# TIP-OPC: A New Topological Invariant Paradigm for Pixel Based Optical Proximity Correction

Peng Yu and David Z. Pan
The Department of Electrical and Computer Engineering
The University of Texas at Austin
Email: yupeng@cerc.utexas.edu and dpan@ece.utexas.edu

***Abstract*—As the 193nm lithography is likely to be used for 45nm and even 32nm processes, much more stringent requirement will be posed on Optical Proximity Correction (OPC) technologies. Currently, there are two OPC approaches — the model-based OPC (MB-OPC) and the inverse lithography technology (ILT). MB-OPC generates masks which is less complex compared with ILT. But ILT produces much better results than MB-OPC in terms of contour fidelity because ILT is a pixel based method. Observing that MB-OPC preserves the mask shape topologies which leads to a lower mask complexity, we combine the strengths of both methods — the topology invariant property and the pixel based mask representation. To the best of our knowledge, it is the first time that this topological invariant pixel based OPC (TIP-OPC) paradigm is proposed, which fills the critical hole of the OPC landscape and potentially has many new applications. Our technical novelty includes the lithography friendly mask topological invariant operations, the efficient Fast Fourier Transform based cost function sensitivity computation and the TIP-OPC algorithm. The experimental results show that TIP-OPC can achieve much better post OPC contours compared with MB-OPC while maintaining the mask shape topologies.**

## I. Introduction

As the semiconductor industry technology node scales down to 65, 45 or even 32 nm, the pattern printability and the process window are significantly reduced due to the fundamental limit of lithography systems. A variety of Resolution Enhancement Techniques (RET) are developed to address this problem. Optical Proximity Correction (OPC) is one the most important RET techniques.

The current most widely used OPC is Model-Based OPC (MB-OPC) [1–3], where the mask shapes are represented as parametrized polygons such that the shapes can be modified to compensate lithography pattern transfer non-idealities. However, the OPCed result quality is subject to the OPC-recipes (rules for segmenting and tagging), which may be very complex and hard to tune [4]. Also, the parametrized polygons are not flexible enough to represent any possible shapes, which could result in contour fidelity degradation.

Alternatively, Inverse lithography technology (ILT) [5–11] represents mask shapes as pixel images, which allows more flexible mask shape modifications. This approach could result in better image fidelity comparing with MB-OPC. It has also been demonstrated that hardware acceleration and parallel computation can improve the runtime significantly such that this method can be applied to a full chip in practice [12]. However, all the published ILT formulations [7, 8, 11, 13] do not constrain the mask shapes explicitly. The ILT algorithms add Sub-Resolution Assist Features (SRAF) by models. But they may add too many SRAFs, which result in big mask complexity. As an example, Figure 1 shows the target patterns and the mask pattern generated by ILT in [11]. In addition, these approaches do not guarantee the minimum size of SRAFs, while small SRAFs may induce mask inspection problems.

MB-OPC does not add SRAFs, which maintains the mask shape topology. We call the OPC algorithm which maintains mask topologies as topological invariant OPC (TI-OPC). ILT clearly belongs to the other category, topological variant OPC (TV-OPC). Figure 2 shows a cartoon picture of the difference between TI-OPC and TV-OPC.

(a) Target (b) Mask

Figure 1. Mask patterns generated from ILT can be extremely complicated with many small features and SRAFs [11].

Figure 2. A cartoon picture of the TI-OPC (left) and TV-OPC (right) results. The center one is the target. The left mask has the same topology as the target, while the right one is not topologically equivalent to the target.

It is beneficial to combine the advantage of MB-OPC (topological invariance) and the advantage of ILT (pixel based). We call the new paradigm topology invariant pixel based OPC (TIP-OPC). TIP-OPC is OPC-recipe independent and it is much superior compared to the MB-OPC in terms of print contour fidelity. These are important for Design-For-Manufacturability (DFM) analysis. That is, if the designs are not friendly to TIP-OPC, we would know they are definitely not good for MB-OPC, which means designs have to be modified. Otherwise, even if the designs do not have good contour fidelity using MB-OPC, it could be due to imperfection in the OPC recipes. In this case, we should look for better recipes, which may save unnecessary design modifications. We believe the introduction of TIP-OPC could be a fundamental breakthrough in OPC algorithms.

The main contributions of this paper are as follows:

- We introduce a new topological invariant paradigm for pixel based OPC, which provides a trade-off between OPC quality and mask complexity which is not possible in the current methodologies, like MB-OPC and ILT.
- TIP-OPC have finer control over masks than MB-OPC, it can achieve better contour fidelity.
- TIP-OPC explicitly controls the SRAF generations, which results in less complex masks compared to ILT. In this work, we consider the case of not generating any SRAFs. We will extend the algorithm such that it can generate a given number of SRAFs in the future.

