 3.4 (c).

To further reduce loss, GaAs Hyperabrupt high-quality-factor varactor diodes Macom MA46H070-1056 are used across the units. To protect the varactors, each channel is in series with by a $10k\Omega$ series resistor RNCF0603TKY10K0 by Stackpole Electronics Inc.

3.2 Static Reflection Measurement

The static reflection spectrum of a flat metasurface is measured with a single horn RCDLPHA2G18B by RF-Lambda. S_11 is recorded with an Agilent E5071C VNA for 3 cases:

- bare horn, $S_{11,h}$. This is the reflection from the horn itself;
- an aluminum plate, as a perfect electrical conductor (PEC), of the same size, $S_{11,PEC}$. This is the reflection from both the horn and a total reflection of a plane with a 180° phase shift, and
- with metasurface in front of the horn, $S_{11,s}$, all channels set from 0 V to 18 V. This is the reflection from both the horn and the surface.

The complex relative reflection on the surface is thus:

$$R = \frac{S_{11,s} - S_{11,h}}{S_{11,PEC} - S_{11,h}} \tag{3.1}$$

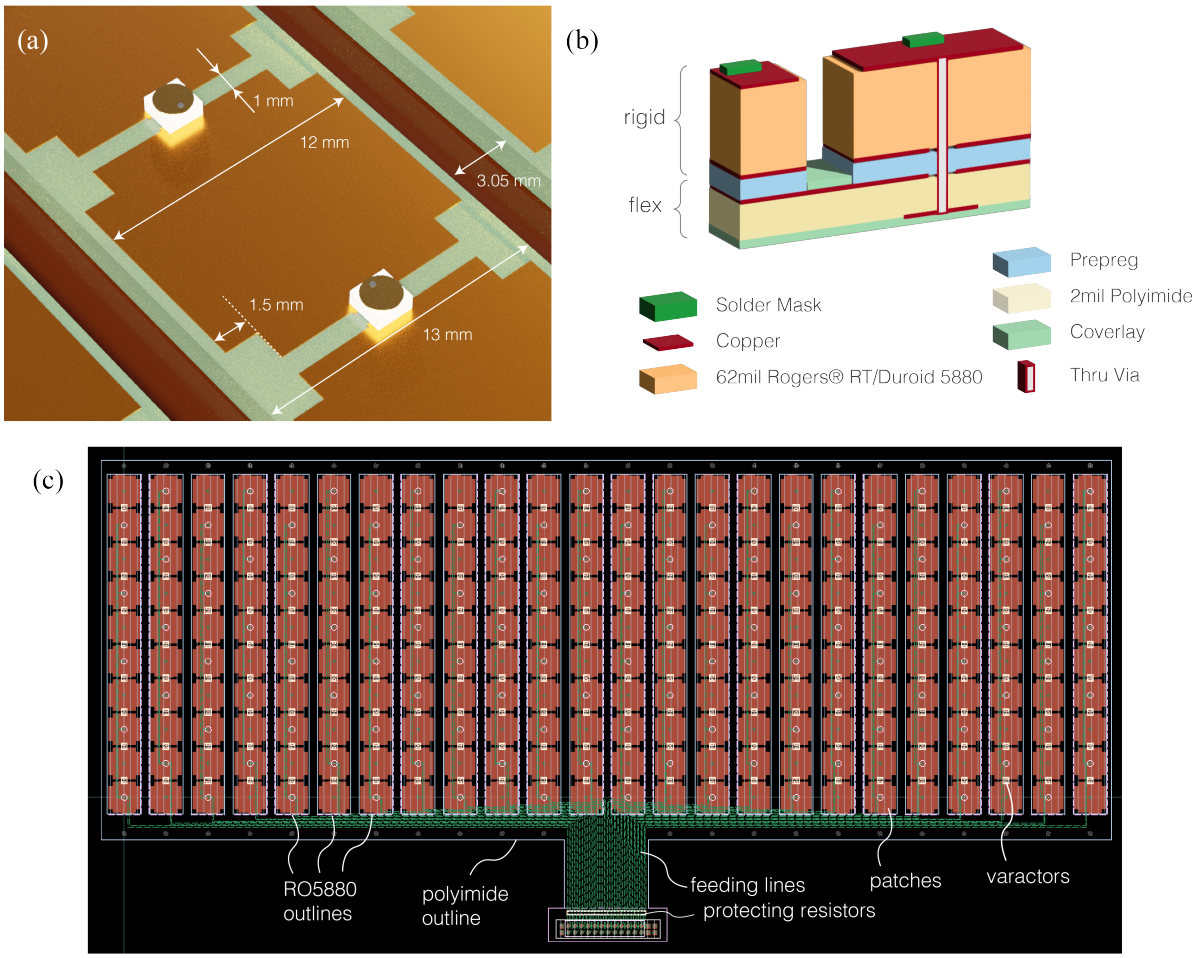

Figure 3.3. Rigid-flex PCB design. (a) Dimension of the unit geometry. (b) The stack-up plot of the rigid-flex board. Dimensions are not proportional. (c) A drawing of the fabricated board. Ground layers, prepreg layers, and coverlays are not shown.

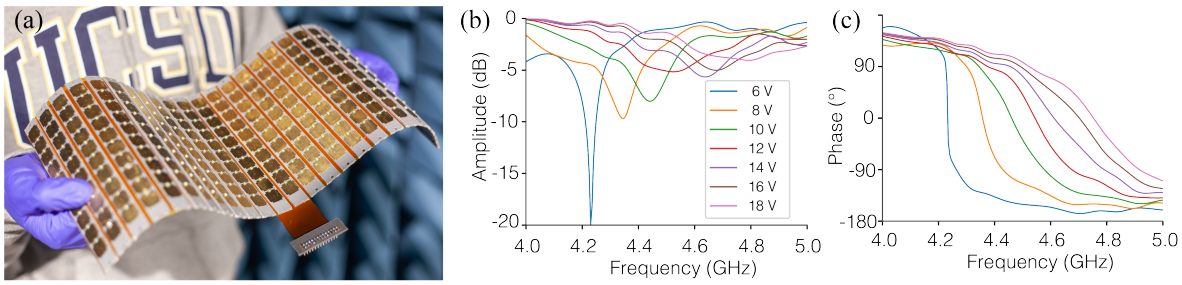

Figure 3.4. Static performance of the prototype (a) Photo of the prototype. (b) and (c) Measured reflection amplitude and phase response of the flat surface under various bias voltages.

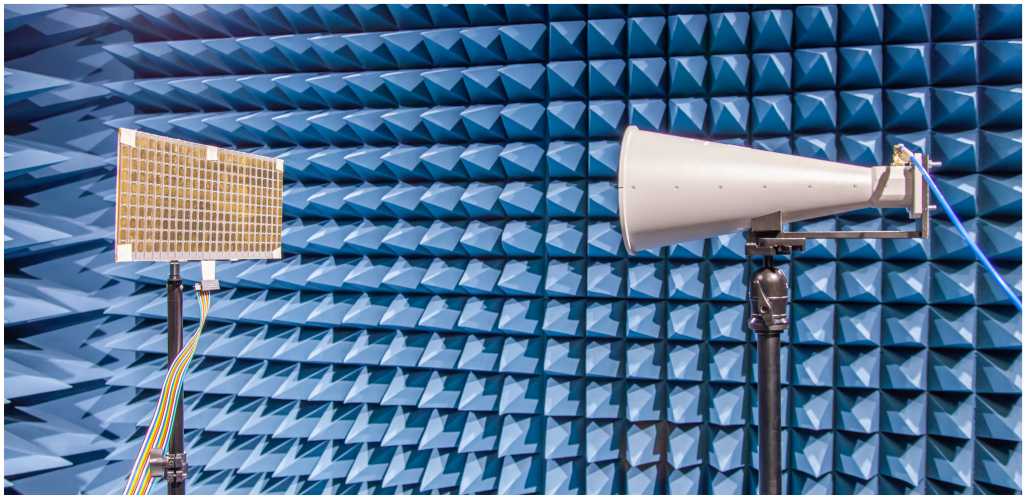

Figure 3.5. Static reflection spectrum measurement setup. The distance between the horn aperture and the patch is 55 cm.

Fig. 3.4 (b) and (c) depict the measured static reflection amplitude/phase response of a flat board. Note that this data is not used throughout the entire inverse design workflow but rather merely as an initial examination and verification of the surface reflection performance. The results show that the board covers a large reflection phase range within the 4.5 GHz - 4.7 GHz frequency band that is necessary to achieve maximum pattern tunability.

3.3 Analytical Challenges

Determining the relationship between the reflection pattern and a given bias combination $D(\theta) = f(\mathbf{V})$ can be challenging, due to the non-linear relationship between bias voltages, local reflection phases, and directivity to certain directions. Additionally, the multireflection effect in the concave regions invalidates any theoretical models that consider the surface as simply a reflective antenna array [45]. Other factors that complicate the problem are, but are not limited to, the varying orientation of each column, coupling effects between unit cells, tolerance of individual lumped components, or in some cases, the presentation of scattering objects near the device. In this scenario, a pure data-driven model is especially advantageous, since it can automatically take into account all these factors by using the real measured data. However, the

extremely large search space of the input variables renders the conventional interpolation or regression methods impractical and make the neural network method the best candidate.

This chapter is based on material in a manuscript to be published: Wen, E., Yang, X., & Sievenpiper, D. F. (2023). “Real-data-driven Real-time Reconfigurable Microwave Reflective Surface”. *arXiv preprint* arXiv:2305.11899. The dissertation author was the primary investigator and author of this paper.

Chapter 4

Real-data-driven Free-form Inverse Design Platform

In Chapter 2 we explain why it is impossible to use a simple FNN to find the underlying pattern-voltage mapping $\mathbf{V} = f^{-1}(\mathbf{D})$: the pitfall is that the design parameters and design goals usually have a multiple-to-one relationship. In this case again, similar or even identical reflection patterns may correspond to very different bias voltage combinations. When the network is trained with gradient descent methods, conflict gradients may arise from data with very similar input (i.e. pattern) but very different labels (i.e. bias voltages), preventing the parameters from converging. For a problem where the input contains less information than the output, as in this case, a generative type of neural network is necessary. Recent large generative models such as diffusion [49] and transformer [50] have shown extremely powerful capability in generating image and language content, even stepping towards artificial general intelligence (AGI) [51], yet for engineering problems on specific tasks, small-scale efficient models are still preferred, among which the tandem architecture has proven a very effective framework [21, 19, 52, 53, 54].

In the same manner, as in Chapter 2, a predictor need to be first trained to solve the analysis problem, in our case, the bias-pattern mapping; then another network, the designer, is trained to handle the synthesis procedure - determining the bias combination given a specific pattern. The designed bias can be fed into the predictor to produce an expected pattern $\tilde{D}$, and the performance can be evaluated by comparing the discrepancy between $\tilde{D}$ with the design

goal D, which serves as the loss function for training designer network. Importantly, the second step does not involve $D - \mathbf{V}$ mapping from any dataset, thus there is expected to be no gradient conflict. Essentially, instead of fitting the reverse function $f^{-1}(\cdot)$, the network aims at seeking any function that simply optimizes the design performance.

