Computer Networks 270 (2025) 111544

Contents lists available at ScienceDirect

Computer Networks

journal homepage: www.elsevier.com/locate/comnet

Learning to shape beams: Using a neural network to control a beamforming antenna

Jose David Fernández-Rodríguez [a,c], Iván García-Aguilar [a], Rafael Marcos Luque-Baena [a,c,*], Ezequiel López-Rubio [a,c], Marcos Baena-Molina [b], Juan Francisco Valenzuela-Valdés [b]

[a] University of Malaga, Bulevar Louis Pasteur, 35, Malaga, 29071, Spain
[b] University of Granada, Periodista Rafael Gomez Montero, S/N, Granada, 18071, Spain
[c] ITIS Software, Calle Arquitecto Francisco Penalosa, 18, Malaga, 29010, Spain

ARTICLE INFO

Keywords:
Beamforming
Neural network
Deep learning

ABSTRACT

The field of reconfigurable intelligent surfaces (RIS) has gained significant traction in recent years in the wireless communications domain, owing to the ability to dynamically reconfigure surfaces to change their electromagnetic radiance patterns in real-time. In this work, we propose utilizing a novel deep learning model that innovatively employs only the parameters of each signal or beam as input, eliminating the need for the entire one-dimensional signal or its diffusion map (two-dimensional information). This approach enhances efficiency and reduces the overall complexity of the model, drastically reducing network size and enabling its implementation on low-cost devices. Furthermore, to enhance training effectiveness, the learning model attempts to estimate the discrete cosine transform applied to the output matrix rather than the raw matrix, significantly improving the achieved accuracy. This scheme is validated on a 1-bit programmable metasurface of size 10×10, achieving an accuracy close to 95% using a K-fold methodology with K=10.

1. Introduction

In recent years, the advancement of technologies based on reconfigurable intelligent surfaces (RIS) has revolutionized the wireless communications arena. This field has emerged as a cutting-edge technology that can transform interaction in the wireless environment. These are composed of a matrix of microscopic elements that can be electronically adjusted to modify the properties of the electromagnetic waves that pass through them, thus allowing signal optimization, eliminating obstacles, and adaptation to changing environments [1,2]. Thanks to the controlled handling in the selective alteration of wave characteristics, such as direction, polarization, amplitude, or phase, it has opened a wide range of possibilities in various applications, such as the optimization of coverage or the improvement in the efficiency and manipulation of signals across a variety of systems [3,4].

As the field advances, significant challenges arise, mainly based on electromagnetic beam shaping and optimizing these RIS. One of the problems relates to the need to generate a series of precise electromagnetic beam configurations in real-time, thus requiring advanced solutions and the integration of new emerging technologies.

Currently, the focal points of RIS research encompass optimization problems and analytical challenges. Optimization problems involve developing joint beamforming and phase optimization algorithms with reduced complexity [5–7]. Analytical concerns include evaluating the performance of RIS systems and conducting comparative studies between RIS and conventional relay systems. For instance, Di Renzo et al. [8] analyzed the similarities and differences between RIS and traditional relays using a plane distance/pathloss model. Björnson et al. [9] introduced a classic two-dimensional single-input single-output (SISO) system model, exploring the number of reflecting elements. Furthermore, Abdullah et al. [10] contributed by presenting an average performance analysis in a hybrid RIS and relay network encompassing various hybrid configurations. However, these studies often neglect the actual vertical positions of the base station, user, and RIS, and they rarely address the numerical verification of the relationship between the placement of RIS and performance enhancements. Consequently, a pressing need exists for RIS deployment analysis to provide practical guidance for its real-world applications.

Programmable metasurfaces have recently been proposed and can dynamically manipulate the electromagnetic waves impinging on them. This is achieved by incorporating active components in the metasurface elements, such as PIN (positive-intrinsic-negative) diodes. By independently changing the state of these components, the phase of the wave reflected by each metasurface element can be adjusted, and complex

* Corresponding author at: University of Malaga, Bulevar Louis Pasteur, 35, Malaga, 29071, Spain.
E-mail address: rmluque@uma.es (R.M. Luque-Baena).

https://doi.org/10.1016/j.comnet.2025.111544
Received 23 May 2024; Received in revised form 24 April 2025; Accepted 7 July 2025
Available online 18 July 2025

spatial waveforms can be created. This provides an additional degree of freedom to control wave propagation in real-time. The reflection phase of the elements can be quantified in binary codes. For example, the 1-bit code “0/1” represents that the element can have two different reflection coefficients, and the 2-bit code “00/01/10/11” indicates four different states in the element’s reflection coefficients. According to the dynamic manipulation of electromagnetic waves using meta-programmable surfaces, these surfaces allow the properties of the unit cells to be adjusted using digital codes, making it possible to generate complex electromagnetic beams in both the spatial and time domains.

The traditional approach to determining the optimal codes required to define the desired beam parameters has involved the use of non-linear optimization algorithms [11], such as genetic algorithms [12] and particle swarm optimization [13], among others. However, the high computational complexity of these methods has posed significant challenges to their real-time implementation [14].

Deep Learning has provided solutions to various fields, such as image and video processing [15,16]. Recently, these techniques have also been applied in the field of RIS with satisfactory results. A variant of the VGG model (a deep learning model designed to perform image classification [17]) was developed by Shan et al. [18] to estimate coding schemes for 1-bit programmable metasurfaces from 2D input matrices representing the radiation pattern for each possible beam configuration; each beam direction can be expressed using two angles, ϕ and θ, as seen in Fig. 1. One-dimensional signals representing the input radiation were used in [19,20]. In these proposals, two models were applied: forward prediction from the metasurface and inverse design from the one-dimensional signal. Similarly, Qiu et al. [21] propose a new methodology (REACTIVE) to automatically estimate the metasurface structure from a set of patterns obtained after applying an autoencoder that encodes the input signal (S-parameters). A similar proposal is made by Shi et al. [22]: an autoencoder is trained to obtain a reduced representation of electromagnetic signal properties and subsequently used to optimize a Support Vector Machine (a classical machine learning model [23]) using Artificial Bee Colony [24] (an optimization meta-heuristic). Despite the previous proposals’ very interesting results, the neural network size in each proposal may be too large to adapt to low-cost devices. On the other hand, the input information may have some redundancy (one-dimensional signals of hundreds of features or 2D maps), as in most cases, they are generated by the ϕ and θ values of the beam on which to estimate the metasurface.

The present work focuses on developing a neural network in which an electromagnetic beam configuration can return the activation pattern of actuator elements necessary to generate such a beam configuration. This approach results in a substantial improvement in the efficiency of RIS through the involvement of machine learning using neural networks, taking as input only the ϕ and θ values for each beam. A deep convolutional neural network has been designed and trained to efficiently compute the necessary element codes. The proposed approach has been validated using a 1-bit meta-programmable surface, thus obtaining consistent results. Consequently, the significant potential of Deep Learning is highlighted, offering a promising solution and opening new avenues for its further application in different fields.

This paper is organized as follows: Section 2 presents the deep learning framework; Section 3 shows numerical and experimental results to benchmark the performance of this scheme; Discussion and Conclusions are presented in Section 4.

2. Methods

2.1. Reconfigurable intelligent surfaces

One of the problems in communications today, and most likely in the future, is that signals experience absorption effects, unwanted reflections, and even scattering due to the elements generally present in everyday propagation environments. As shown in Fig. 2, a RIS can be mounted in strategic, pre-chosen locations to adjust the reflection of the incident wave. They not only illuminate a shaded area but may also be able to increase the received signal strength for the user and minimize other areas [25]. Furthermore, with the help of reconfigurability, interruptions in communications towards target users can be avoided [26]. In this way, a RIS can be deemed to maintain a virtual Line Of Sight (LOS) between the transmitter and receiver. This is precisely where RIS technology becomes an important feature for future generations of communications.

Nowadays, there are several techniques for constructing RIS, such as patch antennas, diode-switching electronics, and mechanically moving elements, among others. The complete set of these elements can modify the phase of the electromagnetic waves impinging on the RIS. Therefore, the design and creation of these RIS represent a significant challenge due to their very high frequency.

At the element level, RIS elements must be properly designed to phase shift a specific value concerning its neighboring cells. However, the required phase shift is in a continuous interval, implying an infinite number of states. Truncation of these phases is necessary, either due to technological limitations or the nature of the devices, thus introducing the resolution parameter of an RIS. For example, for a resolution of 1 bit, the space of possible phase shifts is divided into two zones around 0° and 180°, thus truncating all phase shift values to these two values.

The proposed deep learning model in this article is based on the 1-bit truncation features and 10×10 element size of a previously fabricated [27] and validated [28] RIS prototype, shown in Fig. 3. The unit cell consists of a metallic cylinder whose end has a conical-shaped element with a metallic coating. A linear motion enabled by an electromagnet allows the system to achieve two configuration states corresponding to 1-bit truncation.

2.2. Computing RIS configurations

The availability of a large number of bits directly implies a better resolution at the expense of higher complexity in the computational and control of the RIS. It is crucial to note how the desired target direction is represented in the reflected wave, as this information will be fed into the network. The representation is in the form of a 2D radiation pattern, commonly known as a UV pattern. This diagram represents, in a general way, the spatial distribution of the signal energy, indicating the direction in which the signal has the highest energy.

In order to locate the points with the highest energy, a UV mapping was carried out, which consists of transforming the angle of the beam (θ and ϕ; see Fig. 1) to the corresponding spatial coordinates u and v. This transformation follows the equations:

$$\begin{aligned} u &= \sin\theta\cos\phi \\ v &= \sin\theta\sin\phi. \end{aligned} \tag{1}$$

The output for these radiation patterns is represented as a $N \times N$ element matrix, where each matrix element illustrates the required phase shift in the corresponding RIS cell, using $N = 10$ for the RIS prototype developed in this work. To obtain the phase values of the elements, a brute-force *ad hoc* algorithm was used: a first random configuration of 0/1 values is made, and the energy in the desired direction is obtained. Then, successively and element by element, the state is inverted to recalculate the energy in the desired direction. If the energy increases, the inversion is maintained; otherwise, it is inverted again. After processing all elements, further sweeps are carried out until the stopping criterion is met. This criterion considers whether, in the last iteration of the complete configuration, less than 10% of all RIS states have been inverted. This threshold is set because the computational cost of performing a new sweep does not justify improving the energy value.

It is important to note that, although the algorithm converges relatively quickly, the associated computational cost is high and sustained over time. For this reason, it is not suitable for real-time computations.

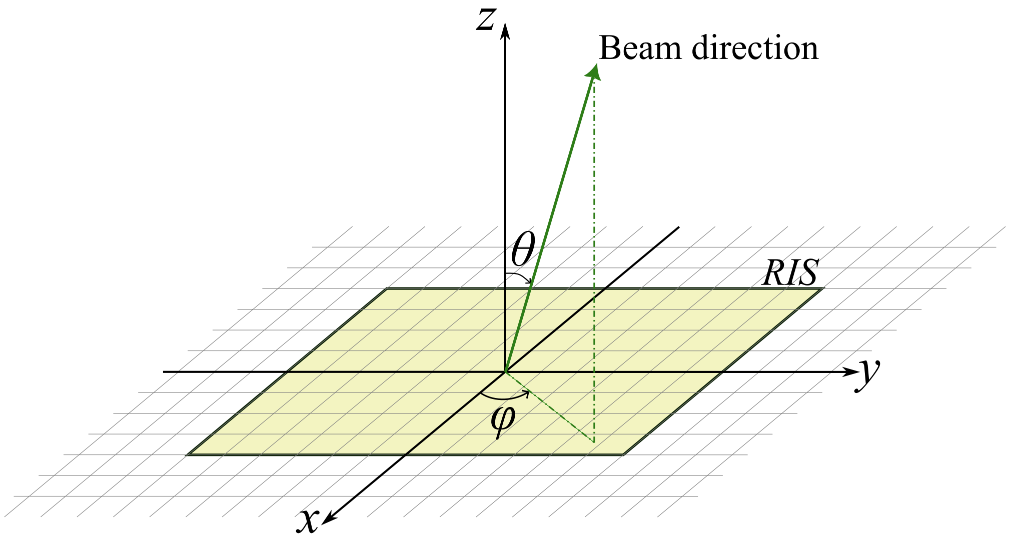


Fig. 1. Representation of the angles ϕ and θ characterizing each beam shaped by the reconfigurable intelligent surface (RIS).

