

Physics-Driven Learning for Manufacturability

WANG, Zixiao

A Thesis Submitted in Partial Fulfilment
of the Requirements for the Degree of
Doctor of Philosophy
in
Computer Science and Engineering

The Chinese University of Hong Kong
July 2026

Thesis Assessment Committee

Professor KING Kuo Chin Irwin (Chair)

Professor YU Bei (Thesis Supervisor)

Professor Farzan FARNIA (Thesis Co-supervisor)

Professor SHAO Zili (Committee Member)

Professor WANG Runsheng (External Examiner)

Abstract

Advanced VLSI technologies impose increasingly strict manufacturability and reliability constraints. Many of the key challenges arise from complex physical effects, such as layout patterning, photoresist behavior, and on-chip thermal phenomena. Accurately modeling these effects is critical for design closure, yet traditional simulation tools are often computationally expensive, difficult to differentiate, and limited by scarce or noisy data.

This dissertation studies data- and physics-driven learning methods for manufacturability in VLSI, with a focus on three representative aspects: layout pattern generation, photoresist modeling, and thermal analysis.

First, for layout pattern generation, we develop discrete diffusion-based generative models that learn manufacturability-aware distributions of layout patterns. These models enable automatic pattern synthesis and optimization under design and process constraints. We further introduce a language-guided pattern customization framework that bridges natural language descriptions with layout generation, allowing human designers to specify high-level intents while the model produces detailed manufacturable patterns.

Second, for photoresist modeling, we propose physics-constrained diffusion models that integrate lithography-domain priors with data-driven learning. By embedding physical knowledge into the generative process, the proposed approach achieves accurate and robust resist prediction under advanced process condi-

tions. Complementing this, we develop an open-source differentiable photore-sist simulator implemented within modern deep learning frameworks, which supports gradient-based calibration and optimization in design-for-manufacturability workflows.

Third, for thermal analysis, we design an efficient random-walk-based method for steady-state point-query thermal estimation. The proposed method leverages the structure of the underlying physical system to significantly accelerate localized thermal analysis while maintaining high accuracy, making it suitable for iterative design and verification.

Together, these contributions demonstrate that combining data-driven learning with physics-based modeling provides an effective paradigm for addressing manufacturability-critical problems in VLSI design. The methods developed in this dissertation offer practical tools and insights for building scalable, accurate, and optimization-friendly models at the intersection of electronic design automation and machine learning.

摘要

隨著 VLSI 製程節點不斷演進，晶片設計面臨日益嚴苛的可製造性與可靠性約束。其中，多項關鍵挑戰源自複雜的物理效應，例如佈局圖形成形、光刻膠響應以及片上熱行為。對這些物理過程進行精確且高效的建模與模擬，是實現設計收斂與製程優化的關鍵。然而，傳統模擬工具往往計算開銷巨大、難以求導，並且在數據有限或含雜訊的情境下表現受限。

本論文聚焦於 VLSI 可製造性的數據與物理驅動學習方法，重點研究三類具有代表性的物理問題：佈局圖形生成、光刻膠建模以及熱分析。

首先，在佈局圖形生成方面，論文提出基於離散擴散的生成模型，用於學習具備可製造性約束的佈局圖形分佈，從而實現自動圖形合成與優化。同時，構建面向自然語言指令的圖形客製化框架，將人類對圖形結構與製程意圖的高層描述，與底層的可製造佈局生成相結合，以提升設計互動性與自動化程度。

其次，在光刻膠建模方面，論文提出融合物理先驗的物理約束擴散模型，將光刻領域知識嵌入生成過程之中，在先進製程條件下實現對光刻膠響應的精確且穩健預測。與此同時，論文開發了基於深度學習框架的開源可微光刻膠模擬器，支援梯度驅動的模型校準與可製造性優化，為設計與製程協同提供新的工具支援。

再次，在熱分析方面，論文設計了高效的隨機游走穩態點查詢熱估計方法，利用物理系統結構特性，顯著加速局部熱分析，同時維持較高精度，適用於反覆迭代的設計與驗證流程。

綜合而言，本論文表明，將數據驅動學習與物理建模相結合，為解決 VLSI

設計中關鍵的可製造性問題提供了一條有效途徑。論文所提出的方法，於電子設計自動化與機器學習的交叉領域，為構建兼具可擴展性、準確性與可優化性的模型提供了實務方案與理論啟示。

Acknowledgement

First and foremost, I would like to thank my supervisor, Professor Bei Yu. Professor Yu is a rare scholar in whom personal warmth and academic leadership coexist in equal measure. He guided me onto the path at the intersection of ML and EDA, and over the past five years has consistently encouraged and supported me in pursuing the directions I cared about most. It is thanks to him that I have had the room to reflect on the physical essence underlying the ML $\times$ EDA intersection. These reflections ultimately gave rise to this thesis.

I am equally grateful to my co-supervisor, Professor Farzan Farnia. Professor Farnia tirelessly spent long hours with me discussing paper details and experimental design, and these conversations sparked many of the ideas in this work. The rigor I cultivated during our collaboration, together with the appreciation for the mathematical beauty of machine learning that he instilled in me, will benefit me for a lifetime, and has profoundly shaped my subsequent academic outlook.

I would also like to thank my co-authors, my labmates, and all the friends and mentors who have discussed scholarly questions with me or guided me through the questions of life. I am keenly aware of how limited my own vision and how shallow my accumulation are; the exchanges and collaborations with all of you have given me new perspectives from which to see the world, and have helped me become a better person. I feel fortunate to have shared these pure years of inquiry with each of you.

In addition, I would like to thank a small doggerel quoted in “To the Students,” the preface of my senior-secondary physics textbook (People’s Education Press, Elective 3-1):

Years ago I saw a painting of this lake,
and could not believe such a place was real.
Today, as I cross the lake at last,
even the painter’s brush has fallen short.

It set the heart of a small-town boy on a distant lake; today, having crossed that lake at last, he knows the editors’ words were no exaggeration.

Finally, I thank my wife, Chenyu Wu, for her unwavering love and support, year after year, day after day.

Contents

Abstract	i
摘要	iii
Acknowledgement	v
List of Figures	xiii
List of Tables	xviii
1 Introduction	1
1.1 Physical Design Closure in Advanced VLSI	1
1.2 Physical Modeling Challenges	2
1.2.1 Layout Pattern Generation under Design-Rule Constraints	2
1.2.2 Photoresist Modeling for Pattern Transfer	3
1.2.3 Localized Thermal Estimation for 3D IC Reliability	3
1.3 Limitations of Existing Methods	4
1.4 Physics-Driven Learning and Estimation	5
1.5 Thesis Contributions	7
1.6 Thesis Organization	8
2 Background and Preliminaries	10

2.1	Layout Pattern Generation Background	10
2.1.1	Design Rules and DRC	10
2.1.2	Discrete Layout Representations	11
2.1.3	Discrete Denoising Diffusion (D3PM)	13
2.1.4	Large Language Models and Agents	15
2.2	Lithographic Pattern Transfer and Photoresist Modeling	16
2.2.1	Optical Lithography and Aerial Images	16
2.2.2	OPC and ILT	16
2.2.3	Resist Modeling Landscape	18
2.2.4	Dill Exposure Model	18
2.2.5	Mack Development Model	19
2.2.6	Continuous Denoising Diffusion	20
2.3	Steady-State Thermal Analysis	21
2.3.1	Thermal Integrity in 3D ICs	21
2.3.2	Classical Thermal Simulation Methods	21
2.3.3	Heat-Conduction PDE and Boundary Conditions	22
2.3.4	Feynman–Kac Stochastic Representation	23
2.3.5	Random Walk Discretizations	24
2.4	Chapter Summary	25
3	Layout Pattern Generation via Generative Learning	26
3.1	Introduction	26
3.2	Related Work	29
3.2.1	Rule-Based Pattern Generation	29
3.2.2	Learning-Based Pattern Generation	30
3.2.3	Squish Pattern Representation	31
3.2.4	Large Language Models in EDA	32
3.3	Method Overview	33

3.4	Deep Squish Pattern Representation	34
3.5	Discrete Diffusion for Topology Generation	36
3.5.1	Forward Transition Dynamics	37
3.5.2	Reverse Denoising	38
3.5.3	Training Objective	39
3.5.4	Fast Sampling	40
3.5.5	Topology Pre-filter	41
3.6	Topology Legalization	41
3.6.1	Nonlinear Geometric Assignment	41
3.6.2	Block-Partition Acceleration	43
3.7	Reliable Topology Enhancement	44
3.8	Interactive Customization via Conditional Generation and LLM Orchestration	46
3.8.1	System Overview	47
3.8.2	Expert LLM Agent	47
3.8.3	Property-Conditional Topology Generation	50
3.8.4	Conditional Modification via In-Painting	51
3.8.5	Free-Size Extension via Out-Painting	52
3.9	Experimental Results	54
3.9.1	Experimental Setup	54
3.9.2	Evaluation Metrics	55
3.9.3	Fixed-Size Pattern Generation	56
3.9.4	Free-Size Pattern Generation	60
3.9.5	Flexibility under Varying Design Rules	61
3.9.6	Ablation Studies	62
3.9.7	System Efficiency	64
3.9.8	Evaluation of LLM Agent	66

3.10	Summary	67
4	Physics-Constrained Photoresist Modeling	69
4.1	Introduction	69
4.2	Related Work	72
4.2.1	Classical Resist Models	72
4.2.2	First-Principle and Monte Carlo Simulators	73
4.2.3	Neural Network Surrogates	73
4.2.4	Diffusion Models for Sequential and Video Prediction . . .	74
4.3	Resist Process Recap	74
4.4	TorchResist: A Differentiable Resist Simulator	76
4.4.1	Architecture and Parameterization	77
4.4.2	Differentiable Calibration	77
4.4.3	GPU-Accelerated Parallel Inference	78
4.5	DiffResist for 3D Profile Prediction	78
4.5.1	Problem Formulation and Modeling Paradigms	79
4.5.2	Framework Overview	80
4.5.3	Forward Diffusion Process	80
4.5.4	Reverse Process with Physical Constraints	82
4.5.5	Training and Inference	83
4.5.6	Two-Stage Noise Schedule	85
4.5.7	Efficient Super-Resolution Network	85
4.6	Experimental Results	86
4.6.1	Datasets and Evaluation Metrics	86
4.6.2	TorchResist Evaluation	88
4.6.3	DiffResist Evaluation	91
4.6.4	Ablation Studies	93
4.7	Assumptions and Validity Range	97

4.8	Summary	98
5	Efficient Steady-State Thermal Analysis	101
5.1	Introduction	101
5.2	Related Work	104
5.3	Problem Formulation and Random Walks	107
5.3.1	Random Walk Discretizations	107
5.3.2	The Truncation Challenge	109
5.4	Truncation Residual Identity	109
5.4.1	Stopping-Time Identity	109
5.4.2	Practical Interpretation	111
5.5	FastRW Methodology	111
5.5.1	Single-Point Acceleration via Prior Estimation	111
5.5.2	Prior Estimation Sources	114
5.5.3	Multi-Point Acceleration via Posterior Fusion	114
5.6	Experimental Results	116
5.6.1	Experimental Setup	116
5.6.2	Single-Point and Multi-Point Temperature Estimation	118
5.6.3	Further Analysis	120
5.7	Summary	122
6	Conclusion and Future Work	125
6.1	Summary of Contributions	125
6.2	Lessons Learned	128
6.3	Limitations	130
6.4	Future Research Directions	131
6.5	Closing Remarks	132

A Proofs for Chapter 5	134
A.1 Proof of the Truncation Residual Identity	134
A.2 Bias and MSE of the Plug-in Truncation Correction	136
A.2.1 Single-Path Plug-in Correction	136
A.2.2 MSE Decomposition and Averaging over Paths	137
A.3 Derivation of the Bayesian Posterior Fusion	138
Bibliography	140
Publications Related to This Thesis	161

List of Figures

2.1	Illustration of representative design rules for layout patterns. . . .	11
2.2	Squish Pattern representation of a layout pattern.	12
2.3	Illustration of the denoising diffusion process.	13
2.4	Overview of optical lithography and 3D resist modeling. Left: light is patterned by the mask, focused by the projection optics, and projected onto the photoresist film coating the wafer. Right: the resulting aerial image ($H \times W$) drives a volumetric photochemical response that yields a three-dimensional resist profile ($H \times W \times D$). 17	
3.1	An illustration of the DiffPattern framework for reliable layout pattern generation. The pipeline consists of three phases: Deep Squish Pattern encoding, topology tensor generation via discrete diffusion, and 2D legal pattern assessment through nonlinear legalization.	33
3.2	Illustration of the Deep Squish Pattern Encoding. The expanded topology tensor provides a compact, multi-channel lossless representation. Directly concatenating binary states into a single numerical integer causes unequal bit-weighting and exponential state space explosion, which is circumvented by the multi-channel tensor formulation.	35

3.3	Illustration of the discrete denoising process. Starting from uniform noise (left), the topology tensor is progressively denoised toward a clean binary layout topology (right).	39
3.4	Design-rule constraints used by the legalization solver	43
3.5	Illustration of block-partition acceleration for legalization. Each diagonal block denotes an independent sub-problem. All sub-problems are solved in parallel to provide high-quality initialization for the global solver.	44
3.6	Illustration of reliable topology enhancement. Multiple augmentation methods are applied to each pattern. (f) denotes the random patch within the red box cropped from (e). Only augmented topologies that pass legalization verification are retained for training.	46
3.7	An illustration of the ChatPattern system. The LLM agent interprets natural language requirements and orchestrates the pattern generation backend to produce customized layout libraries.	47
3.8	Working pipeline of the LLM agent in ChatPattern . The agent decomposes user requests into structured sub-tasks, invokes backend APIs, and handles errors autonomously. Dashed lines denote optional paths.	48
3.9	Conditional discrete diffusion for property-specified pattern generation	52
3.10	Pattern In-Painting for localized modification	53
3.11	Mechanism for free-size pattern extension via iterative Out-Painting and bounded In-Painting. Recursive extension supports target sizes larger than the model’s native window.	54

3.12	Complexity distribution comparison. Reliable pattern augmentation enriches the generated pattern distribution, covering regions absent in the real data distribution while maintaining structural validity.	58
3.13	256×256 topology matrices generated by In-Painting extension. .	60
3.14	512×512 topology matrices generated by Out-Painting extension.	61
3.15	Layout patterns instantiated from the identical generated topology under varying DRC design rules. The explicit separation of topology generation from geometric legalization supports rule changes without retraining when the topology remains feasible.	62
3.16	Concatenate-and-Crop probability ablation	63
3.17	Evaluation of In-Painting and Out-Painting extension strategies across different target sizes (ChatPattern). Out-Painting generally achieves higher legality, while In-Painting can provide better diversity.	64
4.1	Schematic of a positive-tone resist process. During exposure a strong acid is generated; the PEB step catalytically alters resist solubility; and the subsequent development step dissolves exposed regions, leaving the final patterned image.	75
4.2	Comparison of modeling paradigms. (a) Direct mapping is computationally expensive due to the 3D output. (b) Sequential modeling predicts each depth layer from the preceding one but accumulates errors. (c) DiffResist injects physical boundary constraints at every reverse step to suppress error propagation.	79

4.3	Illustration of the DiffResist framework. (a) The 2D diffusion stage generates depth-wise concentration maps from random noise, with boundary-condition correction applied at each reverse step. (b) The super-resolution stage restores the full spatial and depth resolution.	81
4.4	Illustration of the TorchResist evaluation dataset built from LithoBench. The region of interest within each mask tile is highlighted by red dashed lines.	87
4.5	Visual comparison of TorchResist predictions against commercial-tool ground truth. All images are rendered at 1 nm/pixel.	89
4.6	TorchResist predictions at different resolutions	90
4.7	TorchResist convergence: the per-voxel absolute error $L_{\text{abs}}(N)$ relative to the 2000-step reference decreases steadily with the number of exposure steps N	91
4.8	Visual comparison of TorchResist output at 50 vs. 2000 exposure steps. (a–d) and (e–h) show two test samples: aerial image, output at 50 steps, output at 2000 steps, and their absolute difference. .	92
4.9	Qualitative comparison on two test samples. For each sample the leftmost subfigure is the ground-truth printed image, and the remaining three subfigures show the absolute difference between the ground truth and the printed image reconstructed from SVG, SVG-RNN, and DiffResist predictions, respectively. Lighter regions indicate larger error.	95
4.10	Per-layer prediction error: DiffResist vs. sequential baselines. . . .	96
4.11	Two-stage vs. standard linear noise schedule on downsampled data.	96
4.12	Ablation of super-resolution network variants.	97

4.13	Error heatmaps at different inference depths. (a) Input aerial image. Top row (b–d): absolute error at the first (top) depth step for each method; bottom row (e–g): absolute error at the last (bottom) depth step. Lighter areas indicate larger error. Only DiffResist maintains low error across the full depth range.	98
5.1	Pyramid-shaped IC geometry illustrating the point-query thermal estimation scenario. In many practical settings only the temperatures at a few queried locations (e.g., potential hotspots) are needed.	102
5.2	Illustration of a random-walk path in FastRW . WOG is used within the heat-source region; WOS accelerates transit through source-free homogeneous regions.	108
5.3	A random walk path in single-point estimation. Standard methods compute the integral from A to B. FastRW terminates at C and estimates the remaining contribution from C to infinity using the prior temperature field.	112
5.4	Random walk paths in multi-point estimation. Paths that traverse other query locations provide cross-point observations that FastRW exploits via Bayesian fusion.	115
5.5	Illustration of the three-layer 3D chip thermal model and the three power-map configurations.	117
5.6	Empirical validation of the truncation residual identity	120

List of Tables

3.1	Pattern diversity and legality on Layer-10001	57
3.2	Comparison of legality and diversity for conditional and free-size pattern generation (ChatPattern). ‘/’ refers to N/A. Topology selection and pattern modification calls are disabled for fair comparison of raw model quality.	59
3.3	Comparison between discrete and continuous diffusion for topology generation (unconditional DiffPattern-Flex , 100,000 patterns, Layer-10001).	63
3.4	Topology sampling efficiency under different fast-sampling factors m (NVIDIA RTX 3090 GPU).	65
3.5	Legalization efficiency comparison (Intel Xeon Gold 6326 CPU). .	65
4.1	Performance comparison of resist models on LithoBench. The lithography model is a commercial tool.	89
4.2	Inference efficiency of TorchResist on a single NVIDIA 3090 GPU.	89
4.3	Performance of TorchResist variants with different open-source lithography models.	90
4.4	Evaluation results on the LithoBench benchmark. EPE values are in nanometers.	94
4.5	Inference latency per sample, measured on an NVIDIA 3090 GPU.	95

5.1	Comparison of temperature estimation accuracy and efficiency. Error and StdDev are in Kelvin. FastRW achieves comparable accuracy with substantially fewer steps and paths.	119
5.2	Multi-point estimation results on Case 1. Larger group sizes yield more cross-point observations, further reducing error and variance.	121
5.3	Per-point time breakdown for FastRW on Case 1.	122
5.4	Trade-off between FEM prior accuracy and random walk efficiency on Case 1. DoF denotes the degrees of freedom of the FEM prior mesh.	122
6.1	Cross-chapter contribution matrix	126

Chapter 1

Introduction

1.1 Physical Design Closure in Advanced VLSI

Advanced very large scale integration (VLSI) technologies continue to push the limits of what can be reliably designed, fabricated, and operated on silicon. As feature dimensions shrink to the single-digit nanometer regime [35] and chip architectures evolve toward heterogeneous three-dimensional integration [82, 99], the gap between design intent and physical outcome widens. Design closure can no longer be understood as a purely logical or geometric task. A candidate design must satisfy nanometer-scale layout rules, survive the lithographic transfer from mask to wafer, and remain reliable under the thermal conditions induced by dense 3D packaging.

This dissertation studies data- and physics-driven computational methods for this broader design–fabrication–reliability chain. Manufacturability remains a central part of the story, especially for layout legality and lithographic printability [61, 129], but the scope is wider than conventional design for manufacturability (DFM). The common goal is to build physical modeling tools that are accurate enough to respect the underlying process, efficient enough to enter itera-

tive EDA workflows, and structured enough to support optimization, generation, or localized analysis. To make this goal concrete, the thesis focuses on three representative physical bottlenecks: layout pattern generation under design-rule constraints, photoresist modeling for lithographic pattern transfer, and localized thermal estimation for 3D IC reliability.

1.2 Physical Modeling Challenges

1.2.1 Layout Pattern Generation under Design-Rule Constraints

The first bottleneck lies at the design representation level: modern data-driven DFM methods require large and diverse libraries of legal layout patterns. Applications such as empirical design-rule exploration [30, 38], layout hotspot detection [9, 26, 119, 121], lithography simulation [43, 128], and optical proximity correction (OPC) [122, 137] all benefit from realistic pattern corpora. In practice, however, production layouts are proprietary and cannot be freely shared, forcing researchers to rely on synthetic generators [24, 27]. Rule-based augmentation approaches [74, 126] expand basic pattern units through flipping and rotation but provide limited structural diversity, while learning-based generators [117, 120, 134] improve diversity but can easily produce topologies that violate industrial design rules. The key tension is therefore not generation alone, but generation under hard legality constraints: a useful pattern generator must explore diverse layout topology while guaranteeing design-rule compliance on the final physical geometries.

1.2.2 Photoresist Modeling for Pattern Transfer

The second bottleneck appears during fabrication, where the designed mask pattern must be transferred onto the wafer through optical lithography and photoresist processing. In extreme ultraviolet (EUV) lithography at 13.5 nm wavelength, photoresist chemistry becomes intrinsically stochastic [4, 14]. High photon energy triggers non-deterministic secondary-electron cascades that produce line-edge roughness, micro-bridging, and contact-hole failures [97, 98], which are difficult to capture with deterministic threshold models developed for larger process nodes [51, 86]. Rigorous first-principle simulations, including Monte Carlo electron-scattering models [41, 42], can reproduce these effects but often require minutes to hours per microscopic clip. Such cost is incompatible with iterative lithography optimization, large-scale process exploration, or learning workflows that require many forward evaluations.

1.2.3 Localized Thermal Estimation for 3D IC Reliability

The third bottleneck emerges after fabrication and packaging. Heterogeneous integration and 3D ICs increase system-level density, but they also intensify thermal challenges because heat removal becomes more difficult in stacked and multi-material structures [67, 70]. Excessive temperature degrades performance, accelerates electromigration, and shortens device lifetime [133]. Steady-state thermal profiling requires solving heat-conduction partial differential equations (PDEs) over heterogeneous 3D domains with mixed boundary conditions. Analytical Green’s-function methods [103, 132] and compact thermal models [34, 91, 130] are efficient but restricted by simplified geometries or lumped abstractions, while finite-element methods (FEM) [44, 45, 46] provide high fidelity at the cost of assembling and solving a global system. During iterative floorplanning or thermal-

aware design exploration, however, designers often need temperatures only at a small number of candidate hotspot locations [93]. This makes full-field re-solving computationally wasteful and motivates localized point-query thermal estimation.

1.3 Limitations of Existing Methods

Although layout generation, resist modeling, and thermal analysis occur at different stages of the IC flow, existing approaches to these tasks share several structural limitations. These limitations reflect a mismatch between the available tools and the needs of modern physical design closure.

High computational cost. Rigorous model-based OPC and inverse lithography technology (ILT) require thousands of forward optical simulations per iteration, consuming substantial CPU-hours for a single critical layer [11, 37, 69, 105]. Detailed resist simulation can require expensive numerical PDE solves or stochastic particle simulation. For thermal analysis, full-chip FEM solvers must discretize the entire 3D volume even when only a few point temperatures are needed. In all three cases, computational cost prevents physical models from being used as frequently as iterative design, calibration, and exploration loops require.

Mismatch with iterative design workflows. Classical tools often expose final predictions but not the structured interfaces needed by modern EDA automation, where differentiable programming and learning-accelerated physical models have become increasingly important [3, 7]. Layout generators may produce samples without a reliable mechanism for enforcing hard DRC legality. Resist simulators implemented as fixed numerical pipelines are difficult to calibrate by gradient-based methods or insert into inverse lithography workflows. Thermal solvers that return a full global field do not exploit the locality of hotspot queries and provide limited reuse across correlated points, despite the sparse-query na-

ture of thermal-aware design exploration [50, 93]. The practical issue is therefore broader than differentiability alone: each domain needs an interface matched to its role in the design loop, whether that interface is a legalization solver, a differentiable physical operator, or a point-query estimator.

Data scarcity and weak physical guarantees. High-quality labeled data is scarce across the design–fabrication–reliability chain. Layout patterns from production designs are protected by foundry non-disclosure agreements, and synthetic alternatives may lack realism or diversity [136]. EUV resist profiles exhibit sample-to-sample variation [4, 14], making clean 3D supervision difficult to obtain. Thermal calibration data from fabricated chips is expensive and often available only at sparse sensor locations. Purely data-driven models trained under these conditions may overfit, hallucinate plausible-looking outputs, or violate known physical constraints such as design-rule legality, resist boundary conditions, and heat-conduction structure.

1.4 Physics-Driven Learning and Estimation

The limitations above suggest that neither conventional physical simulation nor purely data-driven learning is sufficient on its own. Physical simulation provides interpretability and constraint satisfaction, but it is often too slow or too rigid for iterative EDA use. Black-box learning can be fast, but without explicit structure it may require unrealistic amounts of data and may fail precisely where physical correctness matters most. This dissertation therefore adopts *physics-driven learning and estimation* as a unifying methodology.

The central idea is to place domain knowledge at the algorithmic interface where it can most directly improve reliability, efficiency, or controllability. This is broader than adding a physics penalty to a loss function, and it does not

require every method to be a neural network. Across the three technical chapters, physical structure plays three distinct roles:

1. **Physics as hard constraints.** In layout pattern generation, foundry design rules define exact geometric feasibility conditions. The learning model is responsible for exploring diverse discrete topologies, while a white-box legalization solver enforces DRC compliance on accepted physical layouts.
2. **Physics as differentiable operators and boundary conditions.** In photoresist modeling, the Dill exposure and Mack development equations provide compact, interpretable process structure. Rewriting these equations as differentiable tensor operations supports gradient-based calibration, while analytical boundary conditions anchor a diffusion-based predictor for 3D resist profiles.
3. **Physics as stochastic identities.** In thermal estimation, the Feynman–Kac representation connects the heat-conduction PDE to random-walk trajectories. A stopping-time residual identity converts long-horizon random-walk truncation into a continuation-value estimation problem, enabling early termination with prior correction.

These mechanisms differ in form, but they share the same principle: physical knowledge should not be applied only as a post-hoc validation step. Instead, geometric rules, photochemical equations, and stochastic PDE identities can define the representation, estimator, or correction term through which data-driven or randomized computation becomes usable in physical design closure.

1.5 Thesis Contributions

The contributions of this thesis are organized around three pillars across the design–fabrication–reliability flow. Each pillar embeds domain-specific physical structure into a modern computational method, yielding tools that are fast, physically grounded, and compatible with iterative EDA workflows.

1. **Scalable, legality-enforced layout pattern generation.** We propose DiffPattern-Flex [111], a two-stage framework that decouples layout synthesis into topology generation and geometric legalization. A discrete denoising diffusion probabilistic model operates on a compact tensor representation, termed the *Deep Squish Pattern*, to learn and sample diverse layout topologies. A subsequent nonlinear bounding optimization stage enforces strict design-rule compliance, achieving a 100% DRC legality rate among the final accepted legalized outputs in the evaluated setting. Building on this foundation, we further introduce ChatPattern [110], a large-language-model-powered layout generation framework that orchestrates generative tools through natural-language instructions via an agentic pipeline, supporting in-painting, out-painting, and free-size pattern customization.
2. **Physics-constrained, differentiable photoresist modeling.** We develop TorchResist [112], an open-source differentiable photoresist simulator that translates the classical Dill exposure and Mack development equations into a continuous PyTorch computation graph. With fewer than 20 calibratable parameters, it enables gradient-based calibration and seamless integration into inverse lithography workflows. Complementing this, we propose DiffResist [114], a physics-guided continuous diffusion model for predicting stochastic 3D resist profiles. By injecting physical priors (optical aerial images and simulator-derived boundary conditions) into the reverse

diffusion process, the model bridges the gap between deterministic intensity maps and the highly variable resist topographies observed under EUV conditions.

3. **Efficient random-walk thermal analysis.** We introduce FastRW [113], an efficient prior-corrected random-walk method for steady-state thermal PDEs based on the Feynman–Kac stochastic representation. The key theoretical contribution is a *stopping-time truncation residual identity* that exactly decomposes the estimation error of a truncated walk into a computable residual term, enabling an adaptive prior-corrected early termination strategy that eliminates unnecessary walk steps without sacrificing accuracy. We further develop a Bayesian multi-point posterior fusion scheme that aggregates information across correlated spatial queries. Together, these techniques yield a $6.5\times$ speedup over a strong Feynman–Kac random-walk baseline while maintaining high accuracy against FEM reference solutions, making the method practical for on-demand thermal queries during iterative 3D floorplanning.

1.6 Thesis Organization

The remainder of this thesis is organized as follows.

Chapter 2 establishes the background and mathematical preliminaries shared across the subsequent technical chapters. It covers computational lithography fundamentals (optical imaging, OPC, and ILT), photoresist physics (Dill exposure and Mack development models), steady-state thermal PDE formulations with mixed boundary conditions, denoising diffusion probabilistic models in both continuous and discrete settings, and the basics of large language model agents.

Chapter 3 presents the layout pattern generation framework. It details the dis-

crete diffusion formulation for topology generation, the nonlinear legalization system that enforces strict design-rule compliance on accepted outputs, and an LLM-driven customization pipeline for free-size pattern synthesis. Experiments under industrial-grade design rules demonstrate 100% legality for accepted DiffPatternFlex outputs and improved diversity over prior generators.

Chapter 4 develops photoresist modeling methods that combine differentiable simulation with physics-constrained diffusion. It describes the construction of a differentiable PDE simulator and a physics-guided diffusion architecture for stochastic 3D profile prediction under EUV conditions. Evaluations on benchmark resist data and simulator-generated 3D profiles validate the accuracy and efficiency of the proposed models.

Chapter 5 introduces a random-walk method for localized steady-state thermal analysis. It provides the theoretical foundation for mixed-boundary treatment via transition probability construction, the truncation residual identity enabling adaptive early termination, and the Bayesian fusion framework for multi-point queries. Benchmarks on 3D IC thermal testcases confirm the claimed speedup and accuracy.

Chapter 6 summarizes the contributions and findings of this thesis and discusses directions for future research at the intersection of physics-driven learning and electronic design automation.

Chapter 2

Background and Preliminaries

This chapter establishes the physical principles and mathematical foundations used in the three technical chapters that follow. Its organization mirrors the thesis structure: Section 2.1 introduces the layout legality, discrete generative modeling, and LLM-agent background used in Chapter 3; Section 2.2 reviews lithographic pattern transfer, photoresist physics, and continuous diffusion models for Chapter 4; and Section 2.3 formulates the steady-state thermal PDE and its random-walk representation for Chapter 5. This order emphasizes how each technical chapter places physical structure at a different algorithmic interface: design rules for layout generation, photochemical equations for resist modeling, and stochastic PDE representations for thermal estimation.

2.1 Layout Pattern Generation Background

2.1.1 Design Rules and DRC

At the layout level, manufacturability is enforced through *design rules*: technology-specific geometric constraints such as minimum width, minimum spacing, enclosure, and area bounds. These rules are a central component of DFM, which seeks

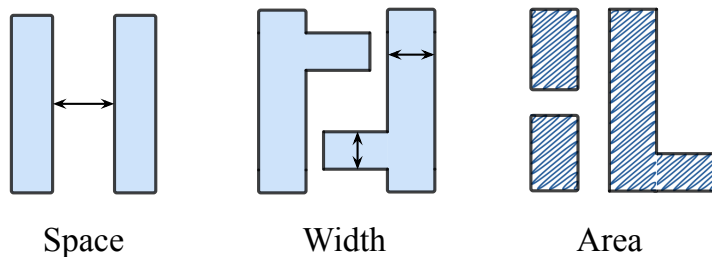


Figure 2.1: Illustration of representative design rules for layout patterns.

to reduce the gap between intended designs and manufacturable silicon [61, 129]. Design rule checking (DRC) verifies whether a layout satisfies all such constraints [30, 38, 55]. Representative width, spacing, and area rules are illustrated in Fig. 2.1. A layout pattern is called *legal* if it passes DRC; otherwise, even a visually plausible generated sample cannot be used as a valid design artifact. The legality requirement is the hard physical constraint enforced by the legalization system in Chapter 3.

2.1.2 Discrete Layout Representations

Although an IC layout is a geometric object composed of polygons, learning-based generators typically operate on compact tensor representations. A drawn mask layer can be viewed as a binary field in which each location indicates either polygon or whitespace. Directly rasterizing the layout at the manufacturing resolution is inefficient because the informative polygon boundaries occupy only a small fraction of the image domain.

The Squish Pattern representation [27] addresses this sparsity through a lossless topology–geometry factorization. It first places horizontal and vertical scan lines along polygon edges, thereby partitioning the layout into a nonuniform grid. The intervals between adjacent scan lines are stored in two positive geometric vectors, Δ_x and Δ_y . The structural content is stored separately in a binary

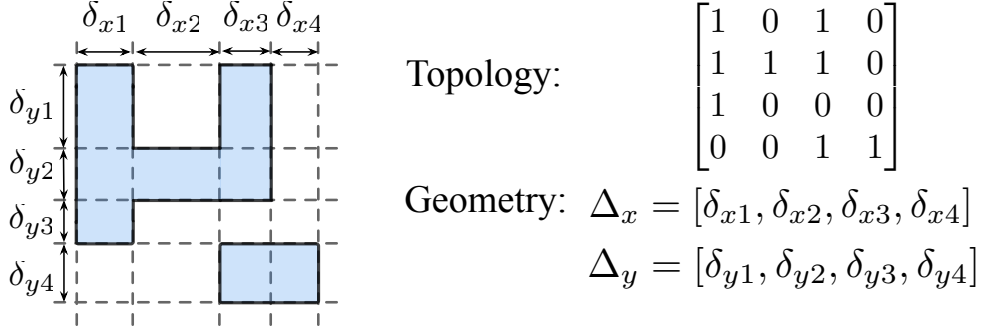


Figure 2.2: Squish Pattern representation of a layout pattern.

topology matrix, where an entry equal to one denotes polygon occupancy in the corresponding grid cell and zero denotes empty space. For compatibility with pixel-based learning pipelines, this topology matrix is commonly padded into a square form [121]. The representation has therefore become a standard compact encoding for layout pattern generation methods [109, 110, 120, 134]. Fig. 2.2 illustrates the resulting decomposition from a polygonal layout to scan-line intervals and a binary topology matrix.

This topology–geometry separation is important for the methods studied in this thesis: the learned model can focus on categorical topology generation, while a deterministic solver assigns physical dimensions under DRC constraints. Building on the same idea, the Deep Squish Pattern introduced in Chapter 3 folds a $\sqrt{C} \times \sqrt{C}$ local patch of the 2D topology matrix into one spatial location with C binary channels, producing a topology tensor $\mathbf{T} \in \{0, 1\}^{C \times M \times M}$. This spatial-to-channel fold reduces the spatial size processed by the neural network while preserving the original logical resolution after unfolding. It also avoids a naive integer encoding of local bit patterns, whose state space grows exponentially and whose bit positions carry unequal numerical weights.

Because the topology tensor is discrete, treating it as a continuous image and then thresholding real-valued outputs is structurally mismatched. The generative

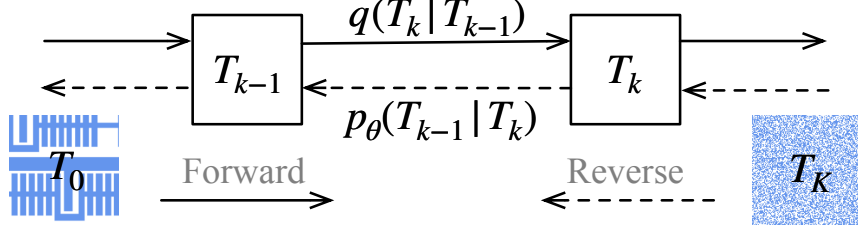


Figure 2.3: Illustration of the denoising diffusion process.

model should instead respect the categorical state space throughout the sampling process. This motivates the discrete diffusion formulation reviewed next.

2.1.3 Discrete Denoising Diffusion (D3PM)

Diffusion probabilistic models [87] define a Markov chain [57] that gradually corrupts data and then learns to reverse this corruption process. The denoising diffusion formulation of Ho et al. [32] popularized this approach for continuous data, but many layout structures are inherently categorical. Discrete denoising diffusion probabilistic models (D3PMs) [1] extend the diffusion framework to categorical variables, making them suitable for binary layout topologies. The generic forward and reverse denoising structure is illustrated in Fig. 2.3; the categorical formulation below instantiates the same Markov-chain idea with discrete transition matrices.

Categorical Forward Process. For a categorical variable $x_t \in \{1, 2, \dots, K\}$, the forward process is defined by a sequence of transition matrices $\mathbf{Q}_t \in \mathbb{R}^{K \times K}$:

$$q(x_t = j \mid x_{t-1} = i) = [\mathbf{Q}_t]_{ij}. \quad (2.1)$$

Using one-hot row vectors, this can be written compactly as

$$q(\mathbf{x}_t \mid \mathbf{x}_{t-1}) = \text{Cat}(\mathbf{x}_t; \mathbf{x}_{t-1} \mathbf{Q}_t). \quad (2.2)$$

The marginal distribution after t steps is obtained by multiplying transition matrices:

$$\bar{\mathbf{Q}}_t = \mathbf{Q}_1 \mathbf{Q}_2 \cdots \mathbf{Q}_t, \quad (2.3)$$

so that $q(\mathbf{x}_t \mid \mathbf{x}_0) = \text{Cat}(\mathbf{x}_t; \mathbf{x}_0 \bar{\mathbf{Q}}_t)$. A common K -category transition is the uniform corruption kernel

$$\mathbf{Q}_t^{\text{uni}} = (1 - \beta_t^{\text{uni}}) \mathbf{I} + \frac{\beta_t^{\text{uni}}}{K} \mathbf{1}\mathbf{1}^\top. \quad (2.4)$$

For binary layout tensors, Chapter 3 uses the equivalent binary-flip parameterization

$$\mathbf{Q}_t^{\text{flip}} = \begin{bmatrix} 1 - \beta_t^{\text{flip}} & \beta_t^{\text{flip}} \\ \beta_t^{\text{flip}} & 1 - \beta_t^{\text{flip}} \end{bmatrix}, \quad (2.5)$$

where β_t^{flip} is the probability of flipping to the other binary state. The two parameterizations have the same binary off-diagonal transition probability when $\beta_t^{\text{flip}} = \beta_t^{\text{uni}}/2$; in Chapter 3, the symbol β_k follows the binary-flip convention.