4.1 Sequential Tandem Neural Network

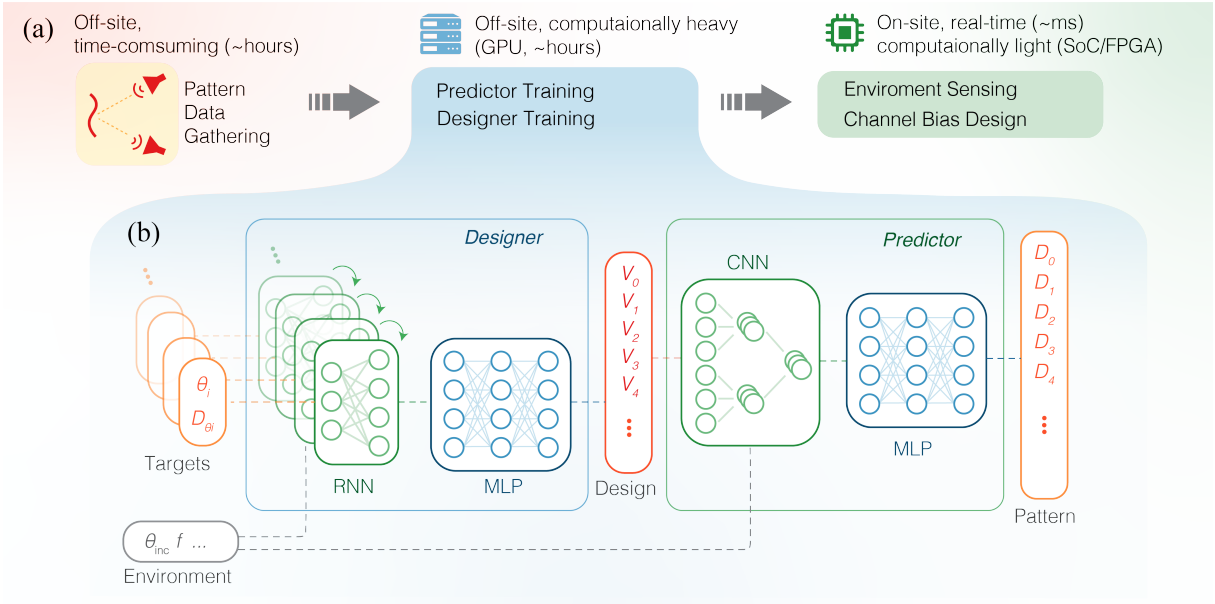

Figure 4.1. Real-data-driven real-time inverse design workflow. (a) The time-consuming and computationally-heavy part is done off-site in the first two steps - data gathering and network training. The pre-trained network can be then deployed to on-site controllers with very limited computing resources to realize the fast-response inverse design. (b) Proposed sequential tandem network architecture. The predictor is first trained with measured pattern data. Then the parameters of the predictor are fixed and random design target sequences are used to train the designer.

Notice in conventional tandem networks reported in Chapter 2, a design target with the exact form as the predictor's output is required, which largely limits its practicality. In many cases, free-form design goals are required, for example, one may want to specify several target directivity values in certain directions without having to construct the entire pattern. To enable this free-form input, we introduce the recurrent neural network (RNN) layer in the designer.

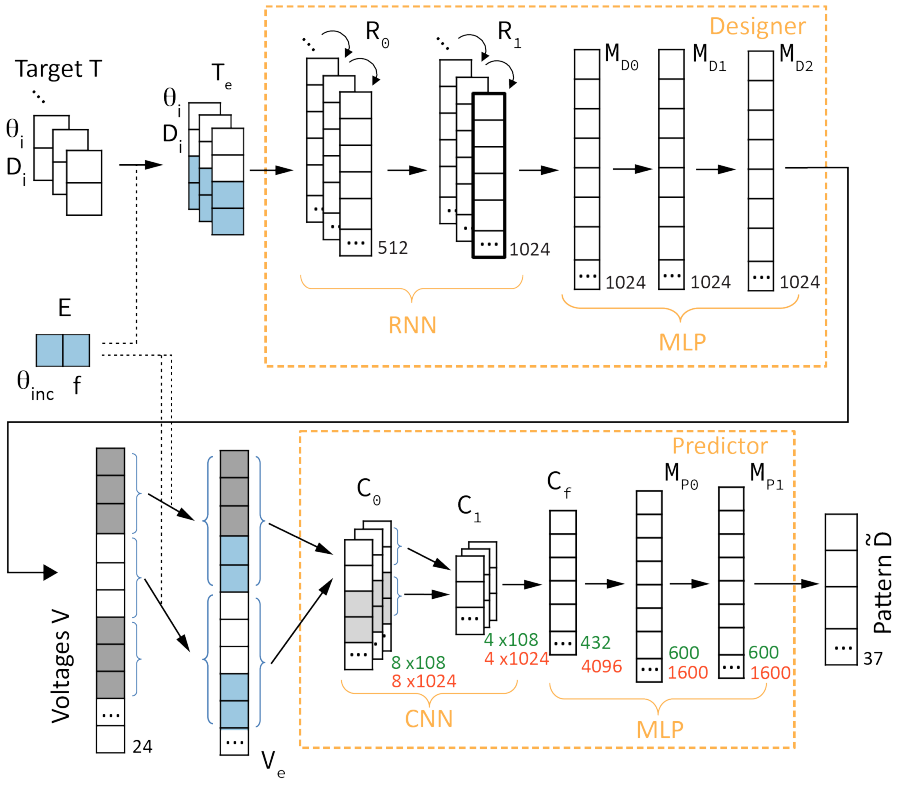

Figure 4.2. Proposed sequential tandem network architecture. Dimensions of tensors in the predictor for cases A,B and cases C,D are in green and red, respectively.

RNN is typically utilized to process temporal signals such as video and speech [55]: the recurrent layer updates itself from a current state as a sequential signal is fed in, resulting in a "memory effect" on all past inputs in the sequence. Here we can use a sequence of design goals as the input, which could be, for example, a sequence with a length of l_t, repeating n_t angle-directivity pairs (θ_i, D_i). Despite that there is no explicit temporal relationship between these design goals, we still expect the layers to "memorize" all those targets within the sequence. In this way, the network can respond to design goals with arbitrary dimensions as long as n_t is below a reasonable threshold to avoid vanishing gradient.

For the predictor, convolutional layers are employed, which is based on the physical knowledge of a linear phased array [56]: the same phase difference between neighboring units should have a similar effect in the far field no matter where they are located within the array, and therefore the parameters can be shared among all adjacent units. The convolutional layers

significantly reduce the number of parameters so that the predictor requires fewer data and suffers less from overfitting.

To allow the surface to operate with changing incident frequency and under different incident angles, this varying environmental information can be also cascaded to the input design goal vectors in the designer, and to the input bias-voltage vectors in the predictor.

Fig. 4.2 shows the detailed layer dimensions and connections. For the predictor:

$$\mathbf{C}_0 = g\left((\mathbf{W}_{C_0}, \mathbf{b}_{C_0}) \otimes_{(3,3)} \mathbf{V}_e\right), \tag{4.1}$$

$$\mathbf{C}_1 = g\left((\mathbf{W}_{C_1}, \mathbf{b}_{C_1}) \otimes_{(2,2)} \mathbf{C}_0\right), \tag{4.2}$$

$$\mathbf{M}_{P0} = g(\mathbf{W}_{M_{P0}} \mathbf{C}_f + \mathbf{b}_{M_{P0}}), \tag{4.3}$$

$$\mathbf{M}_{P1} = g(\mathbf{W}_{M_{P1}} \mathbf{M}_0 + \mathbf{b}_{M_{P1}}), \tag{4.4}$$

$$\tilde{\mathbf{D}} = g(\mathbf{W}_{\tilde{D}} \mathbf{M}_{P1} + \mathbf{b}_{\tilde{D}}), \tag{4.5}$$

where $\mathbf{V}_e$ is the bias voltage vector $\mathbf{V}$ cascaded with the environment vector $\mathbf{E}$, $\mathbf{C}_f$ is flattened form of $\mathbf{C}_1$ and the operator $\otimes_{(k,k)}$ is the 1-D convolution of length k with stride k, i.e. the tensor dimension shrinks by k. The activation function g is ReLU function for the predictor.

The structure of the designer can be expressed by:

$$\mathbf{R}_0^i = h\left(\mathbf{W}_{R_0} \mathbf{T}_e^i + \mathbf{U}_{R_0} \mathbf{R}_0^{i-1} + \mathbf{b}_{R_0}\right), i = 1...l_t, \tag{4.6}$$

$$\mathbf{R}_{0,out}^i = h\left(\mathbf{W}_{R_{0,out}} \mathbf{R}_0^i + \mathbf{b}_{R_{0,out}}\right), i = 1...l_t, \tag{4.7}$$

$$\mathbf{R}_1^i = h\left(\mathbf{W}_{R_1} \mathbf{R}_{0,out}^i + \mathbf{U}_{R_1} \mathbf{R}_1^{i-1} + \mathbf{b}_{R_1}\right), i = 1...l_t, \tag{4.8}$$

$$\mathbf{R}_{1,out}^i = h\left(\mathbf{W}_{R_{1,out}} \mathbf{R}_1^i + \mathbf{b}_{R_{1,out}}\right), i = 1...l_t, \tag{4.9}$$

$$\mathbf{M}_{D0} = g\left(\mathbf{W}_{M_{D0}} \mathbf{R}_{1,out}^{-1} + \mathbf{b}_{M_{D0}}\right), \tag{4.10}$$

$$\mathbf{M}_{Di} = g(\mathbf{W}_{M_{Di}} \mathbf{D}_{i-1} + \mathbf{b}_{M_{Di}}), i = 1, 2, \tag{4.11}$$

$$\mathbf{V} = g(\mathbf{W}_V \mathbf{M}_{D2} + \mathbf{b}_V), \tag{4.12}$$

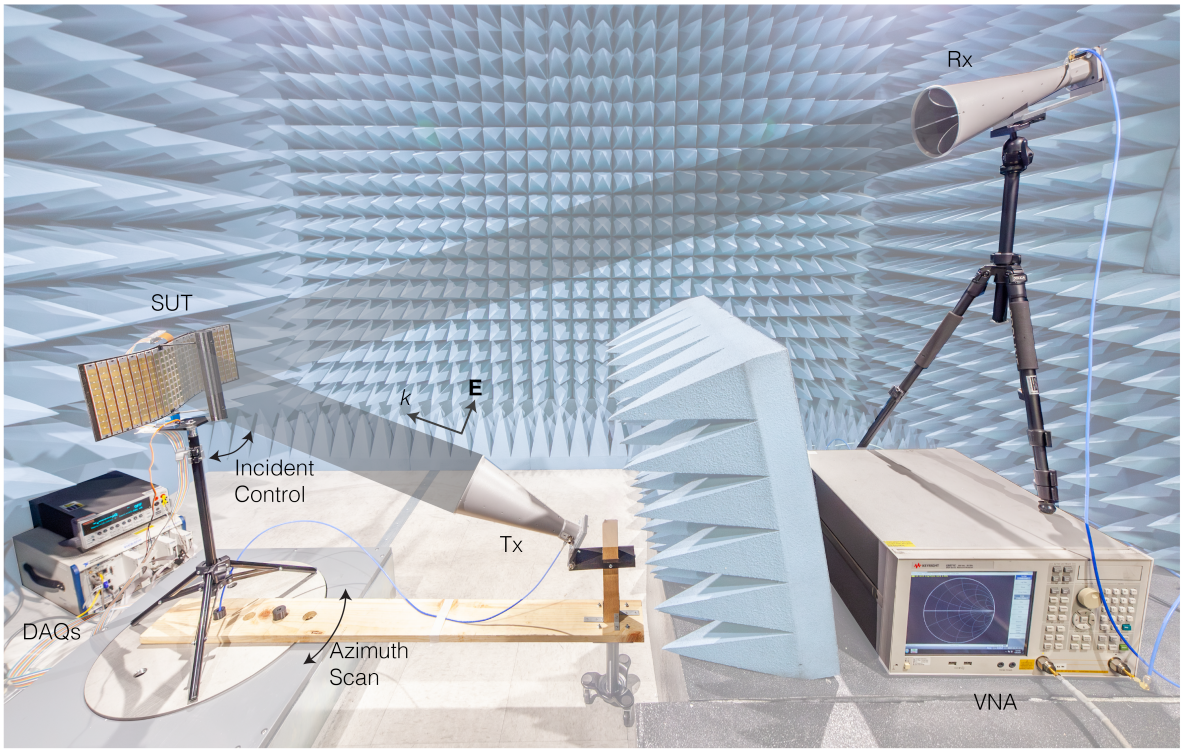

Figure 4.3. Experiment setup for pattern data gathering. A vertically polarized beam is excited with a horn antenna Tx, and its specular reflection from the surface under test (SUT) is received by another horn antenna Rx. The intensity and phase are recorded by a vector network analyzer (VNA). Both SUT and Tx antenna are attached to a servo motor to realize azimuth pattern scanning. Another servo motor is used to rotate the SUT to simulate incident angle changes. Revered bias-voltages of 24 channels are generated and applied to the board with data acquisition (DAQ) cards.

where $\mathbf{T}_e$ is the bias target vector $\mathbf{T}$ with each row cascaded with the environment vector $\mathbf{E}$, the -1 superscript denotes the last column of the matrix (final step), and the activation function for recurrent layers $h(\cdot)$ are tanh function and $g(\cdot)$ still ReLU.

