Context-OPCGAN: Context-Aware OPC Modeling with Generative Adversarial Networks

Xin Hong[†‡], Shuhan Wang[†‡], Yajuan Su[†‡*], Xiaojing Su[†‡], Bojie Ma[†‡], Zixi Liu[†‡], Xiaohuan Ling[†‡], Yuqin Wang[†‡], Pengyu Ren[†‡], Yujie Jiang[†‡], Zhanzi Chen[†‡], Tianao Chen[†‡], and Yayi Wei[†‡*]
[†]Institute of Microelectronics of The Chinese Academy of Sciences, Beijing, China
[‡]University of Chinese Academy of Sciences, Beijing, China
*Corresponding Author Email: suyajuan@ime.ac.cn, weiyayi@ime.ac.cn

***Abstract*—Optical proximity correction (OPC) is a crucial step in compensating for the optical proximity effect which causes pattern distortion in contemporary integrated circuit manufacturing. Traditional OPC methodologies rely on lithography modeling and simulation. However, given the escalating intricacies of design and process, there arises a pressing necessity for an innovative OPC approach integrated with deep learning techniques. Such a strategy is imperative to fulfill the dual requirements of correction performance and efficiency in semiconductor manufacturing.**

In this paper, we proposed a framework called Context-OPCGAN, which is based on generative adversarial networks (GANs). Context-OPCGAN incorporates surrounding environment considerations into OPC modeling, resulting in enhanced pattern correction performance. Compared to methods that neglect the influence of neighboring pattern topology, Context-OPCGAN achieves an 8% increase in the process variation band (PVBAND). Furthermore, it maintains mask and contour graphics continuity even after slice stitching. Context-OPCGAN exhibits significant potential for efficient OPC modeling, particularly in high-volume IC manufacturing.

I. Introduction

A. Background

The goal of OPC is to accurately replicate the transformation of the mask pattern into the photoresist, while accounting for the non-linear characteristics of mask imaging within the OPC model. Traditional OPC approaches are typically divided into two categories: model-based technology[1, 2] and inverse lithography-based technology (ILT)[3–5]. While both methods exhibit good performance, they necessitate multiple iterations of photolithography simulations during the iterative optimization process, which is both labor-intensive and time-consuming. Consequently, there has been a growing interest among researchers in integrating machine learning into OPC to mitigate these time-consuming aspects.

In the field of computer vision, Generative Adversarial Network (GAN) has shown impressive results in image conversion[6]. In recent years, several researchers have focused on GAN-based OPC methods, particularly in process implementation and model verification, demonstrating the feasibility of applying GAN to OPC.

B. OPC-GAN Flow

Yang[7] initially introduced an OPC-oriented GAN flow that can learn the target layout-mask mapping. While this approach shows promising results, it is limited by its focus on enhancing image conversion accuracy and its reliance on a collection of photomask images. Assembling a set of qualified photomask images proves prohibitively expensive and time-consuming, posing significant challenges for its practical application in the industry. Shao[8] was the first to propose a GAN-OPC model that closely resembles the industrial process. This model encompasses two GANs: OPCnet for OPC and Lithonet for lithography. The training process of this model unfolds in two stages. Initially, Lithonet is trained using collected mask and wafer contour datasets to achieve high accuracy. In the second stage, the parameters of Lithonet are set and cascade Lithonet to OPCnet. That is, the target layout is inputted into OPCnet, which then converts it into a mask. This mask is then inputted into Lithonet, which converts it into a simulated wafer contour. OPCnet performs self-training with the target layout dataset, while iterative optimization aims to minimize discrepancies between the simulated wafer contour and the target layout. This model can construct Lithonet with a limited amount of mask and wafer contour datasets, subsequently relying solely on target layout for self-supervised training to obtain OPCnet. Consequently, it effectively addresses the primary challenge related to data sources.

C. Purpose of Context-OPCGAN

Typically, the training set of a deep-learning-based OPC model consists of a series of layout clips. When these clips are obtained through sequential slicing of the layout, adjacent clips may lose their original adjacency. However, the surrounding layout has a significant influence on the OPC process. Traditionally, the construction of the OPC model has relied on the test pattern. With continuous scanning by the scanning electron microscope (SEM), the continuous layout can be directly utilized for dataset production. Hence, it is feasible to develop an OPC model and dataset production method that considers the surrounding environment. The main contributions of this paper are outlined below:

•Proposing Context-OPCGAN, a model that incorporates the surrounding environment of the layout.