Discrete Reverse Process. The reverse process starts from a noisy categorical prior and iteratively reconstructs the clean topology. The neural network $p_\theta(\tilde{\mathbf{x}}_0 \mid \mathbf{x}_t)$ predicts the distribution of the clean state $\mathbf{x}_0$. The true posterior follows from Bayes' rule:

$$q(\mathbf{x}_{t-1} \mid \mathbf{x}_t, \mathbf{x}_0) = \frac{q(\mathbf{x}_t \mid \mathbf{x}_{t-1}) q(\mathbf{x}_{t-1} \mid \mathbf{x}_0)}{q(\mathbf{x}_t \mid \mathbf{x}_0)}. \quad (2.6)$$

Substituting the predicted clean state gives the parameterized reverse transition:

$$p_\theta(\mathbf{x}_{t-1} \mid \mathbf{x}_t) = \sum_{\tilde{\mathbf{x}}_0=1}^K q(\mathbf{x}_{t-1} \mid \mathbf{x}_t, \tilde{\mathbf{x}}_0) p_\theta(\tilde{\mathbf{x}}_0 \mid \mathbf{x}_t). \quad (2.7)$$

The model is trained to match the true discrete posterior, ensuring that intermediate and final samples remain valid categorical distributions. The original D3PM work used a PixelCNN++ backbone [79]; Chapter 3 adopts a convolutional architecture tailored to binary layout tensors.

2.1.4 Large Language Models and Agents

Beyond batch pattern generation, Chapter 3 also requires an interface that can translate high-level human intent into concrete generation, modification, extension, and legalization calls. This motivates a brief review of LLMs and tool-using agents.

Transformer and LLM Evolution. The foundation of modern LLMs is the Transformer architecture [101], which replaces recurrent computation with parallel self-attention. For input tokens embedded as $\mathbf{X} \in \mathbb{R}^{N \times d}$, the model projects the data into Query ($\mathbf{Q}$), Key ($\mathbf{K}$), and Value ($\mathbf{V}$) matrices:

$$\mathbf{Q} = \mathbf{X}\mathbf{W}_Q, \quad \mathbf{K} = \mathbf{X}\mathbf{W}_K, \quad \mathbf{V} = \mathbf{X}\mathbf{W}_V. \quad (2.8)$$

Scaled dot-product attention is then computed as

$$\text{Attention}(\mathbf{Q}, \mathbf{K}, \mathbf{V}) = \text{softmax}\left(\frac{\mathbf{Q}\mathbf{K}^\top}{\sqrt{d_k}}\right) \mathbf{V}. \quad (2.9)$$

This mechanism enables long-range dependency modeling in natural-language design prompts. BERT [17] demonstrated bidirectional pre-training, while the GPT family adopted autoregressive pre-training, scaling from GPT-2 [72] to GPT-3 [5]. Instruction following and reinforcement learning from human feedback (RLHF) [64] led to GPT-4-class models [60]; open-weight models such as LLaMA [100] and DeepSeek [15] have further broadened access to capable LLMs.

Tool Use and Autonomous Agents. Modern LLMs can invoke external tools rather than only generate text. Toolformer [81] showed that language models can learn API use in a self-supervised manner, while Chain-of-Thought prompting [116] and ReAct [125] combine reasoning traces with executable actions. This agent paradigm has been adopted in EDA: ChatEDA [31] uses an LLM agent to drive tool flows, and ChipNeMo [48] shows the value of domain adaptation for chip-design tasks. In ChatPattern, the LLM agent does not directly emit a full

binary layout tensor; instead, it decomposes user requirements into backend calls for topology generation, in-painting, out-painting, and legalization.

2.2 Lithographic Pattern Transfer and Photore-sist Modeling

2.2.1 Optical Lithography and Aerial Images

Modern semiconductor manufacturing relies on optical lithography to transfer circuit patterns from a photomask onto a silicon wafer coated with photore-sist [18, 51]. The full pattern-transfer chain is illustrated in Fig. 2.4: collimated illumination passes through the photomask and the projection optics and forms an intensity field on top of the resist film, which after exposure and development becomes a three-dimensional topography on the wafer. When the minimum feature size falls below the illumination wavelength, diffraction and optical proximity effects distort the printed image relative to the target layout. The raw projected light field on the wafer, called the *aerial image*, is computed by convolving the mask transmittance function with the optical system’s point-spread function, or equivalently with Hopkins Transfer Cross-Coefficient kernels [66]. As shown on the right of Fig. 2.4, this aerial image is a two-dimensional intensity field of size $H \times W$, while the resist response is a three-dimensional volumetric profile of size $H \times W \times D$; bridging this 2D-to-3D gap is the central modeling task of Chapter 4.

2.2.2 OPC and ILT

OPC compensates for diffraction-induced distortions by modifying the mask geometry. Rule-Based OPC (RB-OPC) [63] applies predefined adjustments such as scatter bars, hammerheads, and corner serifs. Model-Based OPC (MB-OPC) [11,

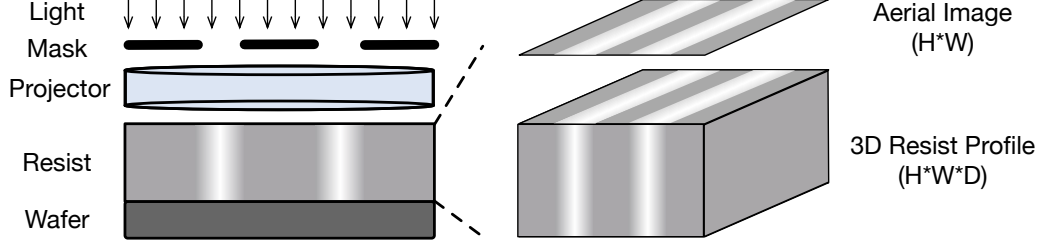


Figure 2.4: Overview of optical lithography and 3D resist modeling. Left: light is patterned by the mask, focused by the projection optics, and projected onto the photoresist film coating the wafer. Right: the resulting aerial image ($H \times W$) drives a volumetric photochemical response that yields a three-dimensional resist profile ($H \times W \times D$).

[137] improves accuracy by fragmenting layout edges and iteratively displacing fragments to reduce edge placement error (EPE) between the simulated contour and the target.

ILT [69] further generalizes mask optimization by treating the mask as a continuous pixel grid:

$$\min_{\mathbf{M}} \|f_{\text{resist}}(\mathbf{H} * \mathbf{M}) - \mathbf{Z}_{\text{target}}\|_2^2 + \lambda \mathcal{R}(\mathbf{M}), \quad (2.10)$$

where $\mathbf{M}$ is the mask transmittance, $\mathbf{H}$ encodes the optical transfer function, f_{resist} is a resist model, $\mathbf{Z}_{\text{target}}$ is the desired wafer pattern, and $\mathcal{R}(\mathbf{M})$ regularizes mask manufacturability. Solved with gradient-based methods, ILT can produce free-form curvilinear masks that improve process windows [68]. Recent work has explored machine-learning-accelerated ILT [6, 105] and open-source differentiable lithography frameworks [7], but scalable inverse optimization still depends strongly on the speed and differentiability of the forward resist model.

2.2.3 Resist Modeling Landscape

The photoresist film mediates the transfer of the aerial image into a physical 3D topography on the wafer [51], as previewed on the right of Fig. 2.4. Accurate resist modeling is essential for predictive lithography simulation [23, 86]. Compact approaches approximate the resist response by thresholding the aerial image, either with a fixed threshold or with variable-threshold corrections [29, 73]. Neural-network surrogates have also been used to approximate wafer contours or resist development behavior [115, 131]. These approaches are fast, but they may discard the photochemical structure needed for robust extrapolation and gradient-based calibration. The following two subsections therefore review the classical Dill exposure and Mack development models, which form the physical basis for TorchResist and DiffResist.

2.2.4 Dill Exposure Model

Following the Dill model [19], light absorption in the resist obeys the Lambert–Beer law. Neglecting reflections, the intensity $\mathbf{I}(h, t)$ at depth h from the resist–air interface after exposure time t satisfies

$$\frac{\partial \mathbf{I}(h, t)}{\partial h} = -\mathbf{I}(h, t) [a_1 \mathbf{m}_1(h, t) + a_2 \mathbf{m}_2(h, t) + a_3 \mathbf{m}_3(h, t)], \quad (2.11)$$

where a_i and $\mathbf{m}_i$ denote the molar absorption coefficients and concentrations of the photoactive inhibitor ($i = 1$), base resin ($i = 2$), and reaction product ($i = 3$), respectively. The photolytic destruction of the inhibitor follows

$$\frac{\partial \mathbf{m}_1(h, t)}{\partial t} = -\mathbf{m}_1(h, t) \mathbf{I}(h, t) C, \quad (2.12)$$

where C is the fractional decay rate per unit intensity. The surface intensity is set by the aerial image: $\mathbf{I}(0, t) = \mathbf{R}(x, y)$.

By imposing uniform initial concentrations, a non-bleaching resin, and conservation of matter, the fractional inhibitor concentration $\mathbf{M}(h, t) = \mathbf{m}_1(h, t)/m_{10}$ yields the consolidated system

$$\begin{aligned}\frac{\partial \mathbf{I}(h, t)}{\partial h} &= -\mathbf{I}(h, t)[A \mathbf{M}(h, t) + B], \\ \frac{\partial \mathbf{M}(h, t)}{\partial t} &= -\mathbf{I}(h, t) \mathbf{M}(h, t) C,\end{aligned}\tag{2.13}$$

where $A = (a_1 - a_3)m_{10}$ and $B = a_2m_{20} + a_3m_{10}$ are grouped Dill parameters. The initial and boundary conditions are

$$\begin{aligned}\mathbf{M}(h, 0) &= 1, \\ \mathbf{M}(0, t) &= e^{-\mathbf{R}Ct}, \\ \mathbf{I}(h, 0) &= \mathbf{R}e^{-(A+B)h}, \\ \mathbf{I}(0, t) &= \mathbf{R}.\end{aligned}\tag{2.14}$$

The top-layer condition $\mathbf{M}(0, T) = e^{-\mathbf{R}CT}$ after total exposure duration T becomes a physical boundary constraint in DiffResist. For chemically amplified resists used in EUV lithography, additional acid-diffusion and catalytic-amplification steps occur during post-exposure bake [41, 42]; achieving uniform critical dimensions at sub-20 nm remains a major materials challenge [96, 97].

2.2.5 Mack Development Model

Once the exposure-modified concentration profile $\mathbf{M}(h, T)$ is obtained, the resist is immersed in developer solution. The Mack bulk dissolution-rate model [52] maps concentration to local development rate:

$$\mathbf{r} = r_{\max} \frac{(a+1)(1-\mathbf{M})^n}{a+(1-\mathbf{M})^n} + r_{\min},\tag{2.15}$$

where $r_{\max}$ and $r_{\min}$ are the maximum and minimum dissolution rates, n governs the reaction order, and $a = \frac{n+1}{n-1}(1-m_{\text{TH}})^n$ controls the inflection point through

the threshold concentration m_{TH} . The developer front propagates through the resist volume; at every spatial location (x, y, z) , the arrival time $\mathbf{T}$ satisfies the Eikonal equation $|\nabla \mathbf{T}| = 1/\mathbf{r}$. This equation can be solved with fast-marching level-set methods [62, 83] or path-based acceleration techniques [13]. The final developed profile is the iso-contour $\mathbf{T}(x, y, z) = t_{\text{dev}}$.

2.2.6 Continuous Denoising Diffusion

DiffResist models 3D resist profile prediction as a continuous generative process. Continuous denoising diffusion probabilistic models (DDPMs) corrupt real-valued data with Gaussian noise and learn to reverse the corruption [32]. The broader diffusion family has connections to non-equilibrium statistical physics [87], accelerated sampling [88], score matching [89], and stochastic differential equations [90]; a survey is provided in [123].

Forward Process. Given a data point $\mathbf{x}_0 \sim q(\mathbf{x}_0)$, the forward process adds Gaussian noise over T steps according to a variance schedule $\beta_1, \dots, \beta_T$:

$$q(\mathbf{x}_t \mid \mathbf{x}_{t-1}) = \mathcal{N}(\mathbf{x}_t; \sqrt{1 - \beta_t} \mathbf{x}_{t-1}, \beta_t \mathbf{I}). \quad (2.16)$$

Let $\alpha_t = 1 - \beta_t$ and $\bar{\alpha}_t = \prod_{s=1}^t \alpha_s$. The marginal at step t is

$$q(\mathbf{x}_t \mid \mathbf{x}_0) = \mathcal{N}(\mathbf{x}_t; \sqrt{\bar{\alpha}_t} \mathbf{x}_0, (1 - \bar{\alpha}_t) \mathbf{I}), \quad (2.17)$$

or equivalently $\mathbf{x}_t = \sqrt{\bar{\alpha}_t} \mathbf{x}_0 + \sqrt{1 - \bar{\alpha}_t} \boldsymbol{\epsilon}$ with $\boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$.

Reverse Process and Objective. The generative model approximates the reverse conditional with

$$p_{\theta}(\mathbf{x}_{t-1} \mid \mathbf{x}_t) = \mathcal{N}(\mathbf{x}_{t-1}; \boldsymbol{\mu}_{\theta}(\mathbf{x}_t, t), \boldsymbol{\Sigma}_{\theta}(\mathbf{x}_t, t)). \quad (2.18)$$

In practice, the network predicts the injected noise $\boldsymbol{\epsilon}_{\theta}(\mathbf{x}_t, t)$. The simplified train-

ing loss is

$$\mathcal{L}_{\text{simple}}(\theta) = \mathbb{E}_{t, \mathbf{x}_0, \epsilon} \left[\left\| \epsilon - \epsilon_{\theta}(\sqrt{\bar{\alpha}_t} \mathbf{x}_0 + \sqrt{1 - \bar{\alpha}_t} \epsilon, t) \right\|^2 \right]. \quad (2.19)$$

The denoising network is commonly implemented as a U-Net [77]. Latent diffusion models [76] reduce cost by applying diffusion in compressed latent spaces, and the encoder–decoder interpretation is related to variational autoencoders [20]. In Chapter 4, the continuous diffusion formalism is adapted to a depth-wise resist profile generation problem and constrained by the boundary condition in equation 2.14.

2.3 Steady-State Thermal Analysis

2.3.1 Thermal Integrity in 3D ICs

Thermal integrity is a first-order design concern in modern ICs. Rising power density and complex 3D packaging create hotspots that degrade performance and reliability [34, 67, 99]. A broad survey of chip-level thermal simulation methods can be found in [94]. Many design-space exploration tasks, including hotspot identification and thermal-aware placement [133], require repeated thermal evaluation but often need temperatures only at sparse locations rather than a full-chip field.

2.3.2 Classical Thermal Simulation Methods

Classical deterministic approaches include analytical Green’s-function methods [103], finite-element and multigrid solvers [45], alternating-direction implicit (ADI) schemes [106], and compact RC-network models such as HotSpot [34] and PACT [130]. These methods are accurate and widely used, but they typically compute the full temperature field. Stochastic random-walk methods instead estimate the solution

locally and are therefore well suited to point-query settings [71, 118]. Chapter 5 builds on this point-query viewpoint and focuses on reducing the long-path cost of reflected Feynman–Kac estimators.

2.3.3 Heat-Conduction PDE and Boundary Conditions

The steady-state temperature distribution $T(\mathbf{x})$ over a three-dimensional heterogeneous chip domain $\Omega \subset \mathbb{R}^3$ satisfies the heat-conduction equation with mixed boundary conditions:

$$\begin{aligned} -\nabla \cdot (k(\mathbf{x})\nabla T(\mathbf{x})) &= p(\mathbf{x}), & \mathbf{x} \in \Omega, \\ T(\mathbf{x}) &= \phi_D(\mathbf{x}), & \mathbf{x} \in \partial\Omega_D, \\ \frac{\partial T}{\partial \mathbf{n}} &= \phi_N(\mathbf{x}), & \mathbf{x} \in \partial\Omega_N, \\ \frac{\partial T}{\partial \mathbf{n}} + c(\mathbf{x})T(\mathbf{x}) &= \phi_R(\mathbf{x}), & \mathbf{x} \in \partial\Omega_R, \end{aligned} \tag{2.20}$$

where $p(\mathbf{x})$ is volumetric heat generation density and $k(\mathbf{x})$ is thermal conductivity. The boundary $\partial\Omega$ is partitioned into Dirichlet, Neumann, and Robin subsets. Dirichlet boundaries model prescribed temperatures such as ideal heat sinks; Neumann boundaries model prescribed heat flux or adiabatic surfaces; and Robin boundaries model convective or radiative exchange with the environment.

At a material interface Γ_{ij} between conductivities k_i and k_j , physical continuity requires

$$\begin{aligned} T_i &= T_j, \\ k_i \frac{\partial T_i}{\partial \mathbf{n}_i} &= k_j \frac{\partial T_j}{\partial \mathbf{n}_i}. \end{aligned} \tag{2.21}$$

Within each homogeneous material subdomain where k is constant, the PDE can be normalized to the Poisson form used by the stochastic representation:

$$\begin{aligned} -\frac{1}{2}\nabla^2 T(\mathbf{x}) &= f(\mathbf{x}), \\ f(\mathbf{x}) &= \frac{p(\mathbf{x})}{2k}. \end{aligned} \tag{2.22}$$

Material heterogeneity is then handled through interface continuity and, after discretization, conductance-weighted random-walk transition probabilities. The point-query problem studied in Chapter 5 is:

Problem 1. Given the heat-conduction problem defined by equation 2.20 with mixed boundary conditions and a set of query locations $\{\mathbf{x}_q\}$ in Ω , estimate the temperature $T(\mathbf{x}_q)$ at each query location.

2.3.4 Feynman–Kac Stochastic Representation

Under the normalized Poisson convention in equation 2.22, the Feynman–Kac formula connects the deterministic PDE to expectations over stochastic processes [22, 39, 59]. Let X_t denote a standard reflecting Brownian motion (SRBM) in Ω , initialized at $X_0 = \mathbf{x}$. The process reflects at Neumann and Robin boundaries and is absorbed upon hitting a Dirichlet boundary. With the normalized reflected Brownian-motion convention associated with generator $\frac{1}{2}\nabla^2$, the outward-normal sign and boundary-local-time normalization are absorbed into the coefficients ϕ_N , ϕ_R , and c . The temperature admits the representation

$$\begin{aligned} T(\mathbf{x}) = & \mathbb{E} \left\{ \int_0^\infty \hat{e}_c(t) f(X_t) dt \right\} \\ & + \mathbb{E} \{ \hat{e}_c(t_D) \phi_D(X_{t_D}) \} \\ & + \mathbb{E} \left\{ \int_0^\infty \hat{e}_c(t) \phi_N(X_t) dL(t) \right\} \\ & + \mathbb{E} \left\{ \int_0^\infty \hat{e}_c(t) \phi_R(X_t) dL(t) \right\}, \end{aligned} \tag{2.23}$$

where t_D is the first hitting time of the Dirichlet boundary and $L(t)$ is the boundary local time, which increases when the process reflects at Neumann or Robin boundaries [21, 139]. The multiplicative Feynman–Kac weight is

$$\hat{e}_c(t) = \exp \left(- \int_0^t c(X_s) dL(s) \right), \tag{2.24}$$

with $c(\mathbf{x}) = h(\mathbf{x})/k(\mathbf{x})$ evaluated on Robin boundaries. The four terms in equation 2.23 correspond to volumetric heat accumulation, Dirichlet absorption, Neumann flux contribution, and Robin convective contribution.

In practice, the infinite integrals are truncated. Simulation terminates when either the path hits a Dirichlet boundary or the weight $\hat{e}_c(t)$ decays below a pre-defined threshold. Chapter 5 studies how to correct this truncation in reflection-dominated regimes.

2.3.5 Random Walk Discretizations

The continuous SRBM is approximated by discrete random walks. Two complementary discretizations are used in this thesis.

Walk-on-Grid (WOG). In heat-source regions, the domain is discretized into a uniform voxel grid with per-voxel material and power-density values [36, 71]. The Brownian motion is approximated by random jumps between adjacent grid nodes with conductance-weighted transition probabilities:

$$\begin{aligned} P(a, b) &= \frac{G(a, b)}{\sum_{x \in \text{adj}(a)} G(a, x)}, \\ G(a, b) &= \frac{A_{ab}}{d_{ab}^{(a)}/k_a + d_{ab}^{(b)}/k_b}, \end{aligned} \tag{2.25}$$

where $\text{adj}(a)$ is the set of nodes adjacent to a , $G(a, b)$ is the conductance across the shared interface, A_{ab} is interface area, $d_{ab}^{(a)}$ and $d_{ab}^{(b)}$ are distances from voxel centers to the shared interface, and k_a, k_b are the local thermal conductivities.

Walk-on-Spheres (WOS). In homogeneous source-free regions, the Poisson equation reduces to Laplace's equation. The mean-value property of harmonic functions gives

$$T(\mathbf{x}) = \frac{1}{4\pi} \int_0^{2\pi} \int_0^\pi T(r, \omega, \theta) \sin \theta \, d\theta \, d\omega. \tag{2.26}$$

The WOS algorithm [56] exploits this property by jumping from the center of a sphere to a uniformly sampled point on its surface, with the sphere radius chosen to avoid crossing boundaries or material interfaces. Recent variants such as Walk on Stars [80] extend sphere-jumping ideas to Neumann boundaries, and GPU implementations [49] improve throughput for large IC geometries. Following established thermal random-walk practice [47, 124], Chapter 5 applies WOG in heat-source regions and WOS in source-free regions.

2.4 Chapter Summary

This chapter has organized the necessary background in the same order as the technical chapters. Section 2.1 introduced design rules, categorical layout representations, D3PMs, and LLM agents for the layout generation methods in Chapter 3. Section 2.2 reviewed lithographic transfer, OPC/ILT, Dill–Mack photoresist physics, and continuous diffusion models for the photoresist modeling framework in Chapter 4. Section 2.3 formulated the steady-state thermal PDE, its Feynman–Kac representation, and the WOG/WOS discretizations used by FastRW in Chapter 5. Together, these preliminaries instantiate the thesis theme introduced in Chapter 1: physical knowledge can appear as hard design constraints, differentiable process equations, or stochastic representations, depending on where it best supports the design–fabrication–reliability flow.

Chapter 3

Layout Pattern Generation via Generative Learning

3.1 Introduction

This chapter addresses the design-stage bottleneck identified in Chapter 1: the scarcity of diverse, design-rule-compliant layout data that underpins data-driven manufacturability analysis. Drawing on the discrete diffusion models and LLM agent framework introduced in Chapter 2, we develop a physics-driven generative pipeline in which foundry design rules are enforced as hard structural constraints rather than learned implicitly from data.

The compilation of diverse, rule-compliant layout pattern libraries is a foundational prerequisite for data-driven DFM methodologies. Advanced applications, ranging from hotspot detection [9, 26, 119] and lithography simulation [8, 43, 128] (see Chapter 4) to empirical design rule exploration [30, 38, 55], heavily depend on the availability of expansive layout datasets to effectively train high-capacity machine learning models. However, the proprietary nature of industrial layout geometries makes acquiring such datasets prohibitively difficult. Consequently,

researchers have turned to synthetic layout generation to bridge this data gap.

Early attempts at layout synthesis relied predominantly on rule-based heuristics, which expanded a small set of basic geometric units via rudimentary augmentations such as flipping, rotating, and random concatenations. These approaches inherently suffer from limited structural diversity and fail to capture the complex, holistic interactions present in actual production layouts. To overcome these limitations, learning-based generative models have been recently introduced [109, 117, 120, 134]. These models learn the underlying distribution of a training set to synthetically produce novel geometries.

Traditional learning-based methods can be broadly categorized into pixel-based and sequential-based paradigms. Sequential-based methods model layouts as ordered polygon sequences composed of vertices and edges. While effective for simple shapes, they struggle with the intricate, multi-polygon spatial correlations inherent to advanced nodes. Conversely, pixel-based approaches treat the layout as a binary image synthesis problem. To mitigate the computational inefficiency caused by the geometric sparsity of layout images, researchers introduced the *Squish Pattern* technique [27], a lossless compression scheme that encodes high-resolution layouts into compact 2D binary topology matrices and corresponding geometric interval vectors.

Despite the efficiency gains provided by the Squish Pattern representation, existing pixel-based generative models (typically based on continuous state spaces like GANs [28] or standard diffusion models [32]) still exhibit significant shortcomings along three escalating dimensions. First, they attempt to map continuous generative outputs to binary discrete states (0 for space, 1 for shape) through arbitrary thresholding, which severely degrades sample quality and causes rule compliance to plummet. Second, they generate fixed-size topologies unconditionally, rendering them incapable of responding to specific, localized design requirements

such as pattern repair and conditional layout extension. Third, they offer no intuitive mechanism for specifying complex, multi-step generation requirements, forcing users to manually orchestrate low-level tool parameters.

This chapter presents a multi-tier framework that addresses these shortcomings with two connected components: **DiffPattern-Flex** supplies the compact representation, discrete generator, and legalization engine for fixed-size pattern libraries, while **ChatPattern** reuses this backend to support conditional and language-guided customization.

1. **Compact Representation.** A *Deep Squish Pattern* encoding (Section 3.4) compresses sparse layout geometries into multi-channel binary tensors, drastically reducing the spatial dimensionality the neural network must process while preserving lossless layout resolution.
2. **Discrete Generation.** A discrete diffusion model (DiffPattern-Flex [111], Section 3.5) synthesizes diverse topologies natively in the binary state space, eliminating the thresholding artifacts inherent to continuous generative models. Reliable topology enhancement (Section 3.7) further enriches the training distribution.
3. **Constraint-Based Legalization.** A nonlinear programming system (Section 3.6) assigns physical dimensions to generated topologies under hard design-rule constraints, enforcing DRC compliance on accepted legalized outputs. Because the legalization is decoupled from the generative model, changes in foundry rules require no retraining when the generated topology remains feasible.
4. **Interactive Customization.** Property-conditional generation, localized In-Painting for defect repair, and iterative Out-Painting for larger patterns are orchestrated by an expert LLM agent (**ChatPattern** [110], Section 3.8)

that accepts natural-language specifications, transforming the batch generator into a flexible, interactive pattern design tool.

The remainder of this chapter is organized as follows. Section 3.2 surveys the related work. Section 3.3 provides a high-level tour of the generation pipeline and its training-time and deployment-time extensions. Section 3.4 details the Deep Squish Pattern encoding. Section 3.5 describes the core discrete diffusion model for topology generation. Section 3.6 presents the deterministic legalization system that converts generated topologies into DRC-clean layout patterns. Section 3.7 introduces reliable topology enhancement, a training-time data enrichment procedure that reuses the legalization system as a structural-validity gate. Section 3.8 extends the generator with conditional control and natural-language-driven orchestration. Finally, Section 3.9 presents comprehensive empirical evaluations, followed by a summary in Section 3.10.

3.2 Related Work

This section reviews the landscape of layout pattern generation methods and the emerging role of LLMs in electronic design automation, positioning the contributions of this chapter within the existing literature.

3.2.1 Rule-Based Pattern Generation

The earliest approaches to layout pattern library construction were purely rule-based [74, 126]. These methods begin with a small set of hand-crafted basic layout units and expand them through simple geometric augmentations, namely flipping, rotation, and random concatenation of unit cells. The augmented fragments are then randomly assembled to form larger layout compositions. While

straightforward to implement and inherently legal (since the base units are DRC-clean), rule-based generators suffer from a fundamental diversity bottleneck: the combinatorial space of augmented patterns remains tethered to the original seed library. As a result, the generated patterns fail to capture the rich, holistic structural correlations present in real production layouts, and the overall diversity and quantity of generated patterns consistently falls short of downstream application requirements.

3.2.2 Learning-Based Pattern Generation

To overcome the diversity limitations of rule-based approaches, learning-based generative models have been increasingly adopted for layout pattern synthesis. These methods learn the underlying statistical distribution of a training set and synthesize novel geometries by sampling from the learned distribution. Existing learning-based approaches can be broadly categorized into pixel-based and sequential-based paradigms.

GAN-Based Methods Generative Adversarial Networks (GANs) [28] were among the first deep generative models applied to layout pattern synthesis. Yang et al. [120] proposed DeepPattern, which uses a convolutional auto-encoder (CAE) to generate topology matrices in the Squish Pattern representation. Zhang et al. [134] improved upon this with LegalGAN, incorporating a variational convolutional auto-encoder (VCAE) coupled with a GAN-based adversarial loss. While LegalGAN significantly improved legality rates (from 3.74% with a plain CAE to 84.51% with VCAE), the generated topologies still lacked the sharpness and structural fidelity required for advanced-node design rules. A fundamental issue is that GANs operate in continuous latent spaces and must apply post-hoc thresholding to convert real-valued outputs into binary topology matrices, which

inevitably introduces artifacts at the discretization boundary.

Transformer-Based Methods Wen et al. [117] proposed LayoutTransformer, a sequential-based method that models layout patterns as ordered polygon sequences. Each polygon is decomposed into a series of vertices and directed edges, and a Transformer decoder autoregressively generates new polygon sequences. While LayoutTransformer achieved 89.73% legality and competitive diversity, the sequential formulation struggles to capture the intricate multi-polygon spatial correlations inherent to dense, advanced-node layouts. Additionally, the autoregressive generation paradigm imposes a strict ordering on inherently unordered polygon sets, introducing biases that limit topological variety.

Continuous Diffusion Approaches The emergence of DDPMs [32, 88] has revolutionized image synthesis. However, directly applying continuous diffusion models to layout topology generation introduces a critical mismatch. Layout topologies are inherently binary: each entry in the topology matrix is either 0 (space) or 1 (shape). Continuous diffusion models generate real-valued outputs that must be binarized through ad-hoc thresholding (e.g., $x > 0.5 \implies 1$). This thresholding is fundamentally misaligned with the categorical nature of the data, forcing the neural network to allocate parameter capacity to modeling arbitrary real values rather than the discrete structural boundaries required for legal IC layouts. An earlier version of DiffPattern [109] demonstrated that discrete diffusion [1] substantially outperforms continuous-then-threshold approaches, achieving near-perfect legality by operating directly in the binary state space.

3.2.3 Squish Pattern Representation

The Squish Pattern representation [27] is a lossless encoding technique that addresses the inherent geometric sparsity of layout images. It divides a layout pat-

tern into a grid based on horizontal and vertical scan lines aligned with polygon edges. The intervals between adjacent scan lines are recorded in two geometric vectors, Δ_x and Δ_y , while the structural presence or absence of a polygon within each grid cell is encoded in a binary topology matrix. This separation of topology from geometry is central to the generation framework in this chapter: the generative model synthesizes diverse topologies, while a separate legalization stage assigns DRC-compliant geometric dimensions. The Squish Pattern has been widely adopted by subsequent pixel-based methods [109, 110, 120, 121, 134] as the standard compact representation for layout pattern generation.

3.2.4 Large Language Models in EDA

Recent advances in LLMs have catalyzed a wave of research exploring their application to electronic design automation tasks. Zhou et al. [138] proposed DB-GPT, which adapts LLMs for database management tasks through tool-augmented generation. He et al. [31] introduced ChatEDA, demonstrating that LLMs can serve as autonomous agents capable of invoking EDA tool flows through natural language interfaces. These works leverage the ReAct [125] paradigm, in which the LLM decomposes complex user requests into structured sub-tasks and iteratively executes tool calls while reasoning about intermediate results.

Applying LLMs as autonomous layout pattern library builders, however, requires bridging the semantic gap between natural language design specifications and the low-level numerical parameters required by generative models. Unlike typical EDA tool flows that operate on well-defined file formats, layout pattern customization involves nuanced requirements—such as specifying local modifications to accommodate specific applications [95, 135], conditional generation under different design rules, and free-size extension—that demand both domain expertise and flexible tool orchestration. The interactive customization framework

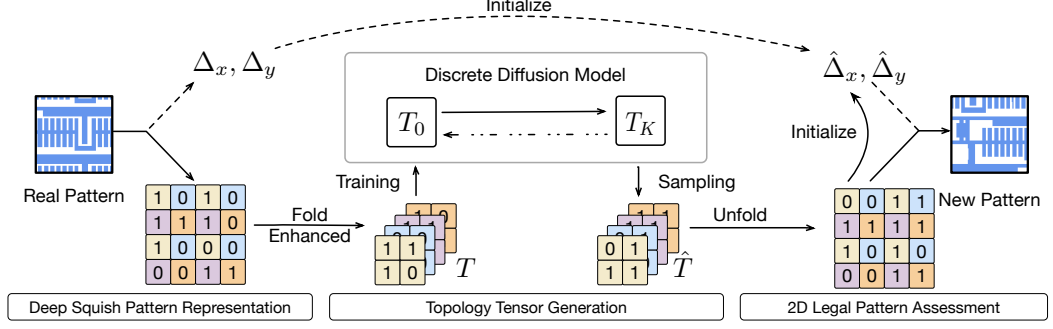


Figure 3.1: An illustration of the DiffPattern framework for reliable layout pattern generation. The pipeline consists of three phases: Deep Squish Pattern encoding, topology tensor generation via discrete diffusion, and 2D legal pattern assessment through nonlinear legalization.

developed in Section 3.8 addresses this challenge.

3.3 Method Overview

Fig. 3.1 summarizes the full generation pipeline that ties together the components introduced in this chapter. At inference time, the pipeline runs left to right: an input layout is first encoded into the *Deep Squish Pattern* tensor format (Section 3.4); a *discrete diffusion model* (Section 3.5) iteratively denoises a uniform noise tensor into a clean binary topology tensor; a lightweight pre-filter discards rare degenerate configurations; and the surviving tensor is unfolded back into a 2D topology matrix and passed through a constrained *legalization* solver (Section 3.6) that assigns physical dimensions under the foundry’s design rules, producing a DRC-clean layout pattern.

Two further components close the loop. During training, *reliable topology enhancement* (Section 3.7) enriches the dataset with augmented topologies that are gated by the same legalization solver, ensuring that only structurally valid

samples enter the training distribution. During deployment, an *expert LLM agent* (Section 3.8) orchestrates conditional generation, In-Painting repair, and Out-Painting extension on top of the core generator, transforming it from a fixed-size batch synthesizer into an interactive pattern design tool. The remaining sections detail each component in the order in which it appears in the inference pipeline, followed by the training-side enhancement and the LLM-driven customization layer.

3.4 Deep Squish Pattern Representation

A critical challenge in pixel-based layout generation is the inherent sparsity of the geometric information. Because real layouts consist of distinct polygons separated by space, treating a layout simply as a high-resolution grid results in enormous computational overhead. The classical *Squish Pattern* method [27] provides a lossless compression alternative by dividing the layout based on horizontal and vertical scan lines aligned with the polygon edges. The intervals between these scan lines are stored in two continuous vectors, Δ_x and Δ_y , while the structural presence of a polygon within the grid is recorded in a discrete, two-dimensional binary topology matrix.

While the standard Squish Pattern significantly reduces memory overhead, the topology matrix still acts as a single-channel 2D mask—a representation that is inefficient for modern generative architectures. Diffusion models, in particular, scale quadratically with spatial resolution through self-attention and convolutional operations, yet are relatively insensitive to the channel dimension. Under the standard Squish representation, the generative model must therefore operate at the full spatial resolution of the topology matrix, forgoing an opportunity for lossless spatial compression.

To exploit this asymmetry, we introduce the *Deep Squish Pattern* representation (illustrated in Fig. 3.2), which trades unused channel capacity for reduced spatial resolution. This is a key algorithmic contribution of this chapter: unlike prior Squish variants that merely tune the grid resolution or patch size, Deep Squish Pattern performs a principled spatial-to-channel fold that expands the model’s effective receptive field while keeping the per-channel state space strictly binary—avoiding both the bit-weighting pathology and the exponential state-space explosion of naive integer encoding.

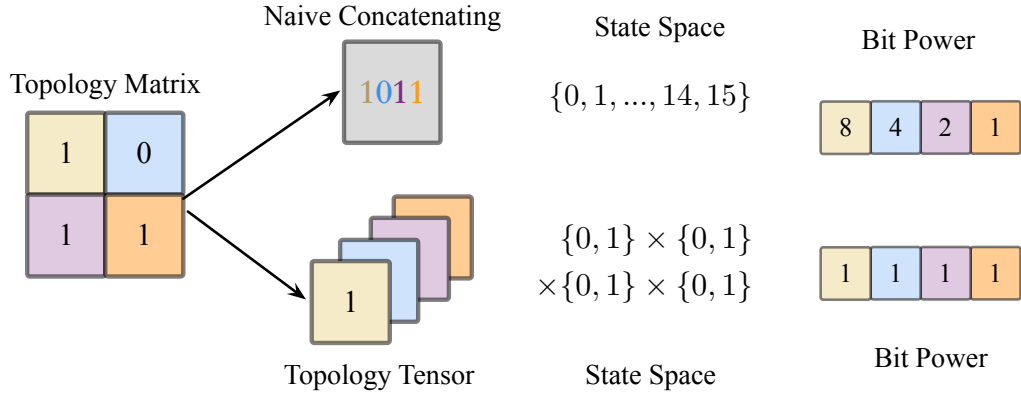


Figure 3.2: Illustration of the Deep Squish Pattern Encoding. The expanded topology tensor provides a compact, multi-channel lossless representation. Directly concatenating binary states into a single numerical integer causes unequal bit-weighting and exponential state space explosion, which is circumvented by the multi-channel tensor formulation.