1-4244-1382-6/07/$25.00 ©2007 IEEE

- Our algorithm is enabled by efficient techniques on topological invariant mask operations and fast FFT based cost function sensitivity computation.
- We derive, for the first time, the optimal tile size for dense simulation method.

The rest of this paper is organized as follows. We review the lithography fundamentals for sparse and dense simulation methods in Section II. In Section III, we show the optimal value of the tile size parameter and show that the dense simulation method could be faster than the sparse simulation method for dense simulation applications. In Section IV, we formally define topology invariance and derive topology invariant pixel based mask operations with lithographic considerations. In Section V, we describe the contour based cost function and the TIP-OPC algorithm. Section VI shows the experimental results, followed by the conclusions in Section VII.

## II. Lithography Preliminaries

Any lithography system can be described as a function $T[\cdot]$,

$$r = T[m], \tag{1}$$

which transfers the mask transmission function $m(x, y)$ to the normalized post development photoresist thickness function $r(x, y)$ ($0 \leqslant r \leqslant 1$). The $(x, y)$ in $m(x, y)$ and $r(x, y)$ denotes the locations on the mask and the wafer respectively. In reality, masks can only take discrete transmission values. For example, binary masks are allowed to take only 0 or 1. We denote the region where $m$ equals 1 as $M$ and the region where $m$ equals 0 as $\overline{M}$, which is $M$'s complement.

As a simplification, the transfer from the mask pattern to the photoresist pattern on the wafer can be divided into the following two steps.

### A. Aerial Image Formation

The aerial image intensity is described by the Hopkins' Equation [14]. It can be simplified by the kernel decomposition [3, 15–18], which has the following two formulations. In the first formulation, the image intensity $I(x, y)$ is computed in spatial domain as

$$I(x, y) = \sum_{p=0}^{P-1} \sigma_p |h_p * m|^2, \tag{2}$$

where $h_p(x, y)$'s are called optical kernels, which incorporate the optical system information, and $*$ is the convolution operator. In the second formulation, the image intensity $I(x, y)$ is computed in the frequency domain using FFT as

$$I(x, y) = \sum_{p=0}^{P-1} \sigma_p \Big| \mathcal{F}\big[H_p \mathcal{F}^{-1}[m]\big] \Big|^2, \tag{3}$$

where $\mathcal{F}$ and $\mathcal{F}^{-1}$ are the Fourier and inverse Fourier transform operators and

$$H_p = \mathcal{F}^{-1}[h_p]. \tag{4}$$

These two methods are called the *sparse* method and the *dense* simulation method, respectively.

### B. Photoresist Development

In this work, we use the *constant threshold model*. We will extend the work using other models in the future. In the constant threshold model model, the photoresist thickness $r(x, y)$ can be written as

$$r(x, y) = D\big(I(x, y)\big), \tag{5}$$

where the function $D(\cdot)$ has the following form

$$D(I) = \begin{cases} 0 & I \geqslant I_{\text{th}} \\ 1 & I < I_{\text{th}} \end{cases}. \tag{6}$$

## III. Analysis of the Complexities of Sparse and Dense Simulation Methods

Lithography simulation used to be sparsely performed in OPC software [3]. As technology scales, it becomes desirable to perform lithography simulation on more locations to avoid the printing of any unnecessary features [19–21]. Dense simulating is also preferred to take advantage of the latest photoresist modeling advancement [22]. We will demonstrate that it is better to use the dense simulation method than to use the conventional sparse simulation method for the dense simulation [21].

In lithography simulation, the images are simulated on grid points. We study a chip discretized to a grid of $L \times L$ points. We define a few useful terms below:

- $s$ is the percentage of grid points where simulations are performed. $s = 1$, when the simulations are performed on every grid points (extremely dense).
- $v$ is the ratio between the total number of polygon vertexes and the total number of mask pixels.
- $K^2$ is the size of the spatial kernels ($h_p$'s).

We can derive the complexity for sparse simulation method [3] as follows.

**Lemma 1** (Sparse simulation method complexity)**.** *The total number of complex number operations using* (2) *is approximately*

$$T_{sparse} = f(K) P L^2, \tag{7}$$

*where* $f(K)$ *is written as*

$$f(K) = svK^2. \tag{8}$$

For dense simulation, the chip of size $L \times L$ is divided into tiles of size $(K' - K) \times (K' - K)$ to save memory usage (Figure 3). Each

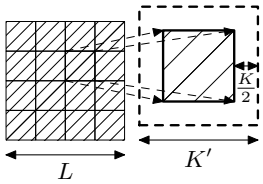



Figure 3. A chip with size $L \times L$ is divided into many tiles of size $K' - K$, where $K$ is the same as that of sparse simulation and $K'$ is a tunable parameter. Each tile is zero-padded to of size $K'$.

tile is then zero-padded to of size $K' \times K'$ [23]. Let us denote the complexity of the FFT of a $K' \times K'$ array as $CK'^2 \log K'$, where $C$ is a constant depending on the implementation details of the FFT algorithm [24]. We can derive the complexity for dense simulation method as follows.