4.2 Experiment

Here we demonstrate four scenarios in which the conformal metasurface may operate, with increasing complexity as follows:

A) the simplest case of a flat surface operating under a normally incident plane wave at a single frequency;

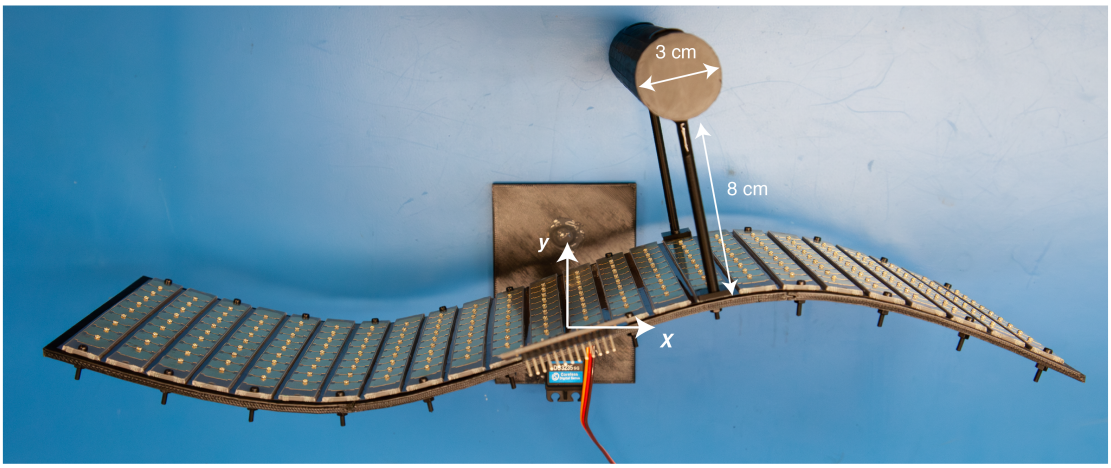

Figure 4.4. Photo of the curved surface being measured. The anchor points for the spline curve are [(-19.8,0),(-7.6,-3.0),(7.6,2.0),(19.8,0)] , units in cm. The length of the cylinder in the z direction is 20 cm.

B) a curved surface, under normal incidence, at a single frequency;

C) a curved surface working in a varying environment: with incident wave angles ranging from -30° to 30°, and a frequency band from 4.5GHz to 4.7GHz

D) a curved surface under a varying incident angle and frequency, with a plastic scattering object present in front of the surface, disturbing the reflective pattern.

For these four cases, the flexible surface is mounted on a 3-D printed backplane made of PLA plastic fastened with nylon screws. The curve surface for cases B,C and D is a spline with 4 anchor points. The 2-D scattering object for case D is a 3-D PLA cylinder. The dimensions are given in Fig. 4.4.

4.2.1 Pattern Data Gathering

The pattern data are gathered using a setup in an anechoic chamber shown in Fig. 4.3. The distance between the Tx and the SUT is 75 cm and the distance between the SUT and the Rx is 190 cm. The specular angle in the elevation plane is 25°.

The test bench is controlled by a single controller PXIe-8135 by National Instruments (NI) running Python 3.7. For the bias voltage supply, three 8-channel 16-bit DAQ cards NI

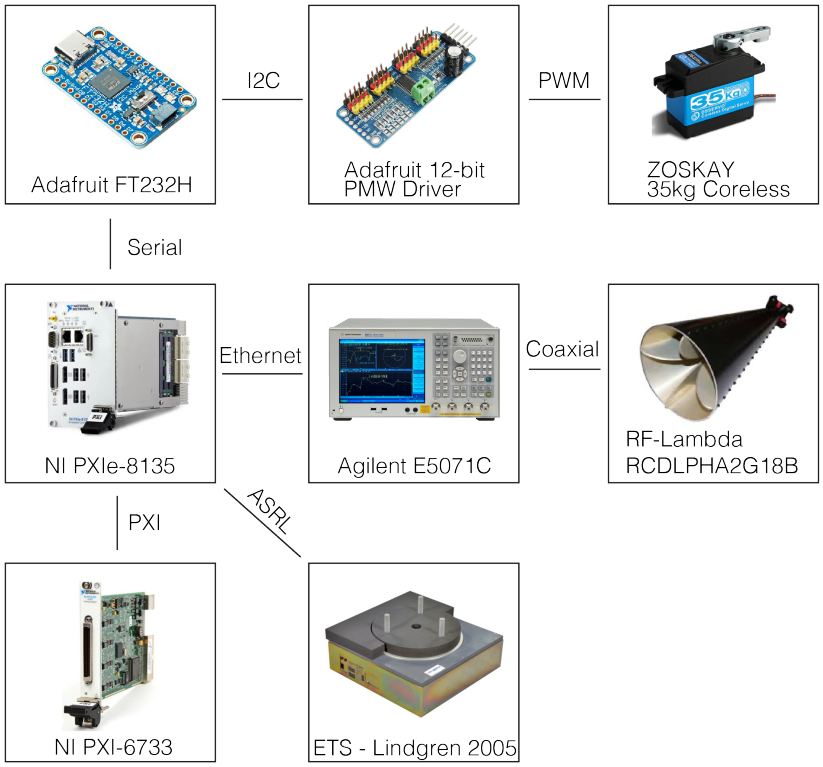

Figure 4.5. Interconnections between the devices.

PXI-6733 and a d.c. source Keithley 2410 are used. A servo motor ETS Lindgren 2005, attaching the SUT (with a tripod) and the Tx antenna (with a wooden arm), is employed to realize azimuth pattern scanning. Another small servo motor ZOSKAY DS3235SG is used to rotate the SUT to simulate incident angle changes. This motor is driven by an Adafruit FT232H breakout and an Adafruit 12-bit PMW driver. The interconnections between all equipment are shown in Fig. 4.5

The transmission between the Tx and the Rx along the azimuth plane, from 3 GHz to 5.5 GHz, is measured for 3 cases:

- blank case $S_{21,b}$, where the surface is not mounted on the supporting structures. This includes the ambient noise, scattering from the supporting structures, and direct talk between Tx and Rx;
- SUT case $S_{21,s}$. This is the direct reflection along with all noises mentioned above.

- aluminum board $S_{21,PEC}$. This is for calculating the directivity of the board with the transmission intensity.

The loop of sampling, from outer to inner is:

1) θ scanning, by azimuth motor rotation.

2) θ_{inc} scanning, by incident motor rotation (for case C and case D).

3) Different bias voltages, by DAQs.

4) Frequency, done by VNA itself.

The wait time for step 1 and step 2 are set to 10 s and 4 s, respectively, to reduce the impact of mechanical vibration. The wait time between different sets of bias is 20 ms. To reduce the unwanted noise or environmental scatterings, all received signal is transferred to the time domain with inverse Fourier transformation to apply a 5 ns time gate, then transferred back to the frequency domain, i.e.

$$S_{21,gated}(f) = \mathscr{F}\left[\mathscr{F}^{-1}\left(S_{21}(f)\right)W(t)\right], \tag{4.13}$$

where the rectangular window in the time domain:

$$W(t) = \begin{cases} 1, & t_{\text{start}} \leq t \leq t_{\text{end}} \\ 0, & \text{otherwise} \end{cases} \tag{4.14}$$

The directivity is thus calculated by:

$$D(\theta) = \left|\frac{S_{21,s}(\theta) - S_{21,b}(\theta)}{S_{21,PEC}(90^\circ) - S_{21,b}(90^\circ)}\right| D_{PEC}(90^\circ), \tag{4.15}$$

where $D_{PEC}(90^\circ) = 8.85$ is the board-side directivity of a PEC aperture.

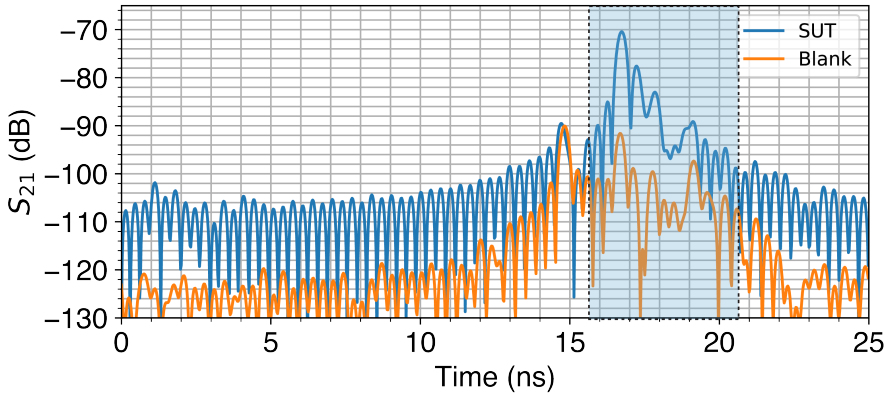

Figure 4.6. Surface response in the time domain. This example is from one sample in case A at $\theta = 90°$. The direct reflection from the SUT can be observed spanning from around 16 ns to 20 ns. The direct talk between the Tx and the Rx is likely the peak appears around 14.5 ns. Highlighted regions are the 5 ns window being applied.

The RCS can be also derived from the directivity with:

$$\sigma(\theta) = A_{PEC}\left(\frac{D(\theta)}{D_{PEC}(90°)}\right)^2, \tag{4.16}$$

where A_{PEC} is the physical aperture of the PEC plate.

The patterns are collected in 5-degree resolution from 0° to 180°, forming a 37-D vector **D**. For cases with a constant environment (cases A and B), 20,000 samples are collected with random combinations of bias voltages ranging from 0 V to 18 V on each channel. For cases C and D, 4,000 random bias samples are collected for each incident angle, with 13 incident angles; 5 random frequencies within the band of interest are sampled for each incident/bias setup, making a total of 260,000 samples. As the response time of the varactor diodes is relatively fast, on the order of nanoseconds, the data collecting speed is mostly limited by the response time of the control/measurement instrument being used, generally on the order of milliseconds. To obtain a stable result for our setup, 20 ms wait time is used in between samples, so the total data collection time is on the order of several hours, details listed in Table 4.4, which is much faster than any full-wave simulation methods can achieve.

4.2.2 Predictor Training

The neural network is implemented using TensorFlow 2.6.0 under Python 3.9.5 and is trained on the UCSD Data Science/Machine Learning Platform (DSMLP) using four Intel XeonGold 6130 CPU @2.1 GHz cores and one NVIDIA GeForce GTX 1080 Ti GPU.

The gathered pattern data is split 80/20 as the training/test set to train and evaluate the predictor. All data, including bias voltages, directivity intensity, incident angle, frequency, and target directions are all normalized to [0,1] in the training process, similar to the process in Chapter 2.2.1. To deal with overfit, L2 regularization is used[57]:

$$L = MSE + \lambda_{L2} \sum_{\text{lay} \in \text{All layers}} \| \mathbf{W}_{\text{lay}} \|_2^2 \tag{4.17}$$

The parameter is optimized with Adam optimizer[26]. The batch size for training is set to 50. Table 4.1 listed the detailed hyperparameter setup used for four cases.

Table 4.1. Predictor training hyperparameters.

Case	Case A	Case B	Case C	Case D
$L2$ weight λ_{L2}	2e-7	5e-7	5e-8	5e-8
Learning rate	1e-3	1e-3	3e-4	3e-4
Epochs	500	500	50	50

The loss function vs. epochs is plotted in Fig. 4.7. The parameters of the epoch with the lowest validation loss are being used. It can be observed that no significant overfit is presented with the regularization.

Fig. 4.8 shows the performance of the trained predictor network. The prediction matches extremely well with measured data in the test set for all four cases, with an error almost close to the noise level, providing great accuracy for the designer training.

