Physics-guided Optimization of Photonic Structures using Denoising Diffusion Probabilistic Models

Dongjin Seo[1†] **Soobin Um**[2†] **Sangbin Lee**[3] **Jong Chul Ye**[2*] **Haejun Chung**[3*]
{dongjin.seo}@yale.edu
{sum,jong.ye}@kaist.ac.kr
{kairus00,haejun}@hanyang.ac.kr

† These authors contributed equally.
* Corresponding author

Abstract

Designing free-form photonic devices is a challenging topic due to the high degree of structural freedom. Here, we present *AdjointDiffusion*, a new algorithm that optimizes photonic structures using adjoint sensitivity analysis and diffusion models. We demonstrate that integrating adjoint gradient values into the denoising process enables the generation of high-performance device structures. Our method can optimize structures with a small number of simulations by incorporating a diffusion model trained on synthetic images that follow fabrication constraints. Compared to conventional algorithms, our approach eliminates the need for intricate binarization and conic filters, overcomes the issue of local optima, and provides a variety of design options. Despite the inherent stochasticity, our algorithm robustly designs high-performance devices and outperforms state-of-the-art nonlinear algorithms.

1 Introduction

Photonics focuses on the manipulation of light waves; designing the appropriate structure of photonic devices plays a key role in achieving effective modulation of light. Inverse design [Molesky et al., 2018] has emerged as a crucial domain within the structural optimization of photonic devices. The field focuses on generating optimal structure that maximizes performance, quantified as a Figure of Merit (FoM). Instead of manually designing geometry based on intuition or trial and error, inverse design algorithms employ optimization techniques to iteratively adjust structural parameters, such as the permittivity (ε) of each structural coordinate.

Adjoint sensitivity analysis [Miller, 2012, Cao et al., 2002] is a kind of inverse design tool that uses two simulations (direct and adjoint) to calculate the FoM gradient with respect to structural parameters. This gradient is referred to as the adjoint gradient. The strength of adjoint sensitivity analysis lies in its simulation efficiency, as it determines the adjoint gradient of every coordinate using only two simulations. Adjoint optimization [Allaire, 2015] refers to a family of gradient-ascent algorithms that utilize the adjoint gradient value. The value is added to the structure coordinate-wise. However, since this method updates each coordinate individually, the generated structure might have numerous coordinates with different values from their neighbors. Such irregularities are undesirable for fabrication, as the presence of numerous small structural components complicates the process and increases the likelihood of fabrication errors. To mitigate this issue, a conic filter, which smooths the structure, is used to ensure a more uniform and manufacturable outcome. In addition, adjoint optimization necessitates binarization since the algorithm generates grayscale structures. This binarization process can unpredictably degrade the FoM. Therefore, meticulous parameter setup is necessary to ensure the final design closely approximates the optimal grayscale solution while

Machine Learning and the Physical Sciences Workshop, NeurIPS 2024.

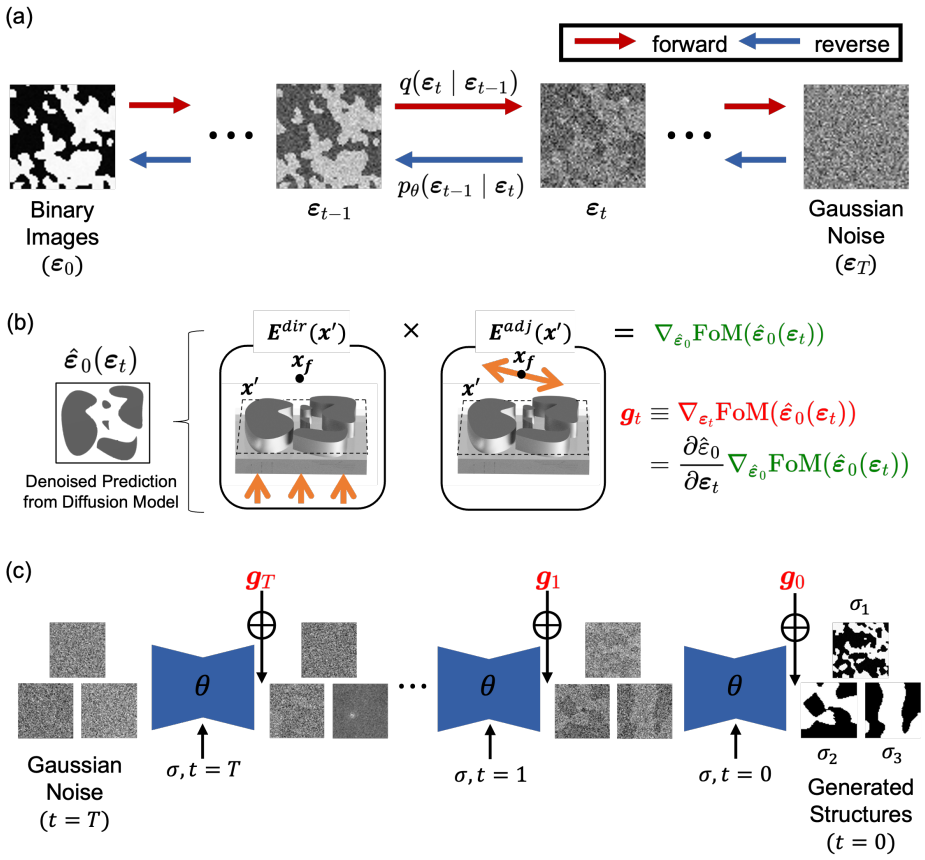

Figure 1: Illustration of the process of integrating adjoint optimization with diffusion models for generating structures. (a) The forward and reverse diffusion process. The forward process (red arrows) starts from binary images ($\hat{\varepsilon}_0$) and adds Gaussian noise until the images are completely noisy. The reverse process (blue arrows) denoises the noisy images step-by-step, using a learned model θ to reconstruct the binary images. (b) Schematics of adjoint sensitivity analysis in our algorithm. The adjoint gradient $\boldsymbol{g}_t$ is calculated for the denoised prediction $\hat{\varepsilon}_0(\varepsilon_t)$by the component-wise product of direct and adjoint fields. (c) The calculated gradient from (b) is added component-wise to the generated structures in the reverse process, where conditional parameters σ and t are applied.

meeting binary constraints. Moreover, gradient-based optimization is inherently prone to local optima issues. Although nonlinear algorithms can address this issue, they might not be suitable for more complex setups.

2 Method

2.1 Problem setup

We set our target problem as a bending waveguide shown in Figure 2. Bending waveguides are electromagnetic waveguides—structures that confine and direct wave propagation— specifically designed to guide waves along a curved or bent path. We utilize Si ($\varepsilon_r = 11.6$) and SiO_2 ($\varepsilon_r = 2.1$) as composing materials of the design region. The incident source is a Gaussian wave with wavelength 1.55 μm. The FoM of a bending waveguide is calculated as conversion efficiency, or $\frac{I_{output}}{I_{input}}$.