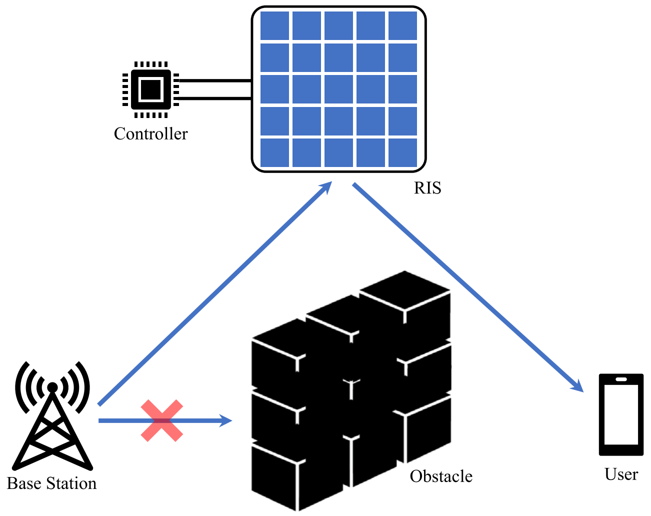


Fig. 2. A RIS being used to redirect a beam from a base station to a user, i.e., creating a virtual Line Of Sight (LOS).

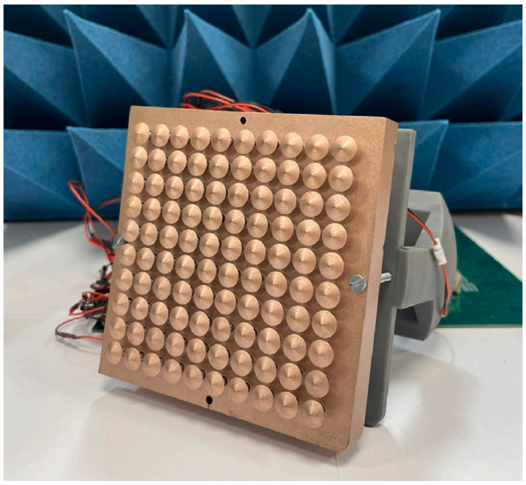


Fig. 3. Photograph of 10 × 10 RIS prototype with 1-bit reconfiguration.

2.3. Proposed deep neural network model

The RIS prototype considered in this work can shape beams in a range of angles. Considering that the RIS is in a horizontal coordinate system pointing towards the zenith (azimuth $\phi = 0°$, elevation $\theta = 0°$), the RIS can shape the beam to point to an array of directions within $\phi \in [-89° \ldots 90°]$ and $\theta \in [-25° \ldots 25°]$. Each reconfigurable element in the RIS can be in any of two states $\{0, 1\}$, corresponding to angles 0° and 180° (see Fig. 8 for some examples). It is important to note that the beam is shaped by the differences in phase between the elements of the RIS device (not by the neural network, whose task is to generate a configuration pattern for the elements of the RIS device). As a corollary, if we take the state of each RIS element to be a Boolean value and apply the Boolean NOT operator to all of them, the resulting beam orientation will remain the same. Regarding the training of the neural network, this is a fundamental property of the mapping between angles ϕ, θ and the grid patterns for the RIS.

The RIS should be able to be driven from a low-powered device. For this purpose, we propose a lightweight neural network architecture designed to accept as input two angles and to generate as output a 10 × 10 grid of values representing the state of the RIS. Angles ϕ, θ at the input are encoded as sin/cos pairs (as it is common with angular values in the context of deep learning), so the input is a 4-dimensional vector $\sin(\phi), \cos(\phi), \sin(\theta), \cos(\theta)$, as depicted in Fig. 4. The network has an architecture based on a convolutional decoder, with a first fully-connected layer to increase the dimensionality of the input from 4 to 30. Then, three deconvolutions are applied to increase the width and height, and finally, two convolutions to gradually decrease the number of channels, from $8N$ (N being an architecture parameter) to 1. The number of layers and per-layer channel depths is chosen based on experience with convolutional decoders with small outputs [29]. Each one of these layers is followed by batch normalization and an activation function (noted as *Activation1/2* in Fig. 5). For the last layer, the activation function depends on the specific configuration of the network. For the others, preliminary tests were conducted with ReLU, leaky ReLU, and SiLU activation functions, finding that SiLU [30] trains significantly better than the others, so it was used in all subsequent tests (described in the next Section). Channel depth in convolutional and deconvolutional layers is modulated using a parameter N, with an architecture that starts with $8N$ channels and gradually decreases that number up to the output, with only one channel. Stride size is 1 for all convolutional and deconvolutional layers, while padding size is 0 for deconvolutional layers and 1 for convolutional ones.

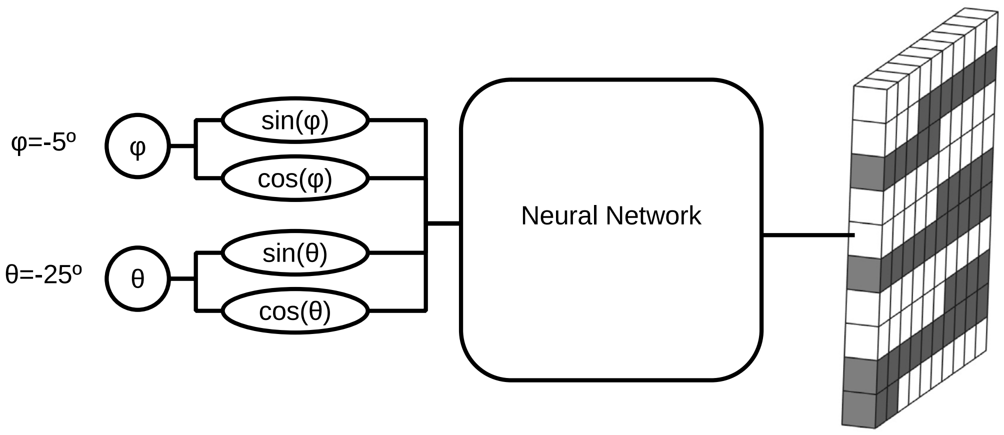


Fig. 4. Our proposed RIS can shape beams whose orientation is characterized by two angles, ϕ and θ. For a specific pair of angles ϕ, θ, our simplest proposed neural network variant takes as inputs their sines and cosines and generates as output a 2D grid of Boolean values to configure the elements of the RIS for that beam. In this figure, we show as the output the ground truth for the beam with $\phi = -5°, \theta = -25°$. For a diagram showing the proposed neural network architecture, see Fig. 5.

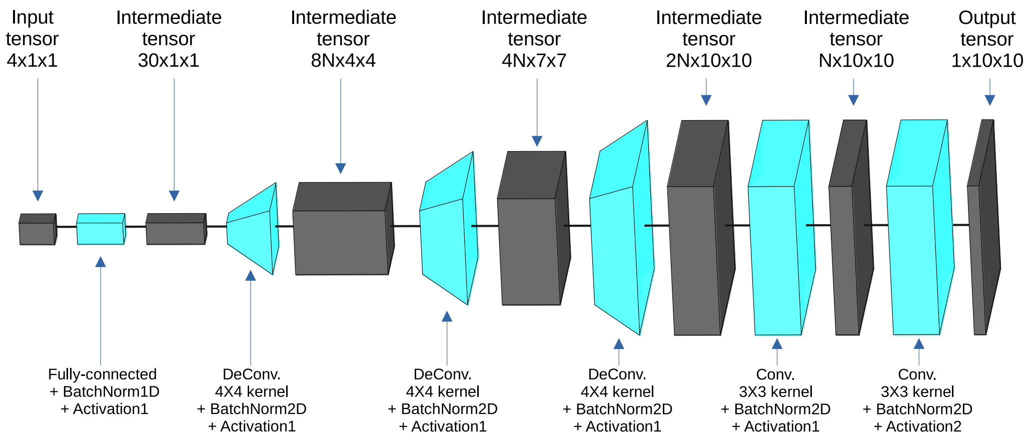

Input tensor 4x1x1
Intermediate tensor 30x1x1
Intermediate tensor 8Nx4x4
Intermediate tensor 4Nx7x7
Intermediate tensor 2Nx10x10
Intermediate tensor Nx10x10
Output tensor 1x10x10
Fully-connected + BatchNorm1D + Activation1
DeConv. 4X4 kernel + BatchNorm2D + Activation1
DeConv. 4X4 kernel + BatchNorm2D + Activation1
DeConv. 4X4 kernel + BatchNorm2D + Activation1
Conv. 3X3 kernel + BatchNorm2D + Activation1
Conv. 3X3 kernel + BatchNorm2D + Activation2

Fig. 5. Graphical representation of the family of deep learning networks proposed in this work. Input/intermediate/output tensors are depicted as dark prisms with labels above, detailing tensor sizes (number of channels X height X width). The network is composed of a sequence of layers. Each layer transforms a tensor into the next one and is depicted as a prism or frustum in light blue, with labels below explaining details for each layer. Padding is 1 for convolutions, 0 for deconvolutions, and the stride is 1 for all. Note that the same proposed architecture is used for all variants (with output as either Boolean patterns or the DCTs of such patterns; see text for details). (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

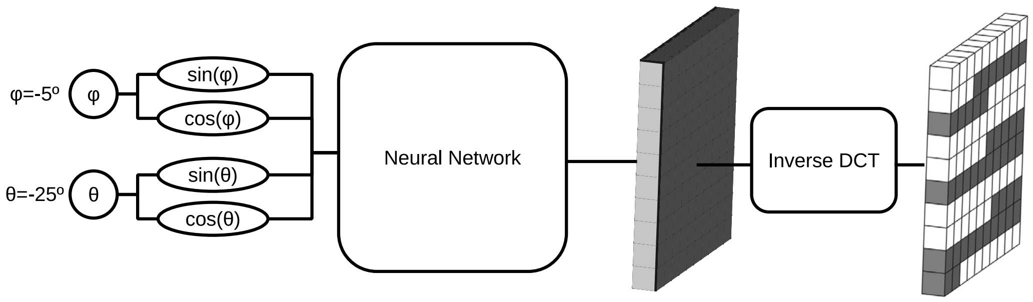


Fig. 6. Same as Fig. 4, but showing the setup for another proposed neural network variant: the network does not output the Boolean values directly, but the DCT transform of the pattern. After an inverse DCT step, this results in the requested grid pattern. Please note that the network has the same architecture, with the same layers and intermediate tensors of the same dimensions; only the expected output is changed concerning the variant shown in Fig. 4.