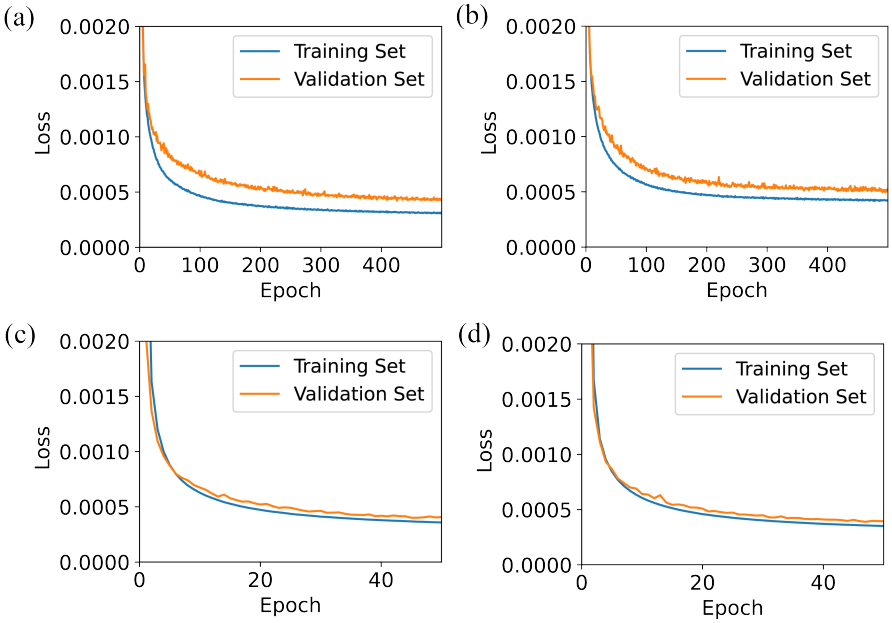

Figure 4.7. Training process of the predictor. (a)-(d) are losses for case A to case D, respectively.

4.2.3 Designer Training

Training the designer network is a non-supervised learning process since the label for any design target is itself: the loss function is defined by the masked means square error (MSE) on target directions:

$$L(\mathbf{T},\tilde{\mathbf{D}}) = \frac{1}{n_t}\sum_{i=1}^{n_t}(D_i - \tilde{D}(\theta_i))^2. \tag{4.18}$$

In this demonstration, we randomly generate sequences with up to five targets. In practice, we find each goal needs to be repeated three to four times in order for the network to fully memorize it, thus for the result to converge, the sequence length l_t is chosen to be 20. Considering the energy distributing effect for multiple targets, the directivity ranges for different target numbers n_t is $[0, D_{PEC}/\sqrt{n_t}]$, where still $D_{PEC} = 8.85$, shown as Table 4.2. For cases A and B, 100,000 samples are generated, and for cases C and D, 260,000 samples.

The data is again split 80/20 for the training/test set. The parameter from the predictor is inherited and fixed during the training of the designer. For the designer, no regularization is used

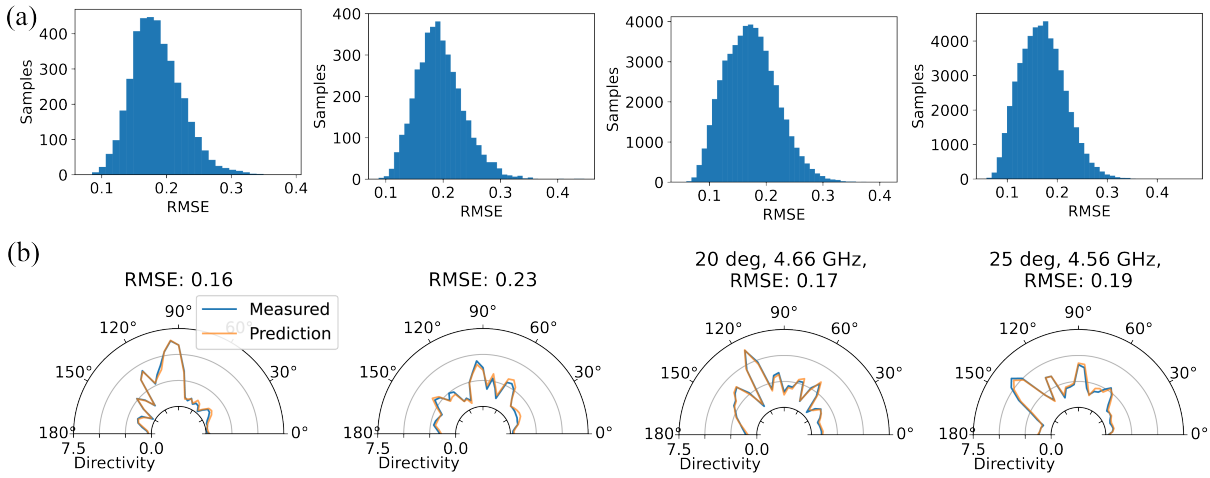

Figure 4.8. Performance of the predictor for four cases. (a) RMSE distribution from case A to case D, in the test set. (b) Visual examples with typical error level.

Table 4.2. Directivity range for different number of targets.

# of targets n_t	1	2	3	4	5
Max directivity	8.85	6.25	5.1	4.43	3.96

since it is always preferred to generate more data to address overfitting and improve generality. The training process of the four cases is plotted in Fig. 4.9.

Table 4.3. Designer training hyperparameters.

Case	Case A	Case B	Case C	Case D
Learning rate	4e-5	6e-5	5e-5	6e-5
Epochs	150	150	120	200

The performance is shown in Fig. 4.10 and Fig. 4.12-4.15. The network performs very well for fewer numbers of targets and still decently well for 3 or more targets, obtaining an average RMSE below 1 for most cases.

It is worth noticing that this performance is evaluated on random targets that is not necessarily physically feasible, such as the existence of a peak and a null in proximity, or strong beams at the end-fire direction. By using a large training set, the physical limit of the surface capability is approached.

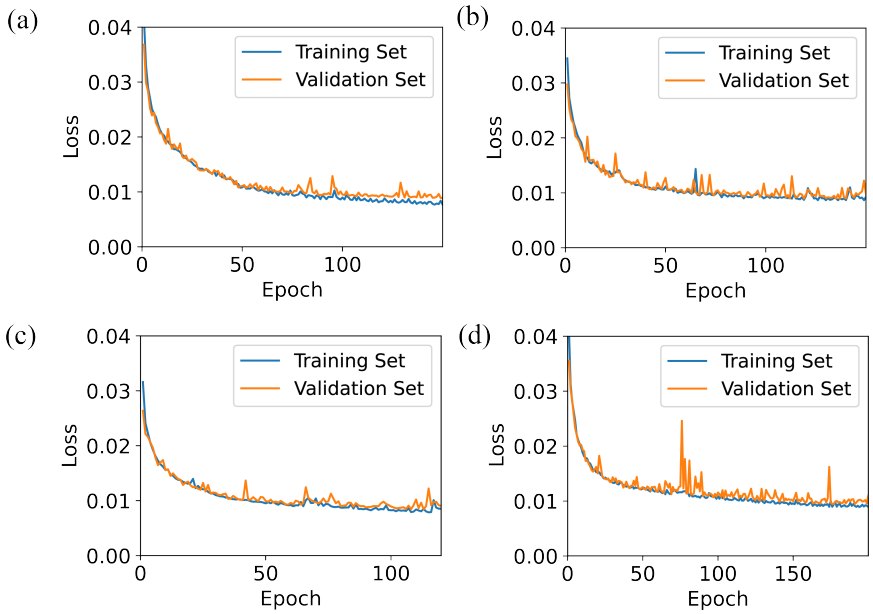

Figure 4.9. Training progress of the designer. (a)-(d) are losses for case A to case D, respectively.

4.2.4 Network Prediction

Compared with the training process, the prediction of the network consumes minimal computing resources. Therefore, this trained network can be deployed on modest micro-controllers with very limited computing power. In Table 4.4 we demonstrate this on a cheap commercially available SoC controller Raspberry Pi 4 with Quad-core Cortex-A72 @1.8GHz and 8GB RAM, running Python 3.7.3 and TensorFlow lite 2.3.0. The speed depends on various factors such as the machine learning platform, batch sizes of input, etc, but generally, the responding time for both designing (with the designer) and evaluation (with the designer + predictor) are on the order of milliseconds per sequence, which can be considered as real-time. The time listed in Table is the average on a dataset with 6000 samples for cases A and B or 6500 samples for cases C and D, processing with a batch size of 500.

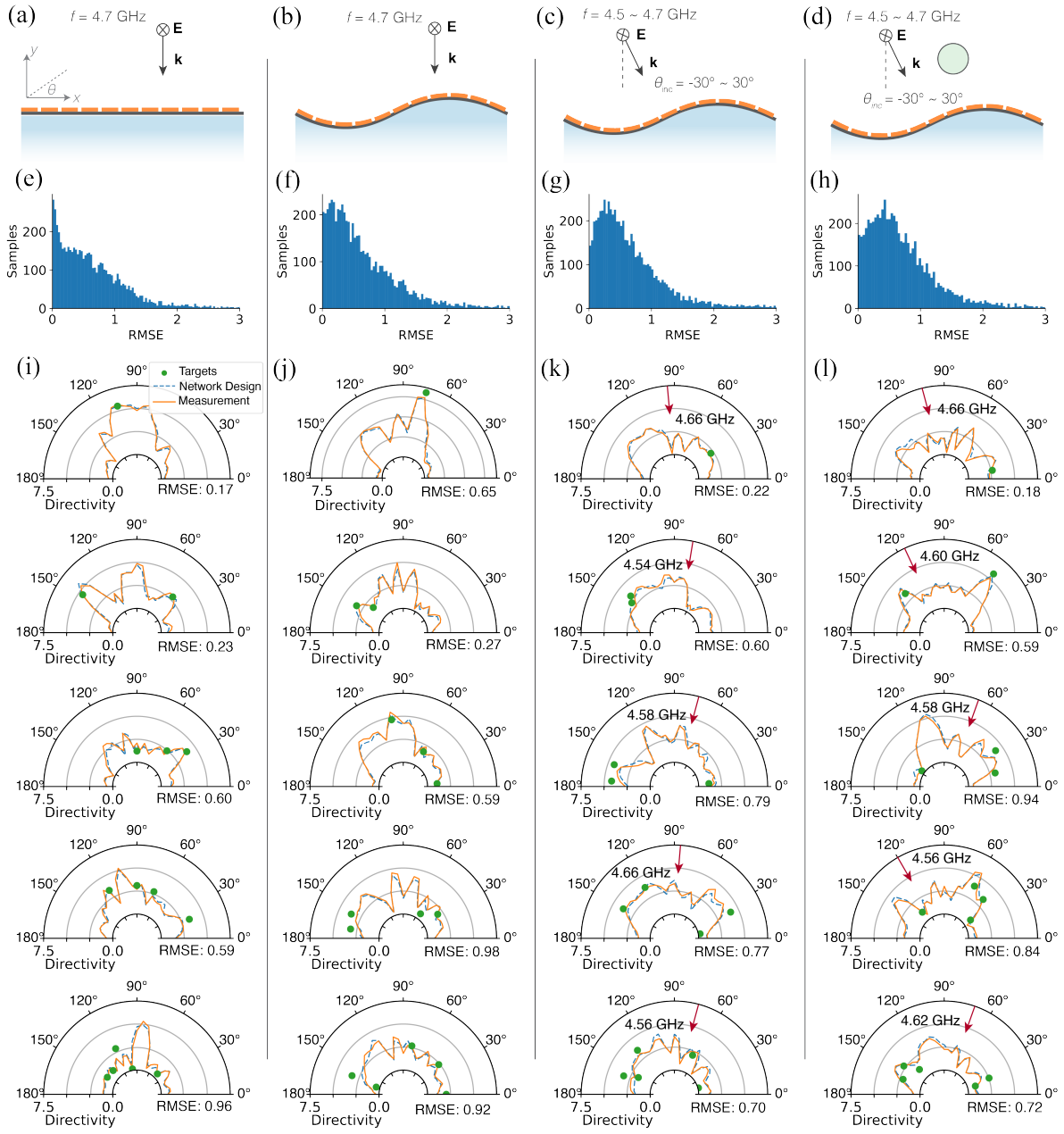

Figure 4.10. Performance of the conformal surfaces inverse design with increasing complexity. (a) and (b) Flat and curved surface, under normal incidence, at constant frequency (c) curved surface, under varying incident angle and frequency, and (d) with scattering object presented. (e)-(h) The root mean square error (RMSE) distribution on designs in the test set. (i)-(l) Visual examples of pattern design with 1 to 5 design targets, with typical error value in its category, excerpted from Fig. 4.12-4.15 where more random-selected visual examples and error distribution in different categories are given. Red arrows in case C and case D indicate the incident direction.