2.2 Physics-guided diffusion models

In this work, we introduce a novel physics-guided diffusion model algorithm, *AdjointDiffusion*, which utilizes the capabilities of DDPM [Ho et al., 2020] for the optimization of photonic structures. Our

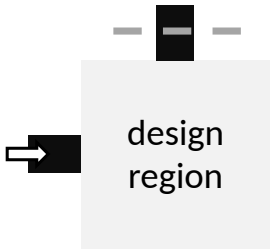

Figure 2: Schematics of the problem setup. The design region has a resolution of 64×64. The incident Gaussian wave propagates from the left waveguide and is induced to the upper waveguide. Both waveguides are composed of Silicon. The boundary condition of the simulation is perfectly matched layer (PML), which absorbs outgoing waves from a computational domain.

method integrates DDPM with physical information by directly applying the adjoint gradient to the inference process. While there have been approaches to integrate diffusion models with inverse design [Zhang et al., 2023, Yang et al., 2024, Vlastelica et al., 2023], to our knowledge, no attempts have been made to combine diffusion models with adjoint gradient values for optimization. The overall process of our algorithm is shown below (see Section B for details):

Dataset construction. We first generate 30K binary grayscale images with 64×64 resolution. Specifically, the pixels of each image were drawn uniformly at random from $\mathcal{U}[0, 1]$ and subsequently discretized to have 0 or 1 by applying a threshold value (i.e., 0.5). Additionally, to incorporate structural properties, we equally divide the dataset into three distinct parts (to produce 3 chunks of 10K images) and applied different Gaussian filters to each chunk. We use standard deviation values of 2, 5 and 8 for the filters.

Model training. On the generated structural data, we train a diffusion model with the ADM framework [Dhariwal and Nichol, 2021]. Specifically, we use 1000 timesteps (i.e., $T = 1000$) with cosine noise schedule [Nichol and Dhariwal, 2021]. For the training objective, we focus on L_{hybrid} proposed in [Nichol and Dhariwal, 2021] The fabrication constraints (i.e., the standard deviations of Gaussian filters) are fed as a conditioning parameter σ. During this stage, the diffusion model learns from the dataset to generate structures suitable for fabrication with respect to given fabrication conditions σ.

Structure generation. To synthesize high-performance photonic structures with enhanced FoM, we incorporate a FoM-maximizing guidance term into the standard sampling process of diffusion models (e.g., ancestral sampling [Ho et al., 2020]). This guidance term is derived through a combination of the pretrained diffusion model and adjoint gradient optimization. Specifically given an intermediate structure $\boldsymbol{\varepsilon}_t$, we predict its posterior mean $\hat{\boldsymbol{\varepsilon}}_0(\boldsymbol{\varepsilon}_t)$ by employing the pretrained diffusion model and Tweedie's formula [Robbins, 1992]. We subsequently calculate the adjoint gradient of the predicted structure $\nabla_{\hat{\boldsymbol{\varepsilon}}_0}\text{FoM}(\hat{\boldsymbol{\varepsilon}}_0(\boldsymbol{\varepsilon}_t))$ with direct and adjoint simulations. Our guidance term g_t is the inner-product of the Jacobian $\frac{\partial \hat{\boldsymbol{\varepsilon}}_0}{\partial \boldsymbol{\varepsilon}_t}$ with the adjoint gradient $\nabla_{\hat{\boldsymbol{\varepsilon}}_0}\text{FoM}(\hat{\boldsymbol{\varepsilon}}_0(\boldsymbol{\varepsilon}_t))$. The process is represented in Figure 1 (b). The calculated guidance term is incorporated in ancestral sampling to optimize FoM of the intermediate structure $\boldsymbol{\varepsilon}_t$, as described in Equation 1. The adjoint gradient is added component-wise to the generated structures during the reverse process, which is illustrated in Figure 1 (c).

The proposed generation process built upon the standard ancestral sampling [Ho et al., 2020] is:

$$\boldsymbol{\varepsilon}_{t-1} = \boldsymbol{\mu}_{\boldsymbol{\theta}}(\boldsymbol{\varepsilon}_t, t, \sigma) + \boldsymbol{\Sigma}_{\boldsymbol{\theta}}^{1/2}(\boldsymbol{\varepsilon}_t, t, \sigma)\boldsymbol{z} + \eta \nabla_{\boldsymbol{\varepsilon}_t} \text{FoM}(\hat{\boldsymbol{\varepsilon}}_0(\boldsymbol{\varepsilon}_t)) \tag{1}$$

where $\boldsymbol{\mu}_{\boldsymbol{\theta}}$ and $\boldsymbol{\Sigma}_{\boldsymbol{\theta}}^{1/2}$ are mean and standard deviation of iterative Gaussian denoising process, respectively, $\boldsymbol{z} \sim N(\mathbf{0}, \boldsymbol{I})$ represents a standard normal random vector, $\boldsymbol{\varepsilon}_t$ is the generated intermediate structure during the denoising process, and $\hat{\boldsymbol{\varepsilon}}_0(\boldsymbol{\varepsilon}_t)$ represents the posterior mean. Note that the update is interpretable as optimizing the target metric (i.e., FoM) with respect to the posterior mean $\hat{\boldsymbol{\varepsilon}}_0(\boldsymbol{\varepsilon}_t)$ (instead of $\boldsymbol{\varepsilon}_t$), sharing the same spirit as successful diffusion-based samplers that work with the posterior mean to incorporate conditional information [Chung et al., 2022a, Um and Ye, 2024].

Post-processing. After completing the reverse process, we post-process the final generated structure $\boldsymbol{\varepsilon}_0$ for further improvements. The post-processing consists of two stages: 1) binarization and 2)

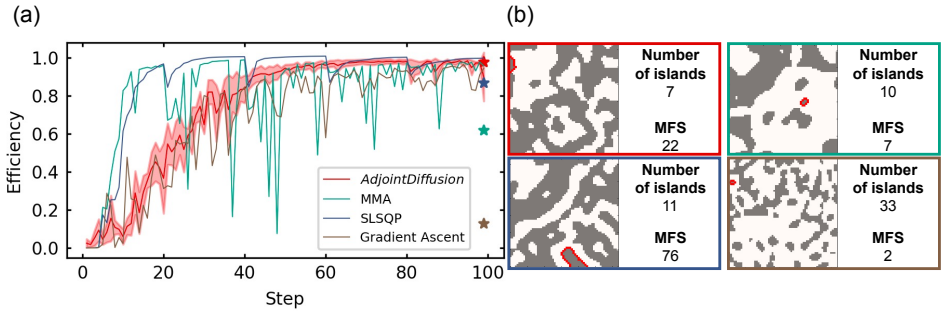

Figure 3: (a) The optimization process of each algorithm for 100 optimization steps. The red line represents the mean efficiency, and the shaded red region shows the standard deviation for *AdjointDiffusion*. The final structure's efficiency after post-processing is marked with star-shaped symbols. (b) Generated structures and the corresponding structural characteristics for each algorithm. The structural components with Minimum Feature Size (MFS) are highlighted with a red boundary.

island deletion. Firstly, since our algorithm does not use conic filters, there is a possibility that a pixel with an adjoint gradient value has an opposite sign from nearby pixels. This issue can degrade the fabricability due to the low minimum feature size (MFS). The problem is also observed in adjoint-based nonlinear algorithms, such as those provided in *NLopt* [Johnson, 2007]. To address this, we apply post-processing after binarization across all algorithms. Additionally, we analyze the FoM change resulting from our post-processing (Section D).