This proposed neural architecture was chosen because it is compact, and convolutional networks are well-suited to both analyze and generate 2D patterns with significant 2D structures. Simpler architectures, such as Multi-Layer Perceptrons, were explored in preliminary experiments but were found to perform badly for this use case, with very high error rates. More sophisticated architectures for image generation (such as Generative Adversarial Networks, Gaussian Diffusion Models, or Image Transformers) were deemed unsuitable to the problem at hand, since they are geared towards significantly larger output 2D patterns, with a substantially increased computational cost both for training and inference.

2.4. Proposed training methods

Several training schemes have been proposed, which are associated with certain variations of the representation of the output and the

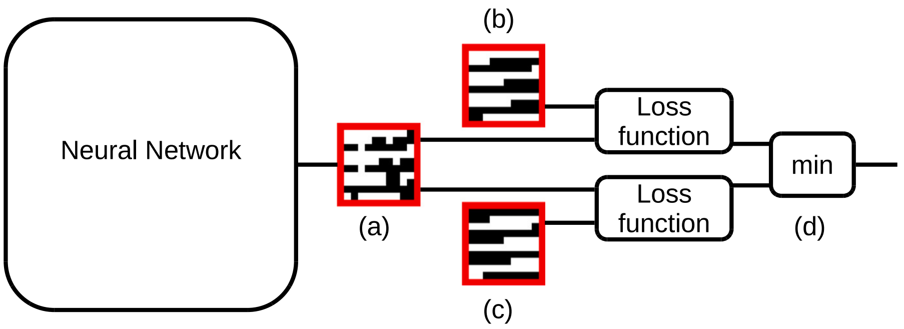


Fig. 7. Proposed training scheme. To train the network, we have to compare the network's output (a) with the ground truth. However, our RIS will produce the same electromagnetic pattern if all the values of the RIS elements are inverted. To account for this, we have found that an effective strategy is to compute the loss concerning both versions of the ground truth (b and c) and take the effective loss as the minimum of both values. In this way, gradients will flow in each case through the version of the ground truth closest to the network's output, and the network self-configures during training to select the best version for each case.

loss function. In the simpler, more straightforward one, we train the network to directly predict a Boolean value for each element in the 10×10 grid of the RIS, using as loss function either binary cross-entropy (BCE) or mean squared error (MSE), and using as the last activation function (noted as *Activation2* in Fig. 5) the logistic function (0/1 sigmoid). At validation time, values are rounded to the nearest integer.

However, the grid patterns corresponding to each beam orientation are relatively hard to learn for a neural network since slight variations in input angles ϕ and θ can result in significant variations in the values of each component of the grid. As the grid patterns in our dataset are typically reminiscent of periodic waves, we also test a more complex variation of our network, training it to predict the discrete cosine transform (DCT) of the grid pattern (changing 0 values to -1 to minimize the DC component of the DCT). This requires two additional steps:

- Using a hyperbolic tangent ($-1/1$ sigmoid) as the last activation function. To obtain the network output, the result is scaled by 1.1 times the maximal DCT values (in absolute terms) observed for our datasets. This scaling is used in order to accelerate the convergence of the training, trying to avoid very low values in the derivative of the loss function for the gradient descent when the output of the network is close to the ground truth values.
- Using an inverse DCT to convert the network output back to a pattern suitable for the RIS (at validation time, the resulting values are rounded to the nearest integer).

This DCT variant (see Fig. 6) is relatively inconvenient since it introduces additional steps to convert the network output into a grid pattern. However, only a few DCT components (corresponding to low to mid frequencies) have large values for most of the grid patterns generally found in our datasets (and these components are the most important in getting a good prediction of the resulting grid pattern). It is slightly easier to train the network to predict these DCT components than the raw grid patterns.

The DCT variant can be trained in two different ways:

- The loss function is the MSE between the DCT of the ground truth and the output of the network, thus directly training it to output the DCT of the expected grid pattern.
- Converting the network output back to a grid pattern with an inverse DCT and then using either a BCE or MSE loss function for the expected grid pattern, indirectly training it to output the DCT of the expected grid pattern.

It was expected that applying the loss function *after* the inverse DCT transform (second option above) would help in the training process, since the DCT components would receive gradients proportional to their respective contributions to the expected grid pattern. However, in preliminary explorations, training was significantly slower (as it required computing gradients through an inverse DCT, a relatively expensive operation), and (more importantly) it seemed to achieve worse results than other training schemes. Therefore, the experiments described in the next Section systematically explored only the first option (taking the MSE with respect to the DCT values of the grid pattern).

Finally, we also consider an alternative divide-and-conquer setup where we use two different neural networks in tandem, both at train and validation time, such that the whole input range is divided among them: one handles inputs in the range $\phi \in [-89^\circ \ldots 0^\circ]$ and the other in the range $\phi \in (0^\circ \ldots 90^\circ]$ (in both cases, combined with all possible values for θ).

Once we have presented the scenarios and loss functions, we have to reconsider an essential aspect of the mapping between input angles and output grid patterns: as stated above, for each beam orientation characterized by a pair of input angles, there is not one but two grid patterns to achieve it, and one is the Boolean negation of the other. This makes the mapping from beam shapes to grid patterns more challenging to learn. Unless we carefully consider how to associate angles and grid patterns, samples with very similar input angles can have very different values for most elements in the output pattern, making for very noisy (and very difficult to learn) output functions for each RIS element. To choose one of the two versions of the mapping for each pair of input angles ϕ, θ, we devised two heuristics: taking the pattern with a minimal number of 1 values and the pattern with minimal mean DCT value. However, it is not clear if any of these heuristics does find a good enough assignment, well-coordinated across most of the field of possible input angles. We can also modify all the loss functions proposed above so that the training process considers finding a good compromise assignment. Instead of measuring the loss between just one of the two possible grid patterns, we measure the loss against both of them and take as the true loss the minimum among them (see Fig. 7). In this way, gradients will flow through the version that gets better loss values in each case, and the training process will automatically choose grid patterns that minimize differences among grid patterns with similar input angles. This technique can also be applied when training with DCT output patterns, simply testing the loss against both DCT patterns (as the DCT is a linear transform, each DCT pattern is the negation of the other). In preliminary tests, we found that modifying the loss functions in this way consistently outperformed the other heuristics, so we chose this as our strategy to solve this problem.

The network's performance at validation time can be measured in two ways. The simplest is the difference between the output of the network and the expected output, measured as the error rate (the number of erroneous values divided by the total number of values in the output). However, this is a very crude measure of the quality of the beam radiating off the RIS. As a better, more physically grounded measure of the beam quality, we use radiation error: for each sample, electromagnetic radiation profiles of the RIS are computed both for

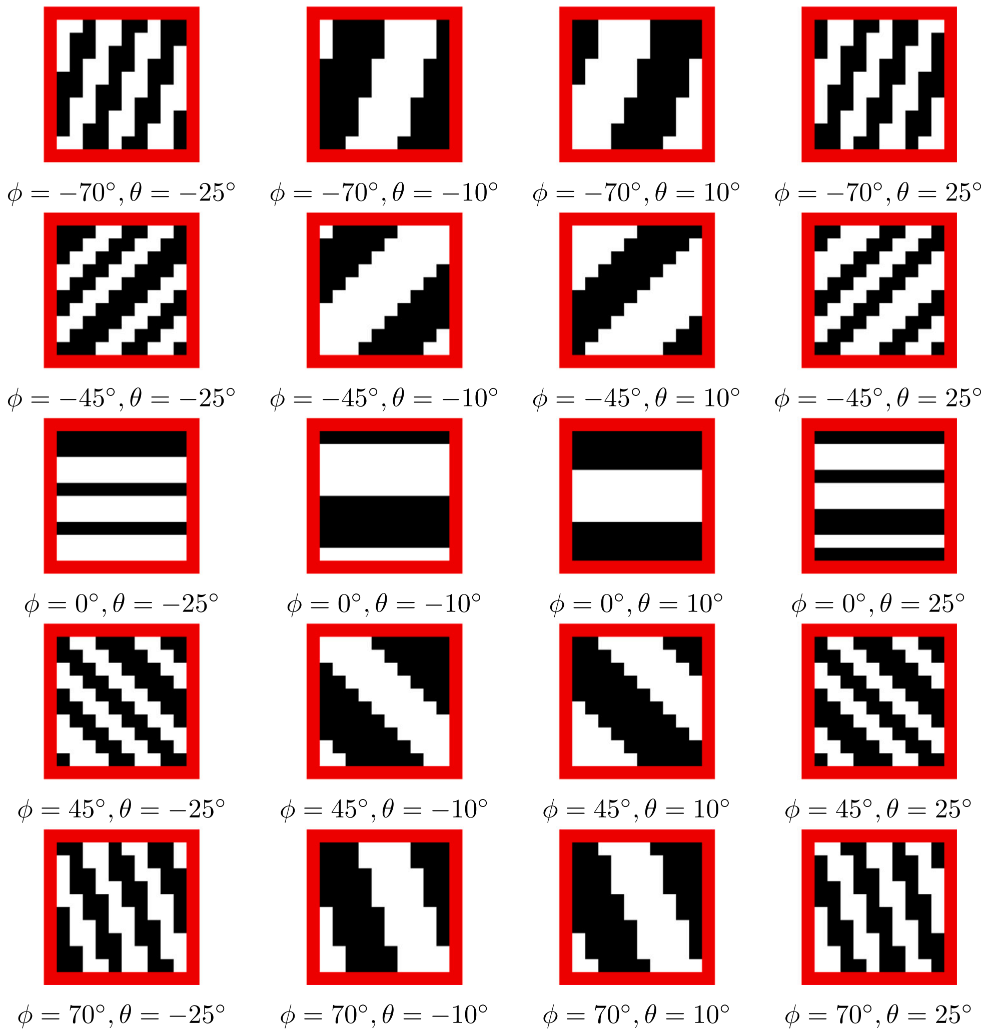

$\phi = -70°, \theta = -25°$
$\phi = -70°, \theta = -10°$
$\phi = -70°, \theta = 10°$
$\phi = -70°, \theta = 25°$
$\phi = -45°, \theta = -25°$
$\phi = -45°, \theta = -10°$
$\phi = -45°, \theta = 10°$
$\phi = -45°, \theta = 25°$
$\phi = 0°, \theta = -25°$
$\phi = 0°, \theta = -10°$
$\phi = 0°, \theta = 10°$
$\phi = 0°, \theta = 25°$
$\phi = 45°, \theta = -25°$
$\phi = 45°, \theta = -10°$
$\phi = 45°, \theta = 10°$
$\phi = 45°, \theta = 25°$
$\phi = 70°, \theta = -25°$
$\phi = 70°, \theta = -10°$
$\phi = 70°, \theta = 10°$
$\phi = 70°, \theta = 25°$

Fig. 8. Various samples from the dataset. Each sample is a pair of angles ϕ, θ characterizing the orientation of a beam and the associated grid pattern for the RIS to get that beam. Please note that the depicted pattern and its negation are equally valid for each pair of angles.

the output of the network and for the ground truth, as measured in a regular grid around the RIS, taking the mean squared error between them. In fact, it is feasible to implement the radiation error in PyTorch as a differentiable function, so that it can be used as the loss function. This has the benefit of more accurately reflecting the quality of the grid patterns and implicitly considering that each grid pattern and its negation lead to the same beam shape. Unfortunately, in all preliminary tests, the training process diverged, with NaN values always appearing before the radiation error improved to values comparable to those obtained with other loss functions. Even after extensive parameter tuning, this problem persisted, so we ended up discarding the use of radiation error as a loss function, using it only at validation time.

3. Results

3.1. Dataset

As specified in Section 2, the RIS can shape the beam to point in an array of directions within $\phi \in [-89°, 90°]$ and $\theta \in [-25°, 25°]$. We build our dataset as a set of samples. Each sample consists of the input to the network (a pair of input angles ϕ and θ representing the desired beam orientation) and its expected output (a 10×10 grid pattern of Boolean values for the RIS). Some samples are shown in Fig. 8.