Consider a localized 2×2 pixel patch within the standard topology matrix, containing 4 binary values. A naive attempt to compress this patch spatially would be to encode the 4 bits into a single integer state (ranging from 0 to 15). However, defining generation over this concatenated integer space introduces numerical instability: the most significant bit applies exponentially more variance than the least significant bit, rendering the model’s loss landscape highly patho-

logical. Furthermore, scaling to a 4×4 patch would explode the discrete state space to 2^{16} , unmanageable for categorical probability prediction.

Our Deep Squish Pattern elegantly bypasses this by executing a spatial-to-channel dimension fold. A $\sqrt{C} \times \sqrt{C}$ spatial patch of the original binary topology matrix is folded into a single spatial point possessing C distinct binary channels, resulting in a compacted topology tensor $\mathbf{T} \in \{0, 1\}^{C \times M \times M}$. In this formulation, the state space for any individual channel remains strictly binary (0 or 1), ensuring the discrete diffusion model only needs to predict dual-state categorical distributions. This technique drastically reduces the spatial dimensionality the neural network must process, accelerating inference times and enhancing global structural coherence, all while maintaining the exact lossless resolution of the original layout configuration. Post-generation, the tensor $\mathbf{T}$ is simply unfolded back into the standard 2D topology matrix prior to geometry legalization. Importantly, the folded topology tensor is treated as a single sample during generation rather than multiple independent samples; hence the deep squish pattern preserves all spatial correlations of the original topology matrix.

3.5 Discrete Diffusion for Topology Generation

With the input layout encoded as a Deep Squish Pattern tensor (Section 3.4), we now describe the generative model that synthesizes new topology tensors. Building on the D3PM framework introduced in Section 2.1.3, we formulate the layout topology generation task as a discrete denoising process over the binary tensor $\mathbf{T}_0 \in \{0, 1\}^{C \times M \times M}$.

While adapting continuous diffusion models to this task by merely applying a post-generation threshold (e.g., $x > 0.5 \implies 1$) is possible, it is fundamentally misaligned with the categorical nature of the data. Such continuous relaxations

force the neural network to allocate parameter capacity to modeling arbitrary real values, ultimately degrading the crisp structural boundaries required for legal IC layouts. The DiffPattern core therefore utilizes a D3PM, preserving categorical integrity throughout the entire generation pipeline.

3.5.1 Forward Transition Dynamics

At any diffusion step $k \in \{1, \dots, K\}$, let $\mathbf{x}_k$ denote the one-hot encoded state of a single localized entry within the topology tensor. The forward process corrupts the data by independently transitioning each entry according to a doubly stochastic Markov matrix $\mathbf{Q}_k$:

$$q(\mathbf{x}_k \mid \mathbf{x}_{k-1}) = \text{Cat}(\mathbf{x}_k; \mathbf{p} = \mathbf{x}_{k-1} \mathbf{Q}_k), \quad (3.1)$$

where Cat denotes the categorical distribution. Because the expanded target state space is strictly binary (0 or 1), a uniform transition matrix is seamlessly applied:

$$\mathbf{Q}_k = \begin{bmatrix} 1 - \beta_k & \beta_k \\ \beta_k & 1 - \beta_k \end{bmatrix}. \quad (3.2)$$

The noise schedule $\beta_k \in (0, 1)$ dictates the rate of structural corruption, typically scaling linearly from β_1 to β_K :

$$\beta_k = \frac{(k-1)(\beta_K - \beta_1)}{K-1} + \beta_1, \quad k = 1, \dots, K. \quad (3.3)$$

As $k \rightarrow K$, the repeated application of $\mathbf{Q}_k$ drives the topology tensor toward a stationary state of uniform noise, where the probability of any entry being 0 or 1 approaches 0.5.

The marginal distribution at step k can be computed analytically without sequentially applying each transition matrix:

$$\mathbf{q}(\mathbf{x}_k \mid \mathbf{x}_0) = \text{Cat}(\mathbf{x}_k; \mathbf{p} = \mathbf{x}_0 \overline{\mathbf{Q}}_k), \quad (3.4)$$

where $\overline{\mathbf{Q}}_k = \mathbf{Q}_1 \mathbf{Q}_2 \dots \mathbf{Q}_k$. This closed-form marginal enables efficient training by directly sampling any intermediate noise level.

3.5.2 Reverse Denoising

To synthesize novel topologies, the forward process is inverted. Commencing from a purely random uniform tensor $\mathbf{T}_K$, a deep neural network parametrized by θ predicts the clean data distribution $p_\theta(\mathbf{x}_0 | \mathbf{x}_k)$. Using Bayes' theorem and the known forward transitions, the step-wise reverse marginal is calculated as:

$$p_\theta(\mathbf{x}_{k-1} | \mathbf{x}_k) = \sum_{\tilde{\mathbf{x}}_0} q(\mathbf{x}_{k-1} | \mathbf{x}_k, \tilde{\mathbf{x}}_0) p_\theta(\tilde{\mathbf{x}}_0 | \mathbf{x}_k), \quad (3.5)$$

where the summation evaluates locally over the finite binary states. The forward posterior $q(\mathbf{x}_{k-1} | \mathbf{x}_k, \mathbf{x}_0)$ appearing in this equation has a closed form derived via Bayes' theorem:

$$\mathbf{q}(\mathbf{x}_{k-1} | \mathbf{x}_k, \mathbf{x}_0) = \text{Cat} \left(\mathbf{x}_{k-1}; \mathbf{p} = \frac{\mathbf{x}_k \mathbf{Q}_k^\top \odot \mathbf{x}_0 \overline{\mathbf{Q}}_{k-1}}{\mathbf{x}_0 \overline{\mathbf{Q}}_k \mathbf{x}_k^\top} \right), \quad (3.6)$$

where $\odot$ denotes the element-wise Hadamard product. The complete generation process proceeds as:

$$p_\theta(\hat{\mathbf{T}}_0 | \mathbf{T}_K) = p_\theta(\hat{\mathbf{T}}_0 | \mathbf{T}_1) \prod_{k=2}^K p_\theta(\mathbf{T}_{k-1} | \mathbf{T}_k), \quad (3.7)$$

where $\hat{\mathbf{T}}_0$ is the newly generated topology tensor. The generated topology is inherently binary with each entry being either zero or one, eliminating any need for post-hoc thresholding. An illustration of the discrete denoising process is provided in Fig. 3.3.

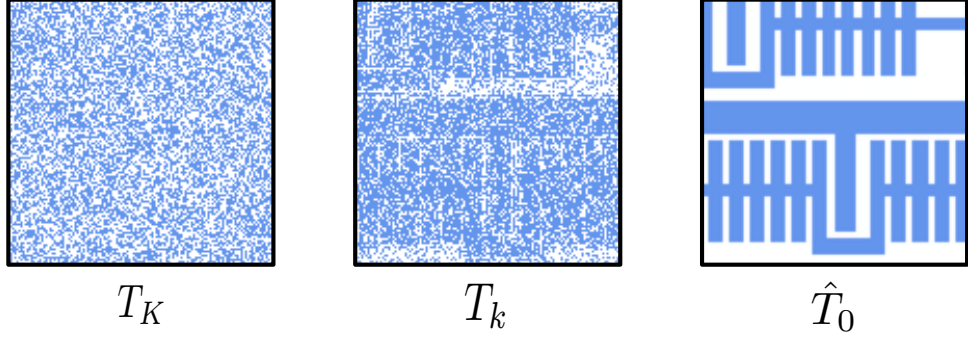


Figure 3.3: Illustration of the discrete denoising process. Starting from uniform noise (left), the topology tensor is progressively denoised toward a clean binary layout topology (right).

3.5.3 Training Objective

With the forward and reverse processes defined, we now describe the training procedure. The objective at each step k is to minimize:

$$L = D_{\text{KL}}(\mathbf{q}(\mathbf{x}_{k-1}|\mathbf{x}_k, \mathbf{x}_0) \parallel \mathbf{p}_{\theta}(\mathbf{x}_{k-1}|\mathbf{x}_k)) - \lambda \log \mathbf{p}_{\theta}(\mathbf{x}_0|\mathbf{x}_k), \quad (3.8)$$

where λ is a hyperparameter balancing the two loss terms. The first term is the KL divergence between the tractable forward posterior $\mathbf{q}(\mathbf{x}_{k-1}|\mathbf{x}_k, \mathbf{x}_0)$ from Eq. (3.6) and the predicted reverse step $\mathbf{p}_{\theta}(\mathbf{x}_{k-1}|\mathbf{x}_k)$ from Eq. (3.5), encouraging the reverse process to match the true posterior. The second term is an auxiliary cross-entropy loss that incentivizes the network to directly predict the clean data distribution $\mathbf{x}_0$.

During training, given a topology tensor $\mathbf{T}_0$, a random target step k is sampled from 1 to K , and a noisy tensor $\mathbf{T}_k$ is obtained via the closed-form marginal in Eq. (3.4). The neural network takes $\mathbf{T}_k$ and the time step embedding k as inputs and predicts the logits of $\mathbf{p}_{\theta}(\mathbf{x}_0|\mathbf{x}_k)$, from which the reverse step probability is computed. The diffusion model is then optimized by gradient descent over the training set; the augmentation procedure that enriches this set is described in

Section 3.7.

3.5.4 Fast Sampling

While standard generation requires executing all K reverse steps iteratively (a computationally burdensome process when $K = 1000$), the strictly Markovian formulation permits accelerated strided sampling. By evaluating the reverse transitions over intervals of m steps simultaneously, the inference latency can be drastically reduced:

$$p_{\theta}(\mathbf{x}_{k-m} \mid \mathbf{x}_k) = \sum_{\tilde{\mathbf{x}}_0} q(\mathbf{x}_{k-m} \mid \mathbf{x}_k, \tilde{\mathbf{x}}_0) p_{\theta}(\tilde{\mathbf{x}}_0 \mid \mathbf{x}_k), \quad (3.9)$$

where $q(\mathbf{x}_{k-m} \mid \mathbf{x}_k, \mathbf{x}_0)$ has a closed form by extending Eq. (3.6):

$$q(\mathbf{x}_{k-m} \mid \mathbf{x}_k, \mathbf{x}_0) = \text{Cat} \left(\mathbf{x}_{k-m}; \mathbf{p} = \frac{\mathbf{x}_k \mathbf{Q}_k^{\top} \odot \mathbf{x}_0 \overline{\mathbf{Q}}_{k-m}}{\mathbf{x}_0 \mathbf{Q}_k \mathbf{x}_k^{\top}} \right). \quad (3.10)$$

This skipping mechanism introduces a controllable trade-off between generative fidelity and computational throughput, allowing rapid synthesis of large-scale topology batches. By performing inference with m steps at a time, topology generation can be accelerated by approximately a factor of m .

The generation-side cost is dominated by the retained reverse diffusion steps and the topology tensor size; for a fixed U-Net backbone, sampling scales with the number of evaluated steps, approximately K/m , and with the tensor resolution CM^2 . The downstream legalization cost is topology dependent because feasibility is determined by a nonlinear constraint system over scan-line variables; block-partition initialization reduces practical runtime, but difficult DRC interactions can still dominate the worst cases.

3.5.5 Topology Pre-filter

After sampling, a lightweight rule-based filter removes degenerate candidates. The main case is the Bow-tie configuration, where two polygons share exactly one grid corner and therefore create ambiguous connectivity. Owing to the high quality of the discrete diffusion outputs, fewer than 0.1% of generated topologies are discarded by this filter in practice. The remaining tensors are then unfolded from the Deep Squish Pattern format into standard 2D topology matrices and legalized via the system described in Section 3.6.

3.6 Topology Legalization

The discrete diffusion model in Section 3.5 produces a binary topology tensor that, after pre-filtering and unfolding, becomes a standard 2D topology matrix—still without physical dimensions. Legalization closes this gap by assigning per-cell scan-line spacings under the foundry’s design rules, transforming a dimensionless topology into a printable, DRC-clean layout pattern. The same legalization machinery is reused in Section 3.7 as a structural-validity gate for training-time data augmentation.

3.6.1 Nonlinear Geometric Assignment

A topology matrix lacks physical dimensions; it merely dictates the relative positioning (shape vs. space) of blocks across a grid. Transforming this dimensionless topology into a printable layout necessitates the assignment of two geometric interval vectors, Δ_x and Δ_y , which define the physical scan-line spacings.

To enforce exact DRC compliance, the assignment of these vectors is formulated as a constrained nonlinear programming problem. Let δ_{xi} and δ_{yj} denote the individual interval entries of Δ_x and Δ_y . The legalization system seeks to

determine legal vectors that fulfill all design rules while minimizing unnecessary global area expansion:

$$\left\{ \begin{array}{ll} \delta_{xi}, \delta_{yj} > 0, & \forall \delta_{xi}, \delta_{yj}; \\ \sum_i \delta_{xi} = \sqrt{C} M, \quad \sum_j \delta_{yj} = \sqrt{C} M; & \\ \sum_{i=a}^b \delta_i \geq Space_{min}, & \forall (a, b) \in \mathcal{S}; \\ \sum_{i=a}^b \delta_i \geq Width_{min}, & \forall (a, b) \in \mathcal{W}; \\ \sum \delta_{xi} \delta_{yj} \in [Area_{min}, Area_{max}], & \forall \text{ Polygon}, \end{array} \right. \quad (3.11)$$

where $Space_{min}$ and $Width_{min}$ are the lower bounds for minimum spacing and minimum width, $\sqrt{C} M \times \sqrt{C} M$ defines the dimensions of the unfolded topology matrix, and $Area_{min}$ and $Area_{max}$ define the permissible area range for each polygon. All design-rule constants are pattern-independent and provided by the foundry specification. The sets $\mathcal{S}$ and $\mathcal{W}$ are pattern-dependent, indicating which pairs of scan lines are constrained by minimum spacing and width rules, respectively. The geometric meaning of the three rule families—minimum spacing between adjacent polygons, minimum width along each axis, and admissible polygon area—is illustrated in Fig. 3.4.

Because this formulation treats the topology matrix as a rigid constraint graph, any solution returned by the nonlinear programming solver satisfies the specified DRC constraints. Importantly, for a given topology matrix and set of design rules, the nonlinear system typically admits multiple solutions, meaning a single topology can be legally instantiated into numerous distinct layout patterns with different geometric dimensions (see Section 3.9.5). In rare cases where the topology structure makes legalization infeasible within a prescribed time budget, the system rapidly flags and discards the topology, precluding illegal geometries from entering the final accepted dataset.

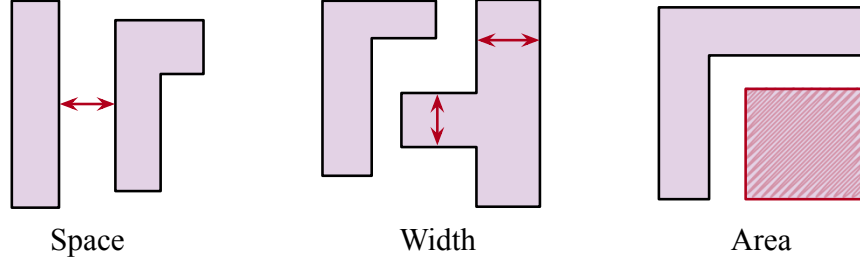


Figure 3.4: Visual key to the design-rule constraints in Eq. (3.11). *Space* measures the distance between adjacent polygons; *Width* measures the size of a shape along one axis; *Area* denotes the area of an individual polygon. Each constraint family in the nonlinear program corresponds to one of these geometric quantities.

3.6.2 Block-Partition Acceleration

The computational complexity of solving Eq. (3.11) scales dramatically with the matrix dimensions. Recognizing that DRC interactions are highly localized (e.g., minimum spacing rules primarily affect adjacent rather than distant polygons), the global optimization problem can be strategically partitioned. As illustrated in Fig. 3.5, the system divides the large topology matrix into several disjoint diagonal sub-blocks. Each sub-block is solved independently and concurrently as an isolated sub-problem. These localized solutions serve as high-quality initializations for the global solver, resulting in empirical speedups exceeding $2.4\times$ while aggressively mitigating late-stage convergence failures.

Additionally, in scenarios where the target pattern and the source pattern follow the same design rules, an existing pair of geometric vectors from the training set can be randomly selected to initialize the nonlinear solver, providing a further acceleration.

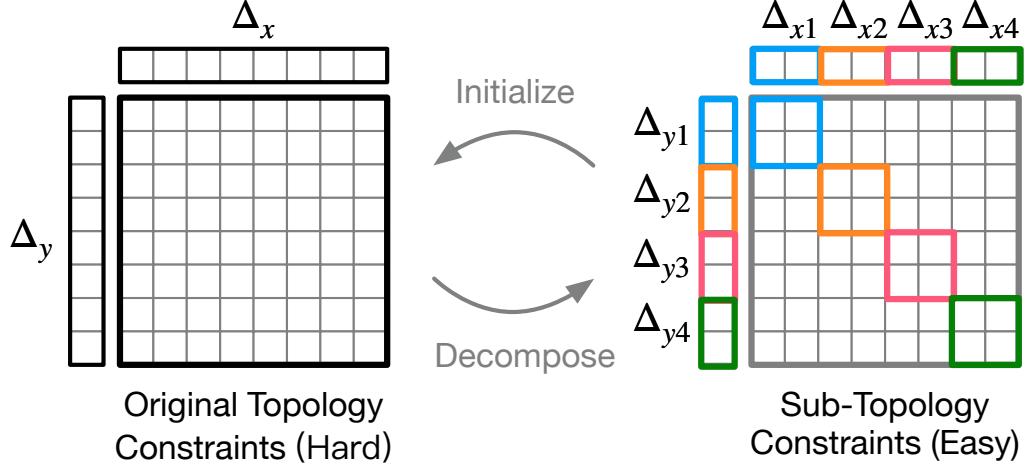


Figure 3.5: Illustration of block-partition acceleration for legalization. Each diagonal block denotes an independent sub-problem. All sub-problems are solved in parallel to provide high-quality initialization for the global solver.

3.7 Reliable Topology Enhancement

The components above describe the inference-time pipeline. We now turn to the training side, where the same legalization system from Section 3.6 is repurposed as a structural-validity gate to enrich the training dataset. Modern generative models benefit significantly from high-quality, diverse training data. While extensive data augmentation techniques have been developed in the computer vision literature [84], directly applying these methods to the layout pattern domain is inadvisable: inappropriate augmentations can compromise the structural validity of topology matrices and produce patterns that violate design rules.

To ensure the reliability of enhanced topology matrices, we propose a pre-checking process that leverages the legalization method from Section 3.6. The reliable topology enhancement procedure is formulated as:

$$\tilde{\mathcal{T}} = \left\{ \tilde{\mathbf{T}} = \Gamma(\mathbf{T}) \mid \Xi(\tilde{\mathbf{T}}) = \text{Success} \right\}, \quad (3.12)$$

where $\mathbf{T}$ and $\tilde{\mathbf{T}}$ represent the topology matrices before and after augmentation, respectively. Here, $\Gamma(\cdot)$ denotes a data augmentation function comprising various enhancement techniques, and $\Xi(\cdot)$ refers to the legalization verification from Section 3.6. Only topologies that successfully pass the legalization check are retained for training; those that fail are discarded.

In our implementation, the augmentation function $\Gamma(\cdot)$ consists of four operations, illustrated in Fig. 3.6:

1. *Random Flip*: horizontal or vertical reflection of the topology matrix.
2. *Random Rotation*: rotation by multiples of 90° .
3. *Symmetric Mirror*: creating a symmetric pattern by mirroring along a selected axis.
4. *Concatenate and Crop*: randomly concatenating multiple topologies and extracting a sub-region, which introduces novel boundary interactions absent in the original data.

These methods were chosen because they preserve geometric plausibility without introducing meaningless random noise, which is rare in practical scenarios and can negatively impact downstream tasks. During training, multiple augmentation methods are applied probabilistically to each sample, and the legalization pre-check $\Xi(\cdot)$ ensures that only structurally valid topologies enter the training set.

The discrete diffusion model described above generates fixed-size topology tensors unconditionally. In practice, however, design teams require finer control: generating patterns that conform to specific process conditions, repairing localized defects without regenerating entire topologies, and synthesizing layouts that exceed the model’s native resolution. The following section extends the core generator with these capabilities.

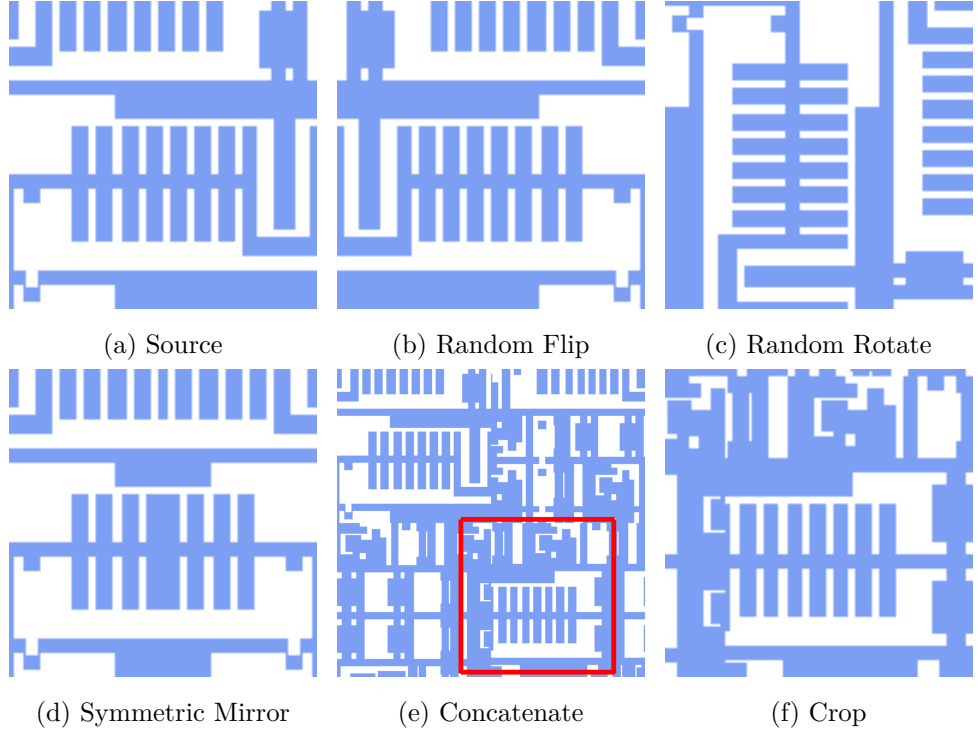


Figure 3.6: Illustration of reliable topology enhancement. Multiple augmentation methods are applied to each pattern. (f) denotes the random patch within the red box cropped from (e). Only augmented topologies that pass legalization verification are retained for training.

3.8 Interactive Customization via Conditional Generation and LLM Orchestration

To support fine-grained, interactive control over the generation process, we augment the core discrete diffusion model with property-conditional generation, localized In-Painting repair, iterative Out-Painting extension, and an expert LLM agent [110] that orchestrates these operations through natural-language instructions.

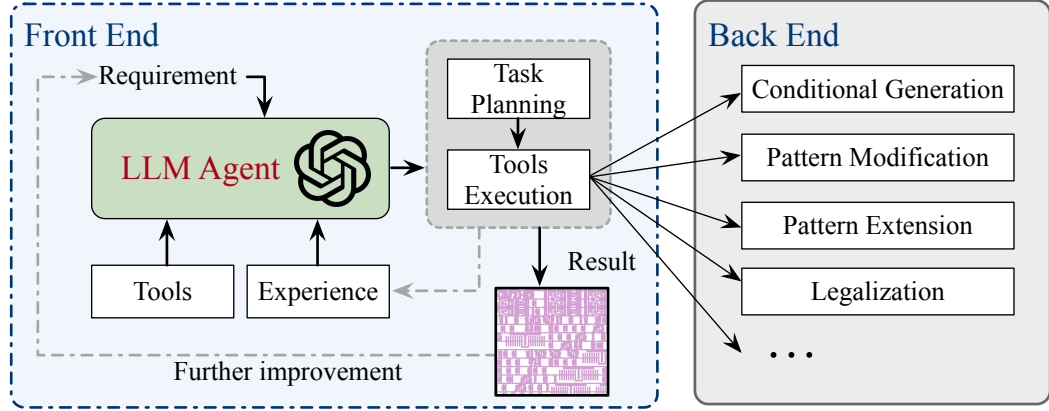


Figure 3.7: An illustration of the ChatPattern system. The LLM agent interprets natural language requirements and orchestrates the pattern generation backend to produce customized layout libraries.

3.8.1 System Overview

As illustrated in Fig. 3.7, ChatPattern is composed of two principal components: (1) an expert LLM agent that interprets user requirements expressed in natural language, breaking down complex demands into manageable sub-tasks; and (2) a flexible, controllable layout pattern generator that supports conditional generation, precise pattern modification, and memory-friendly pattern extension. The LLM agent serves as an indefatigable layout pattern library builder, leveraging the ReAct paradigm described in Section 2.1.4 to autonomously invoke backend generation APIs and troubleshoot failures.

3.8.2 Expert LLM Agent

The agent is designed to be autonomous rather than merely scripted: beyond translating user requests into backend calls, it reasons about intermediate outputs, diagnoses failures (e.g., legalization errors on specific regions), and plans corrective actions such as targeted In-Painting (see the error-recovery case study

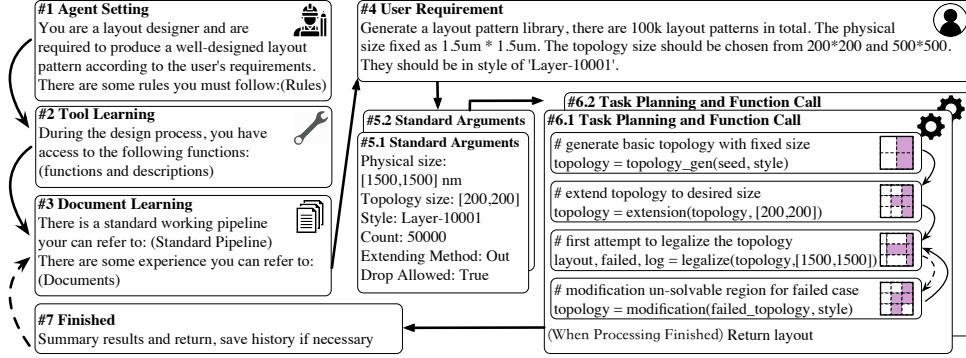


Figure 3.8: Working pipeline of the LLM agent in ChatPattern. The agent decomposes user requests into structured sub-tasks, invokes backend APIs, and handles errors autonomously. Dashed lines denote optional paths.

in Section 3.9.8). This distinguishes the agent from a rule-based arguments-to-program translator and is essential for robust, large-scale pattern library construction. The operational pipeline of the expert LLM agent is depicted in Fig. 3.8. It proceeds through several stages:

Requirement Auto-Formatting Given the complexity of user requests in the layout pattern generation workflow, the LLM agent first translates these requests into a structured format using a predefined template. This template contains all relevant parameters and requirements for use by the pattern generation API functions. The arguments are divided into required ones, which decide the basic parameters (e.g., pattern count and size) of the task, and optional ones, which provide fine-grained control. Importantly, a single user request can be decomposed into several requirement lists, each for one simple sub-task, limiting the complexity of individual tasks and reducing errors.

Task Planning and Execution After formatting, ChatPattern identifies and plans the necessary sub-tasks for fulfilling the user's request. For each planned sub-task, ChatPattern schedules a series of structured tasks, which are then ad-

dressed using various pattern generation tools such as topology generation and legalization. For execution, **ChatPattern** utilizes a Python interpreter or API calls, supported by the layout pattern generation back-end model.

Compared with a rule-based arguments-program translator, the LLM agent can address failed cases based on feedback or logs derived from function calls. For example, in the legalization phase, legalization can easily fail for large topology matrices due to design-rule violations. While simply dropping the failed cases keeps the accepted outputs legal, significant effort in generating topology matrices is wasted. The LLM agent can instead modify the failed regions with different conditions and try different initial states, saving time and computation.

Tool Function Learning and Application The core concept of pattern generation through an LLM agent lies in its ability to operate without directly accessing the generated matrix of zeros and ones, which could surpass the token length limit of the LLM. Instead, the agent relies on the route to the end results and comprehension of overarching characteristics such as complexity, physical dimensions, or error locations. To construct a pattern library, certain fundamental tools are indispensable:

- *Random Topology Generation*: creates random topologies subject to specific conditions.
- *Topology Legalization*: transforms a topology into a compliant layout pattern via [Section 3.6](#).
- *Topology Extension*: augments a topology to a designated size within certain parameters.
- *Topology Modification*: revises parts of a topology, offering a time-efficient alternative to discarding non-compliant topologies.

Additional high-level functions may be integrated to bolster the model’s capabilities, allowing the LLM agent to capitalize on backend tool enhancements without internal changes.

Learning from Documents and Experience Documents are reservoirs of high-level knowledge instrumental in the pattern design process. In the free-size layout pattern generation task, the selection of the extension algorithm plays a pivotal role in determining the quality of the output. Informed algorithm selection can be enhanced through statistical analysis of historical data. Additionally, equipping the model with a standardized operational pipeline at inception can markedly increase performance. The practice of documenting work history and scrutinizing exceptional cases lays the groundwork for the model’s ongoing refinement.

3.8.3 Property-Conditional Topology Generation

Patterns in a dataset do not always share the same design rules or physical properties. For example, the distribution of patterns from the 7-nm manufacturing process differs significantly from that of 130-nm. Establishing separate datasets and training individual models for each pattern type is impractical in practice—each additional process node would multiply the training cost and storage footprint—while directly training a single model on a mixed dataset raises concerns about design-rule conflicts and mode collapse onto the dominant style. Property-conditional generation resolves this tension by letting a single model serve multiple processes: the backbone learns shared structural priors across styles, and the condition vector steers generation toward the requested target distribution at inference time.

Formally, we extend the discrete diffusion model with conditions that specify

the distribution of the generated topology matrix. The k -th step of the condition-specified reverse process is defined as:

$$\mathbf{p}_\theta(\mathbf{x}_{k-1}|\mathbf{x}_k, \mathbf{c}) = \sum_{\tilde{\mathbf{x}}_0} \mathbf{q}(\mathbf{x}_{k-1}|\mathbf{x}_k, \tilde{\mathbf{x}}_0) \mathbf{p}_\theta(\tilde{\mathbf{x}}_0|\mathbf{x}_k, \mathbf{c}), \quad (3.13)$$

where $\mathbf{c}$ denotes the conditions encoding design rules, materials, and manufacturing process. During training, the model optimizes:

$$L = D_{\text{KL}}(\mathbf{q}(\mathbf{x}_{k-1}|\mathbf{x}_k, \mathbf{x}_0) \parallel \mathbf{p}_\theta(\mathbf{x}_{k-1}|\mathbf{x}_k, \mathbf{c})) - \lambda \log \mathbf{p}_\theta(\mathbf{x}_0|\mathbf{x}_k, \mathbf{c}). \quad (3.14)$$

Once trained, a topology matrix with condition $\mathbf{c}$ can be generated by a K -step reverse process:

$$p_\theta(\mathbf{T}_0|\mathbf{T}_K, \mathbf{c}) = p_\theta(\mathbf{T}_0|\mathbf{T}_1, \mathbf{c}) \prod_{k=2}^K p_\theta(\mathbf{T}_{k-1}|\mathbf{T}_k, \mathbf{c}), \quad (3.15)$$

where $\mathbf{T}_K$ is a randomly sampled noise tensor. The condition embedding is extracted through a stack of linear layers and added to the time step embedding to control the style of the generated patterns. Fig. 3.9 schematizes how the condition vector is injected into both the training-time noisy posterior and the inference-time reverse trajectory: the same backbone is shared across styles, while $\mathbf{c}$ steers the trajectory toward the requested distribution at every step.

3.8.4 Conditional Modification via In-Painting

Given an existing, legally verified topology matrix $\mathbf{T}_0^{\text{known}}$ that a user wishes to modify, the In-Painting technique allows surgical modifications to any desired region. Let $\mathbf{M}$ be a binary spatial mask where 1 indicates the region to keep and 0 indicates the region to regenerate. During the reverse discrete diffusion inference, at each step k , we force the known region to match the exact noise

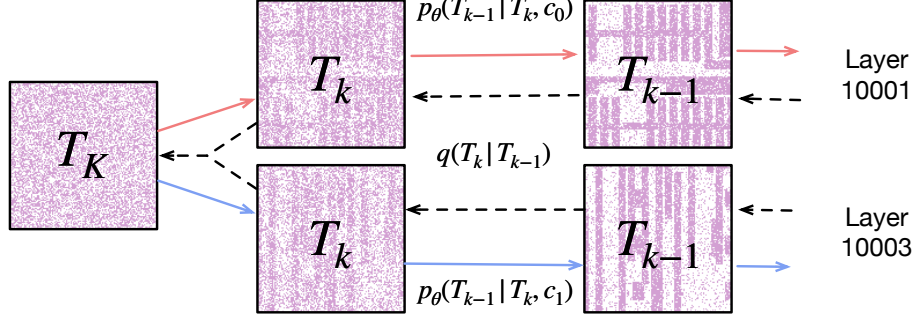


Figure 3.9: Conditional discrete diffusion for property-specified pattern generation. During training, the condition $\mathbf{c}$ is provided alongside the noisy tensor $\mathbf{T}_k$ so that the network learns a class-conditioned reverse step. At inference time, the same $\mathbf{c}$ steers the K -step reverse trajectory from random noise to a topology that matches the requested style.

profile of the source data at step k :

$$\begin{aligned}
\mathbf{T}_{k-1}^{\text{known}} &\sim q(\mathbf{T}_{k-1} \mid \mathbf{T}_0^{\text{known}}) \quad (\text{Forward Analytical}) \\
\mathbf{T}_{k-1}^{\text{unknown}} &\sim p_{\theta}(\mathbf{T}_{k-1} \mid \mathbf{T}_k, \mathbf{c}) \quad (\text{Neural Prediction}) \\
\mathbf{T}_{k-1} &= \mathbf{M} \odot \mathbf{T}_{k-1}^{\text{known}} + (1 - \mathbf{M}) \odot \mathbf{T}_{k-1}^{\text{unknown}},
\end{aligned} \tag{3.16}$$

where $\mathbf{T}_0^{\text{known}}$ shares the design rules with patterns conditioned on $\mathbf{c}$. By strictly maintaining the known geometry and forcing the neural network to predict only the masked regions conditionally, we achieve seamless structural continuity at the mask boundaries. As shown in Fig. 3.10, this technique enables surgical modifications of layouts, useful for hotspot repair or localized rule updating.

3.8.5 Free-Size Extension via Out-Painting

More importantly, by iteratively applying the In-Painting technique to the outer boundaries of a generated patch (Out-Painting), the system can synthesize patterns larger than the fixed resolution ($C \times M \times M$) of the neural network. As

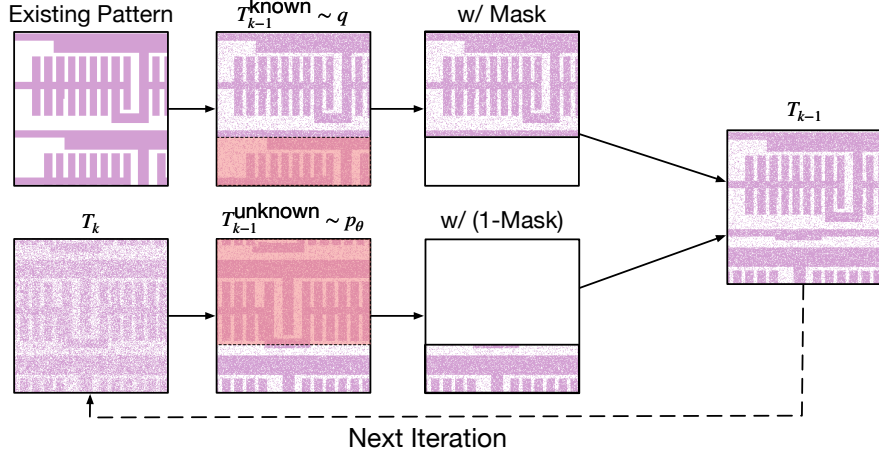


Figure 3.10: Illustration of pattern In-Painting for localized modification. The predefined region (masked) is seamlessly stitched with the newly generated topology, maintaining structural continuity at the boundary.

depicted in Fig. 3.11, the LLM agent can sequence overlapping Out-Painting steps to construct larger layout geometries, while the block-partitioned optimization solver from Section 3.6 attempts to legalize the resulting topology under the specified nanometer-scale rules. The experiments in Section 3.9.4 quantify the practical limits of this extension.

By recursively extending a given pattern, we can synthesize target sizes larger than the model’s native window without loading the full canvas into memory, since only the active region is processed at each step. If we assume the target size of the topology is $[W, H]$ and the window size of the model is $[L, L]$, the number of sampling steps for In-Painting can be calculated as:

$$N_{\text{in}} = \left(2 \left\lceil \frac{W}{L} \right\rceil - 1 \right) \left(2 \left\lceil \frac{H}{L} \right\rceil - 1 \right). \quad (3.17)$$

With a stride S , the number of sampling steps for Out-Painting is:

$$N_{\text{out}} = \left(\left\lceil \frac{W-L}{S} \right\rceil + 1 \right) \left(\left\lceil \frac{H-L}{S} \right\rceil + 1 \right). \quad (3.18)$$

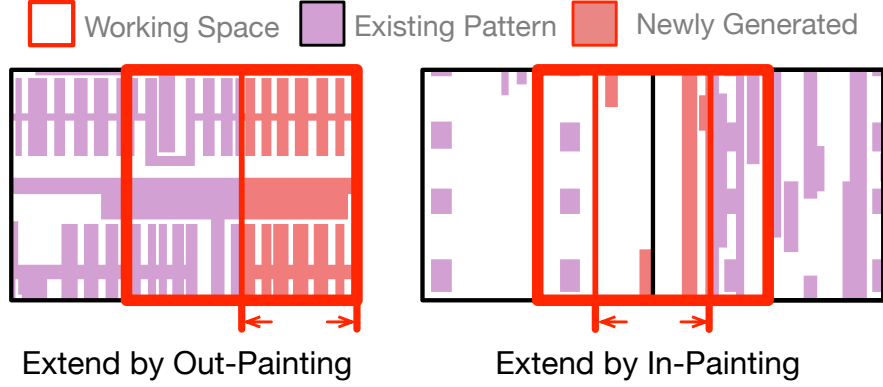


Figure 3.11: Mechanism for free-size pattern extension via iterative Out-Painting and bounded In-Painting. Recursive extension supports target sizes larger than the model’s native window.