Our algorithm exploits the stochastic nature of deep-generative models to overcome local optima while addressing the challenges of conventional adjoint-based methods. Firstly, our framework has minimal reliance on hyperparameters. By utilizing a diffusion model trained on binary data, our sampling process inherently generates binary structures, omitting complex binarization schedules and extensive hyperparameter tuning. Secondly, our algorithm has high computational efficiency, requiring only a small number of adjoint simulations ($\sim 10^2$) compared to existing deep learning methods where data requirement ranges from $\sim 10^5$ [Jiang and Fan, 2019, Park et al., 2023] to $\sim 10^6$ [Seo et al., 2022] for one-dimensional free-form design.

3 Results and Discussion

We provide a comparison of *AdjointDiffusion* with conventional nonlinear adjoint optimization algorithm benchmarks such as MMA [Svanberg, 2002] and SLSQP [Kraft, 1988, 1994], where the details are discussed in Section A. Due to the stochasticity of the generative process, we set the experimental seeds of *Adjointdiffusion* to 3. Each process generates different structures, as shown in Section C.4, providing diverse design options.

Number of simulations. From the viewpoint of computational efficiency, the number of simulations required for the algorithm is an important factor. Adjoint sensitivity analysis requires two simulations per one optimization step, We note that the reverse process of *AdjointDiffusion* can take any step numbers [Nichol and Dhariwal, 2021, Dhariwal and Nichol, 2021] and we set the step numbers as 60, 80, 100, 120, and 140.

Structural characteristics. The feature length is set to three distinct values: 0.224, 0.562, and 0.895, corresponding to Gaussian filter standard deviations of $\sigma = 2$, $\sigma = 5$, and $\sigma = 8$, respectively. The detailed derivation of these values is provided in Section C. Additionally, we quantify the number of 'islands' formed within the generated structure and assess its Minimum Feature Size (MFS).

Figure 3 shows the iterative optimization steps of algorithms. The result shows that *AdjointDiffusion* finds the optimal structure and has robustness to binarization.

In Figure 4, we compare the optimization results of our algorithm with multiple baseline algorithms. The result shows that *AdjointDiffusion* outperforms nonlinear algorithms and a vanilla gradient ascent algorithm (GA). Error bars in the red graphs represent the standard deviation from stochasticity.

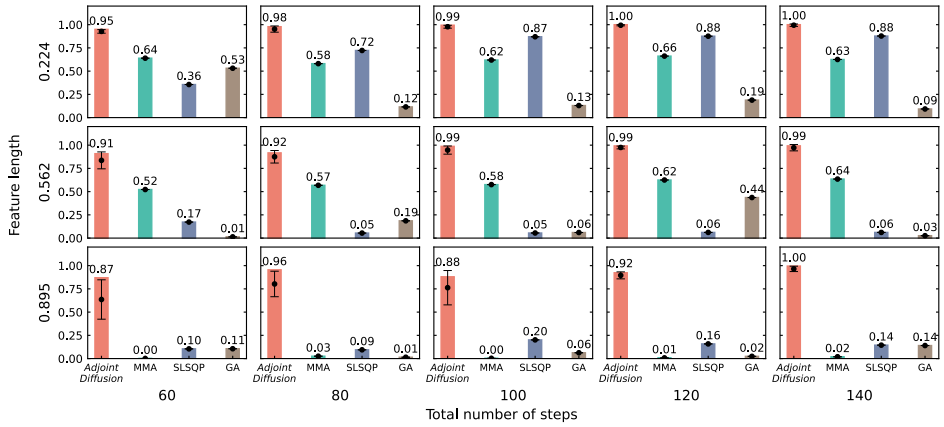

Figure 4: The efficiency of generated structures with different feature lengths and optimization steps

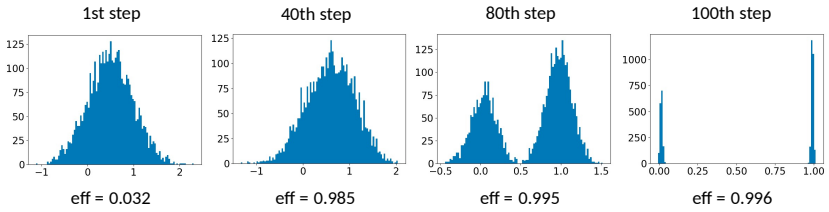

Figure 5: Pixel value distribution of ε_t at each optimization step. By the 40th step, the distribution shifts to increase efficiency to 0.985. By the 80th step, the diffusion prior separates the distribution into two distinct groups, initiating binarization. By the 100th step, the structure is mostly binarized.

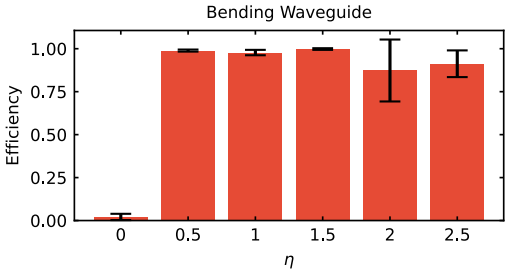

Figure 6: Efficiency of generated final structure from *AdjointDiffusion*. Error bars represent the standard deviation of efficiency.

We analyze the reason for effective binarization observed in our algorithm. Figure 5 shows the histogram of pixel values for the generated structure ε_t at each step t. The analysis implies that the binary prior of our diffusion model naturally performs binarization despite stochasticity, and this process works effectively in conjunction with the adjoint method. This can be interpreted as the manifold-preserving property often observed in gradients with respect to the posterior mean [Chung et al., 2022a, Um and Ye, 2024, Chung et al., 2022b].

Figure 6 illustrates the impact of varying the step size η, which is a coefficient multiplied to the additional adjoint gradient term in diffusion inference, on the performance of *AdjointDiffusion*. At $\eta = 0$, the generation process is efficiency-agnostic, resulting in random generation based only on the datasets. For values such as $\eta = 0.5, 1, 1.5, 2$ and 2.5, optimization is applied to the structures. While performance may decrease when η is 2 or higher, values around $\eta = 1$ show robustness of the algorithm performance, consistently approaching optimal efficiency (1.00 in the graph).

References

Sean Molesky, Zin Lin, Alexander Y Piggott, Weiliang Jin, Jelena Vucković, and Alejandro W Rodriguez. Inverse design in nanophotonics. *Nature Photonics*, 12(11):659–670, 2018.

Owen Miller. *Photonic Design: From Fundamental Solar Cell Physics to Computational Inverse Design*. PhD thesis, EECS Department, University of California, Berkeley, May 2012. URL `http://www2.eecs.berkeley.edu/Pubs/TechRpts/2012/EECS-2012-115.html`.