In our datasets, we arrange the samples in a regular grid of values for $\phi \in [-89° \dots 90°]$ and $\theta \in [-25° \dots 25°]$. In this way, the size of the dataset will be determined by the steps $\Delta\phi$ and $\Delta\theta$. We use 10-fold cross-validation to assess the performance and ability of the network architecture to generalize. Initially, we used steps $\Delta\phi = \Delta\theta = \Delta = 1°$, resulting in datasets with $(25 - (-25) + 1) \cdot (90 - (-89) + 1) = 9180$ samples, since 1° steps were small enough to test the RIS adequately. However, in preliminary tests, networks trained with these relatively small datasets struggled to generalize to unseen samples, with mean error rates consistently above 10% in all experiments. When we decreased the grid steps to $\Delta\phi = \Delta\theta = \Delta = (1/3)°$ (consequently increasing the number of samples to $9180 \cdot 3^2 = 82620$), we found that the proposed architecture generalized substantially better.

3.2. Experimental setup

We study the performance of our network architecture for all combinations of several parameters and architectural choices:

- The scaling factor N for channel depth: we test architectures with $N = 25$ and $N = 50$. For $N = 25$, the neural network has approximately 0.5 million parameters, while $N = 50$ has approximately 1.8 million parameters.
- The network output: either raw binary values (BIN) or the DCT of those values.
- The loss function: for BIN, we test both BCE and MSE; for DCT, only MSE is applicable.

- Divide and conquer: we test configurations with and without the divide-and-conquer strategy described in Section 2.3, resulting in scenarios with either 2 or 1 network, respectively.

Combining all these possibilities, we test 12 different scenarios. For each scenario, we perform 10-fold cross-validation. Additionally, as the dataset is relatively small and covers all physically feasible beam orientations, we also conduct ten independent trainings in each scenario using the entire dataset for both training and validation, in order to assess the network's performance without considering generalization capability. In total, we run 240 trainings.

PyTorch is used to implement the deep learning system.[1] The training uses the Adam optimizer with standard parameters, running for 10,000 epochs in all cases and performing validation every two epochs. To minimize overfitting, uniform random noise in the interval $(-4/2, 4/2)$ is also added to input angles ϕ and θ at training time, but not at validation time.

At validation time, we compute and record the mean results across all validation samples for the error rate. If the new mean error rate improves upon the previous best mean value, we also compute and record the mean value of the radiation error across all validation samples. This is an imperfect methodology because error rate and radiation error are not strongly correlated. There may be times when the mean radiation error gets better even if the mean error rate does not, and the best network weights, as measured by radiation error, may go unnoticed and unrecorded. However, radiation error is many orders of magnitude more computationally expensive than error rate. To illustrate this, it is worth considering that it is much more costly (roughly 5 to 10 times, depending on the specific scenario) to compute the radiation error for the validation samples than performing a whole training epoch in 10-fold cross-validation experiments. Because of this, it is not practical to calculate it at every validation step, so we compromise by computing and recording it only when the error rate improves. We simultaneously record the best network weights according to both the radiation error and the error rate, so at the end of each training, we have the best results by error rate and radiation error.

With this setup, training times on a computer with an i9-10900X CPU, 128 GB of RAM and four 3090 Nvidia GPUs are not strongly dependent on channel depth, number of networks, output format, and validation set size (as the computational load is not GPU-bound in this setup), ranging from approximately 18 to 26 h per training, using one GPU per training (allowing for four simultaneous trainings on the machine). The 240 trainings were completed in a little over two months.

3.3. Experimental results

Table 1 shows a statistical summary of results for 10-fold cross-validation experiments across all scenarios. Each row summarizes the results for one scenario: 10 trainings (corresponding to the ten folds) are run for that scenario; the best results in error rate and radiation error are collected for each training, and the corresponding mean and standard deviation values are reported. Several trends are apparent:

- Error rate is only weakly correlated with radiation error, so it can be used as a rough guide, but cannot act as a substitute metric.
- In almost all cases, larger networks ($N = 50$) perform better than smaller ones ($N = 25$).
- For networks with raw binary output, using MSE loss is almost always better than BCE loss, but only very slightly so, and the effect of network size is relatively small.
- Networks with DCT output consistently outperform networks with raw binary output.

Table 1
Tabulated results for 10-fold cross-validation experiments. Best values in bold. N is the parameter for channel depth, n is the number of networks (over 1 for divide-and-conquer).

Results for 10-fold cross-validation							
Scenario config				Bit count stats		Radiation error stats	
N	Output	Loss	n	Mean	Std	Mean	Std
25	BIN	BCE	1	1.08E−01	±1.30E−03	5.09E−02	±2.70E−03
25	BIN	MSE	1	1.04E−01	±1.19E−03	4.98E−02	±2.17E−03
25	DCT	MSE	1	9.25E−02	±1.18E−03	4.31E−02	±1.74E−03
25	BIN	BCE	2	8.10E−02	±5.33E−03	5.10E−02	±4.92E−03
25	BIN	MSE	2	7.96E−02	±3.63E−03	4.85E−02	±2.12E−03
25	DCT	MSE	2	6.99E−02	±2.32E−03	3.61E−02	±1.69E−03
50	BIN	BCE	1	7.81E−02	±1.34E−03	4.91E−02	±1.55E−03
50	BIN	MSE	1	7.81E−02	±1.63E−03	4.62E−02	±1.97E−03
50	DCT	MSE	1	6.94E−02	±1.62E−03	3.66E−02	±1.12E−03
50	BIN	BCE	2	5.68E−02	±6.25E−03	4.22E−02	±6.16E−03
50	BIN	MSE	2	5.89E−02	±6.58E−03	4.30E−02	±5.78E−03
50	DCT	MSE	2	**5.24E−02**	±4.39E−03	**2.98E−02**	±2.53E−03

- Divide-and-conquer scenarios achieve, on average, better results than using just one network in almost all cases, but at the cost of significantly increased dispersion in the results (i.e., greater standard deviation).

Four comparisons between ground truth and predicted RIS patterns are shown in Fig. 9, all of them from the best network from the first cross-validation fold, whose configuration is shown in the first row of Table 1. For each example, both the corresponding ground truth and the predicted RIS pattern are shown, along with the radiation pattern generated by the RIS, both in 3D and U-V views, with both plots having the same color scheme for radiance power, in decibels. As can be seen from these examples, the correlation between the error rate (ratio of different cells between ground truth and prediction, with respect to the total number of cells) and the radiation error is not strong at low error rates, and depends on the specific patterns being compared in each case. Due to the characteristics of the RIS (1-bit configuration), the desired beam and the specular beam are always generated, as shown in the figure. Note that both ground truth and predicted RIS patterns produce two beams, and beam angles ϕ and θ will correspond to just one of these beams in the ground truth RIS configuration.

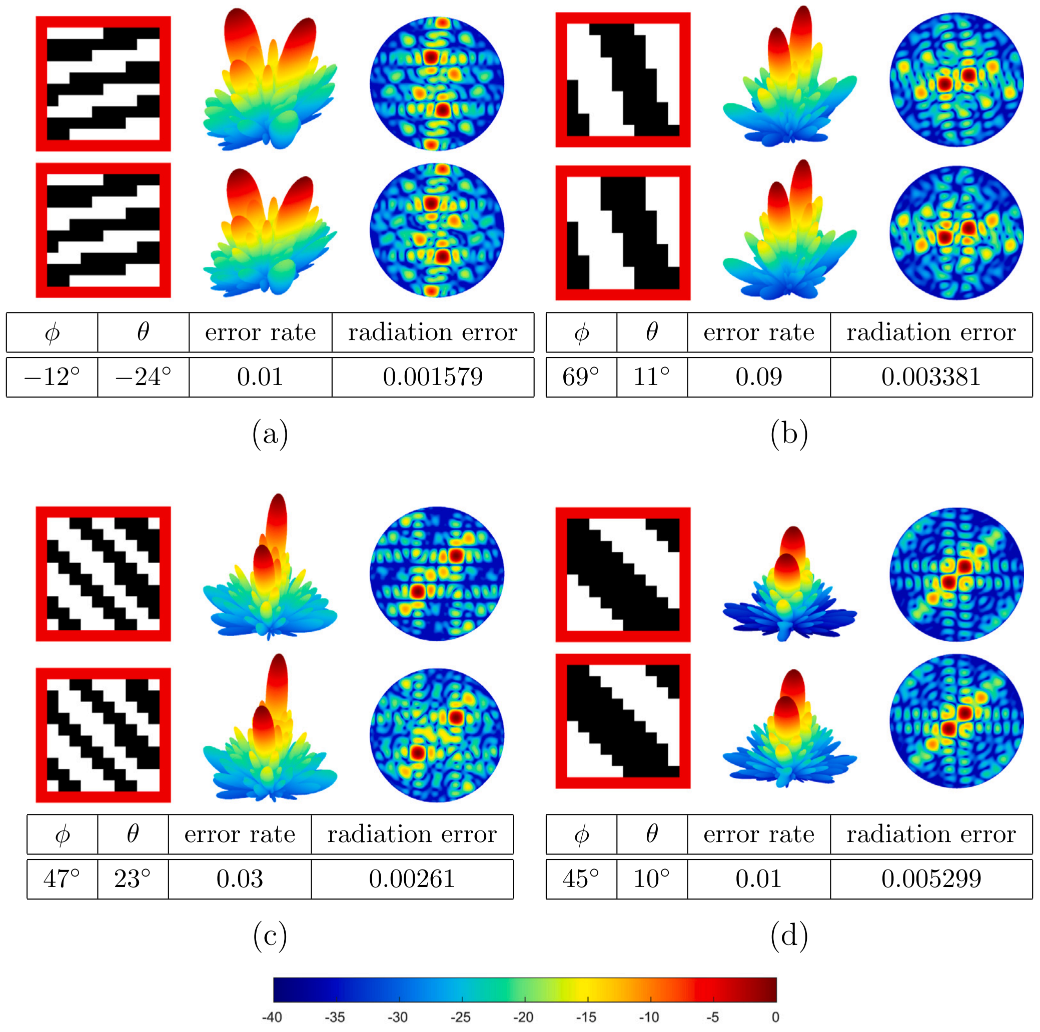
While the experiments discussed in the previous paragraph were intended to characterize and assess the performance of our network architecture (as well as its generalizing capabilities) in a standardized and easily comparable way, we are also interested in the network's performance for the whole range of beam orientations. To more appropriately test this, we run a parallel set of experiments for each scenario but without cross-validation, conducting ten different trainings using the whole dataset both at training and validation time. Results for these experiments are summarized in Table 2. These experiments consistently outperform cross-validation experiments. This is expected since these experiments ask the networks to just model and remember the relationship between beam orientation and grid patterns, while in the case of cross-validation, we are also asking the networks to generalize to validation samples unseen during training. However, performance gains are not uniform: they are much more pronounced for DCT compared to raw binary and for larger networks compared to smaller ones. Curiously enough, the relative performance of BCE vs. MSE in raw binary scenarios remains roughly the same, even if the effect size is small. Metric dispersion is also significantly decreased compared to cross-validation experiments (especially for divide-and-conquer scenarios).