3.9 Experimental Results

3.9.1 Experimental Setup

Datasets. In line with prior research [117, 134], we utilize a dataset comprising small layout pattern images sized $2048 \times 2048 \text{ nm}^2$, derived by segmenting a $400 \times 160 \mu\text{m}^2$ layout map from the ICCAD 2014 contest. Two different layout styles are designated: Layer-10001 (standard benchmark) and Layer-10003 (for testing conditional style specification). The extracted Deep Squish topology tensors have size $C \times M \times M = 16 \times 32 \times 32$, unfolding to a 128×128 binary geometry. For the free-size generation evaluation, larger topology matrices at 256^2 , 512^2 , and 1024^2 are obtained by splitting the layout map into correspondingly larger overlapping tiles.

Diffusion Model Configuration. We adopt a U-Net architecture [77] as the foundation of our discrete diffusion model. The model operates at four spatial resolutions: $[32 \times 32, 16 \times 16, 8 \times 8, 4 \times 4]$. Each stage consists of two con-

volutional residual blocks, with the number of convolutional channels set to $[128, 256, 256, 256]$. At the 16×16 resolution stage, a self-attention module is incorporated. Diffusion time steps k are embedded into each residual block using sinusoidal positional encodings [101]. For **ChatPattern**, layer-style conditions are separately embedded by a small multilayer perceptron and added to the time-step embedding. We set the number of forward diffusion steps to $K = 1000$, with the noise schedule β_k scaling linearly from 0.01 to 0.5. During inference, fast-sampling is deployed with a stride interval of $m = 10$.

Training Details. For the unconditional DiffPattern-Flex model, the discrete diffusion model is trained for 0.5×10^6 iterations on the Layer-10001 dataset with reliable topology enhancement, requiring approximately 20 hours on eight NVIDIA RTX 3090 GPUs. For this model, augmentation probabilities for Random Flip, Random Rotation, Symmetric Mirror, and Concatenate-and-Crop are set to $[0.5, 1.0, 0.5, 0.5]$. For the conditional ChatPattern model, training is extended to 1.0×10^6 iterations on the combined Layer-10001/10003 dataset with class conditions, requiring approximately 250 GPU hours in total. Both models use a batch size of 128, the Adam optimizer with a learning rate of 2×10^{-4} , a dropout rate of 0.1, gradient clipping at 1.0, and a loss coefficient $\lambda = 0.001$.

3.9.2 Evaluation Metrics

Two primary metrics are used to evaluate the quality of generated pattern libraries throughout all experiments.

Definition 3.9.1 (Pattern Diversity). The diversity of a pattern library, denoted by H , is defined as the Shannon entropy of the joint distribution of pattern complexities:

$$H = - \sum_i \sum_j P(c_{xi}, c_{yj}) \log P(c_{xi}, c_{yj}), \quad (3.19)$$

where c_x and c_y are the numbers of scan lines minus one along the x -axis and y -axis of a pattern’s squish representation, respectively, and $P(c_{xi}, c_{yj})$ is the probability of a pattern with complexity (c_{xi}, c_{yj}) being sampled from the library. Higher values of H indicate greater structural variety.

Definition 3.9.2 (Pattern Legality). A layout pattern is considered *legal* if it passes DRC under the specified foundry rules (minimum spacing, minimum width, area bounds as in Eq. (3.11)). The legality rate is the fraction of generated patterns that are DRC-clean:

$$\text{Legality} = \frac{\#\text{Legal Patterns}}{\#\text{Generated Patterns}}. \quad (3.20)$$

3.9.3 Fixed-Size Pattern Generation

We begin by evaluating the quality of fixed-size pattern generation. The unconditional DiffPattern-Flex model is trained on Layer-10001 with reliable topology enhancement. We randomly synthesize 100,000 topologies and apply topology pre-filtering followed by nonlinear legalization to produce the final pattern library. The results are summarized in TABLE 3.1.

DiffPattern-Flex achieves a perfect 100% legality rate among the final assessed patterns, with 100,000 DRC-clean layout patterns produced through topology pre-filtering and nonlinear legalization. This is a decisive improvement over prior learning-based methods: CAE+LegalGAN achieves only 3.74% legality, and even the best prior neural method (LayoutTransformer) reaches only 89.73%. The rule-based approach achieves 24.47% legality, demonstrating the limitations of hand-crafted augmentation.

More importantly, DiffPattern-Flex achieves a diversity score of 11.713, substantially exceeding both the real dataset (10.777) and the previous best model DiffPattern (10.815). This improvement is attributed to the reliable topology

Table 3.1: Pattern diversity and legality comparison on the Layer-10001 benchmark (100,000 generated patterns, unconditional DiffPattern-Flex). ‘—’ denotes inapplicable results. Results of learning-based baselines are from prior works [109].

Set/Method	Generated Topology	Generated Patterns		Legal Patterns	
		Patterns	Diversity ($\uparrow$)	Legality ($\uparrow$)	Diversity ($\uparrow$)
Real Patterns	—	—	—	13,869	10.777
CAE [120]	100,000	100,000	4.588	19	3.787
VCAE [134]	100,000	100,000	10.931	2,126	9.978
CAE+LegalGAN [134]	100,000	100,000	5.847	3,740	5.814
Rule-based [126]	—	100,000	10.256	24,474	10.066
VCAE+LegalGAN [134]	100,000	100,000	9.869	84,510	9.867
LayoutTransformer [117]	—	100,000	10.532	89,726	10.527
DiffPattern [109]	100,000	100,000	10.815	100,000	10.815
DiffPattern-Flex	100,000	100,000	11.713	100,000	11.713

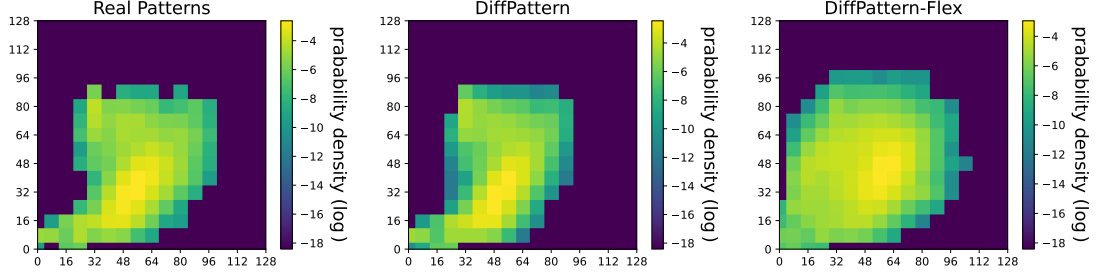


Figure 3.12: Complexity distribution comparison. Reliable pattern augmentation enriches the generated pattern distribution, covering regions absent in the real data distribution while maintaining structural validity.

enhancement described in Section 3.7, which enriches the training distribution with novel structural combinations that pass legalization verification.

The distribution of pattern complexity is visualized in Fig. 3.12. The complexity heatmap generated by DiffPattern-Flex covers regions absent from the real pattern distribution, suggesting that the model can synthesize geometric combinations beyond simple dataset memorization.

To further evaluate the conditional generation capability introduced in Section 3.8.3, we train on the mixed Layer-10001/10003 dataset with class conditions. To isolate raw model quality, the topology selection step is removed from all methods, and pattern modification calls are disabled during both fixed-size and free-size evaluation. Hence, the legality numbers here reflect raw generator or extension performance rather than the full agent recovery pipeline. The comprehensive results are presented in TABLE 3.2.

Fixed-Size Conditional Generation. On the standard 128×128 benchmark, ChatPattern achieves near-perfect legality (99.97%–99.99%) across both layer styles without topology selection, with a marginal diversity improvement over DiffPattern thanks to its ability to leverage a mixed multi-source training dataset with condition specifications. Note that the legality numbers in TA-

Table 3.2: Comparison of legality and diversity for conditional and free-size pattern generation (ChatPattern). ‘/’ refers to N/A. Topology selection and pattern modification calls are disabled for fair comparison of raw model quality.

Task Method	Train*	Size	Layer-10001		Layer-10003		Total†
			Leg. (↑)	Div. (↑)	Leg. (↑)	Div. (↑)	
Real	–	–	10.731	–	8.769	–	10.625
CAE+LegalGAN [134]	L10001		3.74%	5.814	–	–	–
VCAE+LegalGAN [134]	L10001	128 ²	84.51%	9.867	–	–	–
LayoutTransformer [117]	L10001		89.73%	10.527	–	–	–
DiffPattern [109]	L10001/10003		99.97%	10.711	99.98%	8.578	99.98% 10.633
ChatPattern	L10001/10003		99.97%	10.796	99.99%	8.625	99.98% 10.650
Real	–	–	12.702	–	10.696	–	12.695
DiffPattern+Concat. [109]	L10001/10003	256 ²	57.78%	10.719	93.69%	10.511	75.74% 11.706
ChatPattern	L10001/10003		87.36%	11.154	99.78%	10.556	93.57% 11.830
Real	–	–	13.435	–	12.139	–	13.787
DiffPattern+Concat. [109]	L10001/10003	512 ²	0.29%	5.714	40.83%	11.555	20.56% 11.359
ChatPattern	L10001/10003		36.42%	10.401	98.86%	11.620	67.64% 12.133
Real	–	–	13.573	–	12.644	–	14.109
DiffPattern+Concat. [109]	L10001/10003	1024 ²	0.00%	0.000	0.64%	6.926	0.32% 6.926
ChatPattern	L10001/10003		1.19%	6.438	94.96%	11.981	47.80% 11.992

* ‘L10001/10003’ abbreviates ‘Layer-10001/10003’; all training datasets are the 128×128 version.

† We collected generated samples from both Layer-10001/10003 and evaluated them together.

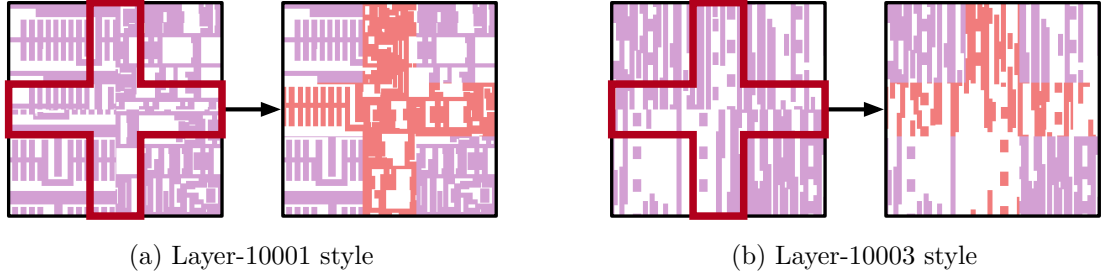


Figure 3.13: 256×256 topology matrices generated by In-Painting extension.

BLE 3.2 (99.97%) are slightly below the 100% in TABLE 3.1 because topology selection is deliberately disabled here for fair comparison.

3.9.4 Free-Size Pattern Generation

Having validated the fixed-size generation quality, we now evaluate the free-size extension capability enabled by the In-Painting and Out-Painting mechanisms described in Sections 3.8.4 and 3.8.5. ChatPattern is designed for free-size pattern generation, a task with high practical relevance. The baseline (DiffPattern with simple concatenation) experiences precipitous legality drops when scaling beyond 128^2 : legality plummets to 0.29% for Layer-10001 at 512^2 and 0.00% at 1024^2 , as naive stitching of independent patches inevitably violates spacing rules at patch boundaries. ChatPattern’s iterative Out-Painting and bounded In-Painting routines maintain substantially higher legality across all tested scales: 87.36% at 256^2 , 36.42% at 512^2 , and 1.19% at 1024^2 for the challenging Layer-10001, with Layer-10003 achieving 94.96% legality at 1024^2 . These results show both the benefit of boundary-aware extension and its limitation: global guidance remains weak for complex oversized Layer-10001 patterns. The generated patterns from In-Painting and Out-Painting are visualized in Figs. 3.13 and 3.14.

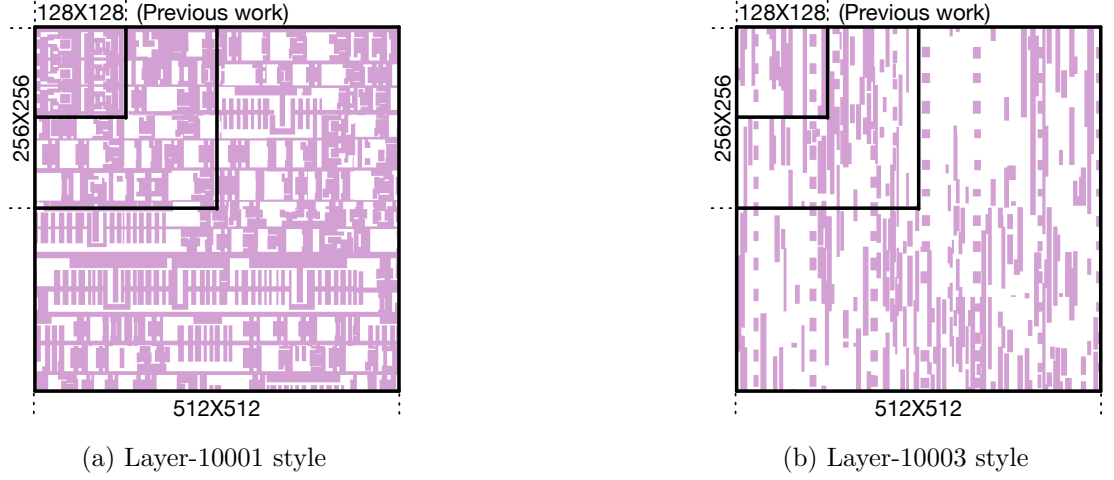


Figure 3.14: 512×512 topology matrices generated by Out-Painting extension.

3.9.5 Flexibility under Varying Design Rules

A key operational advantage of formulating topology generation separately from physical sizing is rule flexibility. As demonstrated in Fig. 3.15, a single synthesized topology can be instantiated under different DRC requirements without retraining the neural network, provided that the topology remains feasible under the updated constraints. When design rules change—such as increasing the minimum width or reducing the maximum polygon area—the trained topology generation model remains directly applicable; only the legalization system in Eq. (3.11) needs to solve under the updated constraints. This decoupling provides practical value as design rules evolve across technology revisions.

Furthermore, for a given topology and fixed set of design rules, the nonlinear system in Eq. (3.11) often admits multiple legal solutions, each corresponding to a different geometric assignment. This means a single topology can yield many distinct, all-legal layout patterns, further amplifying the effective diversity of the generated library.

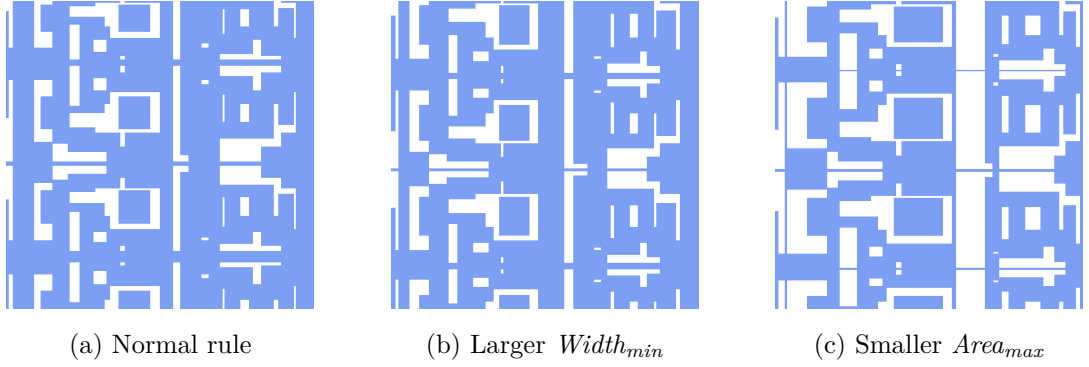


Figure 3.15: Layout patterns instantiated from the identical generated topology under varying DRC design rules. The explicit separation of topology generation from geometric legalization supports rule changes without retraining when the topology remains feasible.

3.9.6 Ablation Studies

We conduct ablation studies to validate three key design choices spanning different layers of the framework: the discrete formulation (generation layer), the augmentation strategy (training pipeline), and the extension strategy (customization layer).

Discrete vs. Continuous Diffusion. Using the unconditional DiffPatternFlex model on Layer-10001, we compare discrete diffusion against a continuous DDPM baseline [32]. Both models use the same U-Net architecture, training protocol, augmentation strategy, and hyperparameters; the only difference is that the continuous baseline generates real-valued outputs that are binarized via a fixed threshold of 0.5. Because the same legalization stage is applied after generation, we focus on diversity as the primary metric for accepted outputs. As shown in TABLE 3.3, discrete modeling improves diversity from 11.294 to 11.713, confirming that operating directly in the categorical state space is more effective for layout topology generation.

Table 3.3: Comparison between discrete and continuous diffusion for topology generation (unconditional DiffPattern-Flex, 100,000 patterns, Layer-10001).

Modeling	Generated Topology	Diversity ($\uparrow$)
Continuous	100,000	11.294
Discrete	100,000	11.713

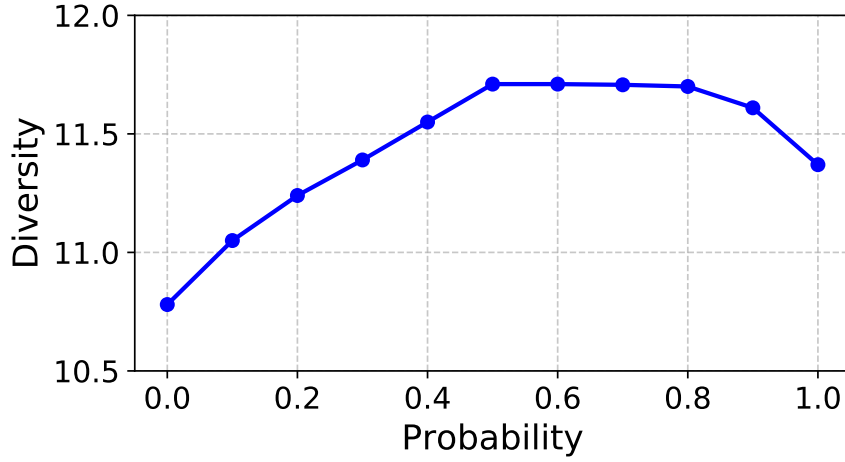


Figure 3.16: The effect of different probabilities of Concatenate-and-Crop augmentation on generated pattern diversity (unconditional DiffPattern-Flex).

Augmentation Probability. Still using the unconditional DiffPattern-Flex model, we investigate the impact of the Concatenate-and-Crop augmentation probability on diversity. By gradually increasing the augmentation probability from 0.0 to 1.0 during training, we retrain the model on the augmented data and evaluate the diversity of the generated samples. As shown in Fig. 3.16, the optimal probability range lies within $[0.5, 0.9]$, with diversity declining at very high probabilities due to the dominance of concatenated boundary artifacts. Based on these findings, we set the probability to 0.5 in our implementation.

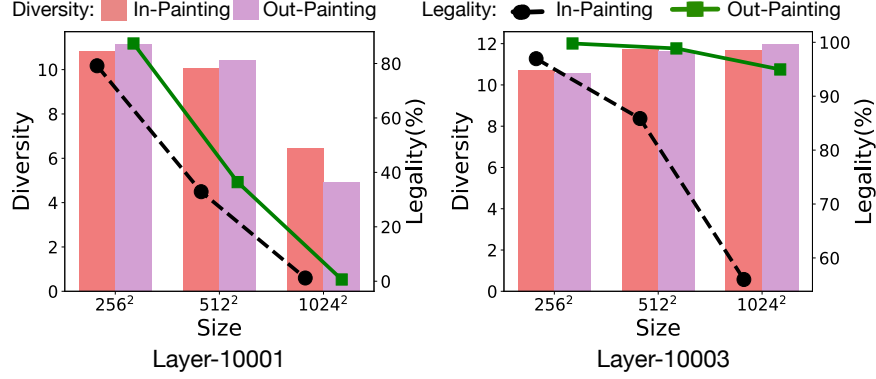


Figure 3.17: Evaluation of In-Painting and Out-Painting extension strategies across different target sizes (ChatPattern). Out-Painting generally achieves higher legality, while In-Painting can provide better diversity.

In-Painting vs. Out-Painting Extension. For the conditional ChatPattern model, the LLM agent can choose between In-Painting and Out-Painting strategies for free-size extension. Fig. 3.17 presents a comparative evaluation of both approaches. The results indicate that Out-Painting typically yields better legality, as it generates new regions conditioned on an existing legal boundary. In contrast, In-Painting excels in diversity under certain conditions, as it modifies interior regions that can exhibit more varied structural configurations. In practice, the LLM agent leverages documentation containing these statistical insights to select the optimal extension strategy for each sub-task.

3.9.7 System Efficiency

Efficiency dictates the practical deployment viability of large-scale layout generation frameworks. Since topology generation and legalization are decoupled in our framework, we evaluate the efficiency of each phase separately.

Topology Sampling Efficiency. TABLE 3.4 summarizes the sampling latency

Table 3.4: Topology sampling efficiency under different fast-sampling factors m (NVIDIA RTX 3090 GPU).

Method	m	Time (s)	Speedup	Diversity
Sampling	1	0.544	$1.00\times$	11.724
Fast-Sampling	5	0.118	$4.61\times$	11.715
Fast-Sampling	10	0.065	$8.37\times$	11.713
Fast-Sampling	20	0.039	$13.95\times$	10.573

Table 3.5: Legalization efficiency comparison (Intel Xeon Gold 6326 CPU).

Method	Extra Information	Time (s)	Speedup
Solving-R	Not required	0.269	$1.00\times$
Solving-E	Required	0.117	$2.30\times$
Solving-D	Not required	0.108	$2.48\times$

under different fast-sampling factors m . The integration of strided discrete sampling ($m = 10$) accelerates the generative forward pass by over $8\times$ with negligible degradation in topological diversity. At $m = 20$, the speedup reaches $13.95\times$ but diversity begins to decline significantly, indicating $m = 10$ as an effective operating point.

Legalization Efficiency. TABLE 3.5 presents the legalization efficiency comparison. Solving-R denotes the baseline with random initialization. Solving-E utilizes historical geometric vectors from patterns with the same design rules as initialization, achieving $2.30\times$ speedup. Solving-D employs the block-diagonal partition strategy described in Section 3.6, achieving $2.48\times$ speedup without requiring any extra information.

3.9.8 Evaluation of LLM Agent

To demonstrate the practical capabilities of the LLM agent in **ChatPattern**, we present two case studies from the pattern generation workflow. These examples evaluate tool orchestration and recovery behavior rather than constituting a comprehensive natural-language benchmark.

Requirement Auto-Formatting. Given a complex user request (e.g., “Generate 50,000 Layer-10001 patterns at 200×200 topology size with 1500×1500 nm physical dimensions, and separately generate 30,000 Layer-10003 patterns at 128×128”), the LLM agent decomposes the request and produces structured requirement lists:

```
# Requirement – subtask 1
## Basic Part: Topology Size: [200, 200], Physical Size: [1500, 1500] nm, Style:
Layer-10001, Count: 50000,
## Advanced Part: Extension Method: Out (Default: Out), Drop Allowed: True
(Default: True), Time Limitation: None (Default: None).
```

Each sub-task maps to a clear, machine-readable specification that the back-end APIs can execute directly. This auto-formatting capability eliminates the need for users to master complex generation tool interfaces.

Autonomous Error Recovery. When the legalization phase fails for a generated topology (especially common for large-size patterns), the LLM agent autonomously diagnoses and recovers from the failure:

```
% In previous attempts, one pattern cannot pass the legalization.
Thought: Since legalization has failed twice in the same region, I will try to
in-paint that specific area with the same style and then attempt legalization again.
Action: Topology_Modification
Action Input: { “topology_path”: ${path}, “upper”: 12, “left”: 56, “bottom”:
```

```
33, "right": 73, "style": "Layer-10001", "seed": 42 }  
% Continue Processing ...
```

This error-handling capability is not hard-coded in the standard operational pipeline; it is enabled by the LLM agent’s ability to interpret tool feedback and select a corrective action. By leveraging feedback from the legalization function (which identifies the problematic region), the agent plans corrective In-Painting actions, saving computational resources compared to simply discarding and re-generating entire topologies.

3.10 Summary

This chapter introduced a multi-tier framework for VLSI layout pattern generation. At the representation and generation layers, the Deep Squish Pattern encoding and discrete diffusion model produce diverse topologies natively in the binary state space, circumventing the thresholding errors intrinsic to prior continuous methods and pushing pattern diversity beyond that of the real dataset (from 10.777 to 11.713). At the legalization layer, a nonlinear programming system enforces compliance with specified industrial design rules on the accepted legalized outputs, achieving 100% legality at the scale of 100,000 patterns. At the customization layer, property-conditional generation, localized In-Painting for defect repair, and Out-Painting for free-size extension—orchestrated by an LLM agent through natural-language instructions—transform the batch generator into a more flexible pattern design tool. The free-size results also show a clear boundary of the method: legality remains high for easier styles but degrades on the more challenging Layer-10001 setting as the target size grows. The resulting design-rule-compliant pattern libraries directly address the data-scarcity bottleneck identified in Chapter 1 and provide a scalable supply of training and

benchmarking data for downstream physical design workflows, including the photoresist modeling and thermal-aware design tasks of Chapters [4](#) and [5](#).

With diverse, rule-compliant layout patterns now readily available, the natural next question is how faithfully these patterns can be transferred onto a physical wafer through the lithographic process. Answering this question requires accurate modeling of the photoresist, the light-sensitive material that mediates between the designed mask and the fabricated silicon. The following chapter addresses this challenge by developing differentiable, physics-constrained resist simulation and prediction tools.

Chapter 4

Physics-Constrained Photoresist Modeling

4.1 Introduction

Having established a scalable, design-rule-compliant layout generation pipeline in Chapter 3, we now turn to the fabrication-stage bottleneck posed in Chapter 1: accurately predicting how these layout patterns are transferred onto the silicon substrate through the lithographic process. This chapter embeds the classical Dill–Mack photoresist equations reviewed in Chapter 2 directly into differentiable learning pipelines, continuing the physics-driven learning paradigm with a focus on photochemical modeling.

Optical lithography constitutes the predominant patterning technology in modern semiconductor manufacturing, typically accounting for roughly 30% to 40% of total wafer fabrication expenditures [51]. At the heart of this process lies the photoresist, a light-sensitive polymer film whose spatially varying solubility, induced by exposure and subsequent chemical reactions, ultimately defines the circuit patterns transferred to the silicon substrate. Faithfully predicting the re-

sist response is therefore indispensable for controlling yield and reducing costly re-spins at advanced process nodes.

Resist simulation has historically proceeded along two complementary directions. The first, *2D resist modeling*, focuses on predicting the top-down or cross-sectional contour profile from a pre-computed aerial image. Methods in this category range from simple fixed-threshold approaches [2, 73] and empirical variable-threshold models [29, 53] to data-driven neural-network surrogates [115, 131]. Owing to their computational efficiency, these 2D methods have proven adequate for relatively mature nodes. However, as feature geometries approach the single-digit nanometer regime, secondary-electron scattering, standing-wave interference, and resist-thickness variation demand a more comprehensive treatment of the volumetric photoresist behavior.

The second direction, *3D resist modeling*, seeks to predict the full volumetric inhibitor-concentration profile across the resist depth. Classical analytical simulators [19, 52, 86, 96] compute this profile by numerically solving the coupled reaction-diffusion PDEs that govern exposure, post-exposure bake (PEB), and development. While principled, these methods face two growing obstacles: (i) calibrating the numerous nano-scale reaction parameters becomes progressively more difficult at advanced nodes [97, 98], and (ii) the iterative PDE solvers incur prohibitive computation times, particularly for the fine spatial and temporal resolutions required in EUV lithography.

Recent learning-based approaches attempt to accelerate 3D profile prediction by training neural networks on simulation data. In principle, the task can be viewed as generating a sequence of 2D concentration maps along the depth axis conditioned on a given aerial image, which is reminiscent of video prediction in computer vision [16, 25]. Nevertheless, directly transplanting general-purpose video generation architectures is impractical: such models typically demand mas-

sive training data and compute budgets [33, 85], whereas resist simulation benefits from well-studied photochemical priors that can narrow the model’s prediction task.

This chapter presents a two-tier framework that harnesses these physical priors across complementary levels of abstraction:

1. **TorchResist: A Differentiable Resist Simulator.** We first introduce TorchResist, an open-source, fully differentiable resist simulator built on the PyTorch framework [65]. Rather than relying on black-box neural networks, TorchResist parameterizes the classical Dill exposure and Mack development models with fewer than twenty physically interpretable parameters; automatic differentiation and GPU-accelerated parallel computation enable rapid calibration and inference. The simulator simultaneously outputs 2D binary resist images and 3D depth profiles, serving both as a calibrated analytical modeling tool on commercial-tool benchmarks and as the training-data engine for downstream learning-based methods.
2. **DiffResist: Physics-Constrained Diffusion for 3D Profile Prediction.** Building on the training data generated by TorchResist, we propose DiffResist, a novel framework that reformulates the 3D profile prediction task as a customized 2D diffusion process along the resist depth. Each denoising step in the diffusion model corresponds to one depth level, and physical boundary conditions at the air-resist interface are injected directly into the reverse process to suppress error accumulation. A two-stage noise schedule and a lightweight super-resolution network further enhance both accuracy and efficiency.

The remainder of this chapter is organized as follows. Section 4.2 surveys related work on resist modeling and generative sequential prediction. Section 4.3

briefly recaps the underlying physics of resist exposure and development. Section 4.4 details the architecture and calibration procedure of TorchResist. Section 4.5 presents the DiffResist methodology: its forward and reverse diffusion formulations, training and inference algorithms, and super-resolution module. Section 4.6 provides comprehensive experimental evaluations of both tools, and Section 4.8 concludes the chapter.

4.2 Related Work

4.2.1 Classical Resist Models

The foundations of analytical resist modeling were laid by Dill *et al.* [19], who formulated the exposure process as coupled ordinary differential equations governing light absorption and inhibitor decomposition in the resist film. The subsequent development step was formalized by Mack [52] through a nonlinear dissolution-rate model that maps the local inhibitor concentration to a developer etch rate. Together, the Dill–Mack framework has remained the standard analytical description of positive-tone resist behavior for over four decades [51]. Extensions to chemically amplified resists introduced reaction-diffusion dynamics during the post-exposure bake [86, 96], yet the core structure (sequential exposure, PEB, and development PDEs) persists in modern formulations.

At the opposite end of the complexity spectrum, *compact resist models* simplify the full PDE system to a single thresholding operation. Fixed-threshold approaches [2] apply a constant cutoff to the aerial image, while variable-threshold variants [29, 53, 73] introduce image-dependent corrections. These methods are valued for their speed and simplicity but are inherently limited in model capacity: they discard all volumetric information and rely on ad-hoc calibration of the threshold parameters.

4.2.2 First-Principle and Monte Carlo Simulators

For EUV lithography, the dominant energy-deposition mechanism shifts from direct photon absorption to secondary-electron scattering, necessitating stochastic simulation of the electron trajectories within the resist. Monte Carlo methods [97, 98] track individual electron scattering events and subsequent chemical reactions at the molecular level, providing the most physically faithful predictions of EUV resist profiles. Commercial tools such as Prolith, Sentaurus Lithography, and Dr. LiTHO implement combinations of analytical and Monte Carlo engines to handle a range of illumination conditions. Although highly accurate, these first-principle simulators incur computation times ranging from minutes to hours per exposure clip, rendering them impractical for large-scale or iterative optimization workflows.

4.2.3 Neural Network Surrogates

To bridge the gap between the accuracy of physical simulators and the speed required for production, several groups have explored neural-network surrogates for resist and lithography prediction. Zach *et al.* [131] trained shallow networks to approximate the resist development step, while Watanabe *et al.* [115] employed convolutional architectures to predict wafer contours directly from aerial images. More recent efforts replace the full optical-plus-resist pipeline with GPU-accelerated neural surrogates [8, 23] or embed the imaging model into differentiable computational graphs [7, 127], further reducing inference latency. Network-based methods excel at interpolating within their training distribution, yet their generalization to unseen process conditions remains uncertain, and the resulting models typically lack the physical interpretability of analytical formulations. TorchResist addresses this limitation by retaining the full analytical structure

while harnessing differentiable programming for efficient calibration.

4.2.4 Diffusion Models for Sequential and Video Prediction

DDPMs [32, 88] have achieved remarkable success across image synthesis, video generation, and scientific simulation. In the context of sequential prediction, diffusion-based video models [10, 16, 25, 33, 85, 108] generate temporally coherent frame sequences by iteratively denoising along the time axis. The resist depth-profile prediction task shares a structural resemblance: given a boundary condition at one end (the aerial image at the resist surface), the model must produce a sequence of spatially correlated 2D maps along the depth dimension. However, generic video diffusion architectures require massive datasets and computational budgets. DiffResist adapts the diffusion formalism to the resist domain by (i) replacing the standard Gaussian forward process with ground-truth physical layers, (ii) injecting analytical boundary conditions into the reverse process, and (iii) employing a compact two-stage noise schedule tailored to the physics of inhibitor concentration decay. These domain-specific modifications reduce both model capacity and training cost by orders of magnitude relative to general-purpose video diffusion.

4.3 Resist Process Recap

The physical photoresist process includes exposure, PEB, and development, whose detailed background is provided in Section 2.2.3. The methods in this chapter explicitly model the Dill exposure and Mack bulk-development components; PEB dynamics are treated as process background and revisited as future work. The overall workflow is illustrated in Fig. 4.1.

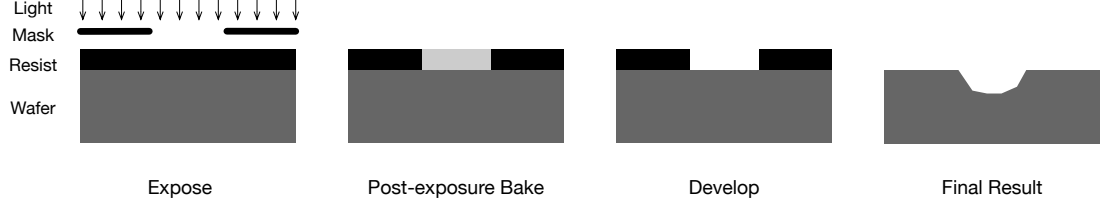


Figure 4.1: Schematic of a positive-tone resist process. During exposure a strong acid is generated; the PEB step catalytically alters resist solubility; and the subsequent development step dissolves exposed regions, leaving the final patterned image.

Exposure Model Following the Dill model [19], the intensity $\mathbf{I}(h, t)$ at depth h and the fractional inhibitor concentration $\mathbf{M}(h, t) = \mathbf{m}_1(h, t)/m_{10}$ satisfy the consolidated system

$$\begin{aligned} \frac{\partial \mathbf{I}(h, t)}{\partial h} &= -\mathbf{I}(h, t) [A \mathbf{M}(h, t) + B], \\ \frac{\partial \mathbf{M}(h, t)}{\partial t} &= -\mathbf{I}(h, t) \mathbf{M}(h, t) C, \end{aligned} \quad (4.1)$$

where A , B , and C are grouped physical constants (see Section 2.2.3 for their definitions in terms of individual molar absorption coefficients). The boundary and initial conditions are

$$\begin{aligned} \mathbf{M}(h, 0) &= 1, \\ \mathbf{M}(0, t) &= e^{-\mathbf{R} C t}, \\ \mathbf{I}(h, 0) &= \mathbf{R} e^{-(A+B)h}, \\ \mathbf{I}(0, t) &= \mathbf{R}. \end{aligned} \quad (4.2)$$

In particular, the top-layer boundary condition $\mathbf{M}(0, T) = e^{-\mathbf{R} C T}$ after the total exposure duration T will serve as the *physical constraint* injected into the DiffResist reverse process (Section 4.5.4).

Development Model The Mack bulk development model [52] converts the post-

exposure concentration profile $\mathbf{M}(h, T)$ into a spatially varying dissolution rate:

$$\mathbf{r} = r_{\max} \frac{(a+1)(1-\mathbf{M})^n}{a+(1-\mathbf{M})^n} + r_{\min}, \quad (4.3)$$

where $r_{\max}$, $r_{\min}$, n , and $a = \frac{n+1}{n-1}(1-m_{\text{TH}})^n$ control the dissolution behavior. The developer front propagation is governed by the Eikonal equation $|\nabla \mathbf{T}| = 1/\mathbf{r}$, solved via fast-marching level-set methods [62, 83]; the final developed profile is the iso-contour $\mathbf{T}(x, y, z) = t_{\text{dev}}$.

From Simulation to Learning The exposure–development pipeline yields accurate predictions but involves iteratively solving coupled PDEs at each depth level, with computation scaling unfavorably with the spatio-temporal resolution. This bottleneck motivates both approaches developed in this chapter: **TorchResist** reformulates the analytical pipeline in a fully differentiable manner to dramatically accelerate calibration and inference, while **DiffResist** replaces the iterative PDE solver entirely with a physics-constrained generative model.

In complexity terms, **TorchResist** scales with the number of spatial grid points, resist-depth levels, and exposure/development integration steps, although GPU batching amortizes independent clips. **DiffResist** reduces inference to a fixed number of depth-wise 2D denoising and super-resolution evaluations, so its cost scales mainly with image area and predicted depth rather than the simulator’s time-step count.