Yang Cao, Shengtai Li, and Linda Petzold. Adjoint sensitivity analysis for differential-algebraic equations: algorithms and software. *Journal of Computational and Applied Mathematics*, 149(1): 171–191, 2002. ISSN 0377-0427.

Grégoire Allaire. A review of adjoint methods for sensitivity analysis, uncertainty quantification and optimization in numerical codes. *Ingénieurs de l'Automobile*, 836:33–36, July 2015.

Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. *Advances in neural information processing systems*, 33:6840–6851, 2020.

Zezhou Zhang, Chuanchuan Yang, Yifeng Qin, Hao Feng, Jiqiang Feng, and Hongbin Li. Diffusion probabilistic model based accurate and high-degree-of-freedom metasurface inverse design. *Nanophotonics*, 12(20):3871–3881, October 2023. ISSN 2192-8614. doi: 10.1515/nanoph-2023-0292. URL `http://dx.doi.org/10.1515/nanoph-2023-0292`.

Yanyan Yang, Lili Wang, Xiaoya Zhai, Kai Chen, Wenming Wu, Yunkai Zhao, Ligang Liu, and Xiao-Ming Fu. Guided diffusion for fast inverse design of density-based mechanical metamaterials, 2024.

Marin Vlastelica, Tatiana López-Guevara, Kelsey Allen, Peter Battaglia, Arnaud Doucet, and Kimberley Stachenfeld. Diffusion generative inverse design, 2023.

Prafulla Dhariwal and Alexander Nichol. Diffusion models beat gans on image synthesis. *Advances in neural information processing systems*, 34:8780–8794, 2021.

Alexander Quinn Nichol and Prafulla Dhariwal. Improved denoising diffusion probabilistic models. In *International conference on machine learning*, pages 8162–8171. PMLR, 2021.

Herbert E Robbins. An empirical bayes approach to statistics. In *Breakthroughs in statistics*, pages 388–394. Springer, 1992.

Hyungjin Chung, Jeongsol Kim, Michael T Mccann, Marc L Klasky, and Jong Chul Ye. Diffusion posterior sampling for general noisy inverse problems. *arXiv preprint arXiv:2209.14687*, 2022a.

Soobin Um and Jong Chul Ye. Self-guided generation of minority samples using diffusion models. *arXiv preprint arXiv:2407.11555*, 2024.

Steven G. Johnson. The NLopt nonlinear-optimization package. `https://github.com/stevengj/nlopt`, 2007.

Jiaqi Jiang and Jonathan A Fan. Global Optimization of Dielectric Metasurfaces Using a Physics-Driven Neural Network. *Nano Letters*, 19(8):5366–5372, 2019.

Chaejin Park, Sanmun Kim, Anthony W. Jung, Juho Park, Dongjin Seo, Yongha Kim, Chanhyung Park, Chan Y. Park, and Min Seok Jang. Physics-informed reinforcement learning for sample-efficient optimization of freeform nanophotonic devices, 2023.

Dongjin Seo, Daniel Wontae Nam, Juho Park, Chan Y. Park, and Min Seok Jang. Structural optimization of a one-dimensional freeform metagrating deflector via deep reinforcement learning. *ACS Photonics*, 9(2):452–458, 2022. doi: 10.1021/acsphotonics.1c00839.

Krister Svanberg. A class of globally convergent optimization methods based on conservative convex separable approximations. *SIAM journal on optimization*, 12(2):555–573, 2002.

Dieter Kraft. A software package for sequential quadratic programming. *Forschungsbericht- Deutsche Forschungs- und Versuchsanstalt fur Luft- und Raumfahrt*, 1988.

Dieter Kraft. Algorithm 733: TOMP–fortran modules for optimal control calculations. *ACM Transactions on Mathematical Software*, 20:262–281, 1994. doi: 10.1145/192115.192124.

Hyungjin Chung, Byeongsu Sim, Dohoon Ryu, and Jong Chul Ye. Improving diffusion models for inverse problems using manifold constraints. *Advances in Neural Information Processing Systems*, 35:25683–25696, 2022b.

Ardavan F Oskooi, David Roundy, Mihai Ibanescu, Peter Bermel, John D Joannopoulos, and Steven G Johnson. Meep: A flexible free-software package for electromagnetic simulations by the fdtd method. *Computer Physics Communications*, 181(3):687–702, 2010.

Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In *Medical image computing and computer-assisted intervention–MICCAI 2015: 18th international conference, Munich, Germany, October 5-9, 2015, proceedings, part III 18*, pages 234–241. Springer, 2015.

Vikash Sehwag, Caner Hazirbas, Albert Gordo, Firat Ozgenel, and Cristian Canton. Generating high fidelity data from low-density regions using diffusion models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11492–11501, 2022.

A Baseline Algorithms

Adjoint optimization. Adjoint optimization [Miller, 2012] uses adjoint sensitivity analysis to compute the adjoint gradient and update each coordinate with the gradient value. For example, taking an optical lens as an example (Figure 1(b)), the FoM for optimization is the intensity of the electric field ($\boldsymbol{E}^{\text{dir}}$) at a focal point ($\boldsymbol{x}_f$) generated by an incident plane wave from below:

$$\text{FoM} = \frac{1}{2}|\boldsymbol{E}^{\text{dir}}(\boldsymbol{x}_f)|^2. \tag{2}$$

First, we define the structural parameter as a permittivity indicator (ε), where $\varepsilon = 1$ represents a material with higher permittivity (e.g., Si) and $\varepsilon = 0$ represents a material with lower permittivity (e.g., Air). For convenience, we refer to this value as permittivity. Initially, we set all coordinate permittivity values to 0.5 as the initial guess for the structure. Next, we determine the figure of merit (FoM) value through a direct simulation induced by an incident plane wave. In the direct simulation, the electric field at the target point $\boldsymbol{E}^{\text{dir}}(\boldsymbol{x}_f)$ is computed and subsequently used as amplitudes of the adjoint sources, while $\boldsymbol{E}^{\text{dir}}(\boldsymbol{x}')$ at the design area is saved for the purpose of adjoint gradient calculation.