•Analyzing a critical methodological issue in existing GAN-OPC processes: the omission of mask thresholding.

•Designing a novel slicing method that includes the surrounding environment for model validation.

II. Context-OPCGAN Framework

A. Context-Aware Slicing Methods

As depicted in Figure 1, the continuous layout is initially segmented into large slices of 1024 × 1024. Subsequently, each of these large slices is subdivided into 9 smaller slices of 256 × 256. The 256 × 256 area serves as the region for the OPC operation, while the area outside the 256 × 256 area is considered as the surrounding environment. The dimensions of this surrounding area are carefully determined to maintain an appropriate ratio with the central area. In our experiment, we extract squares from the same center of the aforementioned 256 × 256 slice. These squares possess varying side lengths, resulting in clips with side lengths of 320, 384, 448, and 512 respectively.

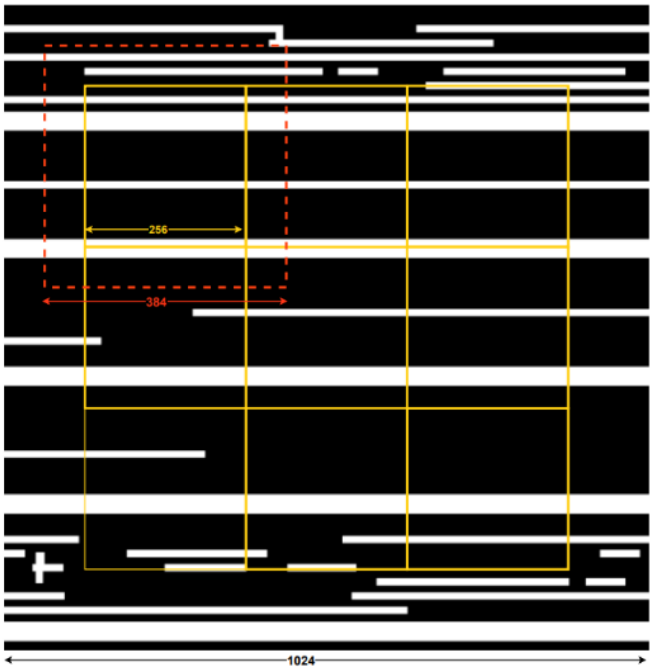

Fig. 1. Context-Aware Clips Cut

B. Structure of Context-OPCGAN

The architecture of the generator and discriminator are shown in Table I. OPCnet and Lithonet use the same generator generator and discriminator. The training process of Context-OPCGAN is divided into two stages as shown in Figure 2. In the first stage of training for Lithonet, the training set is a pair of clips of mask and wafer contour. The objective is to ensure that the clips derived from the input mask clips once transformed by Lithonet, closely resemble the actual wafer contour. We use L2 loss to measure this resemblance. The loss of the generative network is shown as eq1.

$$L_g = log(1 - D(Lithonet(x))) + \frac{1}{n}\|x - y\|^2 \quad (1)$$

In the second stage of training for OPCnet, only target layout clips are required for the training set. Initially, OPCnet

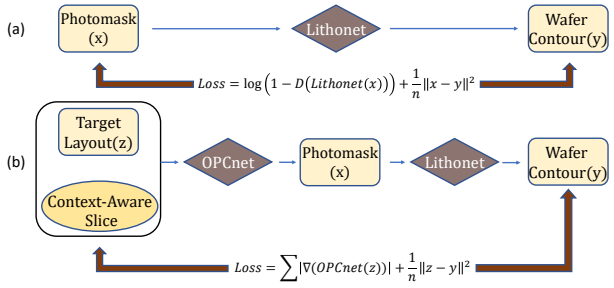

Fig. 2. Training Process of Context-OPCGAN (a)Training for Lithonet (b) Training for Cascaded Context-OPCGAN