4.4 TorchResist: A Differentiable Resist Simulator

Conventional resist simulators implement the exposure–development pipeline in Section 4.3 through explicit numerical integration. While accurate, such implementations are rarely differentiable and cannot be driven by gradient-based

calibration or embedded into learning-based optimization workflows. To close this gap, we develop **TorchResist**, an open-source resist simulator whose computation graph, spanning from aerial-image input through Dill exposure, Mack development, and thresholding, is expressed in PyTorch [65] and is therefore automatically differentiable.

4.4.1 Architecture and Parameterization

TorchResist faithfully encodes the Dill exposure equations (Eqs. (4.1) and (4.2)) and the Mack bulk development model (Eq. (4.3)) into a differentiable computational graph. The constants A , B , C governing the exposure model, together with the development parameters $r_{\max}$, $r_{\min}$, m_{TH} , and the binarization threshold τ , constitute the parameter set θ , totaling fewer than twenty scalar values, which is orders of magnitude fewer than a typical neural-network surrogate. This compact parameterization acts as a strong structural regularizer, conferring excellent generalization even when calibrated on small datasets.

4.4.2 Differentiable Calibration

Given a calibration dataset $\{(\mathbf{R}_i, \mathbf{W}_i)\}_{i=1}^N$ of aerial-image and ground-truth wafer-image pairs, the calibration objective is

$$\theta^*, \tau^* = \arg \min_{\theta, \tau} \|\mathbf{W} - \Gamma(f_{\theta}(\mathbf{R}), \tau)\|_0, \quad (4.4)$$

where f_{θ} denotes the full resist simulation pipeline and $\Gamma(\cdot, \tau)$ is the binary thresholding operator. Because the L_0 -norm and the hard threshold are non-differentiable, we relax the objective via a sigmoid approximation and binary cross-entropy (BCE):

$$\mathcal{L} = \text{BCE}(\sigma(s f_{\theta}(\mathbf{R}) - s \tau), \mathbf{W}), \quad (4.5)$$

where $\sigma(\cdot)$ is the sigmoid function and s is a sharpness scaling factor that controls the steepness of the binarization boundary. Since every computational step in f_θ is implemented via differentiable PyTorch operations, the gradient $\nabla_\theta \mathcal{L}$ is obtained automatically, and standard optimizers such as Adam [40] can be employed for rapid convergence.

4.4.3 GPU-Accelerated Parallel Inference

Beyond calibration, the differentiable formulation brings a crucial practical advantage: all per-pixel computations within the exposure and development stages are embarrassingly parallel and can be fully vectorized on modern GPU hardware. This enables **TorchResist** to process an entire $2\mu\text{m} \times 2\mu\text{m}$ mask tile in 0.04s at 7 nm/pixel resolution on a single NVIDIA 3090 GPU. Furthermore, the output is not limited to a 2D binary image; **TorchResist** simultaneously produces a 3D depth-resolved inhibitor-concentration profile $\mathbf{M}(h, T)$, which can be directly consumed by downstream applications requiring volumetric resist information.

4.5 DiffResist for 3D Profile Prediction

While **TorchResist** provides a differentiable simulator for the modeled Dill–Mack process, its forward model advances the coupled exposure equations through many sequential time steps that cannot be parallelized, and at fine spatial grids each step incurs non-trivial cost. In the 3D profile setting of Section 4.6.3, convergence is measured against a 2000-step simulator reference, which costs about 77s per sample on an NVIDIA 3090. In this section we present **DiffResist**, a learning-based framework that amortizes this cost by bypassing the iterative PDE solver entirely, reformulating the 3D profile prediction as a physics-constrained 2D diffusion process. Building on the continuous DDPM framework detailed in Section 2.2.6,

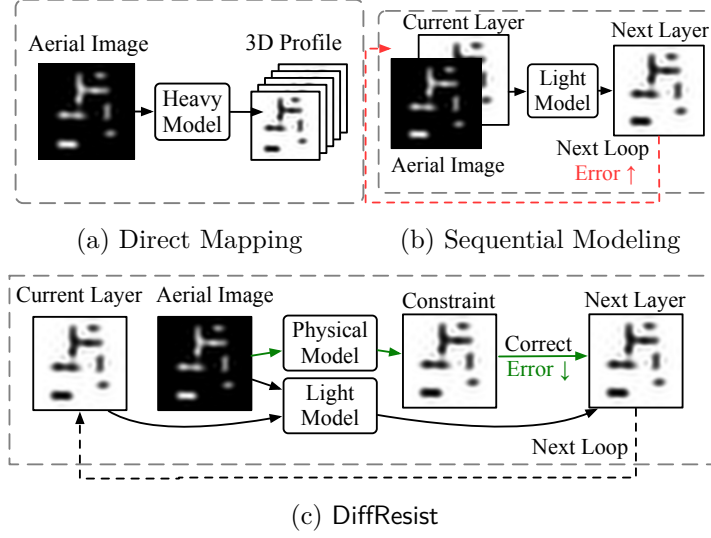


Figure 4.2: Comparison of modeling paradigms. (a) Direct mapping is computationally expensive due to the 3D output. (b) Sequential modeling predicts each depth layer from the preceding one but accumulates errors. (c) DiffResist injects physical boundary constraints at every reverse step to suppress error propagation.

DiffResist adapts the denoising diffusion formalism to the specific structure of resist depth profiles.

4.5.1 Problem Formulation and Modeling Paradigms

The goal is to predict the full fractional inhibitor concentration volume $\mathbf{M} \in [0, 1]^{H \times W \times D}$ from the aerial image $\mathbf{R} \in \mathbb{R}_{\geq 0}^{H \times W}$. Three modeling paradigms are illustrated in Fig. 4.2:

- **Direct mapping** learns a single function $\mathbf{M} = f_{\theta}(\mathbf{R})$, but the enormous output dimension ($H \times W \times D$) demands prohibitive model capacity and training time.
- **Sequential modeling** decomposes the 3D prediction into a Markov chain

over depth layers $\mathbf{M}_d$, with $d \in \{0, \dots, D-1\}$ (larger d closer to the wafer):

$$\mathbf{M}_0 = f_\theta(\mathbf{M}_1, 1) = f_\theta(f_\theta(\mathbf{M}_2, 2), 1) = \dots \quad (4.6)$$

While computationally lighter, errors accumulate across the sequential predictions, degrading the deeper layers.

- **DiffResist** retains the sequential structure but injects physical boundary constraints (derived in Section 4.3) into the reverse diffusion process, thereby anchoring the top-layer prediction and suppressing the error cascade.

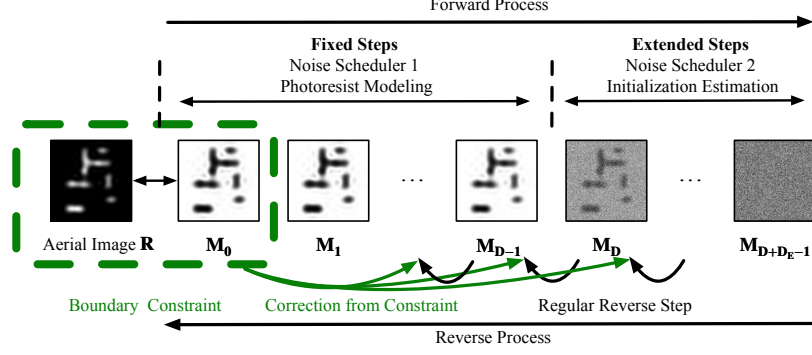
4.5.2 Framework Overview

DiffResist comprises two cascaded stages illustrated in Fig. 4.3:

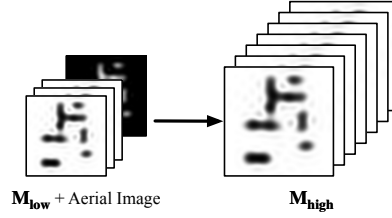
1. A *2D diffusion stage* operates on downsampled inputs to generate the depth-wise sequence of fractional inhibitor concentrations via a customized diffusion model.
2. A *super-resolution stage* restores the original spatial and depth resolution using a lightweight residual network.

4.5.3 Forward Diffusion Process

In DiffResist, the forward process defines a Markov chain over $D + D_E$ states indexed by a virtual “diffusion depth” $d \in \{0, 1, \dots, D + D_E - 1\}$. The first D states correspond to the physical resist depth layers, $\mathbf{M}_0$ at the air-resist interface and $\mathbf{M}_{D-1}$ closest to the substrate; the remaining D_E extended states are auxiliary noise layers that bridge the bottom physical layer to an isotropic Gaussian prior.



(a) 2D Diffusion Stage



(b) Super-Resolution Stage

Figure 4.3: Illustration of the DiffResist framework. (a) The 2D diffusion stage generates depth-wise concentration maps from random noise, with boundary-condition correction applied at each reverse step. (b) The super-resolution stage restores the full spatial and depth resolution.

Transitions follow the standard DDPM kernel with a fixed variance schedule $\beta_1, \dots, \beta_{D+D_E-1}$:

$$q(\mathbf{M}_d | \mathbf{M}_0) := \prod_{i=1}^d q(\mathbf{M}_i | \mathbf{M}_{i-1}), \quad (4.7)$$

$$q(\mathbf{M}_d | \mathbf{M}_{d-1}) := \mathcal{N}(\mathbf{M}_d; \sqrt{1 - \beta_d} \mathbf{M}_{d-1}, \beta_d \mathbf{I}),$$

with β_d chosen so that the terminal state $\mathbf{M}_{D+D_E-1}$ approximates $\mathcal{N}(\mathbf{0}, \mathbf{I})$.

Crucially, for the physical layers $0 < d < D$, the ground-truth concentration

values are directly available from the training data:

$$q(\mathbf{M}_d|\mathbf{M}_0) = \delta(\mathbf{M}_d - \mathbf{M}_d^{\text{data}}), \quad (4.8)$$

where $\delta(\cdot)$ denotes the Dirac delta. Only the extended layers $D \leq d < D + D_E$ follow the standard Gaussian forward diffusion:

$$q(\mathbf{M}_d|\mathbf{M}_0) = \mathcal{N}(\mathbf{M}_d; \sqrt{\bar{\alpha}_d} \mathbf{M}_0, (1 - \bar{\alpha}_d) \mathbf{I}), \quad (4.9)$$

with $\alpha_d = 1 - \beta_d$ and $\bar{\alpha}_d = \prod_{i=1}^d \alpha_i$.

4.5.4 Reverse Process with Physical Constraints

The reverse process generates each upper layer $\mathbf{M}_{d-1}$ from the lower layer $\mathbf{M}_d$, starting from a random sample $\mathbf{M}_{D+D_E-1} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$:

$$p_\theta(\mathbf{M}_{d-1}|\mathbf{M}_d) = \mathcal{N}(\mathbf{M}_{d-1}; \boldsymbol{\mu}_\theta(\mathbf{M}_d, d), \sigma_d^2 \mathbf{I}). \quad (4.10)$$

Following DDPM [32], the mean $\boldsymbol{\mu}_\theta$ is expressed in terms of the noise prediction network $\boldsymbol{\epsilon}_\theta$:

$$\boldsymbol{\mu}_\theta(\mathbf{M}_d, d) = \frac{1}{\sqrt{\bar{\alpha}_d}} \left(\mathbf{M}_d - \frac{\beta_d}{\sqrt{1 - \bar{\alpha}_d}} \boldsymbol{\epsilon}_\theta(\mathbf{M}_d, d) \right), \quad (4.11)$$

yielding the one-step update:

$$\mathbf{M}_{d-1} = \boldsymbol{\mu}_\theta(\mathbf{M}_d, d) + \sigma_d \mathbf{z}, \quad \mathbf{z} \sim \mathcal{N}(\mathbf{0}, \mathbf{I}). \quad (4.12)$$

Standard diffusion applies Eq. (4.12) without any domain knowledge, leading to three deficiencies for resist modeling: (i) the aerial image condition $\mathbf{R}$ is not incorporated; (ii) the specific physics of the resist process is ignored; and (iii) errors accumulate across sequential predictions. To address all three, we decompose

$\mathbf{M}_{d-1}$ into explicit Markov terms:

$$\begin{aligned} \mathbf{M}_{d-1}(\mathbf{M}_d, d) = & \underbrace{\frac{(1 - \bar{\alpha}_{d-1})\sqrt{\alpha_d}}{\sqrt{1 - \bar{\alpha}_d}} \cdot \boldsymbol{\epsilon}_\theta(\mathbf{M}_d, d)}_{\text{Directional correction}} \\ & + \underbrace{\sqrt{\bar{\alpha}_{d-1}} \hat{\mathbf{M}}_0(\mathbf{M}_d, d)}_{\text{Top layer estimation}} \\ & + \underbrace{\sigma_d \mathbf{z}}_{\text{Random noise}}, \end{aligned} \quad (4.13)$$

and replace the estimated top layer $\hat{\mathbf{M}}_0$ with the *exact* boundary condition from Eq. (4.2):

$$\mathbf{M}_0(\mathbf{R}) = e^{-\mathbf{R}CT}, \quad (4.14)$$

where C is the known decay constant and T the total exposure time. Substituting this physical constraint yields the non-Markovian corrected reverse step:

$$\begin{aligned} \mathbf{M}_{d-1}(\mathbf{M}_d, d, \mathbf{R}) = & \underbrace{\frac{(1 - \bar{\alpha}_{d-1})\sqrt{\alpha_d}}{\sqrt{1 - \bar{\alpha}_d}} \cdot \boldsymbol{\epsilon}_\theta(\mathbf{M}_d, d, \mathbf{R})}_{\text{Directional correction}} \\ & + \underbrace{\sqrt{\bar{\alpha}_{d-1}} \mathbf{M}_0(\mathbf{R})}_{\text{Corrected top layer}} \\ & + \underbrace{\sigma_d \mathbf{z}}_{\text{Random noise}}, \end{aligned} \quad (4.15)$$

where the noise prediction network now conditions on $\mathbf{R}$ as well. Because the concentration profile is a deterministic function of $\mathbf{R}$, randomness is removed for the physical layers ($0 < d < D$) by setting $\sigma_d = 0$; the extended layers adopt the DDPM variance schedule $\sigma_d^2 = \frac{1 - \bar{\alpha}_{d-1}}{1 - \bar{\alpha}_d} \beta_d$.

4.5.5 Training and Inference

Training Following the simplified DDPM objective [32], we optimize:

$$L_{\text{simple}}(\theta) = \mathbb{E}_{d, \mathbf{R}, \epsilon, \mathbf{M}} [\|\boldsymbol{\epsilon}(d) - \boldsymbol{\epsilon}_\theta(\mathbf{M}_d, d, \mathbf{R})\|^2], \quad (4.16)$$

Algorithm 1 DiffResist Training

- 1: **repeat**
- 2: $\mathbf{M}_0, \mathbf{R} \sim q(\mathbf{M}_0, \mathbf{R});$
- 3: $d \sim \text{Uniform}(\{1, \dots, D + D_E - 1\});$
- 4: Get $\mathbf{M}_d$ from Eq. (4.8) or Eq. (4.9);
- 5: Get $\epsilon(d)$ from Eq. (4.17) or Eq. (4.18);
- 6: Take gradient descent step on

$$\nabla_{\theta} \|\epsilon(d) - \epsilon_{\theta}(\mathbf{M}_d, d, \mathbf{R})\|^2;$$

- 7: **until** converged
-

where d is uniformly sampled from $\{1, \dots, D + D_E - 1\}$. For the extended layers ($d \geq D$), the target noise is the standard Gaussian perturbation:

$$\epsilon(d) = \frac{\mathbf{M}_d - \sqrt{\bar{\alpha}_d} \mathbf{M}_0}{\sqrt{1 - \bar{\alpha}_d}}. \quad (4.17)$$

For the physical layers ($0 < d < D$), the ground-truth concentration values are known, yielding a *fixed* noise target:

$$\epsilon(d) = \frac{\sqrt{1 - \bar{\alpha}_d}}{(1 - \bar{\alpha}_{d-1})\sqrt{\alpha_d}} [\mathbf{M}_{d-1} - \sqrt{\bar{\alpha}_{d-1}} \mathbf{M}_0]. \quad (4.18)$$

The training procedure is summarized in Algorithm 1.

Sampling and Inference Starting from a random sample $\mathbf{M}_{D+D_E-1} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$ and the aerial image $\mathbf{R}$, the reverse process generates $\mathbf{M}_{0:D+D_E-1}$ by repeatedly applying Eq. (4.15). Only the first D layers $\mathbf{M}_{0:D-1}$ are retained as the predicted concentration profile; the additional D_E steps serve as a stochastic warm-up that allows the model to refine its initial estimate before the physical prediction begins. The sampling procedure is given in Algorithm 2.

Algorithm 2 DiffResist Sampling

- 1: Given $\mathbf{R}$, $\mathbf{M}_{D+D_E-1} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$;
 - 2: **for** $d = D + D_E - 1$ **down to** 2 **do**
 - 3: Get $\mathbf{M}_{d-1}$ from $\mathbf{M}_d$ with Eq. (4.15);
 - 4: Record $\mathbf{M}_{d-1}$;
 - 5: **end for**
 - 6: Record $\mathbf{M}_0(\mathbf{R})$ with Eq. (4.14);
 - 7: **return** Last D items in record;
-

4.5.6 Two-Stage Noise Schedule

In standard diffusion models the noise schedule β_d follows a single monotonic profile (e.g., linear or cosine). We empirically find that the two-phase nature of DiffResist, where physical layers are followed by extended noise layers, benefits from a *two-stage* schedule:

$$\beta_d = \begin{cases} C_1, & 0 < d < D, \\ C_1 + \frac{d-D}{D_E-1}(C_2 - C_1), & D \leq d < D + D_E, \end{cases} \quad (4.19)$$

where $C_1 \ll C_2$ are constants. The low noise floor C_1 during the physical layers preserves fine-grained concentration details, while the linearly increasing noise during the extended layers smoothly connects the last physical layer to a near-isotropic Gaussian.

4.5.7 Efficient Super-Resolution Network

To reduce training and inference cost, the diffusion stage operates on spatially and depth-wise downsampled data, using reduction factors $(\lambda_D, \lambda_H, \lambda_W)$ along the depth, height, and width axes, respectively. The low-resolution output $\mathbf{M}_{\text{low}}$

is then upsampled by a lightweight residual super-resolution network:

$$\mathbf{M}_{\text{high}} = g(\mathbf{M}_{\text{low}}, \mathbf{R}) + \text{UP}(\mathbf{M}_{\text{low}}), \quad (4.20)$$

where g is a trainable convolutional module conditioned on the aerial image $\mathbf{R}$, and UP denotes non-parametric bilinear interpolation. The residual connection anchors the upscaled output to the diffusion prediction, ensuring that the super-resolution network only needs to learn the high-frequency correction. After upscaling, the final binary printed image $\mathbf{P}$ is computed from the predicted high-resolution concentration $\mathbf{M}$ via the bulk development model described in Section 4.3.

4.6 Experimental Results

This section evaluates both TorchResist and DiffResist on the LithoBench benchmark [136].

4.6.1 Datasets and Evaluation Metrics

TorchResist Dataset The TorchResist evaluation uses the MetalSet subset of LithoBench, comprising 16,472 tiles of size $2048 \times 2048 \text{ nm}^2$. For each tile, the central $812 \times 756 \text{ nm}^2$ region is cropped, and both the aerial image and the ground-truth resist image are generated at 7 nm/pixel resolution using commercial lithography tools. A random 80/20 split defines the test and calibration sets, consistent across all threshold baselines. Fig. 4.4 illustrates a representative tile, with the central region of interest highlighted by red dashed lines.

DiffResist Dataset The DiffResist evaluation uses the same LithoBench mask source, with a 3D profile generation setting tailored to depth-resolved inhibitor concentration prediction. Only the central $1024 \times 1024 \text{ nm}^2$ valid region of each

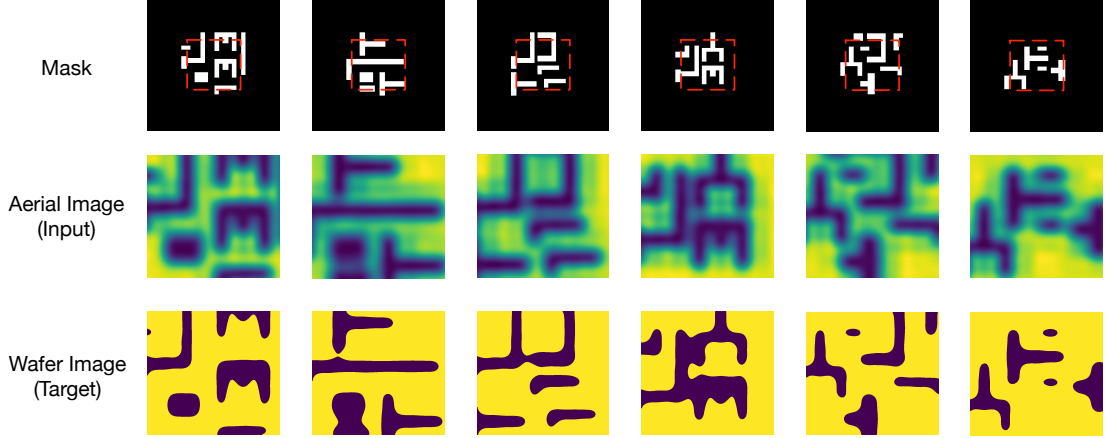


Figure 4.4: Illustration of the TorchResist evaluation dataset built from LithoBench. The region of interest within each mask tile is highlighted by red dashed lines.

tile is retained. Aerial images are generated with the ICCAD13 lithography simulator, and 3D fractional inhibitor concentration profiles are generated by TorchResist. The horizontal resolution is 7 nm/pixel and the vertical resolution is 1 nm/pixel, yielding padded volumes of size (75, 147, 147) before the downsampling described in Section 4.6.3. The dataset is split randomly into 80% training and 20% test samples.

Evaluation Metrics We employ three quantitative metrics:

- **Pixel Difference (PD):** The normalized L_0 -norm between the predicted and ground-truth binary resist images, expressed as a percentage.
- **EPE-mean:** The average edge placement error across all edges in a tile, in nanometers.
- **EPE-max:** The maximum edge displacement within a tile.

Reported values are averaged over all tiles in the test set. For 3D profile predic-

tion, we additionally report the MSE of the predicted fractional inhibitor concentration $\mathbf{M}$ against the ground-truth volume.

4.6.2 TorchResist Evaluation

Calibration Setup The absorption coefficient is set to $\alpha = A + B = 6.186/\text{nm}$ with $A = 0$, resist thickness is 75 nm, the development order is $n = 5$, and the binarization sharpness factor $s = 6$. The remaining parameters are calibrated via Adam [40] over 9 epochs, with an initial learning rate of 10^{-2} decayed by $0.3\times$ every 3 epochs and a batch size of 16. Calibration completes in approximately one hour on a single NVIDIA A100.

Baseline Methods We compare TorchResist against two threshold-based approaches:

- **Fixed Threshold** [2]: applies a constant threshold to the aerial image.
- **Variable Threshold** [73]: uses an adaptive threshold $\tau_{\text{var}} = c_1 + c_2 R_{\text{max}}$, where c_1, c_2 are calibrated constants and R_{max} is the local aerial-image maximum.

Accuracy Results TABLE 4.1 summarizes the comparison. TorchResist achieves a PD of 0.22%, roughly halving the error of the best threshold baseline, while simultaneously providing 3D depth simulation, a capability absent from all threshold methods.

Inference Efficiency and Scale Robustness TABLE 4.2 reports inference latency at two resolutions. At 7 nm/pixel a single tile is processed in 0.04 s; switching to the full 1 nm/pixel resolution incurs a $\sim 49\times$ cost increase due to the larger grid. Importantly, all model parameters remain unchanged across

Table 4.1: Performance comparison of resist models on LithoBench. The lithography model is a commercial tool.

Model	PD (%)	EPE-mean	EPE-max	Diff.	3D Sim.
Fixed Thres. [2]	0.59	1.52	4.45	✗	✗
Var. Thres. [73]	0.49	1.21	3.95	✗	✗
TorchResist	0.22	0.73	2.87	✓	✓

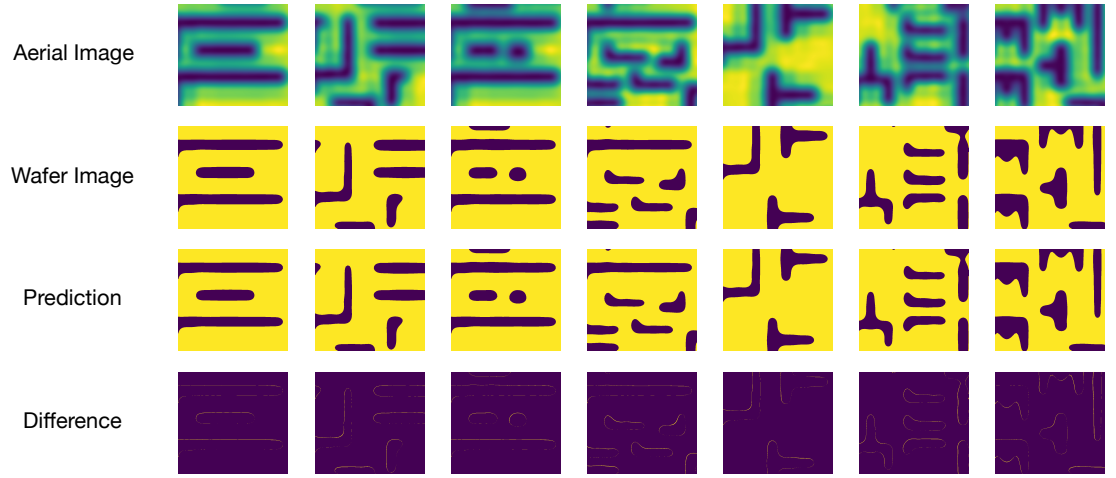


Figure 4.5: Visual comparison of TorchResist predictions against commercial-tool ground truth. All images are rendered at 1 nm/pixel.

resolutions; the average pixel difference between outputs at both scales is only 0.17%, confirming the scale robustness of the physics-based formulation.

Table 4.2: Inference efficiency of TorchResist on a single NVIDIA 3090 GPU.

Resolution	Cost Time (s)	Ratio
7 nm/pixel	0.04	1.00×
1 nm/pixel	1.98	48.96×

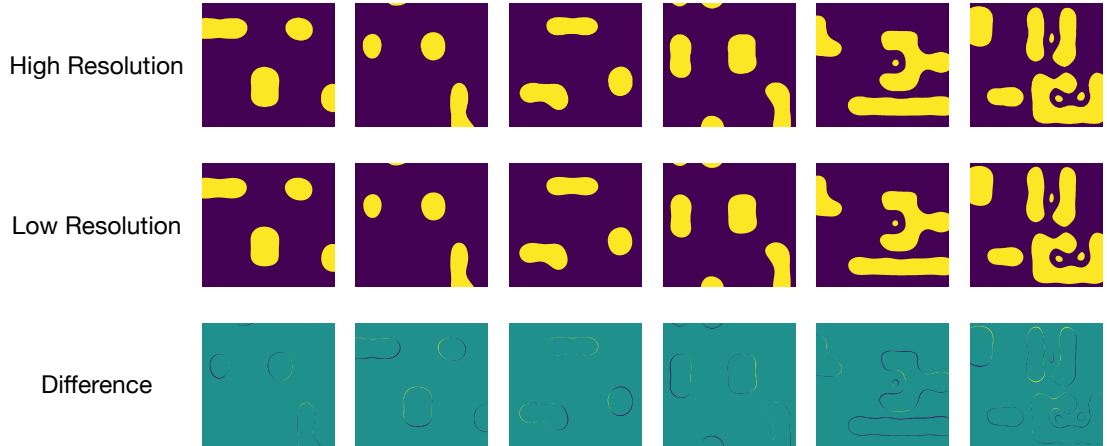


Figure 4.6: Comparison of TorchResist predictions at different resolutions, demonstrating scale robustness.

Extension to Open-Source Lithography Models To broaden the applicability of TorchResist, we also calibrate two additional variants using aerial images from the open-source lithography simulators FULT [127] and ICCAD13 [2], yielding TorchResist – F and TorchResist – I, respectively. Their evaluation results are listed in TABLE 4.3. These variants are calibrated against simulator-derived aerial images and commercial-tool resist labels, so the numbers are best interpreted as within-front-end calibration results rather than direct comparisons to the main commercial-lithography setting.

Table 4.3: Performance of TorchResist variants with different open-source lithography models.

Litho Model	Resist Model	PD (%)	EPE-mean (nm)	EPE-max (nm)
FULT [127]	TorchResist – F	1.77	7.03	31.91
ICCAD13 [2]	TorchResist – I	3.39	10.62	59.98

4.6.3 DiffResist Evaluation

Ground-Truth Generation The 3D resist profile ground truth is generated by TorchResist itself, run in non-differentiable forward mode with fixed (calibrated) parameters. Because a single exposure in our setup lasts approximately 10 s, 50 exposure steps (the value reported in the original TorchResist work for 2D contour accuracy) proved insufficient for the full 3D concentration volume. We therefore measure convergence through the per-voxel absolute difference

$$L_{\text{abs}}(N) = \frac{1}{HWD} \sum_{h,w,d} |\mathbf{M}_{h,w,d}^N - \mathbf{M}_{h,w,d}^{2000}| \quad (4.21)$$

between the output at N steps and the $N = 2000$ reference. As shown in Fig. 4.7, $L_{\text{abs}}(N)$ decreases steadily with N , and Fig. 4.8 illustrates that the residual between 50- and 2000-step outputs is visually non-negligible. We therefore adopt the 2000-step output as the training ground truth, at a cost of 76.9 s per sample on an NVIDIA 3090.

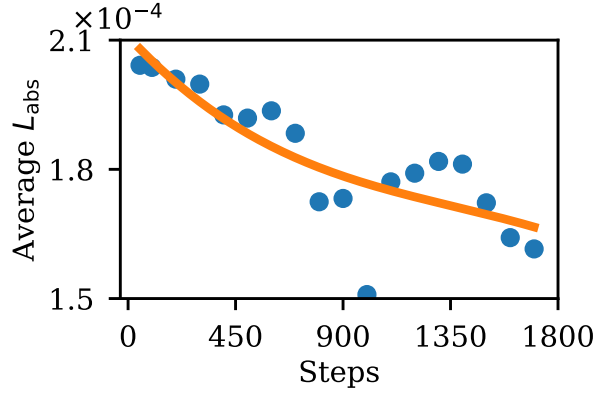


Figure 4.7: TorchResist convergence: the per-voxel absolute error $L_{\text{abs}}(N)$ relative to the 2000-step reference decreases steadily with the number of exposure steps N .

Model Configuration We adopt a U-Net backbone [77] for the noise prediction

network ϵ_θ . The two-stage noise schedule uses variance bounds $C_1 = 1 \times 10^{-4}$ and $C_2 = 2 \times 10^{-2}$. The original data dimensions $(75, 147, 147)$ are downsampled with factors $(\lambda_D, \lambda_H, \lambda_W) = (5, 3, 3)$ to $(15, 49, 49)$, giving $D = 15$ diffusion steps plus $D_E = 5$ extended steps. The super-resolution network has 0.48M parameters. Both the diffusion and super-resolution networks are trained for 50 epochs with batch size 32, using Adam [40] at a learning rate of 10^{-4} .

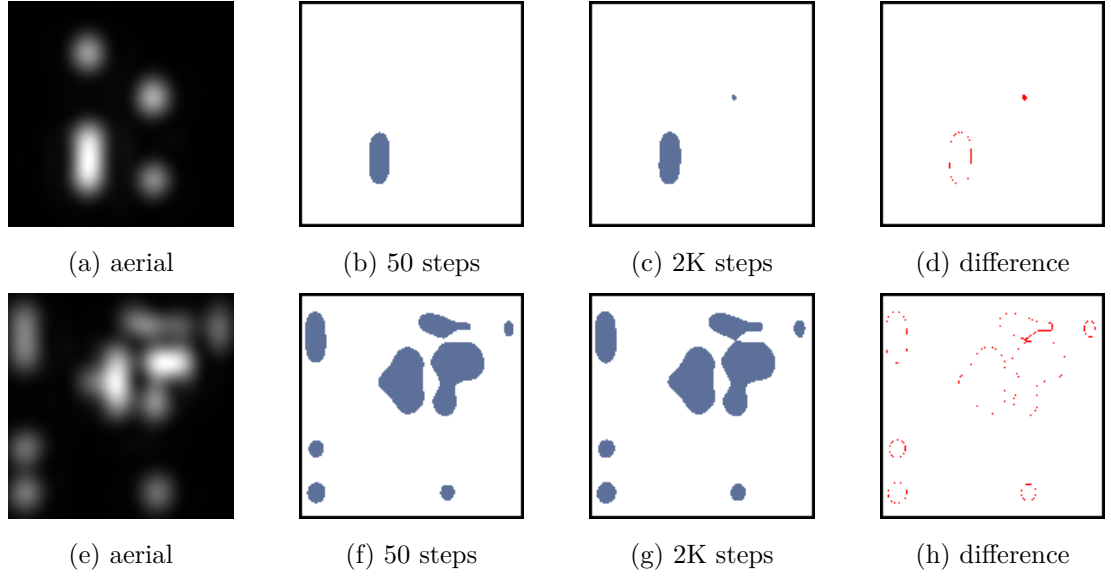


Figure 4.8: Visual comparison of TorchResist output at 50 vs. 2000 exposure steps. (a–d) and (e–h) show two test samples: aerial image, output at 50 steps, output at 2000 steps, and their absolute difference.

Baseline Methods We compare against four baselines spanning both paradigms:

- **Sequential:** SVG [16] (LSTM-based prior) and SVG-RNN (standard RNN variant [78]).
- **Direct mapping:** 3D-DDPM [32] (1000 training steps, 50 inference steps) and 3D-DDIM [88] (same model, DDIM sampler).

All baselines are reimplemented and trained for 100 epochs with tuned hyperparameters. The direct-mapping baselines use the same downsampled input resolution followed by bilinear upsampling.

Quantitative Results TABLE 4.4 presents the main comparison. The direct-mapping methods 3D-DDPM and 3D-DDIM carry far larger parameter counts, FLOPs, and training costs, yet still fail to reach competitive accuracy. The sequential baselines (SVG, SVG-RNN) and DiffResist share comparable model size, FLOPs, and training cost, but DiffResist achieves the best accuracy among the evaluated methods. In particular, DiffResist achieves 0.85% PD and 2.56 nm EPE-mean, substantially surpassing the strongest baseline SVG-RNN (1.02% PD, 4.62 nm EPE-mean).

Qualitative Results Fig. 4.9 visualizes printed images computed from the predicted concentration profiles via the development model. The difference maps highlight the zones of disagreement with the ground truth: DiffResist shows markedly lower residual error than both SVG and SVG-RNN across diverse pattern geometries.

Inference Efficiency TABLE 4.5 compares inference latency. The lightweight design of DiffResist yields a $\sim 15\times$ speedup over 3D-DDPM while simultaneously achieving far superior accuracy (2.56 nm vs. 10.17 nm EPE-mean).

4.6.4 Ablation Studies

We analyze the contribution of each key component in DiffResist.

Effect of Physical Constraint The boundary correction at the air-resist interface anchors the top-layer prediction and prevents error accumulation. Fig. 4.10 compares the per-layer MSE of DiffResist against the two sequential baselines:

Table 4.4: Evaluation results on the LithoBench benchmark. EPE values are in nanometers.

Method	Parameters	FLOPs	Training Cost (hrs)	MSE	PD (%)	EPE-mean (nm)	EPE-max (nm)
3D-DDPM	25.61M	15.53T	146.2	4.14×10^{-2}	19.10	10.17	40.47
3D-DDIM	25.61M	15.53T	146.2	3.79×10^{-2}	18.54	9.86	39.52
SVG	8.73M	99.91G	7.3	1.41×10^{-3}	2.18	7.73	19.14
SVG-RNN	7.15M	99.82G	7.5	7.39×10^{-4}	1.02	4.62	17.06
DiffResist	6.71M	69.57G	7.0	4.17×10^{-4}	0.85	2.56	13.64

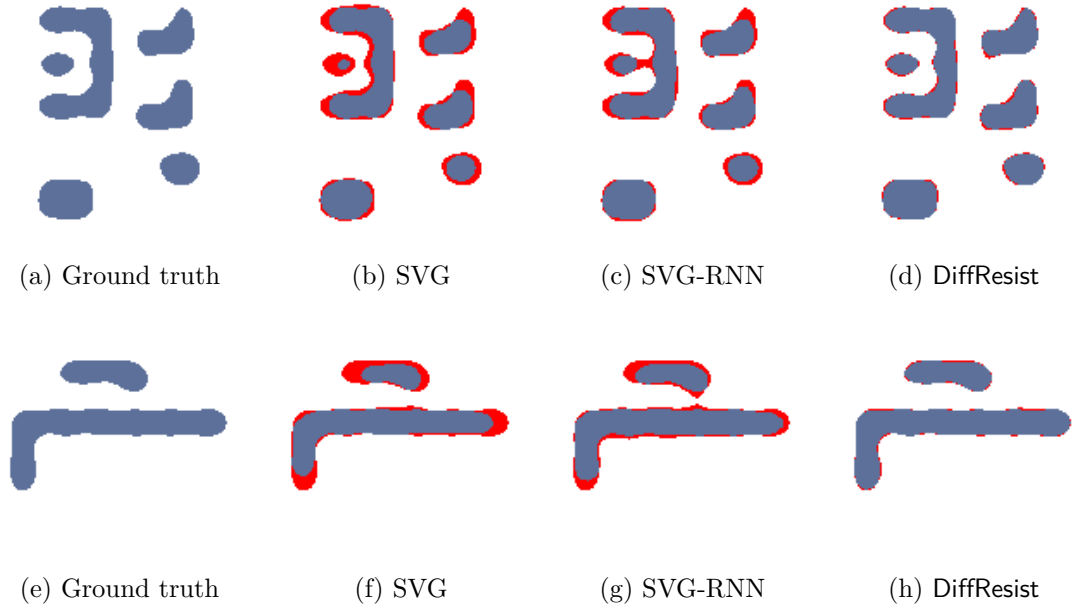


Figure 4.9: Qualitative comparison on two test samples. For each sample the leftmost subfigure is the ground-truth printed image, and the remaining three subfigures show the absolute difference between the ground truth and the printed image reconstructed from SVG, SVG-RNN, and DiffResist predictions, respectively. Lighter regions indicate larger error.