The variation in FoM by the transmitted electric field is

$$\begin{aligned}
\delta\text{FoM} &= \frac{1}{2}\left[\boldsymbol{E}^{\text{dir}}(\boldsymbol{x}_f) + \delta\boldsymbol{E}(\boldsymbol{x}_f)\right]\overline{\left[\boldsymbol{E}^{\text{dir}}(\boldsymbol{x}_f) + \delta\boldsymbol{E}(\boldsymbol{x}_f)\right]} - \frac{1}{2}\boldsymbol{E}^{\text{dir}}(\boldsymbol{x}_f)\overline{\boldsymbol{E}^{\text{dir}}(\boldsymbol{x}_f)} \\
&= \frac{1}{2}\left[\boldsymbol{E}^{\text{dir}}(\boldsymbol{x}_f)\overline{\delta\boldsymbol{E}(\boldsymbol{x}_f)} + \overline{\boldsymbol{E}^{\text{dir}}(\boldsymbol{x}_f)}\delta\boldsymbol{E}(\boldsymbol{x}_f) + |\delta\boldsymbol{E}(\boldsymbol{x}_f)|^2\right] \\
&\approx \text{Re}\left[\overline{\boldsymbol{E}^{\text{dir}}(\boldsymbol{x}_f)}\delta\boldsymbol{E}^{\text{dir}}(\boldsymbol{x}_f)\right].
\end{aligned} \tag{3}$$

The variation of the electric field at point $\boldsymbol{x}_f$, caused by the adjustment in the permittivity of the design space, can be expressed as

$$\delta\boldsymbol{E}^{\text{dir}}(\boldsymbol{x}_f) = \mathbf{G}(\boldsymbol{x}_f, \boldsymbol{x}')\mathbf{P}^{\text{ind}}(\boldsymbol{x}') = \mathbf{G}(\boldsymbol{x}_f, \boldsymbol{x}')\delta\boldsymbol{\varepsilon}(\boldsymbol{x}')\boldsymbol{E}^{\text{dir}}(\boldsymbol{x}'), \tag{4}$$

where $\boldsymbol{x}$ and $\boldsymbol{x}'$ indicate the positions in the monitor and the design space, respectively. $\mathbf{P}^{\text{ind}}(\boldsymbol{x}')$ indicates the polarization density, which is induced by the variation of the permittivity $\delta\boldsymbol{\varepsilon}(\boldsymbol{x}')$. $\overleftrightarrow{\mathbf{G}}(\boldsymbol{x}_f, \boldsymbol{x}')$ is a Green's function which represents the electric field at the point $\boldsymbol{x}_f$ generated by the unit dipole at the point $\boldsymbol{x}'$. The formula for the change in FoM becomes

$$\delta\text{FoM} = \text{Re}\left[\mathbf{G}(\boldsymbol{x}_f, \boldsymbol{x}')\overline{\boldsymbol{E}^{\text{dir}}(\boldsymbol{x}_f)}\delta\boldsymbol{\varepsilon}(\boldsymbol{x}')\boldsymbol{E}^{\text{dir}}(\boldsymbol{x}')\right]. \tag{5}$$

The adjoint field $\boldsymbol{E}^{\text{adj}}$ can be expressed as

$$\boldsymbol{E}^{\text{adj}}(\boldsymbol{x}') = \mathbf{G}(\boldsymbol{x}_f, \boldsymbol{x}')\overline{\boldsymbol{E}^{\text{dir}}(\boldsymbol{x}_f)}. \tag{6}$$

Subsequently, the amplitude of the adjoint dipole source becomes $\overline{\boldsymbol{E}^{\text{dir}}(\boldsymbol{x}_f)}$. The adjoint dipole sources backpropagate through the designable region and generate $\boldsymbol{E}^{\text{adj}}(\boldsymbol{x}')$. Finally, the gradient of the FoM with respect to changes in $\boldsymbol{\varepsilon}(\boldsymbol{x}')$ within the design region is computed as

$$\frac{\delta\text{FoM}}{\delta\boldsymbol{\varepsilon}(\boldsymbol{x}')} = \text{Re}[\boldsymbol{E}^{\text{dir}}(\boldsymbol{x}')\boldsymbol{E}^{\text{adj}}(\boldsymbol{x}')] \tag{7}$$

which is called the adjoint gradient value. This gradient indicates how the design parameters should be adjusted to improve FoM. Typically, this adjustment is performed using a gradient ascent algorithm with a learning rate η:

$$\boldsymbol{\varepsilon}_{i+1} = \boldsymbol{\varepsilon}_i + \eta\nabla_{\boldsymbol{\varepsilon}_i}\text{FoM}(\boldsymbol{\varepsilon}_i). \tag{8}$$

After each step, since the generated structures are in grayscale, a binarization process is required. Binarization is scheduled to operate less intensively at the beginning and more intensively towards the end. This process is governed by two factors, β and $\boldsymbol{\zeta}$:

$$\boldsymbol{\varepsilon}' = \frac{\tanh\left(\beta * \boldsymbol{\zeta}\right) + \tanh\left(\beta * \left(\boldsymbol{\varepsilon} - \boldsymbol{\zeta}\right)\right)}{\tanh\left(\beta * \boldsymbol{\zeta}\right) + \tanh\left(\beta * \left(\mathbf{1} - \boldsymbol{\zeta}\right)\right)} \tag{9}$$

β is a scalar that modulates the extent of binarization applied to the structures. β starts small in the initial stages and progressively increases, meaning more binarization is applied as the optimization proceeds. $\boldsymbol{\zeta}$ serves as a threshold value for binarization, with every element being 0.5. $\mathbf{1}$ is an all-ones matrix. Note that all operations (multiplication, subtraction, tanh) are performed element-wise on the matrices.

Adjoint optimization with nonlinear optimization algorithm. Gradient ascent algorithms fall into local optima in nonconvex objective functions. To resolve this issue, researchers utilize nonlinear optimization algorithms that search the entire function space for the global optimum. From the open-source library named *NLopt* [Johnson, 2007], we import MMA (Method of Moving Asymptotes) [Svanberg, 2002] and SLSQP (Sequential Least Squares Quadratic Programming) [Kraft, 1988, 1994] as baseline algorithms. Both are gradient-based local optimization algorithms that are generally used in adjoint optimization due to their robustness in handling complex constraints and non-convex problems.

1) MMA creates a local approximation using the gradient of the function along with a quadratic penalty term to ensure cautious approximations. The key idea is that the approximation is both convex and separable, making it easy to solve using a dual method. This solution provides a new candidate point that is then evaluated. If the approximations are conservative, the process restarts at the new point; if not, the penalty is increased, and the problem is re-optimized.

2) SLSQP is a sequential quadratic programming (SQP) algorithm designed to solve nonlinear optimization problems by breaking them down into simpler quadratic programming (QP) subproblems. It works by using a second-order Taylor expansion to approximate the objective function and linearize the constraints. The algorithm iteratively refines the solution until it meets the convergence criteria. If the criteria are not met, the process is repeated.

B Implementation Details of *AdjointDiffusion*

Network architecture. The neural network architecture we used for our diffusion backbone is based on U-Net [Ronneberger et al., 2015]. More precisely, we employed an improved version of U-Net presented in [Dhariwal and Nichol, 2021], which is specifically tailored for diffusion models. For architectural design choices, We mostly followed the default setting recommended in the original codebase of [Dhariwal and Nichol, 2021][1]. A distinction is that we used the number of input channels as 1, since we consider grayscale images for inputs. Structural information σ is first added with time embedding t and fed in the form of adaptive group normalization (i.e., the same approach as in [Dhariwal and Nichol, 2021]). See Figure 7 for an overall illustration.