TABLE I
Structure of Generator and Discriminator

Type	Layer	Filter	Stride	Output-size
Discriminator	Input	-	-	256×256×1
	Conv-1	5×5×32	2	128×128×32
	Conv-2	5×5×64	2	64×64×64
	Conv-3	5×5×128	2	32×32×128
	Conv-4	5×5×256	2	16×16×256
	Output	5×5×1	1	8×8×1
Generator	Input	-	-	256×256×32
	Conv-1	5×5×32	2	128×128×32
	Conv-2	5×5×64	2	64×64×32
	Conv-3	5×5×128	2	32×32×64
	Conv-4	5×5×256	2	16×16×256
	Resnet	-	-	64×64×128
	Deconv-1	5×5×256	1	16×16×256
	Deconv-2	5×5×128	1	32×32×128
	Deconv-3	5×5×64	1	64×64×64
	Deconv-4	5×5×32	1	128×128×32
	Output	5×5×1	1	256×256×1

undergoes pre-training to imbue the model with the capability to generate the initial contour. This pre-training phase utilizes a training set comprising target layout clips and their corresponding versions processed by Gaussian noise.The objective function is the same as eq1. Pre-training enables OPCnet to generate well-defined polygons. Subsequently, OPCnet and Lithonet are cascaded, wherein the output of OPCnet serves as the input for Lithonet. The parameters trained during the first stage are imported into Lithonet and remain constant throughout the training process, while the pre-trained parameters are imported into OPCnet. The objective of the cascaded network is to maximize the similarity between the input target layout and the output wafer contour. The loss of the generative network is shown as eq2.

$$L_g = \frac{1}{n}\|z - Lithonet(OPCnet(z))\|_2 + \sum|\bigtriangledown(OPCnet(z))| \quad (2)$$

C. Mask Thresholding

During the training process of GAN, the mask needs to be processed. In the first stage of Lithonet training, binarization pixel value of mask is employed as input, so inputting a binary image to Lithonet can get the most realistic output. However, during the second training stage, OPCnet generates

graphics with continuous pixel values. Following the cascading process, the output graphics are then fed into the pre-trained Lithonet. For rigorousness, it is necessary to perform threshold processing during the training process. However, binary processing of the output from OPCnet would lead to the disappearance of neural network gradient backpropagation. Hence, an approximate processing method is adopted. Obviously, the threshold value will affect the number of remaining pixels in the processed graph. We evaluated the effect of model training with the threshold value ranging from 0.1-0.5, ultimately setting the final threshold at 0.2.

D. Performance Parameters

In the study of the GAN-OPC problem, PVBAND is defined as the difference between the photoresist profile and the target layout[9]. Since the layout graphics are all binarized, the value of each pixel in the target layout is 0 or 1, which can be regarded as the true value. Then the difference can be described as the prediction accuracy of each pixel of the photoresist profile multiplied by the area of each pixel.

$$PVBAND = (FP + FN) \times pixel - area \tag{3}$$

This parameter serves as metrics to gauge the performance of OPC models in analogous works.

III. Experimental Results

A. Dataset Collection

The dataset used in the experiment is the ICCAD2 subset of the ICCAD2012 dataset, and the layout size in the dataset is about $13000nm^2$. GAN-based OPC is based on pixel processing. Therefore, during the feature extraction from the layout, a density-based extraction method is utilized to procure the feature matrix of the layout.

When the clip size is larger, fewer clips are obtained through division, resulting in each slice containing more polygons. Intuitively, the more polygons in each clip, the more information the model can obtain during training. However, when the pixel size becomes excessively large, considering that the layout is a binary graph and the error is quantified by the number of pixels, the desired result may not be accurately achieved. Simultaneously, when the pixel size is excessively small, the number of polygons contained in each slice is few, and the model may not acquire sufficient information during training. Figure 3 illustrates clips with pixel sizes set to 10nm, 4nm, and 1nm. Considering the impact on model training and measurement accuracy, the pixel size of this experiment is set to 4nm.

B. Training Process of Context-OPCGAN

Figure 4(a) illustrates the curve of L2 loss versus training progress for training Lithonet in the first stage. It can be seen that at about 2500 training batches, the L2 loss converges to around 0.015. At this time, the wafer contour converted by Lithonet and its corresponding mask is shown in Figure 4(b).