Table 4.5: Inference latency per sample, measured on an NVIDIA 3090 GPU.

Method	Time (s)	Acceleration	EPE-mean (nm)
3D-DDPM	6.45	1.00×	10.17
DiffResist	0.44	14.66×	2.56

all methods begin with comparable accuracy at the first prediction step, but the baseline errors grow rapidly with depth while DiffResist maintains low error throughout, because $\mathbf{M}_0$ is deterministically fixed by the physical boundary condition in Eq. (4.14).

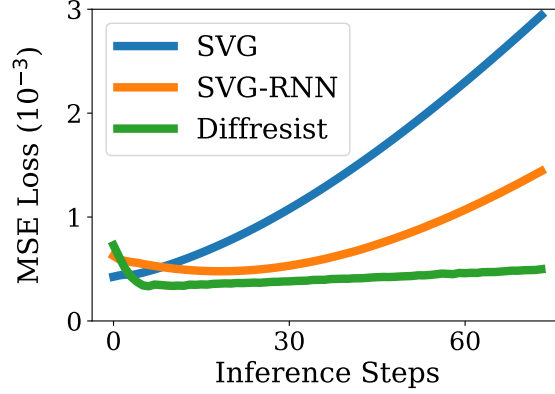


Figure 4.10: Per-layer prediction error: DiffResist vs. sequential baselines.

Two-Stage Noise Schedule Fig. 4.11 compares test-set MSE under the proposed two-stage schedule versus the standard linear schedule. Despite similar training loss, the two-stage schedule achieves notably lower test error, confirming that aligning the noise profile with the physical two-phase structure of DiffResist improves generalization.

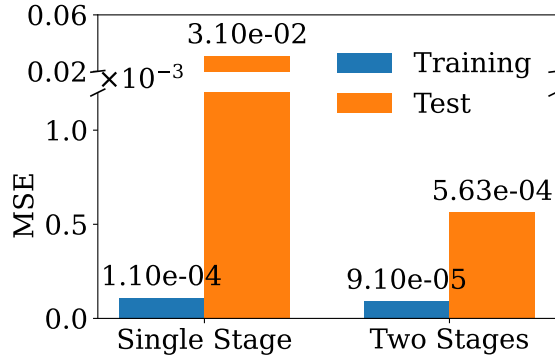


Figure 4.11: Two-stage vs. standard linear noise schedule on downsampled data.

Residual Super-Resolution Network Fig. 4.12 compares three upsampling strategies: non-parametric bilinear interpolation, a non-residual neural network, and the proposed residual design. The residual architecture achieves the best accuracy with the lowest additional parameter count, validating the effectiveness

of anchoring the upscaled output to the diffusion prediction.

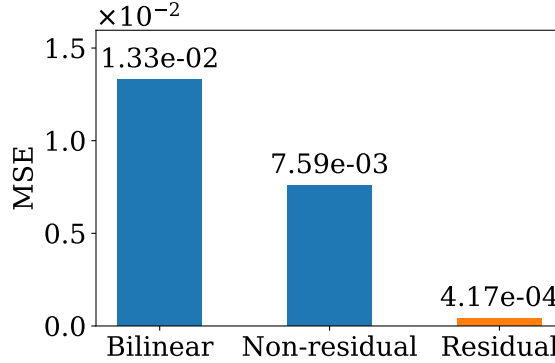


Figure 4.12: Ablation of super-resolution network variants.

Per-Pixel Error Heatmaps To visualize the spatial distribution of the benefit conferred by physical-boundary injection, Fig. 4.13 reports per-pixel error heatmaps at the first and last inference depths for a representative tile. At the top depth step, all three methods (SVG, SVG-RNN, and DiffResist) exhibit comparable residuals. At the bottom depth step, however, the SVG and SVG-RNN residuals spread across the entire tile, whereas the DiffResist residuals remain localized and low in magnitude, corroborating the per-layer MSE curves of Fig. 4.10.

4.7 Assumptions and Validity Range

TorchResist implements the Dill exposure and Mack bulk-development models for calibrated positive-tone resist settings. It captures optical absorption, inhibitor decay, and development-front propagation through compact physical parameters, but it does not explicitly model PEB acid diffusion, EUV secondary-electron stochasticity, etch effects, or chemistry changes outside the calibrated process range.

DiffResist injects the air-resist boundary condition as a hard constraint in

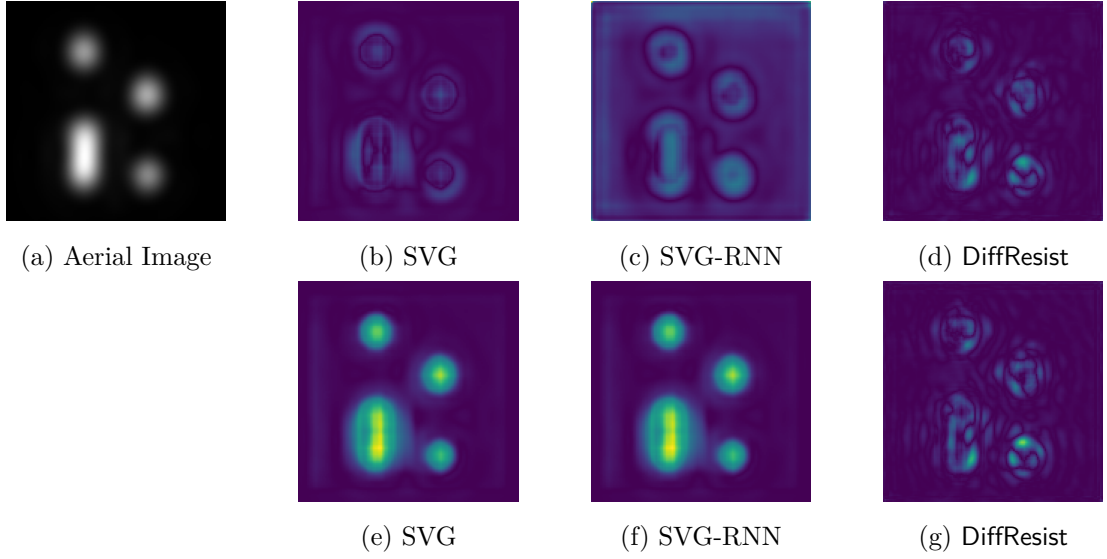


Figure 4.13: Error heatmaps at different inference depths. (a) Input aerial image. Top row (b–d): absolute error at the first (top) depth step for each method; bottom row (e–g): absolute error at the last (bottom) depth step. Lighter areas indicate larger error. Only DiffResist maintains low error across the full depth range.

the reverse process, while the interior depth profile remains a learned prediction. Therefore, physically inconsistent profiles may still appear under out-of-distribution aerial images, process parameters, or resist chemistries; the current validation mainly supports the benchmark and simulator-supervised settings studied in this chapter.

4.8 Summary

This chapter presented a unified two-tier framework for photoresist modeling that bridges physics-based simulation and data-driven learning. At the foundation level, TorchResist provides an open-source, fully differentiable resist simulator

whose compact parameterization (fewer than twenty interpretable constants) enables efficient gradient-based calibration and GPU-accelerated inference, while simultaneously producing both 2D binary resist images and 3D depth profiles. **TorchResist** serves a dual role: as a calibrated analytical model that outperforms threshold-based baselines on commercial-tool benchmarks, and as the simulator used to generate 3D supervision for **DiffResist**.

Building on this foundation, **DiffResist** reformulates the computationally intensive 3D resist profile prediction as a physics-constrained 2D diffusion process along the resist depth. The key innovation is the direct injection of physical boundary conditions from the Dill exposure model into the reverse diffusion step, which deterministically anchors the top-layer concentration estimate and effectively suppresses the error accumulation inherent in sequential modeling approaches. A tailored two-stage noise schedule and a lightweight residual super-resolution network further enhance both prediction accuracy and computational efficiency. Experimental results on the LithoBench benchmark show the best accuracy among the evaluated baselines (0.85% Pixel Difference, 2.56 nm EPE-mean) while providing a $\sim 15\times$ inference speedup over direct 3D diffusion baselines.

A promising direction for future research lies in incorporating additional physical priors, such as PEB diffusion dynamics and etch selectivity constraints, to further tighten the coupling between the generative model and the underlying photochemistry, potentially enabling even higher accuracy and broader generalization across diverse resist chemistries.

The preceding two chapters have addressed manufacturability from the design side, generating diverse layout patterns (Chapter 3) and accurately modeling the lithographic transfer process (this chapter). Once circuits are fabricated, however, their long-term reliability depends critically on thermal management, particularly

in the multi-die stacking and heterogeneous integration architectures that dominate advanced packaging. The next chapter tackles this post-fabrication challenge by developing an efficient, physics-grounded thermal analysis framework.

Chapter 5

Efficient Steady-State Thermal Analysis

5.1 Introduction

The previous two chapters developed physics-driven tools for the pre-fabrication stages of the IC design flow: generating manufacturable layout patterns (Chapter 3) and predicting photoresist behavior during lithographic patterning (Chapter 4). This chapter addresses the third and final pillar of the thesis, namely post-fabrication reliability, by tackling steady-state thermal analysis. Leveraging the Feynman–Kac stochastic representation introduced in Chapter 2, we exploit PDE structure to enable efficient point-query thermal estimation, completing the design–fabrication–reliability triad.

Heterogeneous Integration (HI) technologies, exemplified by 3D Integrated Circuits (3D-ICs) and advanced multi-die packaging, are the primary enablers of continued system-level performance scaling in modern semiconductor manufacturing [82]. As transistor counts and power densities continue to rise, however, thermal effects become an increasingly critical concern: excessive on-chip temper-

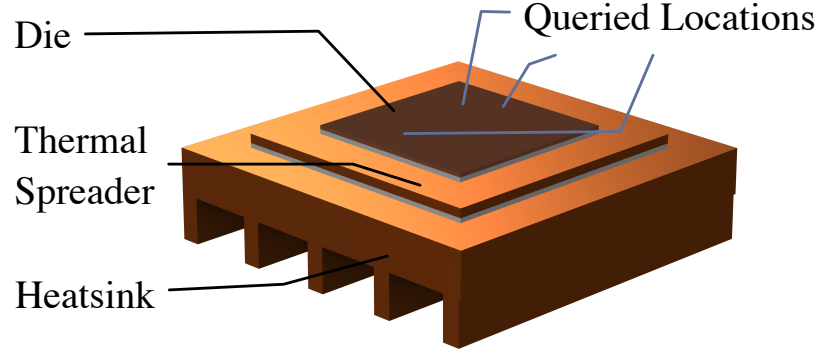


Figure 5.1: Pyramid-shaped IC geometry illustrating the point-query thermal estimation scenario. In many practical settings only the temperatures at a few queried locations (e.g., potential hotspots) are needed.

atures degrade performance, accelerate electromigration, and reduce device lifetime [70]. A reliable thermal analysis capability is therefore indispensable during the design-space exploration phase, where thousands of candidate configurations must be evaluated before the final package design is committed [50, 93, 104].

Steady-state temperature distributions in IC structures are governed by a heterogeneous heat-conduction equation with mixed boundary conditions (Section 2.3), a PDE commonly solved through deterministic numerical pipelines such as analytical Green’s-function methods, finite-element/difference/volume discretizations, and compact RC-network models. A detailed account of these method families, and of stochastic random-walk alternatives, is deferred to the related-work discussion in Section 5.2. In many practical scenarios, however, only the temperatures at a small number of queried locations (for example, the hotspot positions identified by early-stage thermal screening) are needed [133] (Fig. 5.1). Stochastic random-walk methods offer a natural fit for such *point-query* estimation because each query point can be evaluated independently and in parallel [71, 102, 118].

Reflected Feynman–Kac random walks [21, 47, 49, 124] are particularly well suited for mixed-boundary heat conduction problems involving Neumann (adiabatic) and Robin (convective) conditions. Unlike earlier heuristic boundary treatments, these methods compute boundary contributions by tracking the *local time* that the stochastic process spends near reflecting boundaries, yielding principled and accurate estimates. However, in the practically important regime where no Dirichlet boundary is present, the Feynman–Kac multiplicative weight $\hat{e}_c(t)$ decays only upon boundary reflections; the resulting trajectories must therefore be extremely long before the weight drops to a negligible level, severely limiting computational efficiency [80].

This chapter introduces **FastRW**, an efficient random-walk solver for steady-state thermal analysis that overcomes this bottleneck through a quantitative understanding of the truncation error. We first prove a *stopping-time truncation residual identity*: the expected tail contribution after truncating a Feynman–Kac random walk equals the multiplicative weight times the true temperature at the truncation location. This identity converts the truncation-correction problem into a continuation-value estimation task. Leveraging this insight, **FastRW** terminates trajectories early and compensates the residual using an inexpensive auxiliary predictor (e.g., a coarse FEM solution), reducing truncation error by orders of magnitude with negligible overhead. For multi-point estimation, **FastRW** further exploits cross-trajectory interactions via a Bayesian linear-Gaussian fusion that produces uncertainty-aware joint estimates across all query points. Our main contributions are:

1. A stopping-time truncation residual identity that precisely characterizes the expected tail contribution of finite-horizon reflected Feynman–Kac Monte Carlo.
2. **FastRW**, a truncation-corrected random-walk estimator that uses prior esti-

mation to enable principled early termination with a favorable bias–variance tradeoff.

3. A multi-point Bayesian fusion formulation that leverages cross-trajectory observations to produce joint estimates across query points with reduced path counts.
4. Standard 3D chip thermal benchmarks demonstrating over $6.5\times$ measured speedup at comparable accuracy over a strong Feynman–Kac baseline.

5.2 Related Work

This section surveys the main families of thermal analysis methods relevant to IC design, positioning the random-walk approach adopted in this chapter within the broader landscape.

Analytical Methods Analytical approaches to chip thermal modeling leverage Green’s function representations to derive closed-form temperature expressions for layered or rectangular geometries [103, 132]. While computationally efficient for simple configurations, these methods become unwieldy for the complex, heterogeneous multi-material structures typical of modern 3D-IC packages, where material interfaces, non-uniform power maps, and mixed boundary conditions preclude tractable closed-form solutions.

Numerical Discretization Methods FEM, finite difference methods (FDM), and finite volume methods (FVM) discretize the governing PDE on a spatial mesh and solve the resulting linear system [44, 45]. These methods handle arbitrary geometries and material heterogeneity with high accuracy, but their computational cost scales with the total number of mesh elements, typically millions for a full 3D chip model. Acceleration techniques such as ADI schemes reduce the per-step

cost by factoring the system into a sequence of tridiagonal solves [106, 107], yet the fundamental requirement for a global mesh and system-level linear algebra remains. Consequently, these methods are expensive when only a few point-query temperatures are needed.

Compact Thermal Models Compact thermal models (CTMs) abstract the detailed geometry into equivalent resistance–capacitance (RC) networks, transforming the thermal PDE into a SPICE-like circuit simulation [34]. Tools such as 3D-ICE [91] and PACT [130] automate the construction of compact models for multi-die stacks and achieve substantial speedups over full numerical solvers. However, CTMs sacrifice spatial resolution and can introduce modeling errors in regions with steep thermal gradients or complex boundary conditions. Moreover, building an accurate compact model still requires a one-time calibration against a full-field solver.

Stochastic Random-Walk Methods Stochastic methods offer a fundamentally different paradigm: rather than solving for the full temperature field, they estimate the temperature at individual query points by simulating random trajectories of a Brownian motion whose expectation equals the PDE solution. The WOS algorithm [56] exploits the mean-value property of harmonic functions to take large jumps in source-free regions, dramatically reducing the number of simulation steps compared to naive grid-based random walks. Floating random-walk methods extend this idea to handle grounded conductors and capacitance extraction in VLSI design [71, 102, 118].

For thermal problems with mixed boundary conditions, reflected Feynman–Kac random walks provide a principled treatment of Neumann and Robin boundaries by tracking the boundary local time of the stochastic process. The pioneering work of Liang et al. [47] introduced a hybrid random-walk approach

handling adiabatic and convective boundaries. Subsequent improvements by Yang et al. [124] and Dong et al. [21] incorporated the Feynman–Kac multiplicative weight formulation for improved accuracy. Luo et al. [49] demonstrated GPU-accelerated implementations for large-scale thermal analysis. Despite these advances, reflected Feynman–Kac methods suffer from slow convergence in reflecting-dominated regimes (i.e., when no Dirichlet boundary is present), because the multiplicative weight decays only through boundary reflections and trajectories must be extremely long before the weight becomes negligible. This fundamental efficiency bottleneck motivates the truncation-correction approach developed in this chapter.

Truncation, Bias, and Variance Reduction Finite-horizon truncation is a common practical necessity when simulating stochastic representations of elliptic PDE solutions, but it introduces bias in reflection-dominated regimes where absorption is rare. Prior work has studied randomized truncation and debiasing techniques for Monte Carlo estimators, such as Russian roulette [54, 75], to obtain unbiased estimators with controlled variance. Variance-reduction methods including control variates and surrogate-assisted Monte Carlo [58] offer complementary strategies. The contribution of this chapter is distinct: we derive a stopping-time residual identity that exactly characterizes the expected truncation tail given the truncation state, reframing truncation correction as a continuation-value estimation problem and enabling plug-in correction via an inexpensive auxiliary predictor.

Machine-Learning-Based Thermal Surrogates Recent work has explored neural-network surrogates for chip thermal prediction, including convolutional architectures trained on FEM simulation data and physics-informed neural networks. These approaches can produce rapid temperature estimates but typically

require substantial training data and may generalize poorly to unseen geometries or boundary conditions. In this chapter, we adopt a different philosophy: rather than replacing the physics-based solver, we use inexpensive computational surrogates (e.g., coarse FEM solutions) as *priors* within a principled random-walk framework, keeping the Feynman–Kac structure explicit while improving efficiency.

5.3 Problem Formulation and Random Walks

We consider the steady-state thermal problem formulated in Section 2.3: the temperature field $T(\mathbf{x})$ over a heterogeneous 3D chip domain Ω satisfies a heat-conduction equation with mixed Dirichlet, Neumann, and Robin boundary conditions. Within each homogeneous material subdomain, this problem admits the normalized Poisson form equation 2.22 whose Feynman–Kac representation equation 2.23 expresses $T(\mathbf{x})$ as an expectation over reflected Brownian motion paths. The computational task studied in this chapter is the point-query version of Problem 1: estimate $T(\mathbf{x}_q)$ at a small set of query locations $\{\mathbf{x}_q\} \subset \Omega$, without first assembling a global temperature field. This section summarizes the two random-walk discretizations used in the remainder of the chapter (WOG and WOS) and isolates the truncation bottleneck that motivates the FastRW methodology.

5.3.1 Random Walk Discretizations

The continuous reflecting Brownian motion is approximated in practice by discrete random walks. Depending on the local structure, two complementary discretizations are combined (Fig. 5.2).

WOG. Within the heat-source region, the domain is voxelized on a uniform grid

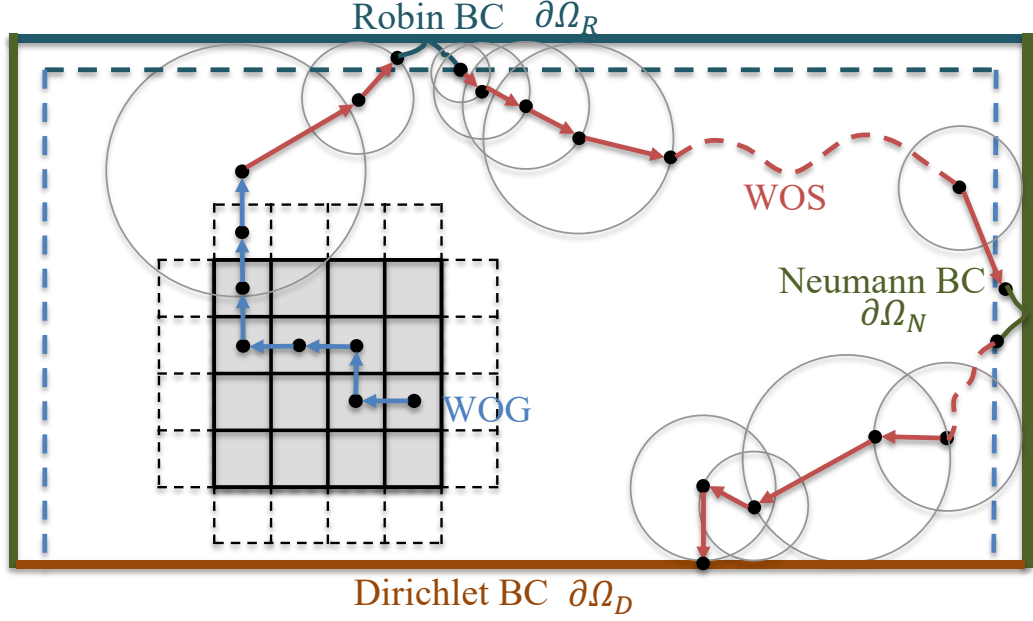


Figure 5.2: Illustration of a random-walk path in FastRW. WOG is used within the heat-source region; WOS accelerates transit through source-free homogeneous regions.

with per-voxel temperature and power-density values. Each step moves between adjacent grid nodes with conductance-weighted transition probabilities as defined in equation 2.25. WOG faithfully tracks volumetric source accumulation but takes small steps, so its cost is governed by how quickly the Feynman–Kac weight $\hat{e}_c(t)$ decays.

WOS. In homogeneous, source-free regions the Poisson equation reduces to Laplace’s equation; the mean-value property equation 2.26 then allows large jumps from the center of a sphere to a uniformly sampled point on its surface, with radius chosen maximally without crossing a material interface or domain boundary. WOS therefore crosses the bulk of the chip package in few steps.

Following [47, 124], we apply WOG inside the heat-source region and WOS

in source-free regions. Crucially, the Feynman–Kac weight $\hat{e}_c(t)$ only evolves along boundary local-time accumulation, so neither discretization shortens the trajectory once a path has entered the reflection-dominated regime—the very regime where truncation bias is most severe.

5.3.2 The Truncation Challenge

In practice, the infinite integrals in the Feynman–Kac representation are truncated: simulation terminates when either (i) the path hits a Dirichlet boundary, or (ii) the multiplicative weight $\hat{e}_c(t)$ drops below a predefined threshold Λ as boundary local time accumulates. When Dirichlet boundaries are present, paths usually absorb before the weight decays appreciably. In the more common and challenging regime with only Neumann/Robin boundaries, however, the weight decays slowly, requiring an enormous number of steps before $\hat{e}_c(t)$ reaches a small threshold. Existing methods rely on empirical heuristics to set the truncation level, lacking a principled guideline for path termination. This bottleneck is the central problem addressed by **FastRW**.

5.4 Truncation Residual Identity

This section formalizes the truncation error caused by finite-horizon approximation of the reflected Feynman–Kac representation and derives a residual identity at a stopping time.

5.4.1 Stopping-Time Identity

We focus on the reflecting-dominated regime where absorption at $\partial\Omega_D$ is unlikely before the weight $\hat{e}_c(t)$ decays appreciably. If a Dirichlet boundary is reached

before the truncation rule triggers, estimation terminates by absorption and no truncation error is incurred.

Consider one Monte Carlo realization of the Feynman–Kac estimator starting from $\mathbf{x}$. For a threshold $\Lambda \in (0, 1)$, let $t_\Lambda = \inf\{t \geq 0 : \hat{e}_c(t) \leq \Lambda\}$ be the truncation time. The single-path contribution decomposes as

$$T_i(\mathbf{x}) = \mathcal{I}_0^{t_\Lambda} + \mathcal{I}_{t_\Lambda}^\infty, \quad (5.1)$$

where $\mathcal{I}_0^{t_\Lambda}$ aggregates all source and boundary terms collected on $[0, t_\Lambda]$, and $\mathcal{I}_{t_\Lambda}^\infty$ is the remaining contribution, which represents the truncation error for this realization.

The following theorem provides an exact characterization of the expected truncation error:

Theorem 1 (Stopping-time truncation residual identity). Let $\{X_t\}_{t \geq 0}$ be the reflected process defined in Section 2.3 with natural filtration $\{\mathcal{F}_t\}_{t \geq 0}$, and let τ be a stopping time such that $\tau < t_D$ on the event of interest. Then the truncation tail satisfies

$$\mathbb{E}[\mathcal{I}_\tau^\infty \mid \mathcal{F}_\tau] = \hat{e}_c(\tau) T(X_\tau). \quad (5.2)$$

A complete proof is provided in Section A.1.

Corollary 1. Given a truncation state X_τ and the corresponding weight $\hat{e}_c(\tau)$, the expected truncation error of the remaining infinite-horizon contribution is exactly $\hat{e}_c(\tau) T(X_\tau)$.

The identity is fundamentally a strong-Markov argument. It holds in continuous time and for the discrete-time Markov chain estimators used in practice, with τ a stopping time for the discrete filtration and $\hat{e}_c$ the corresponding multiplicative weight along the path.

5.4.2 Practical Interpretation

For the threshold rule $\tau = t_\Lambda$, Eq. (5.2) yields

$$\mathbb{E}[\mathcal{I}_{t_\Lambda}^\infty \mid \mathcal{F}_{t_\Lambda}] = \Lambda T(X_{t_\Lambda}). \quad (5.3)$$

This result has a clear physical interpretation: the truncation error corresponds to imposing a Dirichlet boundary condition at the truncation location X_{t_Λ} . Prior methods chose Λ empirically without a quantitative guide; Eq. (5.3) provides a principled criterion: Λ should be selected so that the expected residual $\Lambda \cdot \mathbb{E}[T(X_{t_\Lambda})]$ remains within the desired accuracy tolerance.

Crucially, Eq. (5.3) converts truncation correction into a *continuation-value estimation* problem: if an accurate approximation of $T(X_{t_\Lambda})$ is available, the tail contribution can be corrected rather than computed through continued simulation. This observation is the theoretical foundation of the **FastRW** methodology developed in the next section.

5.5 FastRW Methodology

Building on the truncation residual identity, this section presents **FastRW**, a truncation-corrected random-walk estimator. We first describe how single-point estimation can be accelerated by compensating the truncation residual with an inexpensive prior temperature estimate, and then extend the framework to multi-point estimation through a Bayesian posterior fusion that exploits cross-trajectory observations.

5.5.1 Single-Point Acceleration via Prior Estimation

When the query set $\{\mathbf{x}_q\}$ contains a single point, an unbiased truncation correction requires the exact temperature at the truncation location (Corollary 1),

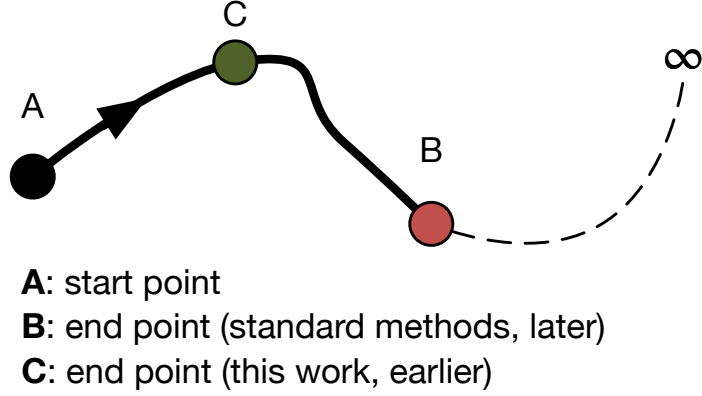


Figure 5.3: A random walk path in single-point estimation. Standard methods compute the integral from A to B. **FastRW** terminates at C and estimates the remaining contribution from C to infinity using the prior temperature field.

which is unknown during simulation. To make truncation correction practical, **FastRW** introduces a pre-computed prior temperature estimate $\tilde{T}(\mathbf{x})$ over a sub-domain $\Omega_s \subseteq \Omega$:

$$\tilde{T}(\mathbf{x}) = T(\mathbf{x}) + \epsilon, \quad \mathbf{x} \in \Omega_s, \quad (5.4)$$

where ϵ denotes the estimation error. With this prior available, each random walk in **FastRW** terminates under one of two conditions:

1. the path reaches a Dirichlet boundary, or
2. the weight $\hat{e}_c(t)$ drops below a threshold Λ *and* the current position lies in Ω_s .

When termination occurs under the second condition at time t , **FastRW** augments the accumulated contribution with the prior estimate:

$$T_i(\mathbf{x}) = \mathcal{I}_0^t + \hat{e}_c(t) \tilde{T}(X_t) = \mathcal{I}_0^t + \mathbb{E}[\mathcal{I}_t^\infty] + \hat{e}_c(t) \epsilon, \quad (5.5)$$

where the second equality follows from Corollary 1. The effective truncation error is thereby reduced from $\hat{e}_c(t) \cdot T(X_t)$ to $\hat{e}_c(t) \cdot \epsilon$. Since typically $\epsilon \ll T(\mathbf{x})$, the

allowable threshold Λ can be chosen much larger than in conventional methods, significantly shortening the required walk lengths (Fig. 5.3).

After N independent paths from the query point $\mathbf{x}$, the estimated temperature is computed by averaging:

$$\bar{T}(\mathbf{x}) = \frac{1}{N} \sum_{i=1}^N T_i(\mathbf{x}), \quad (5.6)$$

with variance $\text{Var}[\bar{T}(\mathbf{x})] = Z(\mathbf{x})/N$, where $Z(\mathbf{x})$ is the per-path variance. Because **FastRW** uses a larger Λ and shorter paths, the per-path variance is also typically smaller than in conventional methods.

For a sparse set of query points, the runtime is governed by the number of queried locations, the number of Monte Carlo paths per location, and the expected path length. **FastRW** reduces the latter two factors through bias-controlled early termination and posterior fusion, but the method is designed for point-query estimation rather than full-field reconstruction.

Principled threshold selection. The plug-in correction in equation 5.5 is generally biased unless $\tilde{T}(X_t)$ is exact. A simple bias bound, whose full derivation together with an MSE decomposition is deferred to Section A.2, shows that under the threshold rule $\hat{e}_c(t) \leq \Lambda$ and uniform predictor error $|\epsilon(\mathbf{x})| \leq \epsilon_{\max}$ over Ω_s ,

$$|\mathbb{E}[\bar{T}(\mathbf{x})] - T(\mathbf{x})| \leq \Lambda \epsilon_{\max}. \quad (5.7)$$

This gives a transparent recipe for choosing the truncation threshold: to reach a bias tolerance δ , it suffices to set $\Lambda \leq \delta/\epsilon_{\max}$, after which the number of paths N is selected to meet the desired variance level via equation 5.6. In contrast to prior methods that fix Λ empirically at a tiny value (e.g., 10^{-4}) to conservatively suppress an uncontrolled residual $\hat{e}_c(t) T(X_t)$, equation 5.7 replaces a conservative safety heuristic by a quantitative bias budget driven by the predictor quality.

5.5.2 Prior Estimation Sources

The prior field $\tilde{T}$ need only be inexpensive and reasonably accurate over the subdomain Ω_s . Several sources are available:

- **Low-resolution FEM:** A coarse finite-element discretization of the entire domain provides a full-field temperature estimate at modest cost. In our experiments, selected coarse discretizations in commercial FEM software (e.g., COMSOL [12]) yield maximum errors below 1 K, which are small compared to typical on-chip temperatures around 350 K and allow Λ to be much larger than in conventional settings.
- **Learning-based surrogates:** Recent neural-network thermal predictors can produce rapid temperature estimates. Even imperfect priors with errors on the order of several kelvin, or coarser rule-of-thumb estimates with larger errors, can still reduce the residual term through equation 5.5.

In this work, we adopt the coarse FEM approach. The WOG method is applied within the source region and WOS in the source-free region; prior predictions are used only in the source region, where the random walk terminates.

5.5.3 Multi-Point Acceleration via Posterior Fusion

When the query set contains multiple locations $\{\mathbf{x}_q\}_{q=1}^M$, conventional methods estimate each temperature independently using only the paths originating from $\mathbf{x}_q$. In the WOG voxelized state space, a cross-point event is recorded when a path enters the voxel or small neighborhood associated with another query location. Under this discrete event definition, Corollary 1 reveals that a path starting at $\mathbf{x}$ and passing through $\mathbf{y}$ at time t provides a noisy linear observation of the

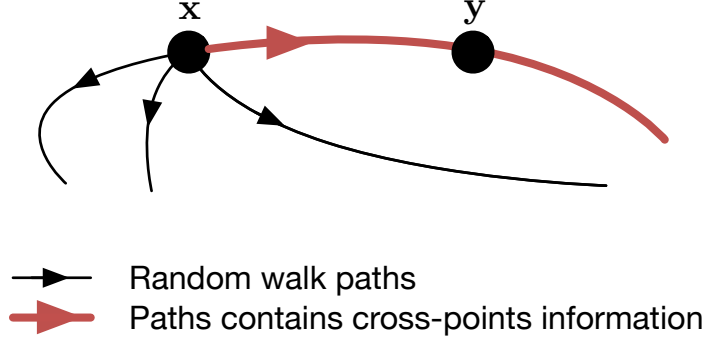


Figure 5.4: Random walk paths in multi-point estimation. Paths that traverse other query locations provide cross-point observations that FastRW exploits via Bayesian fusion.

temperatures at both endpoints:

$$\mathbb{E}[\mathcal{I}_0^t] = T(\mathbf{x}) - \hat{e}_c(t) T(\mathbf{y}). \quad (5.8)$$

Thus, $\mathcal{I}_0^t$ serves as a noisy linear observation of $T(\mathbf{x}) - \alpha T(\mathbf{y})$ with $\alpha = \hat{e}_c(t)$ (Fig. 5.4).

We collect K such cross-point observations from the ensemble of simulated paths:

$$b_k = \mathbf{T}_{i_k} - \alpha_k \mathbf{T}_{j_k} + \eta_k, \quad k = 1, \dots, K, \quad (5.9)$$

where $\mathbf{T} \in \mathbb{R}^M$ with $\mathbf{T}_q \triangleq T(\mathbf{x}_q)$, the indices i_k, j_k identify the starting and visited query points, $\alpha_k = \hat{e}_c(t_k)$, and η_k captures Monte Carlo noise. Stacking yields a linear Gaussian model:

$$\mathbf{B} = \mathbf{A}\mathbf{T} + \boldsymbol{\eta}, \quad \boldsymbol{\eta} \sim \mathcal{N}(\mathbf{0}, \mathbf{R}), \quad (5.10)$$

where $\mathbf{A} \in \mathbb{R}^{K \times M}$ has entry 1 at column i_k and $-\alpha_k$ at column j_k in row k , and the noise covariance is modeled as

$$\mathbf{R} = \text{diag}(\sigma_1^2, \dots, \sigma_K^2), \quad \sigma_k^2 = Z(\mathbf{x}_{i_k}) + \alpha_k^2 Z(\mathbf{x}_{j_k}), \quad (5.11)$$

using the empirical path variances. This diagonal covariance model is an approximation that ignores residual correlations among observations, but it keeps the posterior correction lightweight.

The single-point estimates provide a Gaussian prior:

$$\begin{aligned}\mathbf{T} &\sim \mathcal{N}(\boldsymbol{\mu}_0, \boldsymbol{\Sigma}_0), \\ \mu_{0,q} &= \bar{T}(\mathbf{x}_q), \\ \boldsymbol{\Sigma}_0 &= \text{diag}(Z(\mathbf{x}_1), \dots, Z(\mathbf{x}_M)),\end{aligned}\tag{5.12}$$

and the posterior distribution is

$$\mathbf{T} \mid \mathbf{B} \sim \mathcal{N}(\hat{\mathbf{T}}, \boldsymbol{\Sigma}_{\text{post}}),\tag{5.13}$$

with

$$\boldsymbol{\Sigma}_{\text{post}} = (\mathbf{A}^\top \mathbf{R}^{-1} \mathbf{A} + \boldsymbol{\Sigma}_0^{-1})^{-1},\tag{5.14}$$

$$\hat{\mathbf{T}} = \boldsymbol{\Sigma}_{\text{post}} (\mathbf{A}^\top \mathbf{R}^{-1} \mathbf{B} + \boldsymbol{\Sigma}_0^{-1} \boldsymbol{\mu}_0).\tag{5.15}$$

Under this approximate Gaussian model, the maximum a posteriori (MAP) estimator $\hat{\mathbf{T}}$ corrects the single-point estimates $\boldsymbol{\mu}_0$ using the cross-point observations. By sharing information across query points, **FastRW** reduces the number of paths required per point to achieve the desired precision.

5.6 Experimental Results

This section evaluates **FastRW** on standard 3D steady-state thermal simulation benchmarks [124].

5.6.1 Experimental Setup

Test Cases Following established benchmarks [21, 49, 124], we consider a three-layer 3D chip thermal model (Fig. 5.5): a middle heat-source layer flanked by

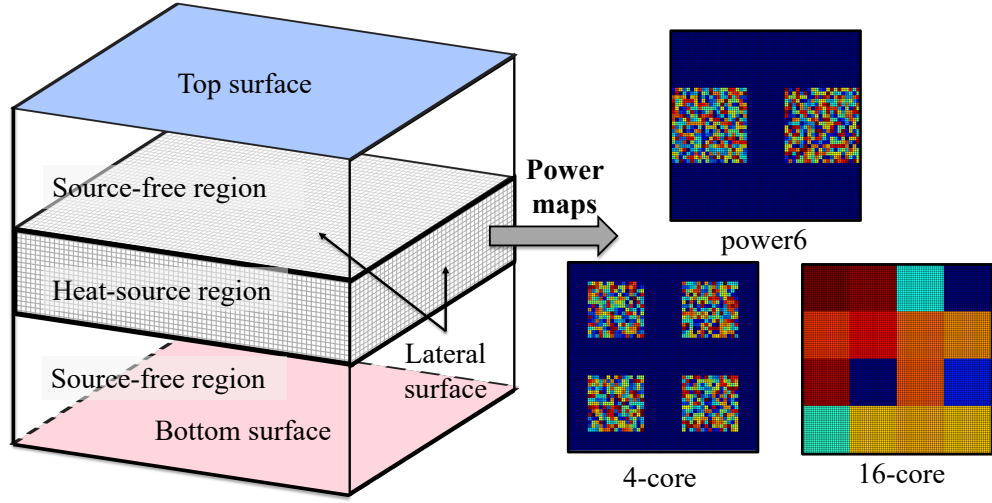


Figure 5.5: Illustration of the three-layer 3D chip thermal model and the three power-map configurations.

source-free top and bottom layers representing the heat transfer medium and substrate, respectively. The chip footprint is $2\text{ cm} \times 2\text{ cm}$. Three thickness configurations are studied:

- Case 1: $(500, 100, 500)\text{ }\mu\text{m}$ (top, middle, bottom).
- Case 2: $(250, 100, 1000)\text{ }\mu\text{m}$.
- Case 3: $(500, 100, 1000)\text{ }\mu\text{m}$.