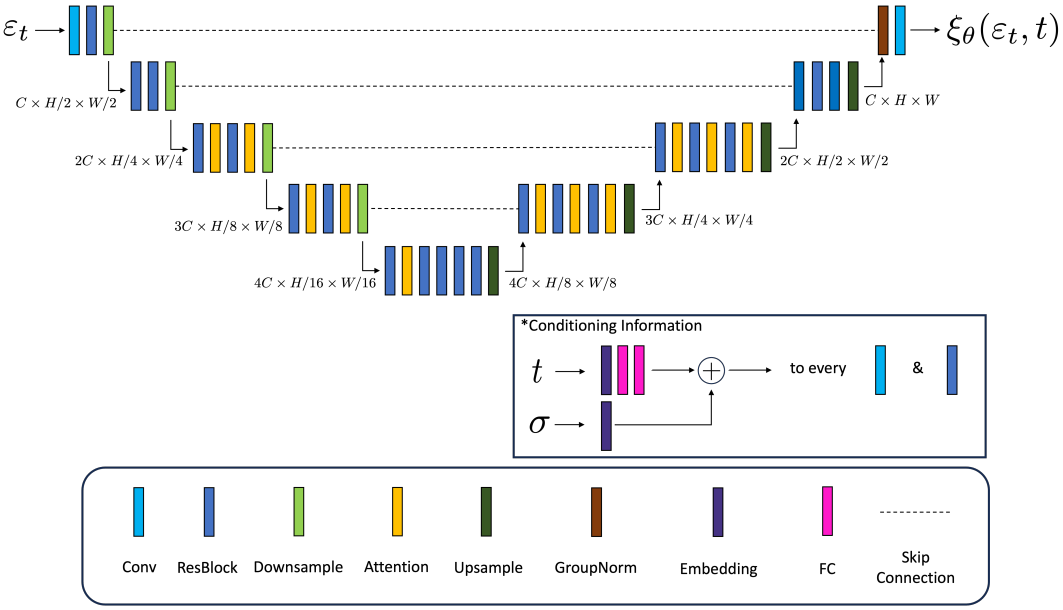

Figure 7: Architecture of the diffusion model employed in *AdjointDiffusion*. Here, channel number $C = 128$ and spatial dimensions $H = 64$, $W = 64$. The model consists of multiple stages with convolutional layers (Conv), residual blocks (ResBlock), downsampling layers (Downsample), attention mechanisms (Attention), and upsampling layers (Upsample). The conditional information, consisting of time step t and structural parameter σ, is embedded and added to each convolutional layer and residual block. The dimensions of the feature maps are progressively reduced and then increased, with skip connections linking corresponding layers across the downsampling and upsampling stages. Group normalization (GroupNorm) is applied to stabilize training.

Training setup. The total number of timesteps is $T = 1000$, and we use cosine noise schedule [Nichol and Dhariwal, 2021]. As in [Dhariwal and Nichol, 2021], For model training, we focus on L_{hybrid} (proposed in [Nichol and Dhariwal, 2021]) to additionally learn the noise variance $\boldsymbol{\Sigma}_{\boldsymbol{\theta}}(\cdot, \cdot)$ for better performance when accelerating the sampling process. The training is performed with a learning rate of 1e-4, and the batch size is 128.

Inference details. We globally use $\eta = 1.0$ and 100 diffusion timesteps for sampling on both tasks (i.e., waveguide and CIS), while admitting variations whenever necessary (e.g., for ablation studies). As in [Sehwag et al., 2022, Um and Ye, 2024], we normalize the gradient (integrated in the reverse process) to have unit l_∞ norm. More precisely, we employ $\nabla^*_{\boldsymbol{\varepsilon}_t} \operatorname{FoM}\left(\hat{\boldsymbol{\varepsilon}_0}\left(\boldsymbol{\varepsilon}_t\right)\right) := \nabla_{\boldsymbol{\varepsilon}_t} \operatorname{FoM}\left(\hat{\boldsymbol{\varepsilon}_0}\left(\boldsymbol{\varepsilon}_t\right)\right) / \|\nabla_{\boldsymbol{\varepsilon}_t} \operatorname{FoM}\left(\hat{\boldsymbol{\varepsilon}_0}\left(\boldsymbol{\varepsilon}_t\right)\right)\|_\infty$ instead of $\nabla_{\boldsymbol{\varepsilon}_t} \operatorname{FoM}\left(\hat{\boldsymbol{\varepsilon}_0}\left(\boldsymbol{\varepsilon}_t\right)\right)$.

Pseudocode. We provide pseudocode which represents training and sampling processes of *AdjointDiffusion*:

[1] `https://github.com/openai/guided-diffusion`

Algorithm 1 DDPM training [Ho et al., 2020]

repeat
  $\boldsymbol{\varepsilon}_0 \sim q(\boldsymbol{\varepsilon}_0)$
  $t \sim \mathrm{Uniform}(\{1, \ldots, T\})$
  $\sigma \sim \mathrm{Uniform}(\{2, 5, 8\})$ ▷ feature length para.
  $\boldsymbol{z} \sim \mathcal{N}(\mathbf{0}, \boldsymbol{I})$
  Take gradient descent step on
    $\nabla_{\boldsymbol{\theta}} \left\| \boldsymbol{z} - \boldsymbol{z}_{\boldsymbol{\theta}}(\sqrt{\bar{\alpha}_t}\boldsymbol{\varepsilon}_0 + \sqrt{1-\bar{\alpha}_t}\boldsymbol{z}, t, \sigma) \right\|^2$
until converged

Algorithm 2 Physics-guided sampling

$\boldsymbol{\varepsilon}_T \sim \mathcal{N}(\mathbf{0}, \boldsymbol{I})$
for $t \leftarrow T$ to 1 **do**
  $\boldsymbol{\xi} \sim \mathcal{N}(\mathbf{0}, \boldsymbol{I})$ if $t > 1$, else $\boldsymbol{\xi} = \mathbf{0}$
  $\boldsymbol{\varepsilon}'_{t-1} \leftarrow \frac{1}{\sqrt{\alpha_t}} \left(\boldsymbol{\varepsilon}_t - \frac{1-\alpha_t}{\sqrt{1-\bar{\alpha}_t}} \boldsymbol{z}_{\boldsymbol{\theta}}(\boldsymbol{\varepsilon}_t, t, \sigma) \right) + \varsigma_t \boldsymbol{\xi}$
  $\boldsymbol{\varepsilon}_{t-1} \leftarrow \boldsymbol{\varepsilon}'_{t-1} + \eta \nabla_{\boldsymbol{\varepsilon}_t} \mathrm{FoM}\left(\hat{\boldsymbol{\varepsilon}_0}\left(\boldsymbol{\varepsilon}_t\right)\right)$
end for
return $\boldsymbol{\varepsilon}_0$

C Calculation Method of Feature Length in Structure Generation

C.1 Feature Length [pixels] from Gaussian filter (in image database)

The Gaussian filter is a technique commonly used in image processing to smooth out and reduce noise. The minimum length of a structure that a Gaussian filter can resolve is related to the filter's standard deviation (σ), and the value determines the extent of blurring.