In the second stage of training, pre-training OPCnet to generate layout polygons is essential to prevent mode collapse.

Fig. 3. Layout Clips with Different Pixel-Size

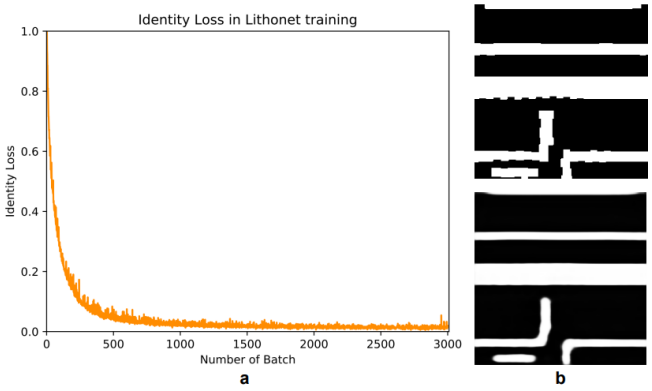

Fig. 4. L2 Loss Curve for Training Lithonet and Transformed Contour

Figure 5 illustrates the pre-training results of OPCnet, showcasing the original target layout alongside its converted graph. Remarkably, the graph can be generated in approximately 1000 training batches, demonstrating rapid training speed.

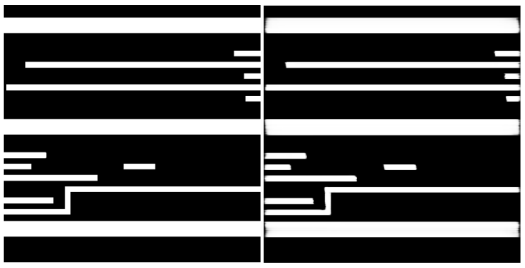

Fig. 5. The Pretrain Output of OPCnet

Despite incorporating a smoothing loss function, the output of OPCnet does not yield a binarized graph. To ensure experimental rigor, we investigated the impact of omitting thresholding on the results. After training, we separately processed the mask output from OPCnet with and without thresholding before inputting it into Lithonet. The experiment results comparing the contours are depicted in Figure 6. In Figure 6, (a) represents the target layout slice, while (b) and (c) display the contour obtained without and with thresholding, respectively, using two different methods. Notably, the difference in the final output graph is minimal. We further compared the averages PVBAND for 150 test slices, with an acceptable error range.

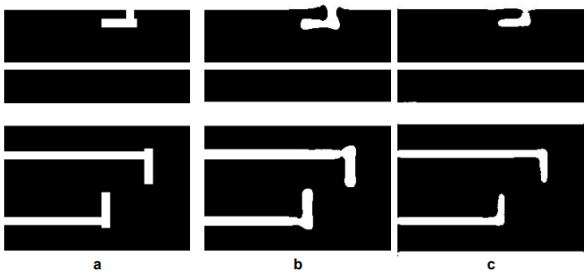

Fig. 6. The Difference Between Whether The Output of OPCnet is Thresholded or Not

C. Performance Evaluation

In our experiments, a training set with varying sizes of surrounding environments and without surrounding environment are constructed. We adhered to the same procedure across experiments involving five distinct training sets. Table 3 shows the average performance parameters of 150 clips after training and testing on the 5 datasets of the experimental design. In Column 256, representing no ambient input, the results show that with an ambient width of 128 pixels (Column 384), PVBAND is optimized by 8%. However, increasing the input size inevitably leads to higher time costs. The time taken to process 150 large slices of 1024 sides is also provided in the table. Context-OPCGAN exhibits certain improvements compared to methods that do not consider the surrounding environment.

TABLE II
STATISTICS OF PVBAND RESULTS

Size	256	320	384	448	512
PVBAND	0.01618	0.01586	**0.01495**	0.01582	0.01526
Time(s)	**219**	248	278	322	364

Figure 7 illustrates the model trained on the 384 dataset, and then spliced into large slices after converting the test set. The image conversion maintains continuity at the connection of small slices, demonstrating that our model is also competent for the application of the real layout.