Table 4.4. Time consumption summary of tasks in the workflow.

		Case A	Case B	Case C	Case D
Data gathering		5h26min	5h27min	14h37min	14h32min
Network training	Predictor	5min	18min	27min	26min
	Designer	46min	1h1min	2h5min	3h44min
Network prediction	Inv. design	5.3ms	5.5ms	7.7ms	7.7ms
	Inv. + evaluation	5.9ms	6.0ms	10.3ms	9.8ms

4.3 Discussion

Being able to specify target directivities in multiple directions makes the surface suitable for numerous applications. One simple example is to reduce the back-scatter of a surface for manipulating RCS in a dynamic environment - by specifying a null at the direction of the incoming wave, the surface can be constantly optimized to reduce the mono-static RCS to a single station, as shown in Fig. 4.11 (a). A more intriguing task is to instruct the surface to cancel out the scattering from an object in front of the surface, keeping it from being detected in certain directions, as in Fig. 4.11 (b). It can also be utilized for intelligent communication applications, performing tasks ranging from creating a single pencil beam, as in Fig. 4.11 (c), to arranging multiple peaks and nulls, completely redistributing the incoming energy as in Fig. 4.11 (d).

The design presented here works over a relatively narrow bandwidth but this is not limited by the model itself: design parameters like substrate thickness can be increased to increase the bandwidth. The efficiency can be improved as the semiconductor technology for varactors develops. Modifications can also be made to the network input to achieve even higher versatility, for example by expending the target pairs with operators to form tokens like $(\theta_i, D_i, \leq)$ representing design goal $D(\theta_i) \leq D_i$. Other types of tasks, such as polarization conversion or holographic pattern, can also be achieved with proper types of surface coating design, by including polarization or near-field data in the input and output of the network.

Considering the versatility of the proposed workflow, we believe this work paves the way for the next generation of tunable EM devices working under extremely complex and

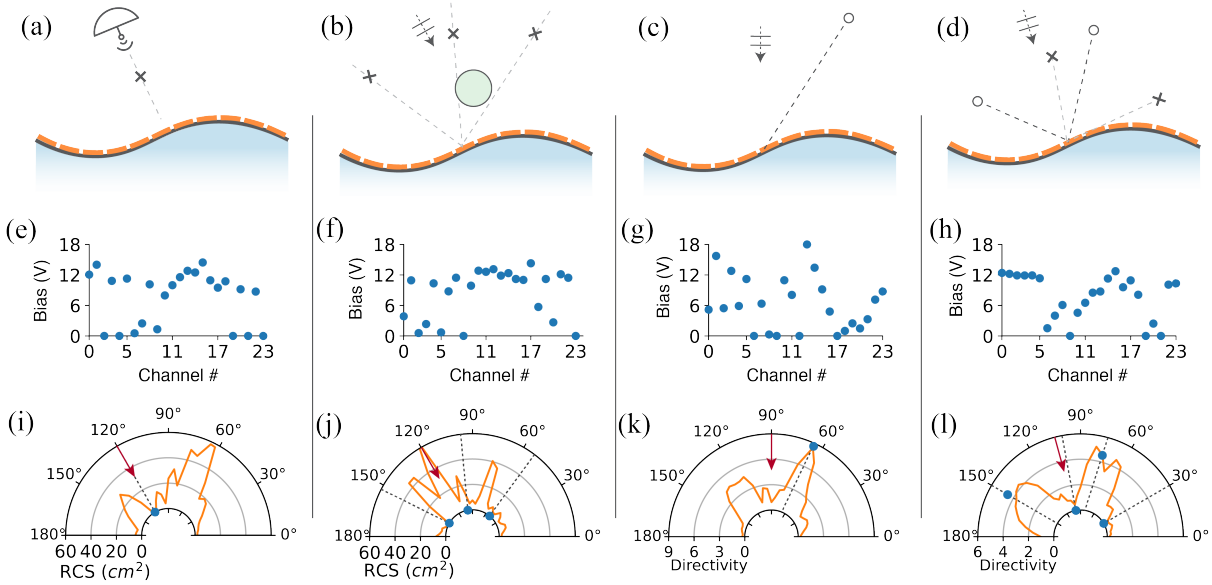

Figure 4.11. Exemplary applications of the free-form design capability. (a) Creating minimal back-scatter for a 30° incident. (b) Creating nulls in 50°, 100° and 150° directions to stealth a scatterer in front of the surface. (c) Creating a pencil beam at 65° direction under a normal incident. (d) Creating beams at 75° and 150° direction and nulls at 30° and 100°, under 15° incident. The operating frequency is 4.6 GHz for these four examples. (e)-(h) Bias voltages on channels 0-23, designed by the designer. (i)-(j) The corresponding patterns.

dynamic environments. In addition, the proposed sequential tandem architecture can potentially be accommodated to any real-time inverse design problem in different science/engineering fields.

4.4 Extended Data Figures

In this section, we include more error analysis and visual examples of the Designer network performance, readers may find the RMSE distribution for the input sequences with different target numbers and random-selected visual examples for case A to case D in Fig. 4.12 -4.15.

In order to ensure generality and reproductivity, we use random seeds 1,2,3,4 when sampling for cases A,B,C and D, respectively.

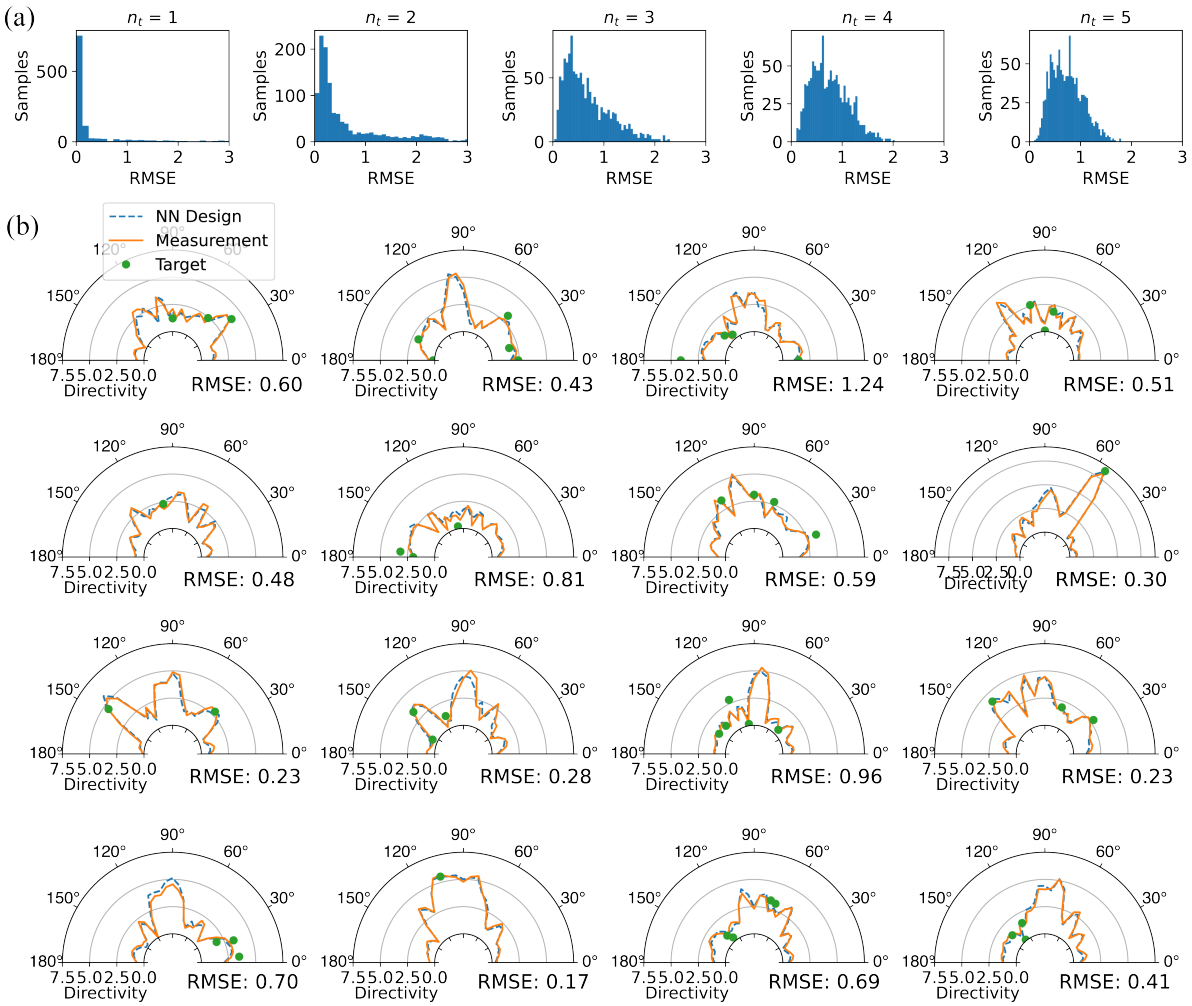

Figure 4.12. Performance of the designer for case A. (a) RMSE distribution on sequence with different target number n_t. (b) Random-selected visual examples, sampling from 6,000 test samples with random seed = 1.

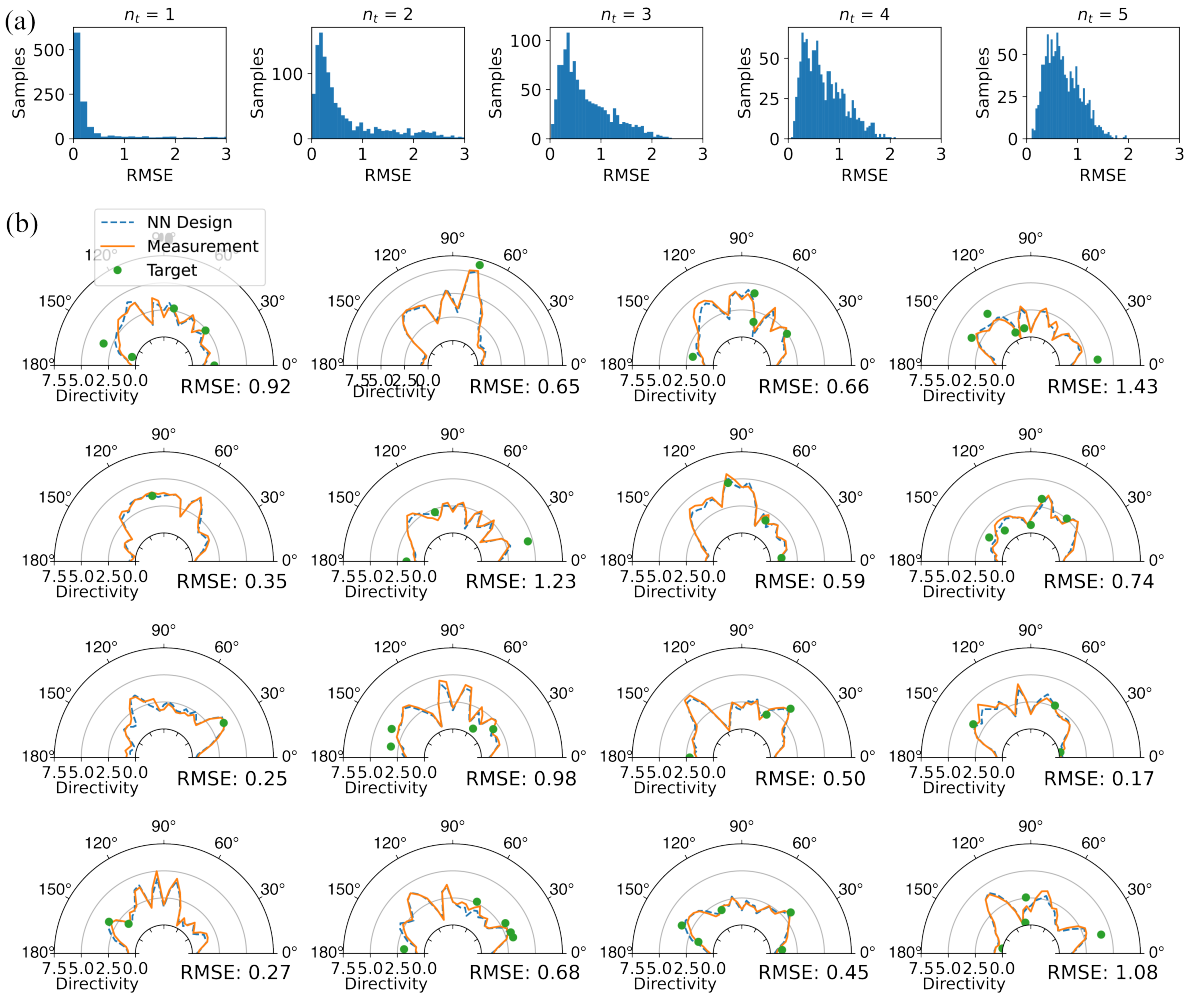

Figure 4.13. Performance of the designer for case B. (a) RMSE distribution on sequence with different target number n_t. (b) Random-selected visual examples, sampling from 6,000 test samples with random seed = 2.