The heat-source region is discretized at $200\text{ }\mu\text{m} \times 200\text{ }\mu\text{m} \times 20\text{ }\mu\text{m}$ resolution for WOG. Thermal conductivities are $395\text{ W}/(\text{K}\cdot\text{m})$ (source-free) and $125\text{ W}/(\text{K}\cdot\text{m})$ (source region), with an ambient temperature of 293.15 K . Three power maps are used: **4-core** (356 W), **16-core** (352 W), and **power6** (174 W).

Robin boundary conditions are imposed on the top and bottom surfaces with convective coefficient $h = 8700\text{ W}/(\text{K}\cdot\text{m}^2)$ for Cases 1–2 and $h = 4900\text{ W}/(\text{K}\cdot\text{m}^2)$ for Case 3. Lateral surfaces are adiabatic Neumann boundaries.

Implementation Details **FastRW** uses a fixed truncation threshold $\Lambda = 0.03$. For multi-point fusion, at most one cross-point observation is recorded every 1000 random walk steps. The random walk engine is implemented in C++ and the Bayesian posterior correction uses the Stan library [92]. All experiments run on a server with two Intel Xeon Silver 4214 CPUs (24 cores each, 2.20 GHz) and 128 GB RAM. Temperature priors are obtained from a commercial FEM tool (coarse mesh); the finest-mesh FEM solution serves as the ground-truth reference.

Baseline Our primary baseline is the path-integral random walk (PIRW) [124], a Feynman–Kac random-walk solver using standard truncation with threshold $\Lambda = 10^{-4}$ as recommended in [49].

5.6.2 Single-Point and Multi-Point Temperature Estimation

We uniformly select 16 query points within the heat-source region and estimate their temperatures using both PIRW and **FastRW**. TABLE 5.1 reports the average absolute error, empirical standard deviation, average path length (steps), and number of paths.

Both methods achieve accurate predictions with errors well below 0.5 K. The efficiency gain of **FastRW** stems from two complementary effects: (i) the much larger truncation threshold ($\Lambda = 0.03$ vs. 10^{-4}) enables early termination, reducing average steps by 2–3 $\times$ (e.g., 5.79e6 $\rightarrow$ 2.20e6 in Case 1); (ii) posterior fusion reduces the number of required paths from 1000 to 400 while maintaining or improving accuracy. Combined, these yield a stable measured speedup of approximately 6.5 $\times$ across all configurations.

Table 5.1: Comparison of temperature estimation accuracy and efficiency. Error and StdDev are in Kelvin. FastRW achieves comparable accuracy with substantially fewer steps and paths.

Case	Power Map	PIRW [124]				FastRW			
		Error/K	StdDev/K	Steps	Paths	Error/K	StdDev/K	Steps	Speedup
1	power6	0.38	0.47	5.79e6	1000	0.36	0.45	2.20e6	6.58 ×
2	4-core	0.43	0.55	5.83e6	1000	0.42	0.54	2.22e6	6.57 ×
3	16-core	0.38	0.46	1.03e7	1000	0.32	0.41	3.93e6	6.55 ×

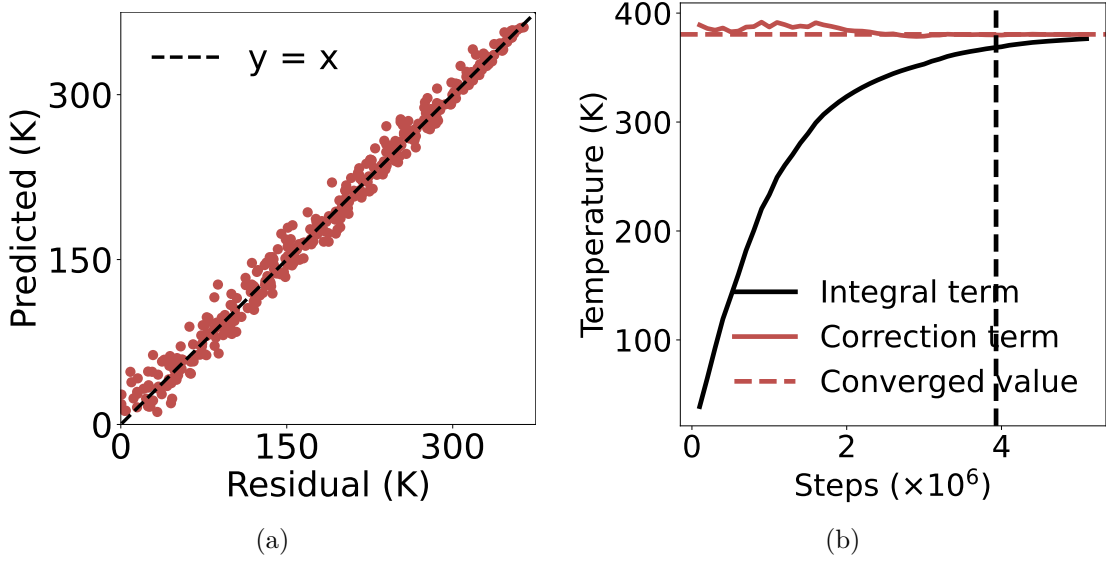


Figure 5.6: (Left) Empirical validation of the truncation residual identity on Case 3: sampled residuals follow the conditional-expectation prediction $\hat{e}_c(t) \cdot T(X_t)$ across a broad range of weights $\hat{e}_c(t) \in [0.03, 0.95]$. (Right) Running accumulated integral $\mathcal{I}_0^t$ (blue) versus the ideal corrected estimate $\mathcal{I}_0^t + \hat{e}_c(t) \cdot T(X_t)$ (orange); the dashed line marks the truncation time.

5.6.3 Further Analysis

Residual Identity Validation To verify Corollary 1 empirically, we sample 400 random-walk trajectories from a query location in Case 3, each terminated at a randomly chosen time such that $\hat{e}_c(t) \in [0.03, 0.95]$. Fig. 5.6 (left) plots the sampled residuals against the conditional-expectation prediction $\hat{e}_c(t) \cdot T(X_t)$; the trend follows the predicted diagonal despite Monte Carlo randomness. Fig. 5.6 (right) illustrates the convergence of the accumulated integral $\mathcal{I}_0^t$ alongside the ideal corrected estimate $\mathcal{I}_0^t + \hat{e}_c(t) \cdot T(X_t)$: the corrected estimate stabilizes far earlier, justifying the early termination strategy when the continuation value is accurately estimated.

Table 5.2: Multi-point estimation results on Case 1. Larger group sizes yield more cross-point observations, further reducing error and variance.

Group Size	Error (K)	StdDev (K)	$\hat{N}$	Speedup
1	0.356	0.446	1000	1.00×
4	0.292	0.379	1385	1.39×
16	0.221	0.281	2519	2.52×

Effect of Group Size on Multi-Point Acceleration We investigate how the number of jointly queried points affects the posterior fusion benefit. For Case 1, the 16 selected points are grouped into sets of size 1, 4, and 16 with $N = 1000$ paths per point. The equivalent number of single-point paths needed to match the fused variance is $\hat{N} = Z(\mathbf{x})/\text{Var}[\hat{T}(\mathbf{x})]$. Results are given in TABLE 5.2.

As the group size increases, more cross-point observations are collected, yielding progressively lower error and variance. At 16 jointly queried points, the Bayesian fusion provides an effective 2.52× additional speedup with negligible computational overhead.

Efficiency Analysis FastRW introduces two additional stages relative to the baseline: prior estimation (FEM) and posterior correction (Bayesian fusion). TABLE 5.3 reports the per-point time breakdown. Even in the most expensive configuration (group size 16), the combined overhead remains below 6% of the random walk time.

Effect of Prior Estimation Accuracy Corollary 1 implies that reducing prior error $|\epsilon|$ permits a larger truncation threshold Λ . TABLE 5.4 examines this accuracy–cost tradeoff by varying the FEM mesh resolution while constraining the expected truncation-correction error to ≤ 0.02 K.

Table 5.3: Per-point time breakdown for **FastRW** on Case 1.

Group Size	Prior (s)	RW (s)	Posterior (s)	Extra Cost (%)
1	0.28	15.41	0.00	1.82
4	0.28	15.41	0.02	1.95
16	0.28	15.41	0.61	5.78

Table 5.4: Trade-off between FEM prior accuracy and random walk efficiency on Case 1. DoF denotes the degrees of freedom of the FEM prior mesh.

Prior		Random Walk		
DoF	Max-Error (K)	Threshold Λ	Steps	Time (s)
1875	1.93	0.01	2.89e6	20.2
4800	0.88	0.02	2.45e6	17.1
12500	0.59	0.03	2.20e6	15.4

A finer FEM prior (12,500 DoF) reduces the maximum error to 0.59 K, enabling $\Lambda = 0.03$ and shortening the average walk time to 15.4 s. For reference, a full FEM solve for Case 1 takes over 450 s per simulation; **FastRW** achieves comparable accuracy in approximately 16 s (including prior and posterior overhead), yielding a roughly $28\times$ speedup.

5.7 Summary

This chapter presented **FastRW**, an efficient random-walk solver for steady-state thermal analysis in heterogeneous IC structures. The central theoretical contribution is a stopping-time truncation residual identity, which establishes that the

expected tail contribution of a truncated Feynman–Kac random walk equals the multiplicative weight times the true temperature at the truncation location. This identity converts the truncation-correction problem into a continuation-value estimation task, enabling principled early termination when an inexpensive prior temperature estimate (e.g., from a coarse FEM solution) is available.

For single-point estimation, compensating the truncation residual with a FEM prior reduces the effective truncation error by over two orders of magnitude, allowing **FastRW** to achieve a roughly $2.6\times$ reduction in path length at comparable accuracy. For multi-point estimation, cross-trajectory observations are aggregated through a Bayesian linear-Gaussian fusion, which provides an additional $2.5\times$ equivalent speedup by sharing information across query points. Combined, these mechanisms yield a measured speedup of over $6.5\times$ on standard 3D chip thermal benchmarks with mixed boundary conditions, at a total overhead below 6%.

Looking ahead, several extensions are worth exploring: integrating transient thermal analysis within the same framework, leveraging GPU parallelism for further acceleration, and replacing the FEM prior with rapid learning-based surrogates to further tighten the bias–variance tradeoff. The truncation residual identity derived here is not specific to thermal problems and applies to any reflected Feynman–Kac Monte Carlo estimator for mixed-boundary elliptic PDEs, suggesting broader applicability in computational physics and engineering design automation.

Together with the layout pattern generation framework of Chapter 3 and the differentiable resist modeling tools of Chapter 4, **FastRW** completes a suite of physics-driven computational methods that collectively span VLSI design, fabrication, and reliability. By exposing structured interfaces to generation, simulation, and localized analysis, these tools open the door to cross-stage co-

optimization across traditionally siloed design tasks, a direction discussed further in Chapter [6](#).

Chapter 6

Conclusion and Future Work

6.1 Summary of Contributions

This dissertation studied how physical structure can be built into computational tools for advanced VLSI physical design closure. The three technical chapters addressed different stages of the design–fabrication–reliability flow: layout pattern generation, photoresist modeling, and steady-state thermal analysis. Their common point is not that learning alone replaces physical models, but that physical knowledge can define the right representation, constraint, or estimator so that each computational component solves a narrower and better-posed problem.

Layout Pattern Generation (Chapter 3). Chapter 3 addressed the need for large, diverse, and design-rule-clean layout pattern libraries. The main contribution is a decoupled generation pipeline. A discrete diffusion model, implemented through DiffPattern-Flex, samples topology tensors in the Deep Squish Pattern representation, while a white-box nonlinear legalization solver assigns physical dimensions under explicit design-rule constraints. This separation is important: the neural model is responsible for topological diversity, and the solver is re-

Table 6.1: Cross-chapter contribution matrix.

Chapter	Contribution	Physical role	Evaluation	Main empirical claim	Publication	Limitation
Ch. 3	DiffPattern-Flex and ChatPattern	Hard design-rule constraints plus language-guided tool orchestration	Layer-10001/10003 benchmarks; CAE, LegalGAN, LayoutTransformer, DiffPattern, and concatenation baselines	100% legality among accepted legalized outputs, diversity score 11.713, and stronger free-size extension than naive concatenation	DAC 2024; TCAD 2025	Legality is scoped to accepted outputs; large free-size Layer-10001 cases remain difficult
Ch. 4	TorchResist and DiffResist	Differentiable Dill-Mack simulation plus hard air-resist boundary injection	LithoBench and simulator-supervised 3D profiles; threshold, sequential, and direct 3D diffusion baselines	0.85% PD, 2.56 nm EPE-mean, and about 15 $\times$ speedup over direct 3D diffusion	SPIE 2025; DATE 2026	Independent 3D experimental validation and cross-chemistry generalization remain limited
Ch. 5	FastRW	Feynman-Kac truncation residual identity with prior-corrected random walks	Standard 3D IC thermal benchmarks; PIRW baseline	More than 6.5 $\times$ measured speedup with average errors below 0.5 K	DATE 2026	Focuses on steady-state point queries and depends on prior quality

sponsible for manufacturability. On the Layer-10001 benchmark, the accepted legalized outputs achieve 100% DRC legality among 100,000 generated patterns, with a diversity score of 11.713. Fast discrete sampling gives an $8.37\times$ sampling speedup at the selected operating point, and block-partition initialization gives a $2.48\times$ legalization speedup.

The chapter further introduced **ChatPattern**, which adds an agentic interface around the pattern generator. Instead of requiring the user to invoke generation, in-painting, out-painting, and legalization routines manually, the LLM agent converts natural-language requirements into structured tool calls. This extends fixed-size pattern generation toward free-size and interactive customization. The strongest result is that overlap-aware in-painting and out-painting substantially outperform naive concatenation when pattern size increases, although the results also show that large-scale free-size generation remains challenging on the more difficult Layer-10001 setting.

Physics-Constrained Photoresist Modeling (Chapter 4). Chapter 4 developed a two-tier resist modeling framework. At the simulator level, **TorchResist** rewrites the classical Dill exposure model and Mack development model as differentiable tensor operations. With fewer than twenty interpretable parameters, it supports gradient-based calibration and GPU-accelerated inference. On LithoBench, **TorchResist** improves the pixel-difference error over threshold-based baselines while retaining a physically interpretable parameterization and supporting 3D depth simulation.

At the learning level, **DiffResist** targets the high cost of generating 3D resist profiles. It reformulates depth-wise inhibitor concentration prediction as a compact 2D diffusion process and injects the air-resist boundary condition derived from the exposure model into the reverse process. This boundary anchoring di-

rectly addresses the error accumulation observed in generic sequential predictors. With a two-stage noise schedule and a residual super-resolution module, DiffResist reaches 0.85% PD and 2.56 nm EPE-mean on LithoBench, while reducing inference time relative to direct 3D diffusion baselines.

Efficient Steady-State Thermal Analysis (Chapter 5). Chapter 5 addressed localized steady-state thermal queries in 3D IC structures. The central result is the stopping-time truncation residual identity for reflected Feynman–Kac random walks. This identity shows that the expected tail after truncation can be written as the remaining multiplicative weight times the continuation value at the truncation state. As a result, the long-path problem is converted into a continuation-value estimation problem.

FastRW uses this identity to terminate random-walk trajectories early and correct the residual with a coarse prior temperature estimate. In the multi-point setting, cross-trajectory observations are combined through a Bayesian linear-Gaussian fusion step. On standard 3D chip thermal benchmarks, this combination reduces path length, reduces the number of required paths, and achieves more than $6.5\times$ measured speedup over a strong Feynman–Kac random-walk baseline while keeping average errors below 0.5 K.

6.2 Lessons Learned

Put hard constraints where they are most reliable. The layout generation results show that manufacturability constraints should not always be learned implicitly. DRC rules are exact, local, and technology specific; treating them as a post-hoc soft loss leaves room for illegal outputs. Separating topology generation from geometric legalization lets the model explore diverse structures while the deterministic solver enforces the rules on the final accepted layouts. This division

of responsibility is a recurring pattern: learning is used where the space is complex and hard to enumerate, while physics or rules are enforced where correctness is binary.

Use physical equations as interfaces, not just as labels. The resist modeling chapter shows that classical equations remain useful even when a learning model is introduced. In `TorchResist`, the Dill–Mack equations become a differentiable simulator with compact, interpretable parameters. In `DiffResist`, the same physics supplies a boundary condition that anchors a generative process. In both cases, the physical model does more than generate labels: it defines the computational interface through which the learning component is constrained.

Exploit identities before adding computation. The thermal chapter illustrates a different use of physics. Rather than training a larger model or simulating longer paths, `FastRW` starts from the structure of the Feynman–Kac representation and derives an identity for the truncation residual. This changes the computational problem itself: instead of waiting for the random-walk weight to decay to a tiny threshold, the method estimates the continuation value at an earlier stopping point. The lesson is that physical and probabilistic structure can sometimes remove computation rather than merely accelerate it.

Taken together, these lessons refine the meaning of physics-driven learning in this dissertation. The goal is not to force every component into a single differentiable graph. The goal is to choose the right role for physics in each subsystem: a hard constraint for layout legality, a differentiable operator and boundary condition for resist modeling, and a stochastic identity for thermal estimation.

6.3 Limitations

Several limitations remain and should be treated as part of the interpretation of the results.

- **Layout generation.** The 100% DRC result applies to the final accepted legalized outputs, not a claim that every raw topology sampled by the neural model is feasible. Infeasible topologies can still be discarded by the legalization stage. Free-size generation also becomes harder as the target size grows: on the challenging Layer-10001 setting, legality drops substantially at 512^2 and 1024^2 , even though **ChatPattern** remains better than naive concatenation. Finally, diversity entropy is only a proxy for pattern variety and does not fully measure industrial realism or downstream utility.
- **LLM-guided customization.** **ChatPattern** demonstrates that an LLM agent can orchestrate generation tools, but the evaluation is closer to a set of representative case studies than to a complete benchmark of natural-language layout design. A systematic task suite, tool-call success rate, recovery rate after legalization failure, and amount of human intervention would be needed to quantify robustness.
- **Photoresist modeling.** **DiffResist** uses high-resolution **TorchResist** outputs as 3D supervision. This is useful for controlled benchmarking, but it is not a substitute for independent experimental 3D resist measurements. Generalization across resist chemistries, process corners, stochastic EUV effects, post-exposure bake dynamics, and etch selectivity remains insufficiently explored.
- **Thermal analysis.** **FastRW** relies on an approximate prior, such as a coarse FEM solution, to estimate the continuation value. The correction is

therefore bias-aware rather than exactly unbiased unless the continuation value is exact. The present experiments focus on steady-state point queries on standard 3D chip benchmarks; transient thermal analysis, full-field reconstruction, and broader package geometries remain open.

- **Cross-chapter integration.** The dissertation demonstrates a common methodology across three physical modeling problems, but it does not yet implement a closed-loop design–fabrication–reliability system in which pattern generation, lithography/resist prediction, and thermal estimation are optimized together.

6.4 Future Research Directions

Physical-feedback generation. A natural next step is to connect layout generation with downstream physical evaluators. `DiffPattern` and `ChatPattern` now emphasize topology diversity and DRC legality. Future systems could also score generated patterns with printability metrics from `TorchResist` or `DiffResist`, together with reliability metrics from thermal analysis. Pattern generation would then become a closed-loop design exploration engine rather than only a way to build pattern libraries. The main challenge is algorithmic: DRC legalization is discrete and partly combinatorial, while printability and thermal objectives are continuous and may be expensive to evaluate. Practical solutions will likely require hybrid optimization, active sampling, and surrogate search rather than a naive gradient through every module.

Stronger validation for resist learning. For resist modeling, the most important future direction is validation beyond simulator-generated supervision. Independent 3D measurements, multiple resist chemistries, process-corner sweeps,

and EUV stochastic effects would provide a clearer picture of where physics-constrained diffusion generalizes and where it fails. Methodologically, additional physical priors could be incorporated into the model, including PEB diffusion, material-dependent reaction rates, etch selectivity, and uncertainty estimates over the predicted profile.

Prior-aware and accelerated random-walk solvers. For FastRW, future work should characterize the effect of prior error more sharply. The truncation identity gives a clean target, but the practical estimator depends on how well the continuation value is approximated. Error bounds that explicitly account for the prior, adaptive threshold selection, and learned continuation-value predictors are promising directions. A GPU implementation of the random-walk engine and extensions to transient thermal analysis or other parabolic/elliptic PDEs would further broaden the method’s applicability.

Human-in-the-loop EDA agents with formal checks. The agentic component of ChatPattern suggests a broader direction for EDA tools that combine natural-language interaction with formally checked backend operations. Future work should move beyond qualitative examples toward benchmarks that measure instruction understanding, tool-call correctness, failure recovery, DRC-clean rate, printability-aware planning, and user effort. Such agents should be evaluated not by fluent explanations, but by whether they produce verifiably correct design artifacts under explicit manufacturing constraints.

6.5 Closing Remarks

The broader message of this dissertation is that physical design closure in VLSI should not be divided into a choice between slow physical simulation and uncon-

strained data-driven prediction. The more useful design principle is to identify which part of the problem should be governed by physical laws, which part should be learned from data, and which interface should connect the two. In layout generation, this interface is the topology–geometry split and the legalization solver. In resist modeling, it is the differentiable Dill–Mack pipeline and the boundary-constrained diffusion process. In thermal analysis, it is the Feynman–Kac truncation identity and the prior-corrected random-walk estimator.

These examples show that physics-driven learning is most effective when physical knowledge is not appended as an afterthought, but placed at the computational stage where it can most directly enforce correctness, reduce uncertainty, or remove unnecessary computation. This principle provides a practical path toward EDA tools that are faster, more interpretable, and more closely aligned with the physics of semiconductor manufacturing.

Appendix A

Proofs for Chapter 5

This appendix provides the deferred proofs for the theoretical results in Chapter 5. Section A.1 proves the stopping-time truncation residual identity (Theorem 1), Section A.2 quantifies the bias and MSE incurred when the ideal continuation value is replaced by a plug-in predictor, and Section A.3 derives the closed-form Bayesian posterior used in multi-point fusion.

A.1 Proof of the Truncation Residual Identity

We restate Theorem 1 for convenience and then give the proof. The notation follows Section 2.3 and Section 5.3: $\{X_t\}_{t \geq 0}$ is the reflected Brownian motion in Ω , $\{\mathcal{F}_t\}_{t \geq 0}$ its natural filtration, $L(t)$ the boundary local time, and $\hat{e}_c(t)$ the Feynman–Kac multiplicative weight defined in equation 2.24.

Proof of Theorem 1. Let τ be a stopping time with $\tau < t_D$ on the event of interest. Define the post- τ shifted process

$$\tilde{X}_s \triangleq X_{\tau+s}, \quad s \geq 0,$$

and let $\tilde{L}(s)$ denote the boundary local time accumulated by $\tilde{X}$ after time τ , so

that $L(\tau + s) = L(\tau) + \tilde{L}(s)$. Introduce the shifted Feynman–Kac weight

$$\tilde{e}_c(s) \triangleq \exp\left(-\int_0^s c(\tilde{X}_u) d\tilde{L}(u)\right). \quad (\text{A.1})$$

By the additivity of the exponent in the local-time integral, the original weight factorises as

$$\hat{e}_c(\tau + s) = \hat{e}_c(\tau) \tilde{e}_c(s). \quad (\text{A.2})$$

Now define $\tilde{\mathcal{I}}_0^\infty$ as the full Feynman–Kac functional (i.e., the right-hand side of equation 2.23 without the outer expectation) evaluated along the shifted path $\tilde{X}$ starting from $\tilde{X}_0 = X_\tau$, with weight $\tilde{e}_c(\cdot)$. By equation A.2, the truncation tail along the original path factors as

$$\mathcal{I}_\tau^\infty = \hat{e}_c(\tau) \tilde{\mathcal{I}}_0^\infty. \quad (\text{A.3})$$

It remains to evaluate the conditional expectation. Since τ is a stopping time for the reflected process, the *strong Markov property* implies that, given $\mathcal{F}_\tau$, the shifted process $(\tilde{X}_s, \tilde{L}(s))_{s \geq 0}$ has the same law as a fresh reflected Brownian motion started from X_τ , and is independent of $\mathcal{F}_\tau$ beyond its dependence through the initial state X_τ . In particular, $\hat{e}_c(\tau)$ is $\mathcal{F}_\tau$ -measurable. Applying the definition of the temperature field via the Feynman–Kac formula equation 2.23 to the shifted process yields

$$\mathbb{E}\left[\tilde{\mathcal{I}}_0^\infty \mid \mathcal{F}_\tau\right] = T(X_\tau). \quad (\text{A.4})$$

Combining equation A.3 and equation A.4, and using the $\mathcal{F}_\tau$ -measurability of $\hat{e}_c(\tau)$, we obtain

$$\mathbb{E}[\mathcal{I}_\tau^\infty \mid \mathcal{F}_\tau] = \hat{e}_c(\tau) \mathbb{E}\left[\tilde{\mathcal{I}}_0^\infty \mid \mathcal{F}_\tau\right] = \hat{e}_c(\tau) T(X_\tau),$$

which is the identity in equation 5.2. □

Remark A.1.1. Although stated in continuous time, the proof relies only on the strong Markov property and the multiplicative structure of the weight. Both

properties hold verbatim for the discrete-time Markov chain estimators used in practice (WOG/WOS), with τ a stopping time for the discrete filtration and $\hat{e}_c$ the corresponding product of per-step weights.

A.2 Bias and MSE of the Plug-in Truncation Correction

This section quantifies the bias and mean-squared error (MSE) introduced when the unknown continuation value $T(X_\tau)$ in the ideal truncation correction is replaced by a prior estimate $\tilde{T}(X_\tau) = T(X_\tau) + \epsilon(X_\tau)$. The key takeaway is that the truncation rule $\hat{e}_c(\tau) \leq \Lambda$ directly controls the bias through the predictor error, yielding the threshold-selection criterion stated in Eq. (5.7).

A.2.1 Single-Path Plug-in Correction

Let τ be the stopping time at which FastRW applies the plug-in correction (e.g., the threshold rule $\tau = t_\Lambda$ from Section 5.4.1), and suppose $\tilde{T}$ is defined over a subdomain Ω_s containing X_τ . The FastRW single-path estimator in Eq. (5.5) is

$$\hat{T}_i(\mathbf{x}) \triangleq \mathcal{I}_0^\tau + \hat{e}_c(\tau) \tilde{T}(X_\tau). \quad (\text{A.5})$$

For comparison, define the ideal (infeasible) truncation-corrected estimator that uses the exact continuation value,

$$\hat{T}_i^*(\mathbf{x}) \triangleq \mathcal{I}_0^\tau + \hat{e}_c(\tau) T(X_\tau), \quad (\text{A.6})$$

which is unbiased for $T(\mathbf{x})$ by Theorem 1.

Theorem 2 (Bias induced by plug-in correction). With $\hat{T}_i(\mathbf{x})$ and $\hat{T}_i^*(\mathbf{x})$ as in Eqs. (A.5) and (A.6),

$$\hat{T}_i(\mathbf{x}) - \hat{T}_i^*(\mathbf{x}) = \hat{e}_c(\tau) \epsilon(X_\tau), \quad (\text{A.7})$$

so the bias of the plug-in estimator is

$$\mathbb{E}[\hat{T}_i(\mathbf{x})] - T(\mathbf{x}) = \mathbb{E}[\hat{e}_c(\tau) \epsilon(X_\tau)]. \quad (\text{A.8})$$

Proof. Subtract Eq. (A.6) from Eq. (A.5) to obtain Eq. (A.7). Taking expectations and using the unbiasedness of $\hat{T}_i^*(\mathbf{x})$ yields Eq. (A.8). $\square$

Corollary 2 (Uniform bound under threshold truncation). Suppose the truncation rule enforces $\hat{e}_c(\tau) \leq \Lambda$ almost surely and the predictor error satisfies $\sup_{\mathbf{x} \in \Omega_s} |\epsilon(\mathbf{x})| \leq \epsilon_{\max}$. Then

$$\left| \mathbb{E}[\hat{T}_i(\mathbf{x})] - T(\mathbf{x}) \right| \leq \Lambda \epsilon_{\max}. \quad (\text{A.9})$$

Equivalently, to meet an absolute bias tolerance δ , it suffices to set $\Lambda \leq \delta/\epsilon_{\max}$.

Proof. Take absolute values in Eq. (A.8) and apply Jensen's inequality together with the two hypotheses: $|\hat{e}_c(\tau) \epsilon(X_\tau)| \leq \Lambda \epsilon_{\max}$ almost surely, hence the same bound holds for its expectation. $\square$

A.2.2 MSE Decomposition and Averaging over Paths

Writing $\Delta(\mathbf{x}) \triangleq \hat{T}_i^*(\mathbf{x}) - T(\mathbf{x})$ for the intrinsic Monte Carlo fluctuation of the ideal estimator, Eq. (A.7) gives the additive decomposition $\hat{T}_i(\mathbf{x}) - T(\mathbf{x}) = \Delta(\mathbf{x}) + \hat{e}_c(\tau) \epsilon(X_\tau)$. Squaring and taking expectations yields

$$\begin{aligned} \mathbb{E}[(\hat{T}_i(\mathbf{x}) - T(\mathbf{x}))^2] &= \mathbb{E}[\Delta(\mathbf{x})^2] + 2 \mathbb{E}[\Delta(\mathbf{x}) \hat{e}_c(\tau) \epsilon(X_\tau)] \\ &\quad + \mathbb{E}[\hat{e}_c(\tau)^2 \epsilon(X_\tau)^2]. \end{aligned} \quad (\text{A.10})$$

Applying Cauchy–Schwarz to the cross term and collecting the resulting square root gives the convenient bound

$$\mathbb{E}[(\hat{T}_i(\mathbf{x}) - T(\mathbf{x}))^2] \leq \left(\sqrt{\mathbb{E}[\Delta(\mathbf{x})^2]} + \sqrt{\mathbb{E}[\hat{e}_c(\tau)^2 \epsilon(X_\tau)^2]} \right)^2. \quad (\text{A.11})$$

Averaging N independent paths as in Eq. (5.6) leaves the bias unchanged and reduces the variance of $\bar{T}(\mathbf{x})$ by a factor $1/N$, motivating the two-stage recipe used in Section 5.5.1: pick Λ to enforce the bias budget via Corollary 2, then scale N to meet the desired variance tolerance.

A.3 Derivation of the Bayesian Posterior Fusion

This section derives the closed-form posterior expressions in Eqs. (5.14) and (5.15). The setup follows Section 5.5.3: let $\mathbf{T} \in \mathbb{R}^M$ collect the unknown temperatures at M query points, and consider the linear Gaussian observation model

$$\mathbf{B} \mid \mathbf{T} \sim \mathcal{N}(\mathbf{A}\mathbf{T}, \mathbf{R}), \quad (\text{A.12})$$

where $\mathbf{A} \in \mathbb{R}^{K \times M}$ encodes the cross-point observations and $\mathbf{R} = \text{diag}(\sigma_1^2, \dots, \sigma_K^2)$ is the heteroscedastic noise covariance. The prior is

$$\mathbf{T} \sim \mathcal{N}(\boldsymbol{\mu}_0, \boldsymbol{\Sigma}_0), \quad (\text{A.13})$$

with $\boldsymbol{\mu}_0$ and $\boldsymbol{\Sigma}_0$ constructed from the single-point Monte Carlo estimates as in Eq. (5.12).

Derivation. By Bayes' rule, the log posterior (up to an additive constant independent of $\mathbf{T}$) is

$$\begin{aligned} \log p(\mathbf{T} \mid \mathbf{B}) &= -\frac{1}{2}(\mathbf{B} - \mathbf{A}\mathbf{T})^\top \mathbf{R}^{-1}(\mathbf{B} - \mathbf{A}\mathbf{T}) \\ &\quad - \frac{1}{2}(\mathbf{T} - \boldsymbol{\mu}_0)^\top \boldsymbol{\Sigma}_0^{-1}(\mathbf{T} - \boldsymbol{\mu}_0) + \text{const.} \end{aligned} \quad (\text{A.14})$$

Expanding the two quadratic forms in $\mathbf{T}$ and collecting terms gives

$$\log p(\mathbf{T} \mid \mathbf{B}) = -\frac{1}{2} \mathbf{T}^\top \underbrace{(\mathbf{A}^\top \mathbf{R}^{-1} \mathbf{A} + \boldsymbol{\Sigma}_0^{-1})}_{\boldsymbol{\Sigma}_{\text{post}}^{-1}} \mathbf{T}$$

$$+ \mathbf{T}^\top (\mathbf{A}^\top \mathbf{R}^{-1} \mathbf{B} + \Sigma_0^{-1} \boldsymbol{\mu}_0) + \text{const.} \quad (\text{A.15})$$

This is a quadratic form in $\mathbf{T}$, so the posterior is Gaussian. Completing the square identifies the posterior covariance (precision inverse) as

$$\Sigma_{\text{post}} = (\mathbf{A}^\top \mathbf{R}^{-1} \mathbf{A} + \Sigma_0^{-1})^{-1}, \quad (\text{A.16})$$

and the posterior mean (equivalently, the MAP estimator, since the posterior is Gaussian) as

$$\hat{\mathbf{T}} = \Sigma_{\text{post}} (\mathbf{A}^\top \mathbf{R}^{-1} \mathbf{B} + \Sigma_0^{-1} \boldsymbol{\mu}_0), \quad (\text{A.17})$$

which are exactly Eq. (5.14) and Eq. (5.15). $\square$

Remark A.3.1. Because $\mathbf{R}$ and Σ_0 are both diagonal, the posterior precision matrix $\Sigma_{\text{post}}^{-1} = \mathbf{A}^\top \mathbf{R}^{-1} \mathbf{A} + \Sigma_0^{-1}$ is sparse when the number of cross-point observations K is moderate relative to M , making the matrix inversion inexpensive in practice.