We also show in Figs. 10 and 11 a summary of the evolution of the best mean error rate and the best mean radiation error for the configuration in the last rows in Tables 1 and 2, respectively (the trends are broadly similar for other configurations, although the curves are less dispersed and the outliers less pronounced in configurations with a single network instead of two). Logarithmic scales on the Y axes

[1] Pretrained models and source code to run them are available at https://github.com/icai-uma/learning-to-shape-beams-using-a-neural-network-to-control-a-beamforming-antenna

Table 2
Tabulated results for experiments with full dataset for training and validation. Best values in bold. N is the parameter for channel depth, n is the number of networks (over 1 for divide-and-conquer).

Results for experiments with full dataset for training and validation							
Scenario config				Bit count stats		Radiation error stats	
N	Output	Loss	n	Mean	Std	Mean	Std
25	BIN	BCE	1	9.02E−02	±1.32E−03	4.38E−02	±5.79E−04
25	BIN	MSE	1	8.32E−02	±9.53E−04	4.08E−02	±1.12E−03
25	DCT	MSE	1	7.03E−02	±1.06E−03	3.51E−02	±7.24E−04
25	BIN	BCE	2	5.27E−02	±1.01E−03	3.39E−02	±1.26E−03
25	BIN	MSE	2	4.91E−02	±1.04E−03	3.19E−02	±1.10E−03
25	DCT	MSE	2	4.12E−02	±8.31E−04	2.14E−02	±1.18E−03
50	BIN	BCE	1	5.04E−02	±1.47E−03	3.40E−02	±1.22E−03
50	BIN	MSE	1	4.66E−02	±1.64E−03	3.06E−02	±2.30E−03
50	DCT	MSE	1	3.85E−02	±1.24E−03	2.18E−02	±6.87E−04
50	BIN	BCE	2	2.42E−02	±1.29E−03	1.97E−02	±2.78E−03
50	BIN	MSE	2	2.42E−02	±7.15E−04	2.05E−02	±1.95E−03
50	DCT	MSE	2	**1.77E−02**	±1.17E−03	**1.07E−02**	±1.07E−03



(a)

ϕ	θ	error rate	radiation error
−12°	−24°	0.01	0.001579

(b)

ϕ	θ	error rate	radiation error
69°	11°	0.09	0.003381

(c)

ϕ	θ	error rate	radiation error
47°	23°	0.03	0.00261

(d)

ϕ	θ	error rate	radiation error
45°	10°	0.01	0.005299

Fig. 9. Comparison between ground truth and four predicted RIS patterns from the first cross-validation fold for the configuration in the first row of Table 1. Four samples are shown, from (a) to (d). For each one, the upper row shows the ground truth for the RIS configuration, and the lower row shows the best prediction after training. Each row shows the corresponding RIS pattern and its associated radiation pattern, shown both in 3D and U-V views. All 3D and U-V plots have the same color scheme (the colorbar is at the bottom), showing the radiance power in each direction in decibels. All 3D plots have the same viewpoint: 37.5° azimuth, 30° elevation. For each sample, the ϕ and θ angles corresponding to the ground truth RIS pattern are shown, as well as the quantitative differences between ground truth and prediction, expressed both as error rate (ratio of different cells) and as radiation error (ratio of differences in radiance power around the surface). (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

are the same in both figures, which makes it easier to compare the results. Error rates steadily improve in all training runs, with no signs of overfitting. Still, improvements also get slower and slower (it was considered infeasible to run all 240 trainings for significantly more epochs to investigate when overfitting begins for different configurations). However, the curves for the radiation error (the primary metric

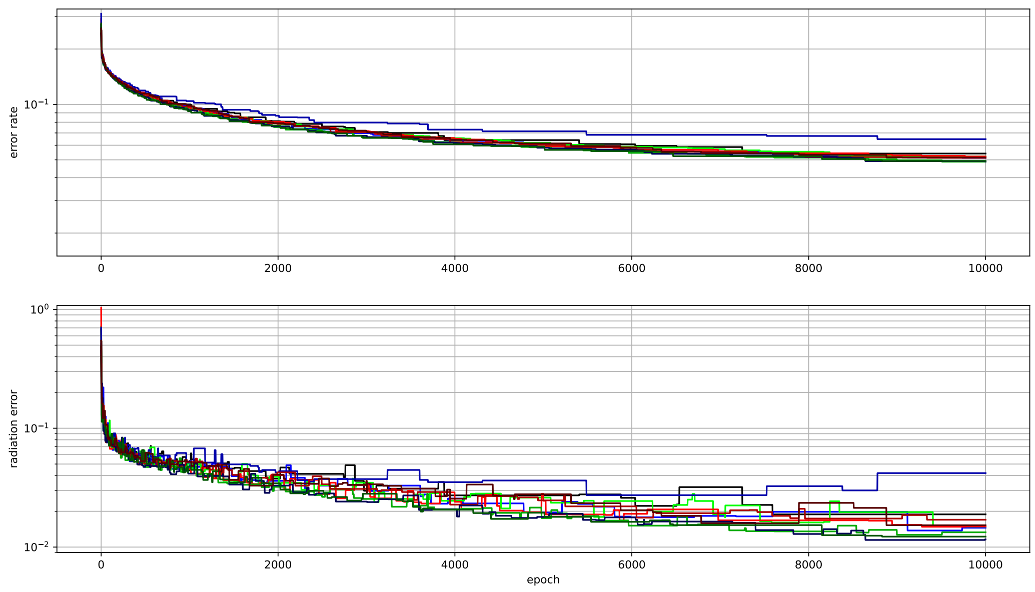


Fig. 10. Summary of the evolution of the best mean error rate and the best mean radiation error during training for ten folds (training with 10-fold cross-validation) with the configuration for the network architecture in the last row in Table 1.

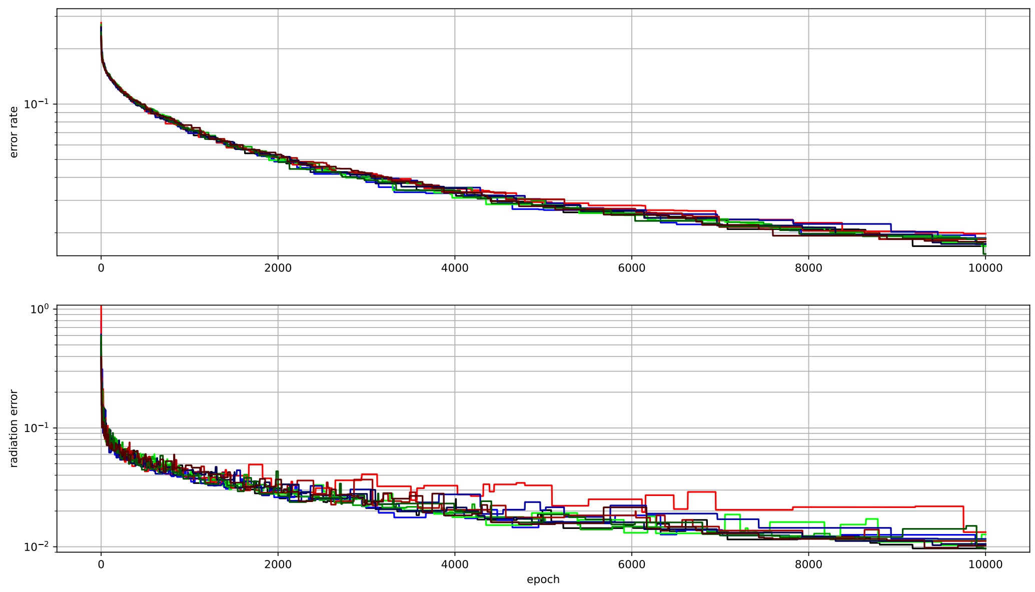


Fig. 11. Summary of the evolution of the best mean error rate and the best mean radiation error during training for ten training runs (validated with the whole dataset) with the configuration for the network architecture in the last row in Table 2.

to measure the performance of the networks) are significantly different, with a much more pronounced elbow (very steep improvements in the first few hundred epochs, followed by prolonged improvements for the rest of the training). Reducing the training time has relatively little effect on this metric.

Since the best results are achieved by configurations where the network is trained to predict the DCT of the RIS configuration pattern, and DCT quantization is typically used for lossy image compression, we also perform an additional experiment: the 10×10 values of the DCT of each RIS configuration pattern are quantized into 1-bit values[2]: since the maximum DCT value across all the dataset is 10 and the minimum one is -9.8, all DCT values are quantized into either -5 or 5. For each sample, after computing the inverse DCT, the resulting values are rounded (since the RIS elements can be only in one of two possible states). Finally, the mean quantization error (with respect to the ground truth) is computed to be 0.243 for all values in RIS configurations patterns across all samples in the dataset.

4. Discussion and conclusions

Despite significant advancements in Reconfigurable Intelligent Surfaces (RIS), the practical implementation of precise real-time configurations for electromagnetic beams remains a considerable challenge.

[2] To be clear, this is just a study of the quantization of the DCT of the ground truth. The output of our proposed networks already undergoes a quantization step where the values are snapped to either 0 or 1, since the outputs are used to configure RIS elements that can be only in one of two possible states. In the case of network architectures that predict the DCT of the configuration pattern, this snapping step happens *after* computing the inverse DCT of the output.

Within this domain, traditional approaches employ non-linear optimization algorithms, such as genetic algorithms and particle swarm optimization, to determine optimal codes that define the desired parameters of the electromagnetic beam on programmable surfaces. While these methods have proven effective for optimization, their computational complexity poses significant challenges for real-time implementation. Our research employs a specifically designed convolutional neural network (CNN) to efficiently compute the codes necessary for actuator elements, yielding consistent and promising results on a 1-bit meta-programmable surface. The use of such models is grounded in their ability to learn and model complex non-linear patterns present in the data, enabling efficient problem-solving compared to traditional methods with high computational complexity.

Unlike traditional approaches that employ non-linear optimization algorithms with high computational complexity, our proposal integrates Deep Learning as an effective tool to determine optimal codes for defining the desired beam parameters. Consequently, this method emphasizes efficiency and implementation speed. This focus on practical implementability not only enhances the broader applicability of our approach but also positions our research as a significant step toward effectively realizing RIS in real-world scenarios.

Regarding our objective of designing a neural network architecture to encode grid patterns for the RIS, we have analyzed a range of scenarios and observed clear trends in the mean radiation errors across them. However, this does not mean that any concrete instance of our network architecture with better scores in that metric is definitely better than others, as they reflect different trade-offs:

- While smaller networks generally exhibit higher radiation errors, they are more suitable for deployment on resource-constrained devices due to their simplicity and faster execution.
- The same applies to architectures with DCT output: although they achieve lower radiation errors, they require a non-trivial postprocessing step that may be difficult to implement on constrained hardware.
- Employing two networks with a divide-and-conquer strategy can yield performance gains similar to increasing channel depth, but at the expense of consistency in output quality.

With respect to DCT-based network configurations, we anticipated significant improvements over configurations with direct Boolean output. For DCT output, most values are near zero, with only a few being substantial. The results are indeed better for DCT output, but only marginally. We hypothesize that this is due to an inherent trade-off: with Boolean outputs, the network must learn numerous parallel, highly diverse mappings between input angles and binary 0–1 values. However, since only two output values are involved, each is relatively simple to learn and does not require precise control. In contrast, DCT outputs, though sparse, demand much finer control to accurately reconstruct the Boolean output after the inverse DCT step.

In multiple qualitative experiments, we have determined that beam quality and orientation are sufficiently accurate (relative to ground truth) when the radiation error is below 0.05. Therefore, on average, our network architecture meets our baseline quality criteria in nearly all scenarios (the exceptions being cross-validated BCE loss for small networks, rows 1 and 4 in Table 1). Nevertheless, by characterizing the radiation error across different scenarios, we can select the most suitable network architecture instance for each use case.