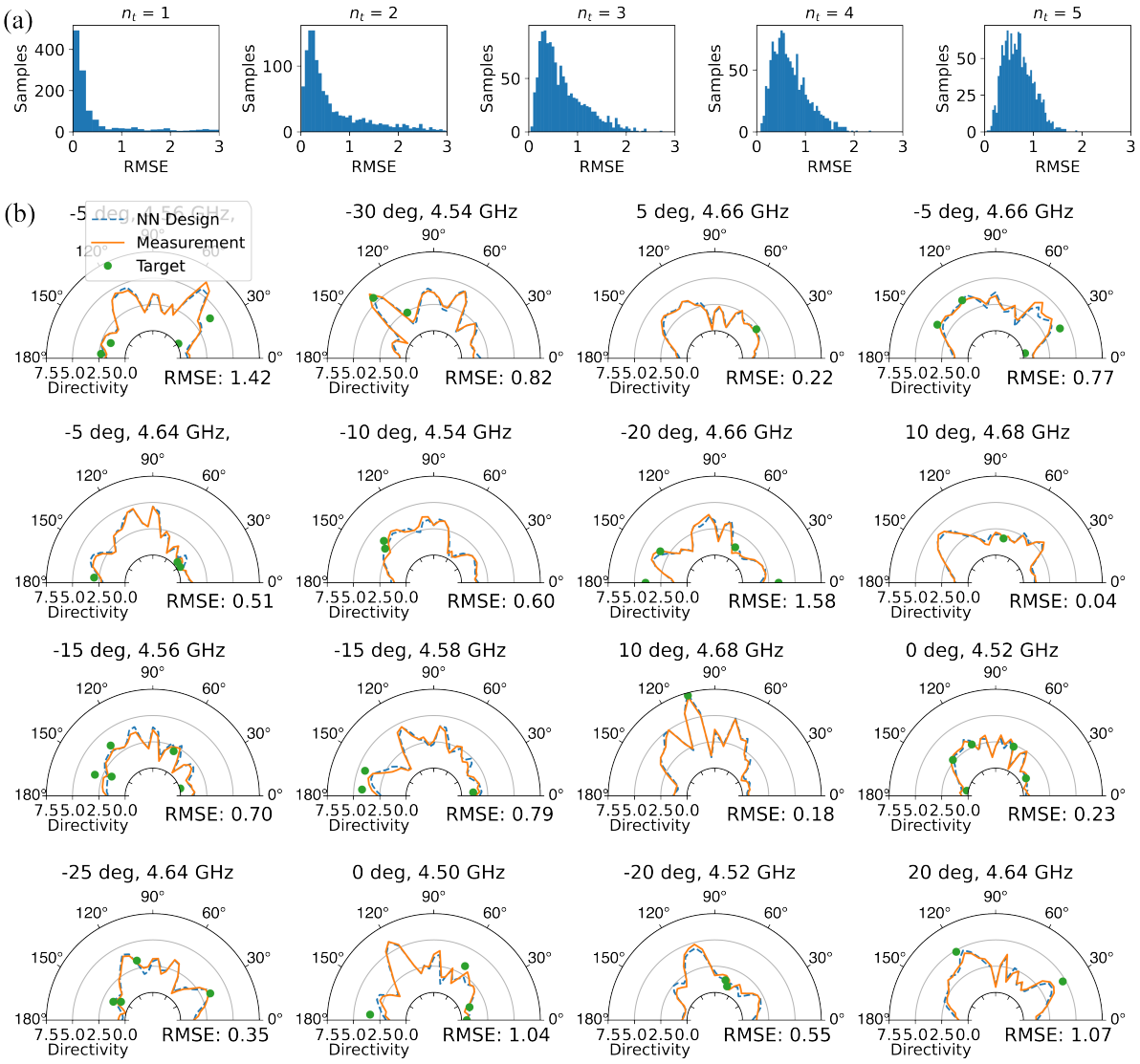

Figure 4.14. Performance of the designer for case C. (a) RMSE distribution on sequence with different target number n_t. (b) Random-selected visual examples, sampling from 6,500 test samples with random seed = 3.

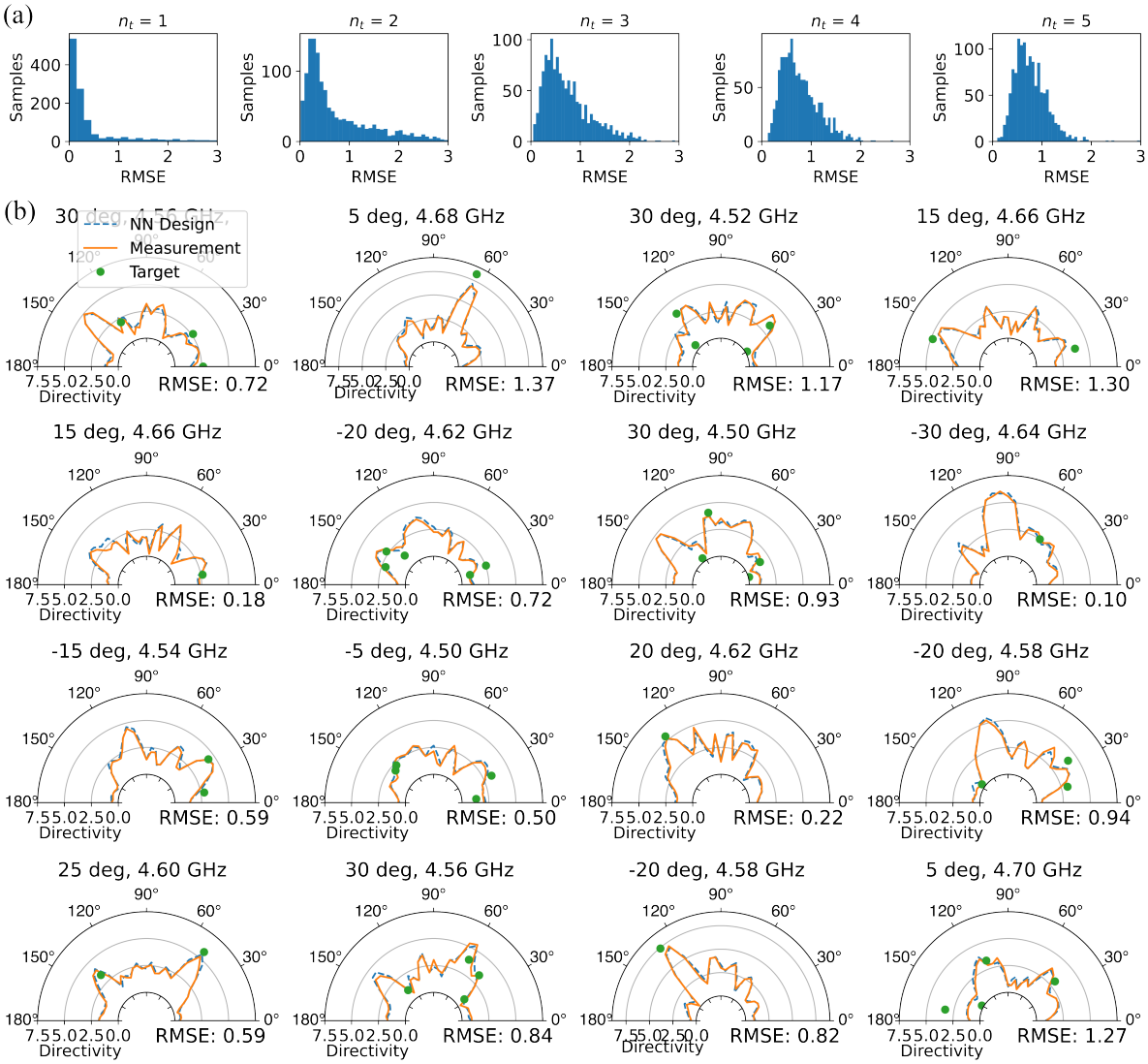

Figure 4.15. Performance of the designer for case D. (a) RMSE distribution on sequence with different target number n_t. (b) Random-selected visual examples, sampling from 6,500 test samples with random seed = 4.

This chapter is based on material in a manuscript to be published: Wen, E., Yang, X., & Sievenpiper, D. F. (2023). “Real-data-driven Real-time Reconfigurable Microwave Reflective Surface”. *arXiv preprint* arXiv:2305.11899. The dissertation author was the primary investigator and author of this paper.

Chapter 5

Conclusion and Possibilities

In the previous chapters, we demonstrate a universal real-data-driven approach of using sequential tandem neural network for the dynamic design of reconfigurable EM metastructure. The workflow, in summary, is as follows:

1. Preparing the neural network:
    - Determining the design variable and observing values;
    - Determining the environment variables;
    - Choosing an appropriate architecture for the predictor based on physical knowledge.
2. Setup the measurement and gathering data needed for training the predictor.
3. Training the predictor and then the designer, adjusting the scale based on the performance requirement and the computing resource limitation.
4. Deploy this pre-trained design network for on-site tasks on the microcontroller.

It can be noticed that this workflow does not limit itself to any particular type of system, thus by involving different types of data in step 1, possibilities are endless. In this chapter, we give several examples of complicated EM problem that is significant but very challenging to handle with conventional optimization methods.

5.1 Controlling Field Distribution in Cavity

One possible accommodation of the model is to manipulate field distribution, such as creating peaks and nulls, at target locations in a closed cavity: the multi-channel phase-tunable metasurface can be attached on one or several sides of cavity walls, disturbing the field distribution.

Considering the chaotic manner of the field distribution in the cavity and the field disturbance effect of the metasurface, it would be extremely difficult for any conventional method to analyze/synthesize the system in a non-statistical way. However, this can be done with the same architecture in Chapter 4 by simply changing the output vector to the field intensity: The intensity data in a cavity can be collected with experiment at locations of interest, under random bias voltage combinations and be used to train a sequential tandem neural network, aiming at inverse design the bias voltages given target field intensity at specified locations.

5.1.1 Disturbing Field in a Cavity

As a proof of concept, imagine a 2-D electrically large cavity shown in Fig. 5.1 with a point source inside, exciting TMz mode (constant E field in the z-direction with vertical polarization). As a rough estimation, we can calculate the field distribution in this quad-literal cavity using the ray launching method. Each ray emitted by the excitation can be represented by a vector $\mathbf{R}_i$, a starting point (x_0^i, y_0^i) and its intensity and phase at its starting point A_i, ϕ_i. As the ray bounces on the boundary of the cavity, the normal vector noted as $\mathbf{n}$, at (x_b, y_b), a reflection ray is generated:

$$\mathbf{R}_i' = \mathbf{R}_i - 2\mathbf{n}(\mathbf{R}_i \cdot \mathbf{n}), \tag{5.1}$$

$$(x_0^{i\,\prime}, y_0^{i\,\prime}) = (x_b, y_b), \tag{5.2}$$

$$A_i' = \alpha A_i, \tag{5.3}$$

$$\phi_i' = \phi_i + kl_i + \beta, \tag{5.4}$$

where α is the reflection rate, β is the reflection phase (π for a PEC wall that is not covered with metasurface), and $l_i = \left|(x_0^i, y_0^i) - (x_b, y_b)\right|$.