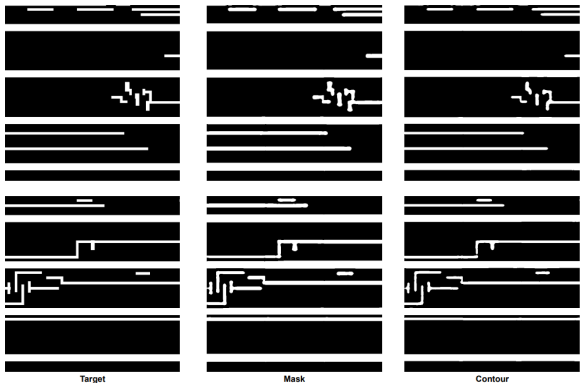

Fig. 7. Mask and Contour Images Obtained Using the Trained Context-OPCGAN

IV. CONCLUSION

This paper introduces Context-OPCGAN, a model that integrates the surrounding environment into the optical proximity correction process. Leveraging generative adversarial networks, this model is tailored for continuous layouts and can dynamically adjust the size of the surrounding environment to account for its influence on OPC. To validate the model's efficacy, we devised a slicing method that incorporates the surrounding environment. We conducted experiments training the model under varying surrounding environment sizes and compared performance parameters. Utilizing the ICCAD2012 dataset with an input slice size of 384, we observed an 8% improvement in PVBAND compared to ignoring the surrounding environment (slice size of 256). Additionally, upon stitching the slices together, our model demonstrated well-maintained continuity in the generated mask and contour graphics.

ACKNOWLEDGMENTS

This work was financially supported by the Strategic Priority Research Program of Chinese Academy of Sciences (Grant No.XDA0330303), National Natural Science Foundation of China (No.62204257 and 62274181). We acknowledge the support from Youth Innovation Promotion Association Chinese Academy of Sciences (No.2021115), Beijing Institute of Electronics, Beijing Association for Science and Technology as well.

REFERENCES

[1] Y.-H. Su, Y.-C. Huang, L.-C. Tsai, Y.-W. Chang, and S. Banerjee, "Fast lithographic mask optimization considering process variation," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 35, no. 8, pp. 1345–1357, 2016.

[2] J. Kuang, W.-K. Chow, and E. F. Young, "A Robust Approach for Process Variation Aware Mask Optimization," in *Design, Automation & Test in Europe Conference & Exhibition (DATE), 2015*, (Grenoble, France), pp. 1591–1594, IEEE Conference Publications, 2015.

[3] A. Poonawala and P. Milanfar, "Mask design for optical microlithography—an inverse imaging problem," *IEEE Transactions on Image Processing*, vol. 16, no. 3, pp. 774–788, 2007.

[4] J.-R. Gao, X. Xu, B. Yu, and D. Z. Pan, "Mosaic: Mask optimizing solution with process window aware inverse correction," in *2014 51st ACM/EDAC/IEEE Design Automation Conference (DAC)*, pp. 1–6, 2014.

[5] Y. Ma, W. Zhong, S. Hu, J.-R. Gao, J. Kuang, J. Miao, and B. Yu, "A unified framework for simultaneous layout decomposition and mask optimization," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 39, no. 12, pp. 5069–5082, 2020.

[6] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial networks," *Commun. ACM*, vol. 63, p. 139–144, oct 2020.

[7] H. Yang, S. Li, Z. Deng, Y. Ma, B. Yu, and E. F. Y. Young, "Gan-opc: Mask optimization with lithography-guided generative adversarial nets," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 39, no. 10, pp. 2822–2834, 2020.

[8] H.-C. Shao, C.-Y. Peng, J.-R. Wu, C.-W. Lin, S.-Y. Fang, P.-Y. Tsai, and Y.-H. Liu, "From ic layout to die photograph: A cnn-based data-driven approach," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 40, no. 5, pp. 957–970, 2021.

[9] X. Xu, T. Matsunawa, S. Nojima, C. Kodama, T. Kotani, and D. Z. Pan, "A machine learning based framework for sub-resolution assist feature generation," in *Proceedings of the 2016 on International Symposium on Physical Design*, ISPD '16, (New York, NY, USA), p. 161–168, Association for Computing Machinery, 2016.