We evaluated a range of architectures using networks of two distinct sizes. The size and complexity of a neural network are typically quantified by the number of trainable parameters. As discussed in Section 3, our networks have approximately 0.5 million parameters when $N = 25$ and approximately 1.8 million when $N = 50$. Accordingly, the more complex models, which use a divide-and-conquer strategy with two networks and $N = 50$, comprise around 3.6 million parameters in total. The most direct comparison can be made with the work of Shan et al. [18], who used a slightly modified VGG16 to generate RIS configuration patterns (instead of the standard one-hot image classification output). VGG16 is significantly larger than our architectures, with around 138 million parameters. This dramatic reduction in model size is feasible because VGG16 is tailored for image analysis (its input being an image) and is designed for a RIS with fewer active elements (10×10 in our case versus 48×48 in theirs). By relying solely on the beam's orientation angles as inputs, we are able to design a considerably streamlined network. Other works cited in Section 1 are less directly comparable, as they apply neural networks to different problems or on different types of programmable surfaces. For instance, Li et al. [20] addresses a programmable surface with only 16 active elements (compared to our 100), using physics-inspired neural units to solve a different problem—estimating a discretized radiation pattern given a particular configuration. Similarly, both Shi et al. [22] and Fu et al. [19] employ deep learning to design 4×4 patches of programmable metasurfaces targeting specific radiation patterns.

4.1. Future work

There are many avenues for future work. We can test our network architecture with reconfigurable surfaces with smaller components, which would result in larger grid patterns, and which we expect to be more challenging to learn for neural networks. Likewise, the components of the reconfigurable surface can be engineered to have more than two different configurations, possibly enabling the study of different output formats and loss functions, thus making the grid patterns more complex. Both of these modifications (smaller components and components with more than two configurations) are expected to enable finer control in the shaping of the beam, so while the grid patterns might become more challenging to learn for the neural network (on account of more extensive patterns with more distinct output values), the resulting radiation errors might be reduced.

CRediT authorship contribution statement

Jose David Fernández-Rodríguez: Writing – original draft, Writing – review & editing, Methodology, Visualization, Investigation, Formal Analysis, Software, Validation. **Iván García-Aguilar:** Writing – original draft, Visualization, Writing – review & editing, Resources, Software. **Rafael Marcos Luque-Baena:** Supervision, Methodology, Writing – original draft, Writing – review & editing, Investigation. **Ezequiel López-Rubio:** Validation, Data curation, Supervision, Conceptualization, Formal analysis, Investigation. **Marcos Baena-Molina:** Software, Resources, Writing – original draft, Data curation, Investigation. **Juan Francisco Valenzuela-Valdés:** Investigation, Validation, Funding acquisition, Project administration, Supervision, Formal analysis.

Funding sources

This work is partially supported by the Autonomous Government of Andalusia (Spain) under project UMA20-FEDERJA-108, project name Detection, characterization and prognosis value of the non-obstructive coronary disease with deep learning, and also by the Ministry of Science and Innovation of Spain, grant number PID2022-136764OA-I00, project name Automated Detection of Non Lesional Focal Epilepsy by Probabilistic Diffusion Deep Neural Models. It includes funds from the European Regional Development Fund (ERDF). It is also partially supported by the University of Málaga (Spain) under grants B1-2021_20, project name Detection of coronary stenosis using deep learning applied to coronary angiography; B4-2023_13, project name Intelligent Clinical Decision Support System for Non-Obstructive Coronary Artery Disease in Coronarographies; B1-2022_14, project name Detección de trayectorias anómalas de vehículos en cámaras de tráfico, and B1-2023_18, project name Sistema de videovigilancia basado en cámaras itinerantes

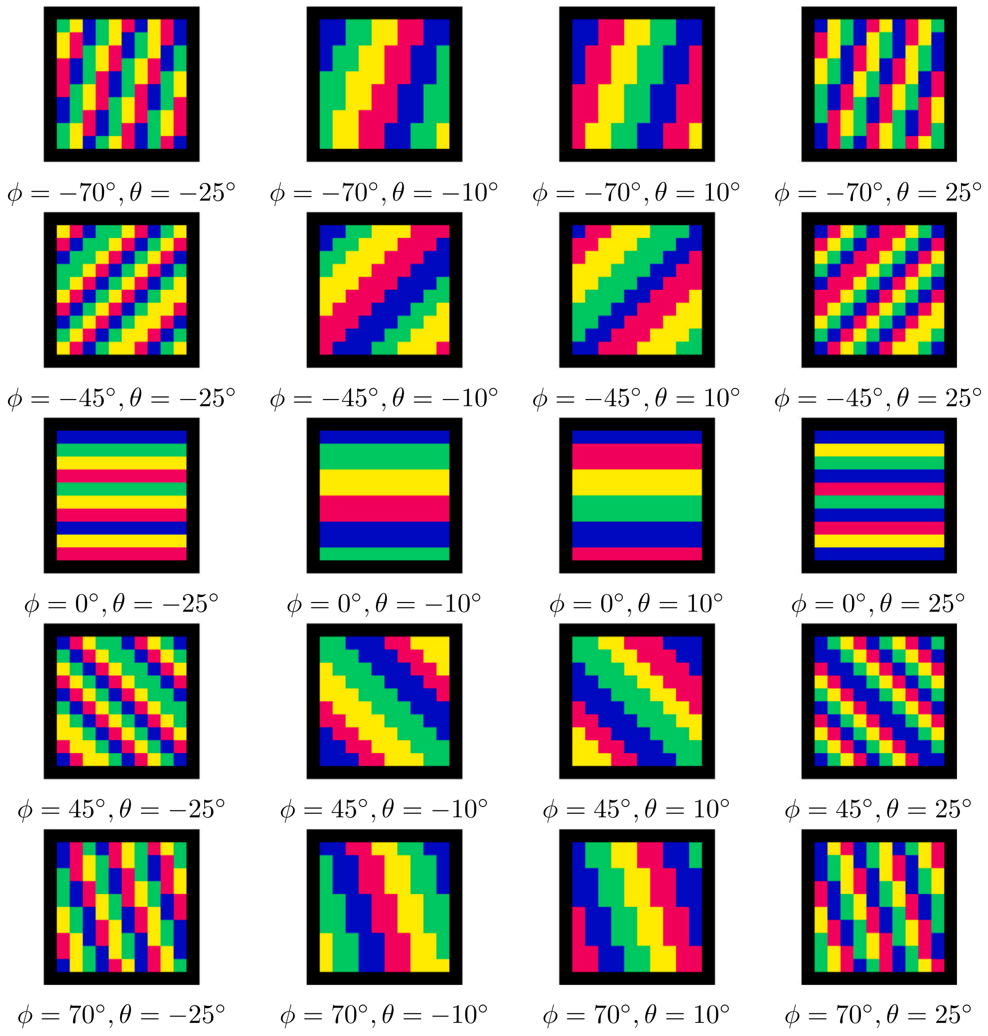

$\phi = -70°, \theta = -25°$
$\phi = -70°, \theta = -10°$
$\phi = -70°, \theta = 10°$
$\phi = -70°, \theta = 25°$
$\phi = -45°, \theta = -25°$
$\phi = -45°, \theta = -10°$
$\phi = -45°, \theta = 10°$
$\phi = -45°, \theta = 25°$
$\phi = 0°, \theta = -25°$
$\phi = 0°, \theta = -10°$
$\phi = 0°, \theta = 10°$
$\phi = 0°, \theta = 25°$
$\phi = 45°, \theta = -25°$
$\phi = 45°, \theta = -10°$
$\phi = 45°, \theta = 10°$
$\phi = 45°, \theta = 25°$
$\phi = 70°, \theta = -25°$
$\phi = 70°, \theta = -10°$
$\phi = 70°, \theta = 10°$
$\phi = 70°, \theta = 25°$

Fig. A.12. Various samples from the 2-bit dataset described in this appendix for a (theoretical) RIS device whose active elements can be configured with four different phase angles: 0° (blue), 90° (green), 180° (yellow) and 270° (red). As in Fig. 8, each sample is a pair of angles ϕ, θ characterizing the orientation of a beam and the associated grid pattern for the RIS to get that beam. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

robotizadas; by the Fundación Unicaja under project PUNI-003_2023, project name Intelligent System to Help the Clinical Diagnosis of Non-Obstructive Coronary Artery Disease in Coronary Angiography; and, by the Ministry of Science and Innovation of Spain, numbers of grants TED2021-129938B-I00, PID2020-112545RBC54, PDC2022-133900-I00 and PDC2023-145862-I00. The authors thankfully acknowledge the computer resources, technical expertise and assistance provided by the SCBI (Supercomputing and Bioinformatics) center of the University of Málaga. They also gratefully acknowledge the support of NVIDIA Corporation with the donation of a RTX A6000 GPU with 48 Gb. The authors also thankfully acknowledge the grant of the Universidad de Málaga, Spain and the Instituto de Investigación Biomédica de Málaga y Plataforma en Nanomedicina-IBIMA Plataforma BIONAND, Spain. Funding for open access charge: Universidad de Málaga / CBUA.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Appendix. Deep learning models for 2-bit RIS devices

The RIS device described in this work has 10 × 10 active elements. Each one of these elements can be configured with a phase angle of either 0° or 180°, i.e., it is a 1-bit RIS. A 2-bit version of the RIS (i.e., a 10 × 10 RIS whose elements can be configured for four different phase angles: 0°, 90°, 180°, and 270°) is theoretically possible, but it is not clear whether it would result in substantially better results given the physical parameters of the device. However, it is possible to explore how to extend the deep learning setup presented in this work to 2-bit RIS configurations. This appendix explores how to do it.

The optimization process described in Section 2.2 can be modified to generate configuration patterns suited to shape beams with a 10 × 10 array of 2-bit RIS elements, i.e. elements able to be configured with four different phase angles, as described in the preceding paragraph. While we have not built such a 2-bit RIS, we can compute configuration patterns to shape the beams for that theoretical device, and we have even done so, as a theoretical exercise, generating a dataset of configuration patterns for a hypothetical 2-bit version of our 10 × 10 RIS device. For maximum comparability between the 1-bit and 2-bit versions, this 2-bit dataset has the same amount of samples as the one for the 1-bit version, with the same input angles encoding the same orientation for each possible beam, but with outputs encoding the 10 × 10 configuration patterns required to shape each beam using 2-bit RIS elements, able to be configured with four different phase angles. Fig. A.12 shows samples from this 2-bit dataset.

Another issue that requires adaptation is that of equivalent RIS patterns: for both 1-bit and 2-bit patterns, the beam is shaped by the

Table A.3
Tabulated results for 10-fold cross-validation experiments for the 2-bit dataset, with the same presentation as in Table 1.

Partial results for 10-fold cross-validation								
Scenario config					Error rate stats		Radiation error stats	
bits	*N*	Output	Loss	*n*	Mean	Std	Mean	Std
2	25	4 Values	MSE	1	1.94E−01	±1.4E−03	3.46E−02	±7.12E−04
2	25	DCT	MSE	1	1.94E−01	±1.48E−03	4.33E−02	±5.53E−04
2	50	4 Values	MSE	1	1.54E−01	±2.21E−03	2.62E−02	±9.1E−04
2	50	DCT	MSE	1	1.58E−01	±1.77E−03	3.54E−02	±1.05E−03

phase differences between RIS elements. In the 1-bit case, this means that each configuration pattern and its Boolean complement induce the same electromagnetic radiation pattern (in practical terms, the same resulting beam). For 2-bit patterns, each RIS element can be configured for any of four different phase angles: 0°, 90°, 180° and 270°. If the phase angles of RIS elements in a 2-bit configuration pattern are all shifted by 90°, the shifted pattern also shapes the radiation pattern into the same beam as the original configuration. Two additional 90° shifts can be applied (applying a fourth one would result in the original configuration pattern), for a total of four different 2-bit configuration patterns, all of them resulting in the same shaped beam. Therefore, to extend the training setup described in Section 2 from 1-bit to 2-bit, the loss for each sample has to be computed with respect to all four equivalent versions of the output pattern, and take the minimum out of these (in the same way as in 1-bit, where the minimum between the two equivalent output patterns is taken).