The full-width at half-maximum (FWHM) of the Gaussian function describes the effective spread of the filter. The FWHM is related to the standard deviation by the following formula:

$$\mathrm{FWHM} = 2\sqrt{2\ln 2} \cdot \sigma \tag{10}$$

which calculates the width of the Gaussian function at half of its maximum value. The Feature Length, which we define as the minimum structure size that can be resolved by the Gaussian filter, is approximately equivalent to the FWHM.

C.2 Feature Length [μm] (in simulation)

In Meep [Oskooi et al., 2010] simulation, the Feature Length in micrometers [μm] is calculated as the Feature Length in pixels divided by the simulation resolution. In our setup, we set the simulation resolution value as 21.

C.3 Calculation

We calculate the Feature Length of each dataset by applying Gaussian filter standard deviations $\sigma = 2, 5, 8$ to Equation 10, with the results presented in Table 1.

Table 1: Relationship between Gaussian filter standard deviation σ, Feature Length in pixels, and Feature Length in micrometers (μm).

σ	Feature Length [pixels]	Feature Length [μm]
2	4.7	0.224
5	11.8	0.562
8	18.8	0.895

C.4 Generated Structures

We provide the generated structures from each algorithm. In Figure 8, the structures produced by *AdjointDiffusion*, MMA, and SLSQP exhibit similar characteristic sizes. For the Gradient Ascent (GA) algorithm, we use the same simulation setup, but the Feature Length is not applied as expected, likely due to an issue with the Conic Filter. However, the smaller Feature Length does not disadvantage GA, so we report the results as they are.

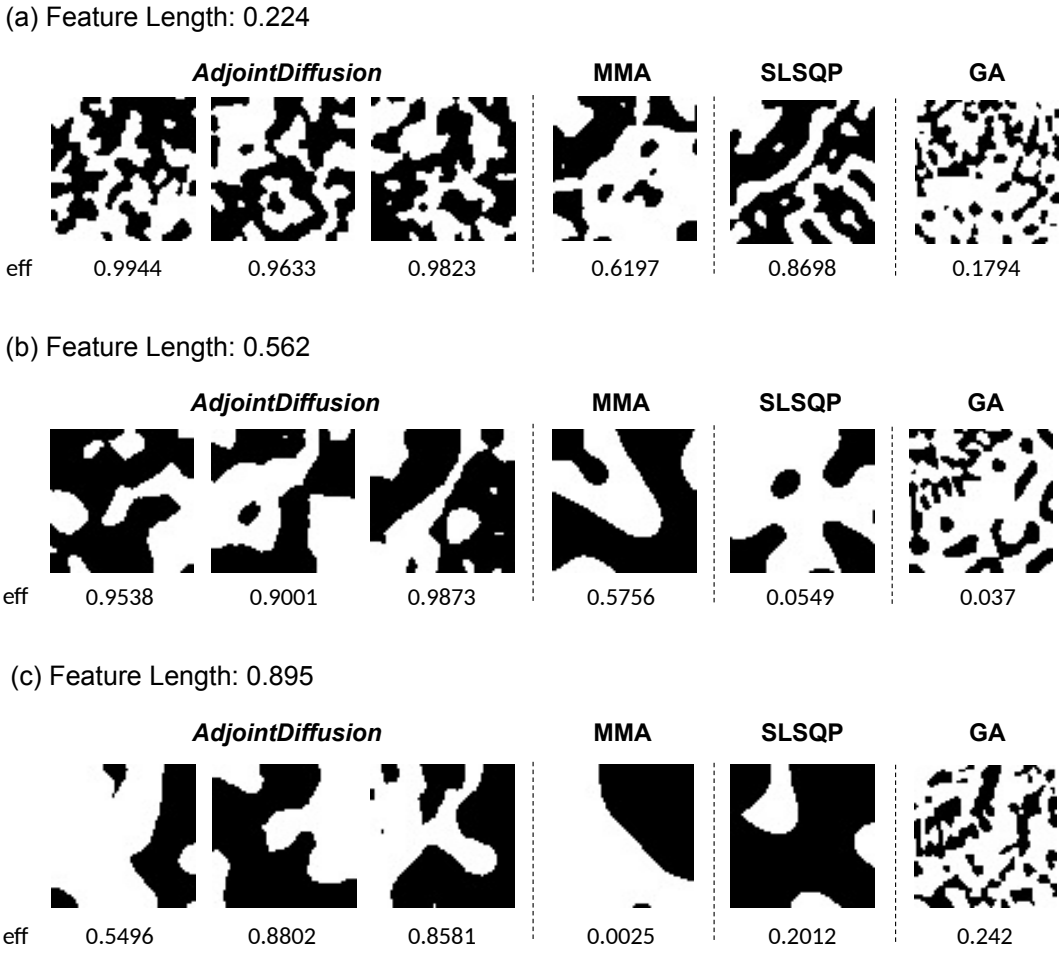

Figure 8: Generated structures for different Feature Lengths [μm] of (a) 0.224, (b) 0.562, and (c) 0.895 using various algorithms. The efficiency (eff) for each algorithm (*AdjointDiffusion*, MMA, SLSQP, GA) is provided below the corresponding structures.

D FoM Degradation from Post-processing

We analyze the degradation of the Figure of Merit (FoM) following post-processing, which includes binarization and island deletion.

D.1 Adjoint Optimization

Using MMA as an example, we observe that, while it achieves a high FoM during the optimization process, the algorithm is particularly vulnerable to post-processing steps, such as binarization and island deletion.

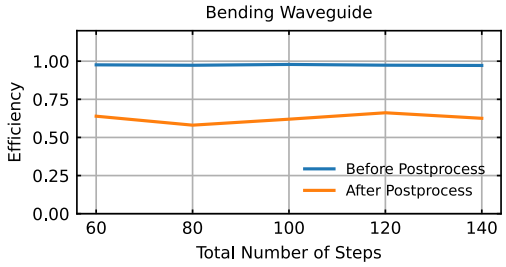

Figure 9: Efficiency comparison before and after post-processing of MMA for the Bending Waveguide over various numbers of optimization steps. The blue line represents efficiency before post-processing, while the orange line represents efficiency after post-processing.

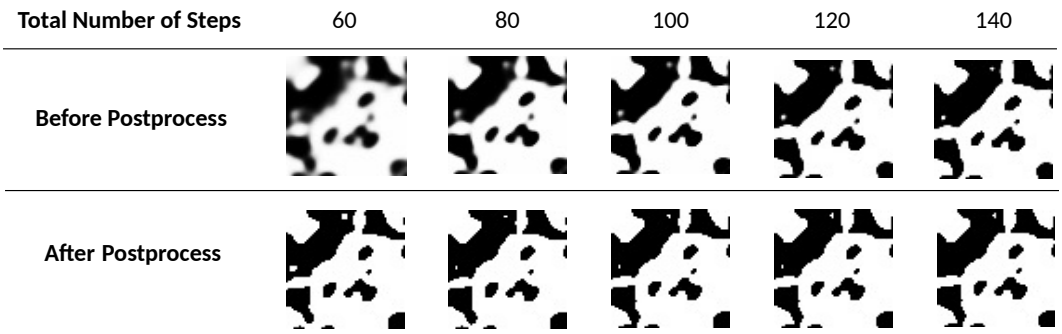

Figure 10: Visual comparison of generated structures before and after post-processing of MMA at different numbers of optimization steps. The top row shows the structures before post-processing, and the bottom row shows the corresponding structures after post-processing.