To calculate the field of a given point, consider a receiving sphere with a radius of r_r around the position of interest, the intensity is the summation of all rays passing through it:

$$I(x_{obs}, y_{obs}) = \sum_{d_i^{obs} <= r_r} A_i e^{jk(\phi_i + l_i^{obs})}, \tag{5.5}$$

where d_i^{obs} is the distance between the observation point and the ray, and l_i^{obs} is the distance between the observation point and the starting point of the ray.

In Fig. 5.1 we calculate the field distribution inside the cavity. In this configuration, we generate 2880 initial rays from the source (0.125° resolution). The reflection rate of the wall and metasurface is set to 0.7. The threshold of intensity for generating the reflection ray is set to 0.1. For field calculation, the radius of the receiving sphere is 0.07λ. We compare the field with or without a 24-channel metasurface attached to one wall, with a random phase coding in this case. It can be observed that the field is strongly disturbed especially near the metasurface.

5.1.2 Possible Setup for Experiment

The metasurface design in Chapter 3 can be easily adjusted to be attached to the 2-D cavity side wall. Since the top and bottom PEC wall of the cavity resembles a pair of periodic boundaries for TMz mode. We implement one row of units as shown in Fig. 5.2, which is realized with a double-layer PCB. The top layer is made with a center patch fed with bias voltages, and the patches on the sides are connected to the ground with a row of vias. The width of the units (height of the cavity) is 2.4 cm, which is small enough to prevent higher-order TM mode for frequencies below 4 GHz. Again, by changing the of each unit, the reflection phase can be freely tuned near its resonant frequency range. The bottom layer of the PCB consists of the feeding network to provide the reverse bias voltage for each patch.

Cavities can be built and used to collect the field destruction data needed for the data-

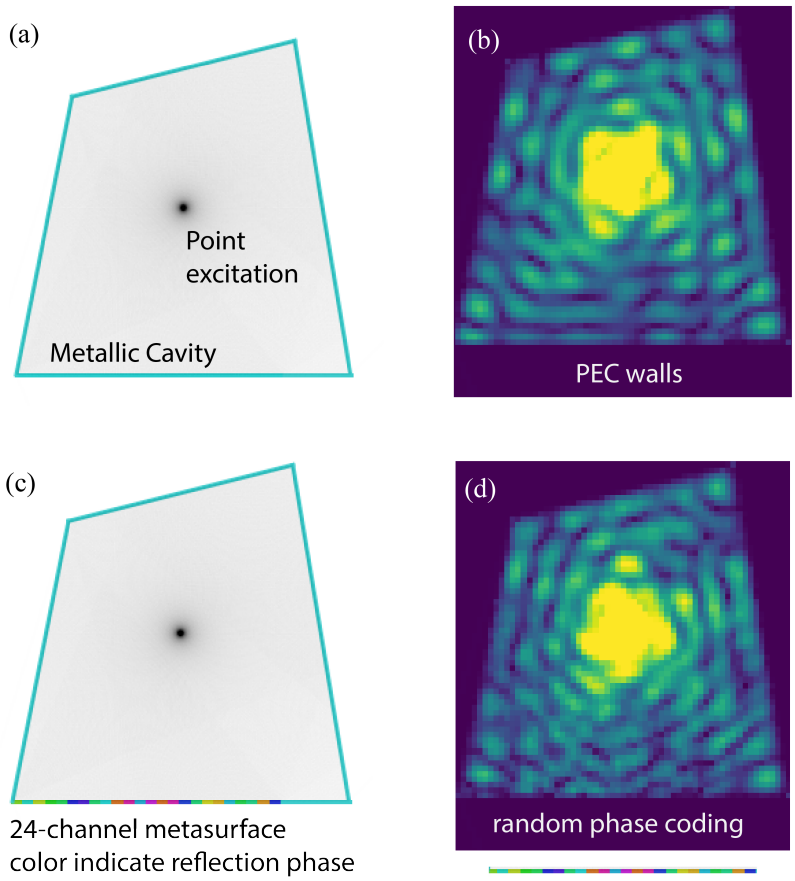

Figure 5.1. Field distribution in a metallic cavity. Excitation is a TMz mode point source in the center of the cavity. (a) and (b) A metallic cavity as a baseline. (c) and (d) a cavity with a phase-tunable metasurface attached on one wall. The phase of each element is represented by different colors.

driven model. The above metasurface design can be attached to one or two of the walls. Fig.5.3 shows two cavities setup with different electrical sizes, consequently different degrees of chaotic character, that can be used for this study. Vertical probes can be used on the top wall of the cavity to excite TMz mode inside the cavity or to observe the field intensity at the locations of interest.

5.2 Including Sensitivity in Inverse Design

When a neural network is used to design the geometry of EM structures, the sensitivity of the performance with respect to design parameters (dimensions) can be high. This may create problems especially when the design involves very fine structures like small pixels, thin arms, or

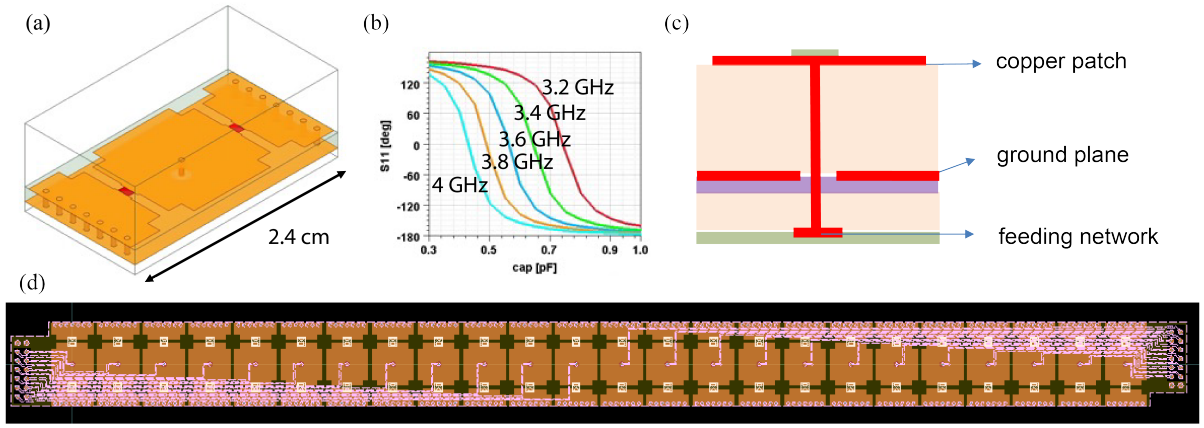

Figure 5.2. TM mode metasurface design (a) Unit-cell design. (b) Simulated reflection phase response. (c) Stack-up of the prototype. (d) Circuit board drawing. Patches are plotted in orange, and Feeding lines are plotted in pink.

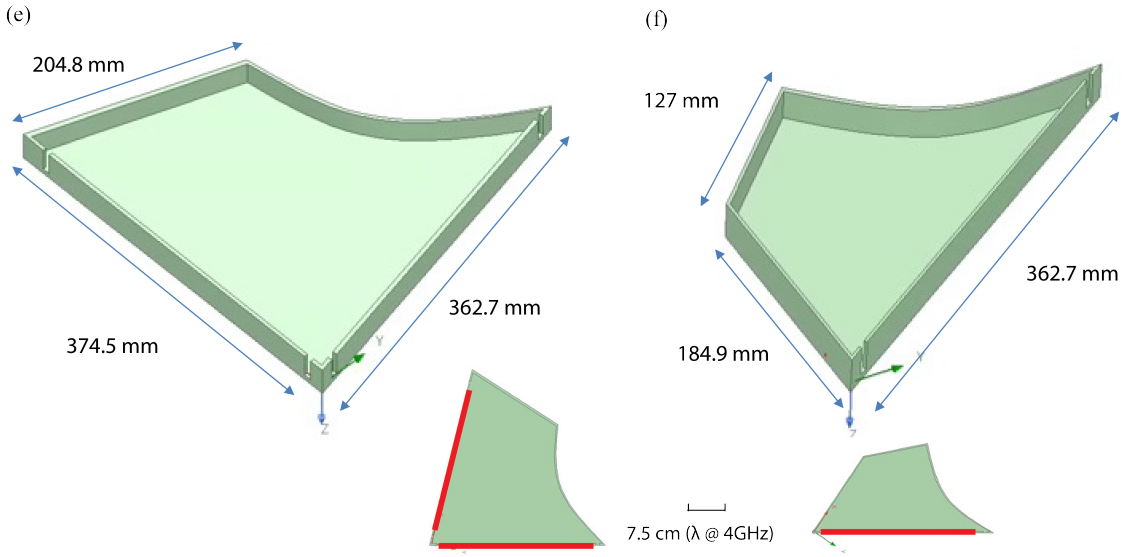

Figure 5.3. Exemplary cavities to be tested. Red lines indicate where the metasurface can be attached. (a) A cavity with 2 walls attaching the metasurface. (c) A relatively small cavity with one wall attaching the metasurface.

slots. It is obvious that certain requirements can usually be realized with different combinations of design parameters, but some of them are less sensitive than others. Thus it can be useful to not only include the performance itself but also the sensitivity of the performance in the network. One possible realization is illustrated in Fig.5.4.

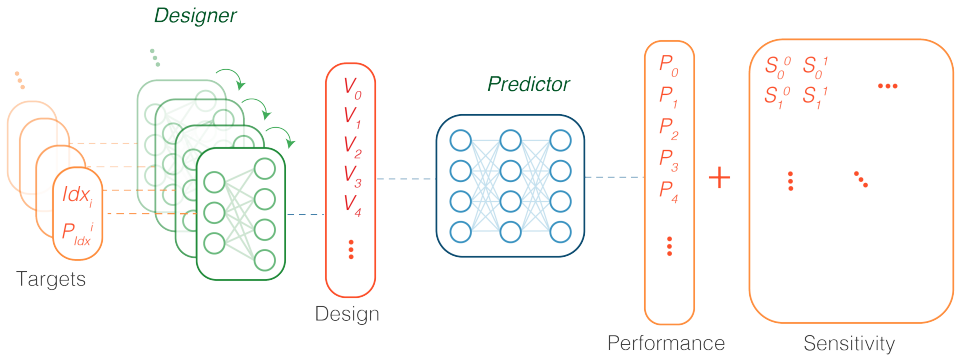

Figure 5.4. Sequential tandem network with sensitivity reduction.

The sensitivity S_i^j represent how sensitive P_i is with respect to V_j, and can be obtained with:

$$S_i^j = \frac{|P_i(V_i + \Delta V) - P_i(V_i)|}{\Delta V}, \tag{5.6}$$

by setting ΔV to a small value.

When training the designer, the loss function should also be designed to reduce the sensitivity, such as:

$$L = L_{mask} + \lambda_s \|\mathbf{S}\| \tag{5.7}$$

where weight λ_s determines the priority of reducing sensitivity.

By including the sensitivity information in both the predictor and the designer, we can hopefully set up the network with a preference to avoid using sensitive design parameter combinations when possible.

Bibliography

[1] R. A. Shelby, D. R. Smith, S. C. Nemat-Nasser, and S. Schultz. Microwave transmission through a two-dimensional, isotropic, left-handed metamaterial. *Applied Physics Letters*, 78(4):489–491, 2001.

[2] D. F. Sievenpiper, M. E. Sickmiller, and E. Yablonovitch. 3d wire mesh photonic crystals. *Physical Review Letters*, 76(14):2480–2483, 1996.

[3] D. Sievenpiper, L. J. Zhang, R. F. J. Broas, N. G. Alexopolous, and E. Yablonovitch. High-impedance electromagnetic surfaces with a forbidden frequency band. *IEEE Transactions on Microwave Theory and Techniques*, 47(11):2059–2074, 1999.

[4] J. Lee and D. F. Sievenpiper. Method for extracting the effective tensor surface impedance function from nonuniform, anisotropic, conductive patterns. *IEEE Transactions on Antennas and Propagation*, 67(5):3171–3177, 2019.

[5] D. F. Sievenpiper, J. H. Schaffner, H. J. Song, R. Y. Loo, and G. Tangonan. Two-dimensional beam steering using an electrically tunable impedance surface. *IEEE Transactions on Antennas and Propagation*, 51(10):2713–2722, 2003.