Some considerations about network architecture should remain the same for better comparability. For example, the architecture of the neural network can remain the same, with the same amount of layers, the same values for the parameter N (the one governing the number of channels in each layers; see Section 2.3) and the same per-layer multipliers to decide the amount of channels in each convolutional layer (except in the last one, whose amount of channels depends on the output encoding, as described below). Therefore, training may be done both with lighter ($N = 25$) and heavier ($N = 50$) networks, to compare how the performance varies with N in both the 1-bit and the 2-bit cases.

The next issue is that of output encodings, and what kind of loss function to use with each one:

- The DCT case would be completely equivalent in 1-bit and 2-bit: for each possible beam, the output from the neural network would also be a 10 × 10 matrix with the DCT transform of the RIS configuration pattern, using MSE loss in both cases.
- The configuration for each RIS element can also be encoded as a specific value uniformly spaced on the 0–1 range, using MSE loss to train the neural network. For 2-bit RIS elements, four uniformly spaced values can be used. Please note that while BCE loss can be applied for the case of 1-bit RIS elements, this is, of course, no longer the case for 2-bit ones.

However, once we have more than two possible values per RIS element, other encodings become possible, such as:

- One-hot encoding trained with cross-entropy loss. One-hot encoding usually yields very good results for classification tasks, and it is worth testing whether it is also good in this case. For 2-bit RIS elements, one-hot encoding entails using four output channels for each RIS element (thus, the output would be a 4 × 10x10 tensor, with 4 output values for each of the 10 × 10 active elements). Each one of the 4 channels represents one possible value, and all output channels must be 0, except the one corresponding to the desired output in each training sample, which must be 1. Therefore, the four phase angles may be encoded for the 2-bit RIS elements as follows: 0° as 0001, 90° as 0010, 180° as 0100, and 270° as 1000. Of course, at inference time, output values will not be perfectly 0 or 1, so the maximum of the channels is taken as the intended result. Note that one-hot encodings have also been successfully used in other tasks where the network outputs a large number of different one-hot-encoded values, such as detecting the per-detected-object class in the YOLOv3 architecture.
- Binary encoding trained with either BCE or MSE loss: The output of the network can be a 2 × 10 × 10 tensor, such that there are two channels for each RIS element, and each channel is expected to output either 0 or 1. Therefore, the phase angles for the RIS elements may be encoded directly using 2 bits: 0° as 00, 90° as 01, 180° as 10, and 270° as 11.

There are practical considerations about the feasibility of testing all possible training setups described in Section 3.3 for the 2-bit case. The issue is the computational cost: during training, most of the computational cost is required in the backpropagation step. Therefore, computing the loss for four different ground truth configurations (instead of two as in the 1-bit case, as described above) roughly doubles the required computational effort. Because of this, training times are substantially longer in this case, and replicating all possible combinations of training setups (N parameter, divide-and-conquer strategy, output encoding) would require an extremely long time, specially considering that the 2-bit version enables even more different output encodings whose performance is worthwhile to explore. In practice, it seems best to drop testing of the divide-and-conquer strategy for the 2-bit case, since it would require excessively long training times.

We have done only a partial, very preliminary study, because of the higher computational load associated with training neural networks for the 2-bit case, as well as the fact that the 2-bit RIS device is purely theoretical. In this preliminary study, we have studied only the DCT case and the one where each output channel encodes 4 uniformly spaced values, both for N = 25 and N = 50, but not for the divide-and-conquer case (n = 2), because of the excessive computational effort required in that case (we have not studied the performance of the one-hot encoding and the 2-bit-binary encodig, either). Tables A.3 and A.4 show the partial results both with 10-fold cross-validation and with the full dataset for 10 trainings, organized the same way as Tables 1 and 2. Fig. A.13 shows a summary of the evolution of the error rate and the radiation error during training in the 10-fold cross-validation case for the configuration using 4 uniformly space values, with N = 25, organized in the same way as Fig. 10. Finally, Fig. A.14 shows two comparisons between the ground truth (first row) and the predictions (second row) in the first fold of the 10-fold cross-validation for the configuration using 4 uniformly spaced values, with N = 25 (it is organized in the same way as Fig. 9).

Trainings with the configuration using 4 uniformly spaced values in 2 bits should be considered roughly equivalent to trainings with BIN/MSE loss in 1 bit. As it can be seen, comparing the partial results for 2-bit with the results in Tables 1 and 2 for 1-bit, DCT is no longer clearly ahead of other network configurations, possibly because the higher number of possible states of the RIS elements makes it necessary to predict the DCTs with higher precision, and relatively small errors in the DCT prediction can lead to greatly increased errors in the resulting RIS configuration patterns (this is also in agreement with the finding that, for the 2-bit case, DCT presents greater standard deviations in the results than the configuration using 4 uniformly space values). Radiation error values are significantly improved with respect to the respective 1-bit cases. More interestingly, even if the radiation error goes down, the error rate goes up. This makes sense, since each RIS element only has two possible states in the 1-bit case, but 4 in the 2-bit case, so there is more room for RIS elements with erroneous states, even if the overall radiation error is decreased because of the finer control in beam shaping afforded by the additional states in the 2-bit RIS elements. Another consequence of using 2-bit instead of 1-bit is that the variability between individual trainings is greatly reduced,

Table A.4
Tabulated results for experiments with full dataset for training and validation for the 2-bit dataset, with the same presentation as in Table 2.

Partial results for experiments with full dataset for training and validation								
Scenario config					Error rate stats		Radiation error stats	
bits	*N*	Output	Loss	*n*	Mean	Std	Mean	Std
2	25	4 Values	MSE	1	1.79E−01	±1.33E−03	3.23E−02	±7.38E−04
2	25	DCT	MSE	1	1.77E−01	±1.93E−03	4.01E−02	±1.1E−03
2	50	4 Values	MSE	1	1.3E−01	±1.56E−03	2.21E−02	±7.25E−04
2	50	DCT	MSE	1	1.29E−01	±2.78E−03	2.93E−02	±6.29E−04

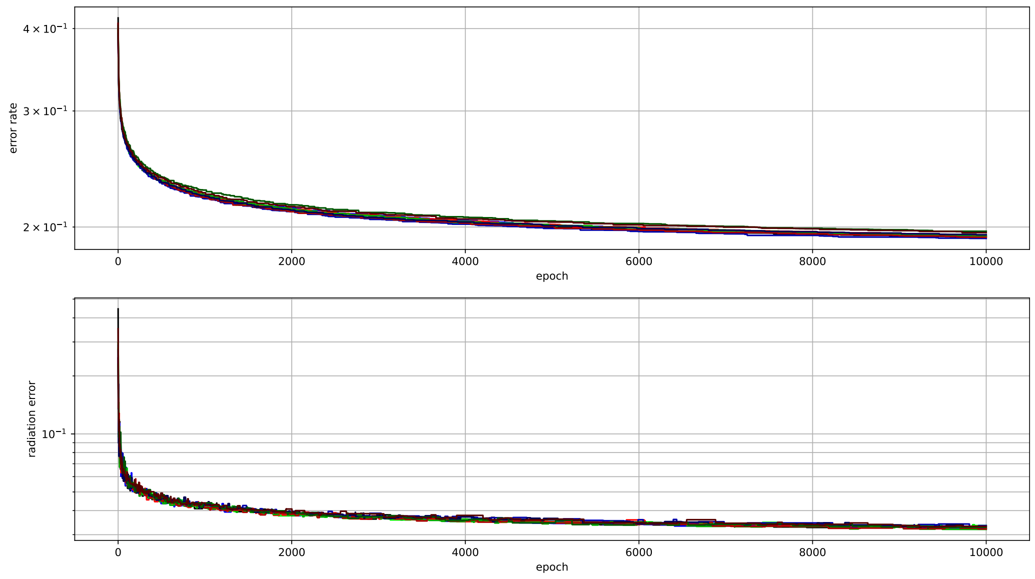
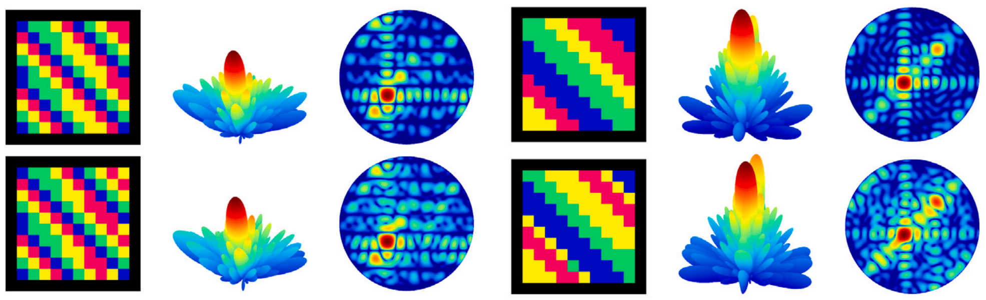


Fig. A.13. Summary of the evolution of the best mean error rate and the best mean radiation error during training for ten folds (training with 10-fold cross-validation) with the configuration for the network architecture in the first row in Table A.3.



ϕ	θ	error rate	radiation error
47°	23°	0.07	0.002021

ϕ	θ	error rate	radiation error
45°	10°	0.25	0.025714

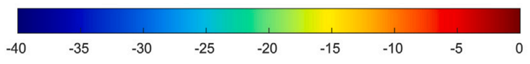


Fig. A.14. Comparison between ground truth and two predicted RIS patterns from the first cross-validation fold for the configuration in the first row of Table A.3. Please see the caption of Fig. 9 for details.

with significantly decreased standard deviations, possibly because the increased amount of states in the RIS elements also makes the output of the system smoother, enabling smoother paths (with less random fluctuations) through the optimization process during the trainings.

Data availability

Data will be made available on request.

References

[1] Q. Wu, R. Zhang, Towards smart and reconfigurable environment: Intelligent reflecting surface aided wireless network, IEEE Commun. Mag. 58 (1) (2020) 106–112, http://dx.doi.org/10.1109/MCOM.001.1900107.

[2] M. Di Renzo, A. Zappone, M. Debbah, M.-S. Alouini, C. Yuen, J. de Rosny, S. Tretyakov, Smart radio environments empowered by reconfigurable intelligent surfaces: How it works, state of research, and the road ahead, IEEE J. Sel. Areas Commun. 38 (11) (2020) 2450–2525, http://dx.doi.org/10.1109/JSAC.2020.3007211.

[3] Z. Zhang, L. Dai, A joint precoding framework for wideband reconfigurable intelligent surface-aided cell-free network, IEEE Trans. Signal Process. 69 (2021) 4085–4101, http://dx.doi.org/10.1109/TSP.2021.3088755.

[4] C. Huang, A. Zappone, G.C. Alexandropoulos, M. Debbah, C. Yuen, Reconfigurable intelligent surfaces for energy efficiency in wireless communication, IEEE Trans. Wirel. Commun. 18 (8) (2019) 4157–4170, http://dx.doi.org/10.1109/TWC.2019.2922609.