Bibliography

- [1] Jacob Austin, Daniel D Johnson, Jonathan Ho, Daniel Tarlow, and Rianne van den Berg. Structured denoising diffusion models in discrete state-spaces. *Proc. NeurIPS*, 34:17981–17993, 2021. [13](#), [31](#)
- [2] Shayak Banerjee, Zhuo Li, and Sani R Nassif. Iccad-2013 cad contest in mask optimization and benchmark suite. In *2013 IEEE/ACM International Conference on Computer-Aided Design (ICCAD)*, pages 271–274. IEEE, 2013. [70](#), [72](#), [88](#), [89](#), [90](#)
- [3] Atilim Gunes Baydin, Barak A. Pearlmutter, Alexey Andreyevich Radul, and Jeffrey Mark Siskind. Automatic differentiation in machine learning: A survey. *Journal of Machine Learning Research*, 18(153):1–43, 2018. [4](#)
- [4] Robert L. Brainard, Peter Trefonas, Jeroen H. Lammers, Charlotte A. Cutler, Joseph F. Mackevich, Alexander Trefonas, and Stewart A. Robertson. Shot noise, LER, and quantum efficiency of EUV photoresists. In *Proceedings of SPIE*, volume 5374, pages 74–85, 2004. [3](#), [5](#)
- [5] Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D. Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler,

- Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. Language models are few-shot learners. In *Advances in Neural Information Processing Systems*, volume 33, pages 1877–1901, 2020. [15](#)
- [6] Tom Cecil, Kyle Braam, Ahmed S. Omran, Ayn Poonawala, Jason Shu, and Clark Vandam. Establishing fast, practical, full-chip ilt flows using machine learning. In *Optical Microlithography XXXIII*, volume 11327 of *Proceedings of SPIE*, page 4, 2020. [17](#)
- [7] Guojin Chen, Hao Geng, Bei Yu, and David Z Pan. Open-source differentiable lithography imaging framework. In *DTCO and Computational Patterning III*, volume 12954, pages 118–127. SPIE, 2024. [4](#), [17](#), [73](#)
- [8] Guojin Chen, Zixiao Wang, Bei Yu, David Z Pan, and Martin DF Wong. Ultrafast source mask optimization via conditional discrete diffusion. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 43(7):2140–2150, 2024. [26](#), [73](#)
- [9] Ran Chen, Wei Zhong, Haoyu Yang, Hao Geng, Xuan Zeng, and Bei Yu. Faster region-based hotspot detection. In *ACM/IEEE Design Automation Conference (DAC)*, pages 1–6, 2019. [2](#), [26](#)
- [10] Shoufa Chen, Mengmeng Xu, Jiawei Ren, Yuren Cong, Sen He, Yanping Xie, Animesh Sinha, Ping Luo, Tao Xiang, and Juan-Manuel Perez-Rua. GenTron: Diffusion transformers for image and video generation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6441–6451, 2024. [74](#)

- [11] Nicolas B. Cobb, Avidesh Zakhori, and Eugene A. Miloslavsky. Mathematical and cad framework for proximity correction. In *Optical Microlithography IX*, volume 2726, pages 208–222. SPIE, 1996. [4](#), [16](#)
- [12] COMSOL AB, Stockholm, Sweden. *COMSOL Multiphysics Reference Manual*, 2023. Version 6.2. [114](#)
- [13] Q Dai, R Guo, S-Y Lee, J Choi, S-H Lee, I-K Shin, C-U Jeon, B-G Kim, and H-K Cho. A fast path-based method for 3-d resist development simulation. *Microelectronic engineering*, 127:86–96, 2014. [20](#)
- [14] Peter De Bisschop. Stochastic printing failures in extreme ultraviolet lithography. *Journal of Micro/Nanolithography, MEMS, and MOEMS*, 17(4): 041011, 2018. [3](#), [5](#)
- [15] DeepSeek-AI, Aixin Liu, Bei Feng, Bing Xue, Bingxuan Wang, Bochao Wu, Chengda Lu, Chenggang Zhao, Chengqi Deng, Chenyu Zhang, Chong Ruan, et al. Deepseek-v3 technical report. *arXiv preprint arXiv:2412.19437*, 2024. [15](#)
- [16] Emily Denton and Rob Fergus. Stochastic video generation with a learned prior. In *ICML*, volume 80 of *Proceedings of Machine Learning Research*, pages 1174–1183. PMLR, 2018. [70](#), [74](#), [92](#)
- [17] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 4171–4186, 2019. [15](#)
- [18] Frederick H Dill. Optical lithography. *IEEE Transactions on Electron Devices*, 22(7):440–444, 1975. [16](#)

- [19] Frederick H Dill, William P Hornberger, Peter S Hauge, and Jane M Shaw. Characterization of positive photoresist. *IEEE Transactions on electron devices*, 22(7):445–452, 1975. [18](#), [70](#), [72](#), [75](#)
- [20] Carl Doersch. Tutorial on variational autoencoders. *arXiv preprint arXiv:1606.05908*, 2016. [21](#)
- [21] Zhixuan Dong, Longlong Yang, Cuiyang Ding, Changhao Yan, Zhaori Bi, Sheng-Guo Wang, Dian Zhou, and Xuan Zeng. ppirw: An efficient and accurate precalculation path integral random walk solver for steady-state thermal simulation with robin boundary conditions. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 44(4):1489–1502, 2025. [23](#), [103](#), [106](#), [116](#)
- [22] Richard Phillips Feynman. Space-time approach to non-relativistic quantum mechanics. *Reviews of modern physics*, 20(2):367–387, 1948. [23](#)
- [23] Tim Fühner. *Artificial evolution for the optimization of lithographic process conditions*. PhD thesis, Friedrich-Alexander-Universität Erlangen-Nürnberg (FAU), 2014. [18](#), [73](#)
- [24] Jhih-Rong Gao, Xiaoqing Xu, Bei Yu, and David Z Pan. MOSAIC: Mask optimizing solution with process window aware inverse correction. In *ACM/IEEE Design Automation Conference (DAC)*, pages 1–6. IEEE, 2014. [2](#)
- [25] Zhangyang Gao, Cheng Tan, Lirong Wu, and Stan Z Li. Simvp: Simpler yet better video prediction. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 3170–3180, 2022. [70](#), [74](#)
- [26] Hao Geng, Haoyu Yang, Lu Zhang, Fan Yang, Xuan Zeng, and Bei Yu. Hotspot detection via attention-based deep layout metric learning. *IEEE*

- Transactions on Computer-Aided Design of Integrated Circuits and Systems (TCAD)*, 41(8):2685–2698, 2022. [2](#), [26](#)
- [27] Frank E Gennari and Ya-Chieh Lai. Topology design using squish patterns. September 9 2014. US Patent 8,832,621. [2](#), [11](#), [27](#), [31](#), [34](#)
- [28] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial networks. *Communications of the ACM*, 63(11):139–144, 2020. [27](#), [30](#)
- [29] Yuri Granik, Nicolas B Cobb, and Thuy Do. Universal process modeling with vtre for opc. In *Optical Microlithography XV*, volume 4691, pages 377–394. SPIE, 2002. [18](#), [70](#), [72](#)
- [30] Zhuolun He, Yuzhe Ma, and Bei Yu. X-check: Gpu-accelerated design rule checking via parallel sweepline algorithms. In *Proc. ICCAD*, pages 1–9, 2022. [2](#), [11](#), [26](#)
- [31] Zhuolun He, Haoyuan Wu, Xinyun Zhang, Xufeng Yao, Su Zheng, Haisheng Zheng, and Bei Yu. Chateda: A large language model powered autonomous agent for eda. *arXiv preprint arXiv:2308.10204*, 2023. [15](#), [32](#)
- [32] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. *Proc. NeurIPS*, 33:6840–6851, 2020. [13](#), [20](#), [27](#), [31](#), [62](#), [74](#), [82](#), [83](#), [92](#)
- [33] Li Hu. Animate anyone: Consistent and controllable image-to-video synthesis for character animation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8153–8163, 2024. [71](#), [74](#)

- [34] Wei Huang, S. Ghosh, S. Velusamy, K. Sankaranarayanan, K. Skadron, and M.R. Stan. Hotspot: a compact thermal modeling methodology for early-stage vlsi design. *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, 14(5):501–513, 2006. [3](#), [21](#), [105](#)
- [35] IEEE. International roadmap for devices and systems (IRDS), 2022 edition. <https://irds.ieee.org/editions/2022>, 2022. [1](#)
- [36] R.B. Iverson, Y.L. Le Coz, B. Kleveland, and S.S. Wong. A multi-scale random-walk thermal-analysis methodology for complex ic-interconnect systems. In *2000 International Conference on Simulation Semiconductor Processes and Devices (Cat. No.00TH8502)*, pages 84–86, 2000. [24](#)
- [37] Bentian Jiang, Lixin Liu, Yuzhe Ma, Bei Yu, and Evangeline F. Y. Young. Neural-ILT 2.0: Migrating ilt to domain-specific and multitask-enabled neural network. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems (TCAD)*, 41(8):2671–2684, 2022. [4](#)
- [38] Jiayi Jiang, Lancheng Zou, Wenqian Zhao, Zhuolun He, Tinghuan Chen, and Bei Yu. Pdrc: Package design rule checking via gpu-accelerated geometric intersection algorithms for non-manhattan geometry. In *ACM/IEEE Design Automation Conference (DAC)*, pages 1–6, 2024. [2](#), [11](#), [26](#)
- [39] Mark Kac. On distributions of certain wiener functionals. *Transactions of the American Mathematical Society*, 65(1):1–13, 1949. [23](#)
- [40] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014. [78](#), [88](#), [92](#)
- [41] Takahiro Kozawa, Akinori Saeki, and Seiichi Tagawa. Modeling and simulation of chemically amplified electron beam, x-ray, and EUV resist processes. *Journal of Vacuum Science & Technology B*, 22(6):3489–3492, 2004. [3](#), [19](#)

- [42] Takahiro Kozawa, Seiichi Tagawa, Hoang B. Cao, Hua Deng, and Michael J. Leeson. Acid distribution in chemically amplified extreme ultraviolet resist. *Journal of Vacuum Science & Technology B*, 25(6):2481–2485, 2007. [3](#), [19](#)
- [43] Jian Kuang and Evangeline FY Young. An efficient layout decomposition approach for triple patterning lithography. In *ACM/IEEE Design Automation Conference (DAC)*, pages 1–6. IEEE, 2013. [2](#), [26](#)
- [44] Scott Ladenheim, Yi-Chung Chen, Milan Mihajlović, and Vasilis Pavlidis. Ic thermal analyzer for versatile 3-d structures using multigrid preconditioned krylov methods. In *2016 IEEE/ACM International Conference on Computer-Aided Design (ICCAD)*, pages 1–8, 2016. [3](#), [104](#)
- [45] Scott Ladenheim, Yi-Chung Chen, Milan Mihajlović, and Vasilis F. Pavlidis. The mta: An advanced and versatile thermal simulator for integrated systems. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 37(12):3123–3136, 2018. [3](#), [21](#), [104](#)
- [46] Peng Li, Lawrence T. Pileggi, Mehdi Asheghi, and Rajit Chandra. IC thermal simulation and modeling via efficient multigrid-based approaches. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems (TCAD)*, 25(9):1763–1776, 2006. [3](#)
- [47] Yuan Liang, Wenjian Yu, and Haifeng Qian. A hybrid random walk algorithm for 3-d thermal analysis of integrated circuits. In *2014 19th Asia and South Pacific Design Automation Conference (ASP-DAC)*, pages 849–854, 2014. [25](#), [103](#), [105](#), [108](#)
- [48] Mingjie Liu, Teodor-Dumitru Ene, Robert Kirby, Chris Cheng, Nathaniel Pinckney, Rongjian Liang, Jonah Alben, Himyanshu Anand, Sanmitra

- Banerjee, Ismet Bayraktaroglu, et al. ChipNeMo: Domain-adapted LLMs for chip design. *arXiv preprint arXiv:2311.00176*, 2023. [15](#)
- [49] Yonghan Luo, Yuyan Wang, Zhixuan Dong, Changhao Yan, Zhaori Bi, Keren Zhu, Sheng-Guo Wang, and Xuan Zeng. Gpu acceleration of a high-precision stochastic solver for steady-state thermal analysis with robin boundary conditions. In *2025 International Symposium of Electronics Design Automation (ISED)*, pages 287–292. IEEE, 2025. [25](#), [103](#), [106](#), [116](#), [118](#)
- [50] Yenai Ma, Leila Delshadtehrani, Cansu Demirkiran, José L. Abellán, and Ajay Joshi. Tap-2.5d: A thermally-aware chiplet placement methodology for 2.5d systems. In *2021 Design, Automation & Test in Europe Conference & Exhibition (DATE)*, pages 1246–1251, 2021. [5](#), [102](#)
- [51] Chris Mack. *Fundamental principles of optical lithography: the science of microfabrication*. John Wiley & Sons, 2007. [3](#), [16](#), [18](#), [69](#), [72](#)
- [52] Chris A Mack. Development of positive photoresists. *Journal of the Electrochemical Society*, 134(1):148–152, 1987. [19](#), [70](#), [72](#), [75](#)
- [53] Chris A Mack. A new fast resist model: the gaussian lpm. In *Design for Manufacturability through Design-Process Integration V*, volume 7974, page 79740B. SPIE, 2011. [70](#), [72](#)
- [54] Don McLeish. A general method for debiasing a monte carlo estimator. *Monte Carlo Methods and Applications*, 17(4):301–315, 2011. [106](#)
- [55] Hossein Modarres and Ronald J Lomax. A formal approach to design-rule checking. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems (TCAD)*, 6(4):561–573, 1987. [11](#), [26](#)

- [56] Mervin E Muller. Some continuous monte carlo methods for the dirichlet problem. *The Annals of Mathematical Statistics*, 27(3):569–589, 1956. [25](#), [105](#)
- [57] James R Norris. *Markov chains*. Number 2 in Cambridge Series in Statistical and Probabilistic Mathematics. Cambridge university press, 1997. [13](#)
- [58] Chris J Oates, Mark Girolami, and Nicolas Chopin. Control functionals for monte carlo integration. *Journal of the Royal Statistical Society Series B: Statistical Methodology*, 79(3):695–718, 2017. [106](#)
- [59] Bernt Oksendal. *Stochastic differential equations: an introduction with applications*. Springer Berlin Heidelberg, 2003. [23](#)
- [60] OpenAI, Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altschmidt, Sam Altman, Shyamal Anadkat, et al. GPT-4 technical report. *arXiv preprint arXiv:2303.08774*, 2023. [15](#)
- [61] Michael Orshansky, Sani Nassif, and Duane Boning. *Design for Manufacturability and Statistical Design: A Constructive Approach*. Springer, 2008. [1](#), [11](#)
- [62] Stanley Osher and Ronald P Fedkiw. Level set methods: an overview and some recent results. *Journal of Computational physics*, 169(2):463–502, 2001. [20](#), [76](#)
- [63] Oberdan W Otto, Joseph G Garofalo, KK Low, Chi-Min Yuan, Richard C Henderson, Christophe Pierrat, Robert L Kostelak, Sheila Vaidya, and

- PK Vasudev. Automated optical proximity correction: a rules-based approach. In *Optical/Laser Microlithography VII*, volume 2197, pages 278–293. SPIE, 1994. [16](#)
- [64] Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll L. Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, John Schulman, Jacob Hilton, Fraser Kelton, Luke Miller, Maddie Simens, Amanda Askell, Peter Welinder, Paul Christiano, Jan Leike, and Ryan Lowe. Training language models to follow instructions with human feedback. In *Advances in Neural Information Processing Systems*, volume 35, pages 27730–27744, 2022. [15](#)
- [65] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. PyTorch: An imperative style, high-performance deep learning library. *Advances in neural information processing systems*, 32:8026–8037, 2019. [71](#), [77](#)
- [66] Y. C. Pati, A. A. Ghazanfarian, and R. F. Pease. Exploiting structure in fast aerial image computation for integrated circuit patterns. *IEEE Transactions on Semiconductor Manufacturing*, 10(1):62–74, 1997. [16](#)
- [67] Massoud Pedram and Shahin Nazarian. Thermal modeling, analysis, and management in vlsi circuits: principles and methods. *Proceedings of the IEEE*, 94(8):1487–1501, 2006. [3](#), [21](#)
- [68] Amyn Poonawala and Peyman Milanfar. Opc and psm design using inverse

- p lithography: a nonlinear optimization approach. In
- Optical Microlithography XIX*
- , volume 6154, page 61543H. SPIE, 2006.
- [17](#)
- [69] Aryn Poonawala and Peyman Milanfar. Mask design for optical microlithography—an inverse imaging problem. *IEEE transactions on image processing*, 16(3):774–788, 2007. [4](#), [17](#)
- [70] Kiran Puttaswamy and Gabriel H Loh. Thermal analysis of a 3d die-stacked high-performance microprocessor. In *Proceedings of the 16th ACM Great Lakes symposium on VLSI*, pages 19–24, 2006. [3](#), [102](#)
- [71] Haifeng Qian, S.R. Nassif, and S.S. Sapatnekar. Power grid analysis using random walks. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 24(8):1204–1224, 2005. [22](#), [24](#), [102](#), [105](#)
- [72] Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. Language models are unsupervised multitask learners. *OpenAI Blog*, 2019. [15](#)
- [73] John Randall, Kurt G Ronse, Thomas Marschner, Anne-Marie Goethals, and Monique Ercken. Variable-threshold resist models for lithography simulation. In *Optical Microlithography XII*, volume 3679, pages 176–182. SPIE, 1999. [18](#), [70](#), [72](#), [88](#), [89](#)
- [74] Gaurav Rajavendra Reddy, Constantinos Xanthopoulos, and Yiorgos Makris. Enhanced hotspot detection through synthetic pattern generation and design of experiments. In *IEEE VLSI Test Symposium (VTS)*, pages 1–6. IEEE, 2018. [2](#), [29](#)
- [75] Chang-han Rhee and Peter W Glynn. Unbiased estimation with square root convergence for sde models. *Operations Research*, 63(5):1026–1043, 2015. [106](#)

- [76] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 10684–10695, 2022. [21](#)
- [77] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In *International Conference on Medical Image Computing and Computer-Assisted Intervention (MICCAI)*, pages 234–241. Springer, 2015. [21](#), [54](#), [91](#)
- [78] David E Rumelhart, Geoffrey E Hinton, and Ronald J Williams. Learning representations by back-propagating errors. *nature*, 323(6088):533–536, 1986. [92](#)
- [79] Tim Salimans, Andrej Karpathy, Xi Chen, and Diederik P Kingma. Pixelcnn++: Improving the pixelcnn with discretized logistic mixture likelihood and other modifications. *arXiv preprint arXiv:1701.05517*, 2017. [14](#)
- [80] Rohan Sawhney, Bailey Miller, Ioannis Gkioulekas, and Keenan Crane. Walk on stars: A grid-free monte carlo method for pdes with neumann boundary conditions. *ACM Transactions on Graphics (TOG)*, 42(4):1–20, 2023. [25](#), [103](#)
- [81] Timo Schick, Jane Dwivedi-Yu, Roberto Dessì, Roberta Raileanu, Maria Lomeli, Eric Hambro, Luke Zettlemoyer, Nicola Cancedda, and Thomas Scialom. Toolformer: Language models can teach themselves to use tools. In *Advances in Neural Information Processing Systems*, volume 36, pages 68539–68551, 2023. [15](#)
- [82] Semiconductor Research Corporation. Microelectronics and advanced pack-

- aging technologies roadmap. Technical report, Semiconductor Research Corporation, 2023. Version 1.0; released October 2023. [1](#), [101](#)
- [83] James A Sethian. Fast-marching level-set methods for three-dimensional photolithography development. In *Optical Microlithography IX*, volume 2726, pages 262–272. SPIE, 1996. [20](#), [76](#)
- [84] Connor Shorten and Taghi M Khoshgoftaar. A survey on image data augmentation for deep learning. *Journal of big data*, 6(1), 2019. [44](#)
- [85] Gaurav Shrivastava and Abhinav Shrivastava. Video prediction by modeling videos as continuous multi-dimensional processes. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 7236–7245, 2024. [71](#), [74](#)
- [86] Mark D Smith, Jeffrey D Byers, and Chris A Mack. The lithographic impact of resist model parameters. In *Advances in Resist Technology and Processing XXI*, volume 5376, pages 322–332. SPIE, 2004. [3](#), [18](#), [70](#), [72](#)
- [87] Jascha Sohl-Dickstein, Eric A. Weiss, Niru Maheswaranathan, and Surya Ganguli. Deep unsupervised learning using nonequilibrium thermodynamics. In *Proceedings of the 32nd International Conference on Machine Learning (ICML)*, volume 37 of *JMLR Workshop and Conference Proceedings*, pages 2256–2265, 2015. [13](#), [20](#)
- [88] Jiaming Song, Chenlin Meng, and Stefano Ermon. Denoising diffusion implicit models. In *Proc. ICLR*, 2021. [20](#), [31](#), [74](#), [92](#)
- [89] Yang Song and Stefano Ermon. Generative modeling by estimating gradients of the data distribution. In *Advances in Neural Information Processing Systems*, volume 32, pages 11895–11907, 2019. [20](#)

- [90] Yang Song, Jascha Sohl-Dickstein, Diederik P. Kingma, Abhishek Kumar, Stefano Ermon, and Ben Poole. Score-based generative modeling through stochastic differential equations. In *International Conference on Learning Representations (ICLR)*, 2021. [20](#)
- [91] Arvind Sridhar, Alessandro Vincenzi, Martino Ruggiero, Thomas Brunschweiler, and David Atienza. 3d-ice: Fast compact transient thermal modeling for 3d ics with inter-tier liquid cooling. In *2010 IEEE/ACM International Conference on Computer-Aided Design (ICCAD)*, pages 463–470, 2010. [3](#), [105](#)
- [92] Stan Development Team. *Stan Modeling Language Users Guide and Reference Manual*, 2023. Version 2.33. [118](#)
- [93] Hameedah Sultan and Smruti R Sarangi. Varsim: A fast process variation-aware thermal modeling methodology using green’s functions. *Microelectronics Journal*, 142:105995, 2023. [4](#), [5](#), [102](#)
- [94] Hameedah Sultan, Anjali Chauhan, and Smruti R. Sarangi. A survey of chip-level thermal simulators. *ACM Comput. Surv.*, 52(2), April 2019. [21](#)
- [95] Shuyuan Sun, Yiyang Jiang, Fan Yang, Bei Yu, and Xuan Zeng. Efficient hotspot detection via graph neural network. In *DATE*, pages 1233–1238, 2022. [32](#)
- [96] James Thackeray, James Cameron, Vipul Jain, Paul LaBeaume, Suzanne Coley, Owendi Ongayi, Mike Wagner, Aaron Rachford, and John Biafore. Pursuit of lower critical dimensional uniformity in euv resists. *Journal of Photopolymer Science and Technology*, 26(5):605–610, 2013. [19](#), [70](#), [72](#)

- [97] James W Thackeray. Materials challenges for sub-20-nm lithography. *Journal of Micro/Nanolithography, MEMS, and MOEMS*, 10(3):033009–033009, 2011. [3](#), [19](#), [70](#), [73](#)
- [98] James W Thackeray, Roger A Nassar, Robert Brainard, Dario Goldfarb, Thomas Wallow, Yayi Wei, Jeff Mackey, Patrick Naulleau, Bill Pierson, and Harun H Solak. Chemically amplified resists resolving 25 nm 1:1 line: space features with euv lithography. In *Emerging Lithographic Technologies XI*, volume 6517, page 651719. SPIE, 2007. [3](#), [70](#), [73](#)
- [99] Anna W. Topol, Douglas C. La Tulipe Jr., Leathen Shi, David J. Frank, Kerry Bernstein, Steven E. Steen, Arvind Kumar, Gilbert U. Singco, Albert M. Young, Kathryn W. Guarini, and Mei Jeong. Three-dimensional integrated circuits. *IBM Journal of Research and Development*, 50(4/5):491–506, 2006. [1](#), [21](#)
- [100] Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, Aurelien Rodriguez, Armand Joulin, Edouard Grave, and Guillaume Lample. LLaMA: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*, 2023. [15](#)
- [101] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Proc. NeurIPS*, 30:5998–6008, 2017. [15](#), [55](#)
- [102] Marios Visvardis, Periklis Liaskovitis, and Efthymios Efstathiou. Deep-learning-driven random walk method for capacitance extraction. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 42(8):2643–2656, 2023. [102](#), [105](#)

- [103] Baohua Wang and Pinaki Mazumder. Accelerated chip-level thermal analysis using multilayer green’s function. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 26(2):325–344, 2007. [3](#), [21](#), [104](#)
- [104] Da-Wei Wang, Bo-Wen Zhang, Le-Tian Wang, Peng Zhang, and Wen-Sheng Zhao. A proposal of fast thermal simulation method for 2.5-d advanced packaging to enable efficient thermal-aware placement optimization. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 45(2):564–575, 2026. [102](#)
- [105] Qijing Wang, Bentian Jiang, Martin D. F. Wong, and Evangeline F. Y. Young. A2-ILT: Gpu accelerated ilt with spatial attention mechanism. In *ACM/IEEE Design Automation Conference (DAC)*, pages 967–972, 2022. [4](#), [17](#)
- [106] Ting-Yuan Wang and C.C.P. Chen. Thermal-adi - a linear-time chip-level dynamic thermal-simulation algorithm based on alternating-direction-implicit (adi) method. *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, 11(4):691–700, 2003. [21](#), [105](#)
- [107] Ting-Yuan Wang and Charlie Chung-Ping Chen. 3-d thermal-adi: a linear-time chip level transient thermal simulator. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 21(12):1434–1445, 2002. [105](#)
- [108] Yunbo Wang, Lu Jiang, Ming-Hsuan Yang, Li-Jia Li, Mingsheng Long, and Li Fei-Fei. Eidetic 3d lstm: A model for video prediction and beyond. In *International conference on learning representations*, 2019. [74](#)

- [109] Zixiao Wang, Yunheng Shen, Wenqian Zhao, Yang Bai, Guojin Chen, Farzan Farnia, and Bei Yu. DiffPattern: Layout pattern generation via discrete diffusion. In *2023 60th ACM/IEEE Design Automation Conference (DAC)*, pages 1–6. IEEE, 2023. [12](#), [27](#), [31](#), [32](#), [57](#), [59](#)
- [110] Zixiao Wang, Yunheng Shen, Xufeng Yao, Wenqian Zhao, Yang Bai, Farzan Farnia, and Bei Yu. Chatpattern: Layout pattern customization via natural language. In *Proceedings of the 61st ACM/IEEE Design Automation Conference*, pages 1–6, 2024. [7](#), [12](#), [28](#), [32](#), [46](#)
- [111] Zixiao Wang, Wenqian Zhao, Yunheng Shen, Yang Bai, Guojin Chen, Farzan Farnia, and Bei Yu. DiffPattern-Flex: Efficient layout pattern generation via discrete diffusion. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 44(12):4679–4690, 2025. [7](#), [28](#)
- [112] Zixiao Wang, Jieya Zhou, Su Zheng, Shuo Yin, Kaichao Liang, Shoubo Hu, Xiao Chen, and Bei Yu. TorchResist: open-source differentiable resist simulator. In *DTCO and Computational Patterning IV*, page 75. SPIE, 2025. [7](#)
- [113] Zixiao Wang, Tianshu Hou, Chenghan Wang, Zhen Zhuang, Tsung-Yi Ho, Farzan Farnia, and Bei Yu. FastRW: An efficient random walk method for steady-state thermal analysis. In *Proc. IEEE/ACM Design, Automation and Test in Europe (DATE)*, 2026. [8](#)
- [114] Zixiao Wang, Jieya Zhou, Xinyun Zhang, Shoubo Hu, Farzan Farnia, and Bei Yu. DiffResist: Physics-constrained diffusion for photoresist modeling. In *Proc. IEEE/ACM Design, Automation and Test in Europe (DATE)*, 2026. [7](#)

- [115] Yuki Watanabe, Taiki Kimura, Tetsuaki Matsunawa, and Shigeki Nojima. Accurate lithography simulation model based on convolutional neural networks. In *Optical Microlithography XXX*, volume 10147, page 101470K. SPIE, 2017. [18](#), [70](#), [73](#)
- [116] Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, Fei Xia, Ed Chi, Quoc V. Le, and Denny Zhou. Chain-of-thought prompting elicits reasoning in large language models. In *Advances in Neural Information Processing Systems*, volume 35, pages 24824–24837, 2022. [15](#)
- [117] Liangjian Wen, Yi Zhu, Lei Ye, Guojin Chen, Bei Yu, Jianzhuang Liu, and Chunjing Xu. LayoutTransformer: Generating Layout Patterns with Transformer via Sequential Pattern Modeling. In *Proc. ICCAD*, pages 1–9. IEEE, 2022. [2](#), [27](#), [31](#), [54](#), [57](#), [59](#)
- [118] Eric Wong and Sung Kyu Lim. 3d floorplanning with thermal vias. In *Proceedings of the Design Automation & Test in Europe Conference*, volume 1, pages 1–6, 2006. [22](#), [102](#), [105](#)
- [119] Haoyu Yang, Jing Su, Yi Zou, Bei Yu, and Evangeline FY Young. Layout hotspot detection with feature tensor generation and deep biased learning. In *ACM/IEEE Design Automation Conference (DAC)*, pages 1–6, 2017. [2](#), [26](#)
- [120] Haoyu Yang, Piyush Pathak, Frank Gennari, Ya-Chieh Lai, and Bei Yu. DeePattern: Layout pattern generation with transforming convolutional auto-encoder. In *ACM/IEEE Design Automation Conference (DAC)*, pages 1–6, 2019. [2](#), [12](#), [27](#), [30](#), [32](#), [57](#)
- [121] Haoyu Yang, Piyush Pathak, Frank Gennari, Ya-Chieh Lai, and Bei Yu.

- Detecting multi-layer layout hotspots with adaptive squish patterns. In *Proc. ASPDAC*, pages 299–304, 2019. [2](#), [12](#), [32](#)
- [122] Haoyu Yang, Shuhe Li, Zihao Deng, Yuzhe Ma, Bei Yu, and Evangeline F. Y. Young. GAN-OPC: Mask optimization with lithography-guided generative adversarial nets. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems (TCAD)*, 39(10):2822–2834, 2020. [2](#)
- [123] Ling Yang, Zhilong Zhang, Yang Song, Shenda Hong, Runsheng Xu, Yue Zhao, Wentao Zhang, Bin Cui, and Ming-Hsuan Yang. Diffusion models: A comprehensive survey of methods and applications. *ACM Computing Surveys*, 56(4):1–39, 2023. [20](#)
- [124] Longlong Yang, Cuiyang Ding, Changhao Yan, Dian Zhou, and Xuan Zeng. A high-precision stochastic solver for steady-state thermal analysis with fourier heat transfer robin boundary conditions. In *2022 IEEE/ACM International Conference On Computer Aided Design (ICCAD)*, pages 1–9, 2022. [25](#), [103](#), [106](#), [108](#), [116](#), [118](#), [119](#)
- [125] Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik Narasimhan, and Yuan Cao. React: Synergizing reasoning and acting in language models. In *International Conference on Learning Representations*, 2023. [15](#), [32](#)
- [126] Wei Ye, Yibo Lin, Meng Li, Qiang Liu, and David Z Pan. LithoROC: lithography hotspot detection with explicit ROC optimization. In *Proc. ASPDAC*, pages 292–298, 2019. [2](#), [29](#), [57](#)
- [127] Shuo Yin, Wenqian Zhao, Li Xie, Hong Chen, Yuzhe Ma, Tsung-Yi Ho, and Bei Yu. Fuilt: Full chip ilt system with boundary healing. In *Proceedings of the 2024 International Symposium on Physical Design, ISPD '24*, page 158

13–20, New York, NY, USA, 2024. Association for Computing Machinery.
[73](#), [90](#)

- [128] Bei Yu, Kun Yuan, Duo Ding, and David Z Pan. Layout decomposition for triple patterning lithography. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems (TCAD)*, 34(3):433–446, 2015.
[2](#), [26](#)
- [129] Bei Yu, Xiaoqing Xu, Subhendu Roy, Yibo Lin, Jiaojiao Ou, and David Z. Pan. Design for manufacturability and reliability in extreme-scaling VLSI. *Science China Information Sciences*, 59:061406, 2016. [1](#), [11](#)
- [130] Zihao Yuan, Prachi Shukla, Sofiane Chetoui, Sean Nemtzow, Sherief Reda, and Ayse K. Coskun. Pact: An extensible parallel thermal simulator for emerging integration and cooling technologies. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 41(4):1048–1061, 2022. [3](#), [21](#), [105](#)
- [131] Franz X Zach. Neural-network-based approach to resist modeling and opc. In *Optical Microlithography XVII*, volume 5377, pages 670–679. SPIE, 2004.
[18](#), [70](#), [73](#)
- [132] Yong Zhan and Sachin S. Sapatnekar. High-efficiency green function-based thermal simulation algorithms. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 26(9):1661–1675, 2007. [3](#), [104](#)
- [133] Yong Zhan, Brent Goplen, and Sachin S Sapatnekar. Electrothermal analysis and optimization techniques for nanoscale integrated circuits. In *Proceedings of the 2006 Asia and South Pacific Design Automation Conference*, pages 219–222, 2006. [3](#), [21](#), [102](#)

- [134] Xiaopeng Zhang, James Shiely, and Evangeline FY Young. Layout pattern generation and legalization with generative learning models. In *Proc. ICCAD*, pages 1–9. IEEE, 2020. [2](#), [12](#), [27](#), [30](#), [32](#), [54](#), [57](#), [59](#)
- [135] Wenqian Zhao, Xufeng Yao, Ziyang Yu, Guojin Chen, Yuzhe Ma, Bei Yu, and Martin DF Wong. Adaopc: A self-adaptive mask optimization framework for real design patterns. In *ICCAD*, pages 1–9, 2022. [32](#)
- [136] Su Zheng, Haoyu Yang, Binwu Zhu, Bei Yu, and Martin Wong. LithoBench: Benchmarking AI computational lithography for semiconductor manufacturing. *Advances in Neural Information Processing Systems*, 36:30243–30254, 2023. [5](#), [86](#)
- [137] Su Zheng, Gang Xiao, Ge Yan, Meng Dong, Yao Li, Hong Chen, Yuzhe Ma, Bei Yu, and Martin Wong. Model-based opc extension in openilt. In *Proc. ISEDA*, pages 568–573. IEEE, 2024. [2](#), [17](#)
- [138] Xuanhe Zhou, Zhaoyan Sun, and Guoliang Li. Db-gpt: Large language model meets database. *Data Science and Engineering*, 9(1):102–111, 2024. [32](#)
- [139] Yijing Zhou, Wei Cai, and Elton Hsu. Computation of local time of reflecting brownian motion and probabilistic representation of the neumann problem. *arXiv preprint arXiv:1502.01319*, 2015. [23](#)

Publications Related to This Thesis

The following publications by the author are directly related to the content of this thesis. An asterisk denotes equal contribution.

1. **Z. Wang**, J. Zhou, X. Zhang, S. Hu, F. Farnia, and B. Yu, “DiffResist: Physics-constrained diffusion for photoresist modeling,” in *Proc. IEEE/ACM Design, Automation and Test in Europe (DATE)*, Verona, 2026.
2. **Z. Wang**, T. Hou, C. Wang, Z. Zhuang, T.-Y. Ho, F. Farnia, and B. Yu, “FastRW: An efficient random walk method for steady-state thermal analysis,” in *Proc. IEEE/ACM Design, Automation and Test in Europe (DATE)*, Verona, 2026.
3. **Z. Wang**, W. Zhao, Y. Shen, Y. Bai, G. Chen, F. Farnia, and B. Yu, “DiffPattern-Flex: Efficient layout pattern generation via discrete diffusion,” *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems (TCAD)*, vol. 44, no. 12, pp. 4679–4690, 2025.
4. **Z. Wang**, J. Zhou, S. Zheng, S. Yin, K. Liang, S. Hu, X. Chen, and B. Yu, “TorchResist: Open-source differentiable resist simulator,” in *Proc. SPIE Advanced Lithography + Patterning*, San Jose, 2025.

5. **Z. Wang***, Y. Shen*, X. Yao, W. Zhao, Y. Bai, F. Farnia, and B. Yu, “ChatPattern: Layout pattern customization via natural language,” in *Proc. ACM/IEEE Design Automation Conference (DAC)*, San Francisco, 2024.
6. **Z. Wang***, Y. Shen*, W. Zhao, Y. Bai, G. Chen, F. Farnia, and B. Yu, “DiffPattern: Layout pattern generation via discrete diffusion,” in *Proc. ACM/IEEE Design Automation Conference (DAC)*, San Francisco, 2023.

Other publications by the author during the doctoral study:

7. **Z. Wang**, L. Jin, Z. Zhuang, R. Chen, and B. Yu, “CSCO: A backside-PDN-aware clock-signal co-optimization framework for improved PPA,” in *Proc. IEEE/ACM International Conference on Computer-Aided Design (ICCAD)*, San Jose, 2026.
8. C. Wang, Z. Yu, S. Zheng, **Z. Wang**, Y. Shi, and B. Yu, “Moreau-ILT: Efficient inverse lithography exploiting weak convexity,” in *Proc. IEEE/ACM International Conference on Computer-Aided Design (ICCAD)*, San Jose, 2026.
9. L. Jin, H. Xu, S. Liang, Z. Zhuang, Z. Hu, **Z. Wang**, B. Yu, R. Chen, and T.-Y. Ho, “BSPDN-Elite: A comprehensive framework for optimizing timing, power and routing resources in BSPDN designs,” in *Proc. ACM/IEEE Design Automation Conference (DAC)*, Long Beach, 2026.
10. W. Xu, S. Chen, Y. Huang, X. Zhang, **Z. Wang**, B. Yu, and N. Zou, “Node2Node: Node adaptation with transformer for cross-node hotspot detection,” in *Proc. IEEE/ACM Design, Automation and Test in Europe (DATE)*, Verona, 2026.

11. R. Fu, L. Shen, Z. Wang, Z. Lei, **Z. Wang**, J. Huang, B. Yu, and T.-Y. Ho, “DCLOG: Don’t cares-based logic optimization using pre-training graph neural networks,” in *Proc. IEEE/ACM Asia and South Pacific Design Automation Conference (ASP-DAC)*, Hong Kong, 2026.
12. X. Zhang*, **Z. Wang***, Y. Kuang, and B. Yu, “Video-based visible-event cross-modal person re-identification for edge AI surveillance systems,” in *Proc. IEEE/ACM Asia and South Pacific Design Automation Conference (ASP-DAC)*, Hong Kong, 2026.
13. **Z. Wang**, F. Farnia, Z. Lin, Y. Shen, and B. Yu, “On the distributed evaluation of generative models,” in *Proc. IEEE/CVF International Conference on Computer Vision Workshops (ICCVW)*, Honolulu, pp. 7703–7712, 2025.
14. Z. Yu, P. Xu, **Z. Wang**, B. Zhu, Q. Wang, Y. Lin, R. Wang, B. Yu, and M. Wong, “SDM-PEB: Spatial-depthwise Mamba for enhanced post-exposure bake simulation,” in *Proc. ACM/IEEE DAC*, San Francisco, 2025.
15. **Z. Wang**, J. Weng, M. Liu, and B. Yu, “FlexPose: Pose distribution adaptation with limited guidance,” in *Proc. AAAI Conference on Artificial Intelligence (AAAI)*, Philadelphia, 2025.
16. W. Zhao, L. Zou, **Z. Wang**, X. Yao, and B. Yu, “HAPE: Hardware-aware LLM pruning for efficient on-device inference optimization,” *ACM Transactions on Design Automation of Electronic Systems (TODAES)*, vol. 30, no. 4, pp. 61:1–61:18, 2025.
17. W. Zhao, S. Yin, C. Bai, **Z. Wang**, and B. Yu, “BAQE: Backend-adaptive DNN deployment via synchronous Bayesian quantization and hardware configuration exploration,” *IEEE Transactions on Computer-Aided Design of*

Integrated Circuits and Systems (TCAD), vol. 44, no. 4, pp. 1394–1405, 2025.

18. G. Chen, **Z. Wang**, B. Yu, D. Z. Pan, and M. D. F. Wong, “Ultrafast source mask optimization via conditional discrete diffusion,” *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems (TCAD)*, vol. 43, no. 7, pp. 2140–2150, 2024.
19. Y. Bai, X. Yao, Q. Sun, W. Zhao, S. Chen, **Z. Wang**, and B. Yu, “GTCO: Graph and tensor co-design for transformer-based image recognition on tensor cores,” *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems (TCAD)*, vol. 43, no. 2, pp. 586–599, 2024.
20. Y. Bai, W. Zhao, S. Yin, **Z. Wang**, and B. Yu, “ATFormer: A learned performance model with transfer learning across devices for large-scale transformers,” in *Proc. Conference on Empirical Methods in Natural Language Processing (EMNLP)*, Singapore, 2023.
21. **Z. Wang**, J. Weng, C. Yuan, and J. Wang, “Truncate-split-contrast: A framework for learning from mislabeled videos,” in *Proc. AAAI Conference on Artificial Intelligence (AAAI, Oral)*, Washington, 2023.

Publication before the doctoral study:

22. H. Ding, **Z. Wang**, and R. Wu, “Enhancing the security of multi-agent networked control systems using QKD based homomorphic encryption,” in *Proc. IEEE Conference on Decision and Control (CDC)*, Miami Beach, 2018.