[6] D. R. Smith, W. J. Padilla, D. C. Vier, S. C. Nemat-Nasser, and S. Schultz. Composite medium with simultaneously negative permeability and permittivity. *Physical Review Letters*, 84(18):4184–4187, 2000.

[7] N. J. G. Fonseca, M. Coudyser, J. J. Laurin, and J. J. Brault. On the design of a compact neural network-based doa estimation system. *IEEE Transactions on Antennas and Propagation*, 58(2):357–366, 2010.

[8] S. Mishra, R. N. Yadav, and R. P. Singh. Directivity estimations for short dipole antenna arrays using radial basis function neural networks. *IEEE Antennas and Wireless Propagation Letters*, 14:1219–1222, 2015.

[9] H. T. He, C. K. Wen, S. Jin, and G. Y. Li. Deep learning-based channel estimation for beamspace mmwave massive mimo systems. *IEEE Wireless Communications Letters*, 7(5):852–855, 2018.

[10] L. L. Li, L. G. Wang, F. L. Teixeira, C. Liu, A. Nehorai, and T. J. Cui. Deepnis: Deep neural network for nonlinear electromagnetic inverse scattering. *IEEE Transactions on Antennas and Propagation*, 67(3):1819–1825, 2019.

[11] Z. Wei and X. D. Chen. Physics-inspired convolutional neural network for solving full-wave inverse scattering problems. *IEEE Transactions on Antennas and Propagation*, 67(9):6138–6148, 2019.

[12] T. Shan, W. Tang, X. W. Dang, M. K. Li, F. Yang, S. H. Xu, and J. Wu. Study on a fast solver for poisson's equation based on deep learning technique. *IEEE Transactions on Antennas and Propagation*, 68(9):6725–6733, 2020.

[13] Y. Sharma, H. H. Zhang, and H. Xin. Machine learning techniques for optimizing design of double t-shaped monopole antenna. *IEEE Transactions on Antennas and Propagation*, 68(7):5658–5663, 2020.

[14] L. Y. Xiao, W. Shao, F. L. Jin, and B. Z. Wang. Multiparameter modeling with ann for antenna design. *IEEE Transactions on Antennas and Propagation*, 66(7):3718–3723, 2018.

[15] L. Y. Xiao, W. Shao, X. Ding, Q. H. Liu, and W. T. Joines. Multigrade artificial neural network for the design of finite periodic arrays. *IEEE Transactions on Antennas and Propagation*, 67(5):3109–3116, 2019.

[16] R. G. Ayestaran, F. Las-Heras, and L. F. Herran. Neural modeling of mutual coupling for antenna array synthesis. *IEEE Transactions on Antennas and Propagation*, 55(3):832–840, 2007.

[17] T. N. Kapetanakis, I. O. Vardiambasis, M. P. Ioannidou, and A. Maras. Neural network modeling for the solution of the inverse loop antenna radiation problem. *IEEE Transactions on Antennas and Propagation*, 66(11):6283–6290, 2018.

[18] L. Y. Xiao, W. Shao, F. L. Jin, B. Z. Wang, and Q. H. Liu. Inverse artificial neural network for multiobjective antenna design. *IEEE Transactions on Antennas and Propagation*, 69(10):6651–6659, 2021.

[19] Z. C. Liu, D. Y. Zhu, S. P. Rodrigues, K. T. Lee, and W. S. Cai. Generative model for the inverse design of metasurfaces. *Nano Letters*, 18(10):6570–6576, 2018.

[20] W. Ma, F. Cheng, and Y. M. Liu. Deep-learning-enabled on-demand design of chiral metamaterials. *Acs Nano*, 12(6):6326–6334, 2018.

[21] D. J. Liu, Y. X. Tan, E. Khoram, and Z. F. Yu. Training deep neural networks for the inverse design of nanophotonic structures. *Acs Photonics*, 5(4):1365–1369, 2018.

[22] C. Cui, W. T. Li, X. T. Ye, P. Rocca, Y. Q. Hei, and X. W. Shi. An effective artificial neural network-based method for linear array beampattern synthesis. *IEEE Transactions on Antennas and Propagation*, 69(10):6431–6443, 2021.

[23] I. P. Gravas, Z. D. Zaharis, T. V. Yioultsis, P. I. Lazaridis, and T. D. Xenos. Adaptive beamforming with sidelobe suppression by placing extra radiation pattern nulls. *IEEE Transactions on Antennas and Propagation*, 67(6):3853–3862, 2019.

[24] W. D. Burnside and K. W. Burgener. High-frequency scattering by a thin lossless dielectric slab. *IEEE Transactions on Antennas and Propagation*, 31(1):104–110, 1983.

[25] S. So, J. Mun, and J. Rho. Simultaneous inverse design of materials and structures via deep learning: Demonstration of dipole resonance engineering using core-shell nanoparticles. *Acs Applied Materials & Interfaces*, 11(27):24264–24268, 2019.

[26] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[27] J. B. Pendry, A. J. Holden, W. J. Stewart, and I. Youngs. Extremely low frequency plasmons in metallic mesostructures. *Physical Review Letters*, 76(25):4773–4776, 1996.

[28] J. B. Pendry, D. Schurig, and D. R. Smith. Controlling electromagnetic fields. *Science*, 312(5781):1780–1782, 2006.

[29] A. B. Li, S. Singh, and D. Sievenpiper. Metasurfaces and their applications. *Nanophotonics*, 7(6):989–1011, 2018.

[30] D. Schurig, J. J. Mock, B. J. Justice, S. A. Cummer, J. B. Pendry, A. F. Starr, and D. R. Smith. Metamaterial electromagnetic cloak at microwave frequencies. *Science*, 314(5801):977–980, 2006.

[31] B. H. Fong, J. S. Colburn, J. J. Ottusch, J. L. Visher, and D. F. Sievenpiper. Scalar and tensor holographic artificial impedance surfaces. *IEEE Transactions on Antennas and Propagation*, 58(10):3212–3221, 2010.

[32] L. L. Huang, S. Zhang, and T. Zentgraf. Metasurface holography: from fundamentals to applications. *Nanophotonics*, 7(6):1169–1190, 2018.

[33] K. D. Wu, P. Coquet, Q. J. Wang, and P. Genevet. Modelling of free-form conformal metasurfaces. *Nature Communications*, 9, 2018.

[34] S. M. Kamali, A. Arbabi, E. Arbabi, Y. Horie, and A. Faraon. Decoupling optical function and geometrical form using conformal flexible dielectric metasurfaces. *Nature Communications*, 7, 2016.

[35] Y. J. Wang, J. X. Su, Z. R. Li, Q. X. Guo, and J. M. Song. A prismatic conformal metasurface for radar cross-sectional reduction. *IEEE Antennas and Wireless Propagation Letters*, 19(4):631–635, 2020.

[36] K. Y. Liu, G. M. Wang, T. Cai, H. P. Li, and T. Y. Li. Conformal polarization conversion metasurface for omni-directional circular polarization antenna application. *IEEE Transactions on Antennas and Propagation*, 69(6):3349–3358, 2021.

[37] Qiqi Dai, Yee Hui Lee, Hai-Han Sun, Genevieve Ow, Mohamed Lokman Mohd Yusof, and Abdulkadir C Yucel. Dmrf-unet: A two-stage deep learning scheme for gpr data inversion under heterogeneous soil conditions. *IEEE Transactions on Antennas and Propagation*, 2022.

[38] J. Q. Jiang, M. K. Chen, and J. A. Fan. Deep neural networks for the evaluation and design of photonic devices. *Nature Reviews Materials*, 6(8):679–700, 2021.

[39] P. Q. Liu, L. Chen, and Z. N. Chen. Prior-knowledge-guided deep-learning-enabled synthesis for broadband and large phase shift range metacells in metalens antenna. *IEEE Transactions on Antennas and Propagation*, 70(7):5024–5034, 2022.

[40] Lianlin Li, Hanting Zhao, Che Liu, Long Li, and Tie Jun Cui. Intelligent metasurfaces: control, communication and computing. *Elight*, 2(1):7, 2022.

[41] Shangyang Li, Zhuoyang Liu, Shilei Fu, Yan Wang, and Feng Xu. Intelligent beamforming via physics-inspired neural networks on programmable metasurface. *IEEE Transactions on Antennas and Propagation*, 70(6):4589–4599, 2022.

[42] L. L. Li, Y. Shuang, Q. Ma, H. Y. Li, H. T. Zhao, M. L. Wei, C. Liu, C. L. Hao, C. W. Qiu, and T. J. Cui. Intelligent metasurface imager and recognizer. *Light-Science & Applications*, 8, 2019.

[43] Zhuo Wang, Hongrui Zhang, Hanting Zhao, Shengguo Hu, Tie Jun Cui, and Lianlin Li. Multi-task and multi-scale intelligent electromagnetic sensing with distributed multi-frequency reprogrammable metasurfaces. *Advanced Optical Materials*, page 2203153, 2023.

[44] Chao Qian, Bin Zheng, Yichen Shen, Li Jing, Erping Li, Lian Shen, and Hongsheng Chen. Deep-learning-enabled self-adaptive microwave cloak without human intervention. *Nature Photonics*, pages 1–8, 2020.

[45] Yuetian Jia, Chao Qian, Zhixiang Fan, Yinzhang Ding, Zhedong Wang, Dengpan Wang, Er-Ping Li, Bin Zheng, Tong Cai, and Hongsheng Chen. In situ customized illusion enabled by global metasurface reconstruction. *Advanced Functional Materials*, 32(19):2109331, 2022.

[46] N. X. Wu, Y. T. Jia, C. Qian, and H. S. Chen. Pushing the limits of metasurface cloak using global inverse design. *Advanced Optical Materials*, 2023.

[47] D. Sievenpiper, J. Schaffner, R. Loo, G. Tangonan, S. Ontiveros, and R. Harold. A tunable impedance surface performing as a reconfigurable beam steering reflector. *IEEE Transactions on Antennas and Propagation*, 50(3):384–390, 2002.

[48] Q. Ma, G. D. Bai, H. B. Jing, C. Yang, L. L. Li, and T. J. Cui. Smart metasurface with self-adaptively reprogrammable functions. *Light-Science & Applications*, 8, 2019.

[49] R. Rombach, A. Blattmann, D. Lorenz, P. Esser, and B. Ommer. High-resolution image synthesis with latent diffusion models. *2022 IEEE/Cvf Conference on Computer Vision and Pattern Recognition (Cvpr)*, pages 10674–10685, 2022.

[50] L. Floridi and M. Chiriatti. Gpt-3: Its nature, scope, limits, and consequences. *Minds and Machines*, 30(4):681–694, 2020.

[51] Sébastien Bubeck, Varun Chandrasekaran, Ronen Eldan, Johannes Gehrke, Eric Horvitz, Ece Kamar, Peter Lee, Yin Tat Lee, Yuanzhi Li, and Scott Lundberg. Sparks of artificial general intelligence: Early experiments with gpt-4. *arXiv preprint arXiv:2303.12712*, 2023.

[52] Z. Zhen, C. Qian, Y. T. Jia, Z. X. Fan, R. Hao, T. Cai, B. Zheng, H. S. Chen, and E. P. Li. Realizing transmitted metasurface cloak by a tandem neural network. *Photonics Research*, 9(5):B229–B235, 2021.

[53] E. D. Wen, X. Z. Yang, and D. F. Sievenpiper. Real-time 2-d beamforming with rotatable dielectric slabs enabled by generative neural network. *IEEE Transactions on Antennas and Propagation*, 70(9):8360–8367, 2022.

[54] S. M. Ren, A. Mahendra, O. Khatib, Y. Deng, W. J. Padilla, and J. M. Malof. Inverse deep learning methods and benchmarks for artificial electromagnetic material design. *Nanoscale*, 14(10):3958–3969, 2022.

[55] Y. Yu, X. S. Si, C. H. Hu, and J. X. Zhang. A review of recurrent neural networks: Lstm cells and network architectures. *Neural Computation*, 31(7):1235–1270, 2019.

[56] Constantine A Balanis. *Antenna theory: analysis and design*. John wiley & sons, 2016.

[57] Y. J. Tian and Y. Q. Zhang. A comprehensive survey on regularization strategies in machine learning. *Information Fusion*, 80:146–166, 2022.