D.2 *AdjointDiffusion*

AdjointDiffusion consistently remains robust to binarization and sometimes even demonstrates an unexpected improvement in FoM after post-processing.

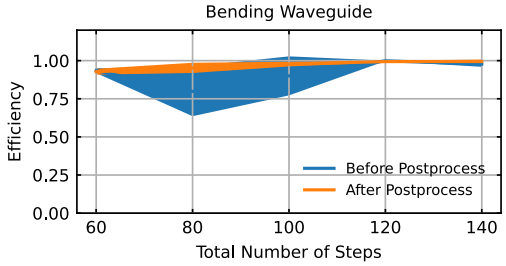

Figure 11: Efficiency comparison before and after post-processing of *AdjointDiffusion* for the Bending Waveguide over various numbers of optimization steps. The blue region represents the mean and standard deviation of efficiency before post-processing, while the orange region represents the mean and standard deviation of efficiency after post-processing.

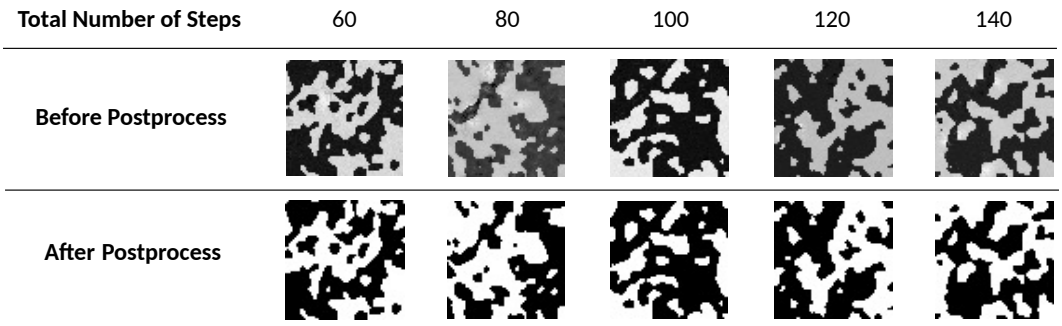

Figure 12: Visual comparison of generated structures before and after post-processing of *Adjoint-Diffusion* at different numbers of optimization steps. The top row shows the structures before post-processing, and the bottom row shows the corresponding structures after post-processing.

E Visualization of Simulation Results

In this section, we present the simulation results. The blue outlines represent the design boundaries, with the red line indicating the source and the blue lines indicating the electric field monitors.

Figure 13, 15, and 17 provide a visualization of the simulation results, showing the evolution of the field distribution at various stages of the simulation. The color gradients illustrate the intensity and phase of wave propagation through the structure. The four parts of the figure correspond to specific points in the simulation timeline, capturing changes in the electromagnetic field as it interacts with the structure.

E.1 Optimized Structure from *AdjointDiffusion*

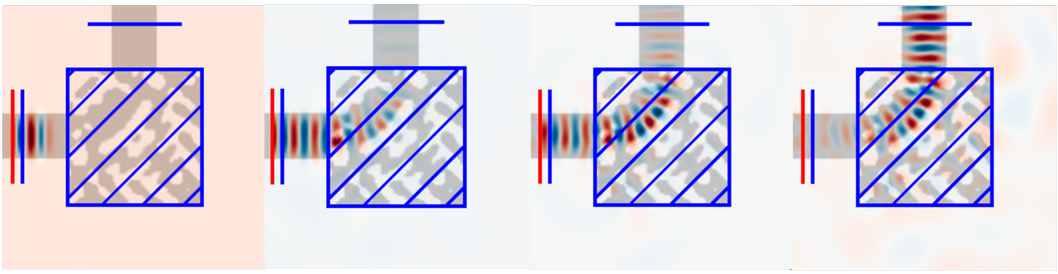

Figure 13: The evolution of the field distribution across different stages of the simulation

E.2 Structures based on Human Intuition

We additionally provide the visualized simulation results of two kinds of design based on human intuition.

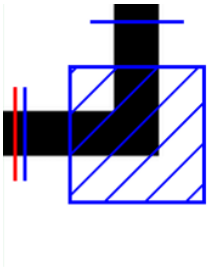

Figure 14: Right-angled waveguide

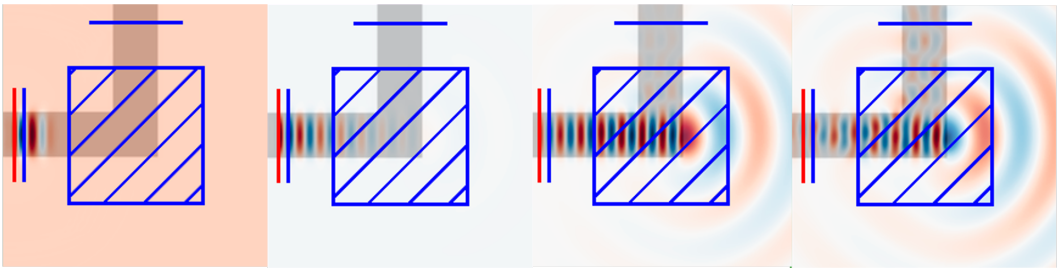

Figure 15: The evolution of the field distribution across different stages of the simulation for structure in Figure 14. The efficiency of this structure is 0.0011.

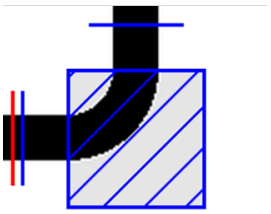

Figure 16: Curved waveguide

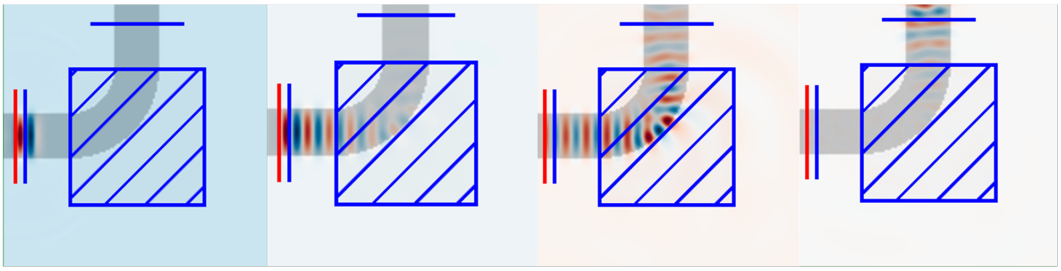

Figure 17: The evolution of the field distribution across different stages of the simulation for structure in Figure 16. The efficiency of this structure is 0.8271.