[5] Q. Wu, R. Zhang, Intelligent reflecting surface enhanced wireless network via joint active and passive beamforming, IEEE Trans. Wirel. Commun. 18 (11) (2019) 5394–5409, http://dx.doi.org/10.1109/TWC.2019.2936025.

[6] M. Fu, Y. Zhou, Y. Shi, K.B. Letaief, Reconfigurable intelligent surface empowered downlink non-orthogonal multiple access, IEEE Trans. Commun. 69 (6) (2021) 3802–3817, http://dx.doi.org/10.1109/TCOMM.2021.3066587.

[7] S. Abeywickrama, R. Zhang, C. Yuen, Intelligent reflecting surface: Practical phase shift model and beamforming optimization, in: ICC 2020 - 2020 IEEE International Conference on Communications, ICC, 2020, pp. 1–6, http://dx.doi.org/10.1109/ICC40277.2020.9148961.

[8] M. Di Renzo, K. Ntontin, J. Song, F.H. Danufane, X. Qian, F. Lazarakis, J. De Rosny, D.-T. Phan-Huy, O. Simeone, R. Zhang, M. Debbah, G. Lerosey, M. Fink, S. Tretyakov, S. Shamai, Reconfigurable intelligent surfaces vs. relaying: Differences, similarities, and performance comparison, IEEE Open J. Commun. Soc. 1 (2020) 798–807, http://dx.doi.org/10.1109/OJCOMS.2020.3002955.

[9] E. Björnson, Ö. Özdogan, E.G. Larsson, Intelligent reflecting surface versus decode-and-forward: How large surfaces are needed to beat relaying? IEEE Wirel. Commun. Lett. 9 (2) (2020) 244–248, http://dx.doi.org/10.1109/LWC.2019.2950624.

[10] Z. Abdullah, G. Chen, S. Lambotharan, J.A. Chambers, A hybrid relay and intelligent reflecting surface network and its ergodic performance analysis, IEEE Wirel. Commun. Lett. 9 (10) (2020) 1653–1657, http://dx.doi.org/10.1109/LWC.2020.2999918.

[11] T. Isernia, A. Massa, A.F. Morabito, P. Rocca, On the optimal synthesis of phase-only reconfigurable antenna arrays, in: Proceedings of the 5th European Conference on Antennas and Propagation, EUCAP, 2011, pp. 2074–2077.

[12] D.E. Goldberg, J.H. Holland, Genetic algorithms and machine learning, Mach. Learn. 3 (2/3) (1988) 95–99, http://dx.doi.org/10.1023/a:1022602019183.

[13] J. Robinson, Y. Rahmat-Samii, Particle swarm optimization in electromagnetics, IEEE Trans. Antennas and Propagation 52 (2) (2004) 397–407, http://dx.doi.org/10.1109/TAP.2004.823969.

[14] S. Caorsi, M. Donelli, A. Lommi, A. Massa, A real-time approach to array control based on a learned genetic algorithm, Microw. Opt. Technol. Lett. 36 (4) (2003) 235–238, http://dx.doi.org/10.1002/mop.10731, URL https://onlinelibrary.wiley.com/doi/abs/10.1002/mop.10731.

[15] I. García-Aguilar, J. García-González, R.M. Luque-Baena, E. López-Rubio, Automated labeling of training data for improved object detection in traffic videos by fine-tuned deep convolutional neural networks, Pattern Recognit. Lett. 167 (2023) 45–52, http://dx.doi.org/10.1016/j.patrec.2023.01.015.

[16] I. García-Aguilar, J. García-González, R.M. Luque-Baena, E. López-Rubio, Object detection in traffic videos: an optimized approach using super-resolution and maximal clique algorithm, Neural Comput. Appl. 35 (26) (2023) 18999–19013, http://dx.doi.org/10.1007/s00521-023-08741-4.

[17] K. Simonyan, A. Zisserman, Very deep convolutional networks for large-scale image recognition, in: Y. Bengio, Y. LeCun (Eds.), 3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings, 2015, pp. 1–14, URL http://arxiv.org/abs/1409.1556.

[18] T. Shan, X. Pan, M. Li, S. Xu, F. Yang, Coding programmable metasurfaces based on deep learning techniques, IEEE J. Emerg. Sel. Top. Circuits Syst. 10 (1) (2020) 114–125, http://dx.doi.org/10.1109/JETCAS.2020.2972764.

[19] J. Fu, Z. Yang, M. Liu, H. Zhang, Y. Zhang, Highly-efficient design method for coding metasurfaces based on deep learning, Opt. Commun. 529 (2023) 129043, http://dx.doi.org/10.1016/j.optcom.2022.129043.

[20] S. Li, Z. Liu, S. Fu, Y. Wang, F. Xu, Intelligent beamforming via physics-inspired neural networks on programmable metasurface, IEEE Trans. Antennas and Propagation 70 (6) (2022) 4589–4599, http://dx.doi.org/10.1109/TAP.2022.3140891.

[21] T. Qiu, X. Shi, J. Wang, Y. Li, S. Qu, Q. Cheng, T. Cui, S. Sui, Deep learning: A rapid and efficient route to automatic metasurface design, Adv. Sci. 6 (12) (2019) 1900128, http://dx.doi.org/10.1002/advs.201900128.

[22] X. Shi, T. Qiu, J. Wang, X. Zhao, S. Qu, Metasurface inverse design using machine learning approaches, J. Phys. D: Appl. Phys. 53 (27) (2020) http://dx.doi.org/10.1088/1361-6463/ab8036.

[23] V.N. Vapnik, The support vector method, in: International Conference on Artificial Neural Networks, Springer, 1997, pp. 261–271.

[24] D. Karaboga, B. Basturk, A powerful and efficient algorithm for numerical function optimization: artificial bee colony (ABC) algorithm, J. Global Optim. 39 (2007) 459–471.

[25] C. Pan, H. Ren, K. Wang, J.F. Kolb, M. Elkashlan, M. Chen, M. Di Renzo, Y. Hao, J. Wang, A.L. Swindlehurst, X. You, L. Hanzo, Reconfigurable intelligent surfaces for 6G systems: Principles, applications, and research directions, IEEE Commun. Mag. 59 (6) (2021) 14–20, http://dx.doi.org/10.1109/MCOM.001.2001076.

[26] P. Zhang, J. Zhang, H. Xiao, H. Du, D. Niyato, B. Ai, RIS-aided 6G communication system with accurate traceable user mobility, IEEE Trans. Veh. Technol. 72 (2) (2023) 2718–2722, http://dx.doi.org/10.1109/TVT.2022.3214818.

[27] M. Baena-Molina, A. Palomares-Caballero, G. Martinez-Garcia, R. Padial-Allue, P. Padilla, J.F. Valenzuela-Valdes, 1-bit RIS unit cell with mechanical reconfiguration at 28 GHz, in: 2024 18th European Conference on Antennas and Propagation, EuCAP, 2024, pp. 1–5, http://dx.doi.org/10.23919/EuCAP60739.2024.10500936.

[28] M. Baena-Molina, A. Palomares-Caballero, G. Martinez-Garcia, J.E. Galeote-Cazorla, A. Ramirez-Arroyo, J.F. Valenzuela-Valdes, 1-bit mechanically reconfigurable metasurface as a beam splitter for indoor environments at 28 GHz, IEEE Antennas Wirel. Propag. Lett. (2025) 1–5, http://dx.doi.org/10.1109/LAWP.2025.3554271.

[29] J.M. Pérez-Bravo, J.A. Rodríguez-Rodríguez, J. García-González, M.A. Molina-Cabello, K. Thurnhofer-Hemsi, E. López-Rubio, Encoding generative adversarial networks for defense against image classification attacks, in: International Work-Conference on the Interplay Between Natural and Artificial Computation, Springer, 2022, pp. 163–172.

[30] S. Elfwing, E. Uchibe, K. Doya, Sigmoid-weighted linear units for neural network function approximation in reinforcement learning, Neural Netw. 107 (2018) 3–11.

Jose David Fernández-Rodríguez received his B.S., M.Sc. and Ph.D. in Computer Science from the University of Málaga (Spain) in 2007, 2009 and 2012, respectively. He has worked on evolutionary algorithms, computational biology, algorithmic composition, 3D printing, and deep learning. He is an associate professor at the University of Málaga. His current interests include computational intelligence, machine learning, deep learning, and image and video processing and analysis.

Iván García Aguilar received his B.Sc. degree in Computer Science,M.Sc. in Software Engineering and Artificial Intelligence, and Ph.D. degree in Computer Engineering from the University of Málaga, Spain, in 2019, 2020 and 2023, respectively. He joined the Department of Computer Languages and Computer Science at the University of Málaga in 2020 and the Department of Computer and Telematic Systems Engineering at the University of Mérida in 2023, where he is currently working as a researcher. His technical interests are deep learning, pattern recognition, image processing, and computer vision.

Rafael Marcos Luque-Baena received the M.S. and Ph.D. degrees in Computer Engineering from the University of Málaga, Spain, in 2007 and 2012, respectively. He moved to Mérida, Spain in 2013, as a lecturer at the Department of Computer Engineering in the Centro Universitario de Mérida, University of Extremadura. Currently, he has come back to the University of Málaga (2016) and has a teaching position as an associate professor in the Department of Languages and Computer Science. He also keeps pursuing research activities in collaboration with other Universities. His current research interests include visual surveillance, image/video processing, neural networks and pattern recognition.

Ezequiel López-Rubio received his M.Sc. and Ph.D. (honors) degrees in Computer Engineering from the University of Málaga, Spain, in 1999 and 2002, respectively. He also received his M.Sc. in Social and Cultural Anthropology and his Ph.D. in Philosophy of Science from the Spanish Distance Education University, Madrid, Spain, in 2010 and 2020, respectively. He joined the Department of Computer Languages and Computer Science, University of Málaga, in 2000, where he is currently a Full Professor of Computer Science and Artificial Intelligence. His technical interests are in deep learning, pattern recognition, image processing and computer vision.

Marcos Baena-Molina was born in Granada, Spain, in 1999. He received his B.Sc. and M.Sc. degrees in telecommunication engineering from the University of Granada (UGR), Spain, in 2022 and 2024, respectively. Since 2023, he has been pursuing a Ph.D. degree with the Smart Wireless Applications and Technologies (SWAT) Research Group in the Department of Signal Theory, Telematics, and Communications at the University of Granada. His current research interests include the design of reconfigurable intelligent surfaces (RIS), reflectarrays (RA), and antennas for mmWave and 5G/6G communications.

Juan F. Valenzuela-Valdés was born in Marbella, Spain. He received the Telecommunication Engineering degree from the Universidad de Malaga, Malaga, Spain, in 2003, and the Ph.D. degree from the Universidad Politecnica de Cartagena, Cartagena, Spain, in 2008. He joined the Department of Information Technologies and Communications, Universidad Politecnica de Cartagena, in 2004. In 2007, he joined EMITE Ing., Murcia, Spain, as the Head of research. In 2011, he joined the Universidad de Extremadura, Merida, Spain, and in 2015, he joined the Universidad de Granada, where he is currently a Full Professor. He is also the Head of the SWAT Research Group, University of Granada, Granada, Spain, and the Co-Head of the Singular Laboratory of electromagnetic characterization of microwave and millimeter devices and antennas. His publication record comprised of more than 150 publications, including 75 JCR (Journal Citation Reports) indexed articles and seven book chapters. He holds several national and international patents. His current research interests include wireless communications, radio frequency devices, antennas, and propagation. Dr. Valenzuela-Valdes received several prizes, including a National Prize to the Best Ph.D. in mobile communications by Vodafone.