**Lemma 2** (Dense simulation method complexity)**.** *The total number of complex number operations using* (3) *is approximately*

$$T_{dense} = Cg(K', K) P L^2, \tag{9}$$

*where*

$$g(K', K) = \frac{K'^2}{(K' - K)^2} \log K'. \tag{10}$$

Since the factor $PL^2$ is common to (7) and (9), we only to need consider $f(K)$ and $Cg(K', K)$ to compare the complexity of both methods.

It is easy to check that there is an optimal tile size $K'_{\text{opt}}$ for any given kernel size $K$ to achieve the minimal $g(K', K)$. Figure 4 shows the optimal tile size $K'_{\text{opt}}$ as a function of $K$, which indicates that $K'_{\text{opt}}$ is almost proportional to $K$. For the range that is shown, we have $K'_{\text{opt}} = 20K$. It is the first time that show there is an optimal tile size for dense simulation method, which is important for us to fine tune dense simulator to achieve the fastest speed.

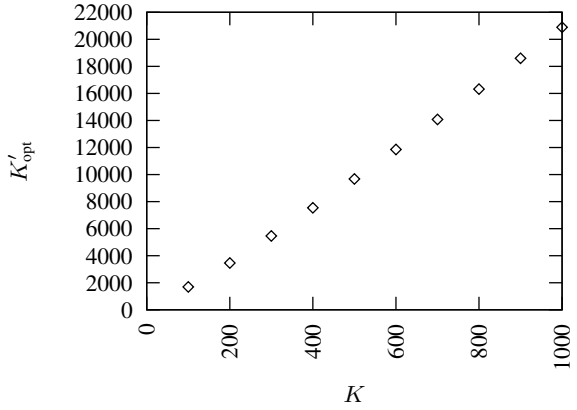



Figure 4. $K'_{\text{opt}}$ as a function of the kernel size $K$.

Figure 5 shows $Cg(K'_{\text{opt}}(K), K)$ as a function of $K$, where we take $C = 4$ and $v = 1/400$ for illustration purposes. We consider the extremely dense simulation case ($s = 1$). It is easy to see that

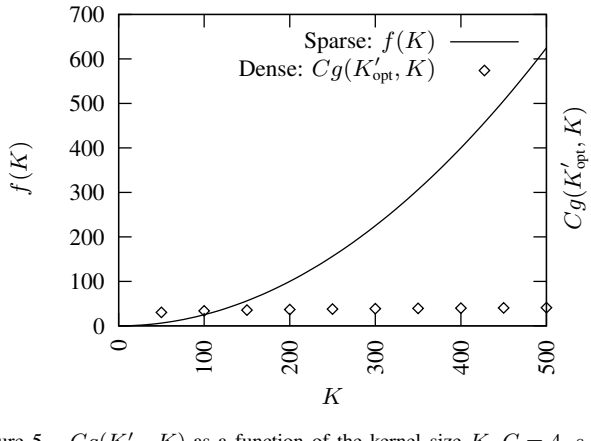



Figure 5. $Cg(K'_{\text{opt}}, K)$ as a function of the kernel size $K$. $C = 4$, $s = 1$ and $v = 1/400$.

dense simulation method will become better than sparse simulation method as for big $K$. As the technology node decreases, bigger kernel size ($K$) is needed for better model accuracy [21]. Using etch models could also increase the kernel size a lot [25]. Dense simulation method is preferred in these cases.

## IV. Topological Invariant Pixel Base Mask Shape Operations with Lithographic Considerations

We discuss the connectivity of binary images and define the *topological invariance*, and list all the single pixel mask operations that preserve the shape topology. We further eliminate the operations which could produce lithography unfriendly shapes. The operations defined here will be used in TIP-OPC.

### A. Connectivity and Topological Equivalence

A lot of papers have been published on thinning of digital image, where some operations are defined to maintain shape topologies [26]. We will modify those operations to fit our specific needs.

A pixel $p$ can be gray (representing value 1) or white (representing value 0). A gray pixel can be turned to white and a white pixel can be turned to gray. These operations are called flipping-off and flipping-on of that pixel.

**Definition 1** (Universal set and complement)**.** A universe set is defined as the set of all the pixels under consideration. For an image of size $m \times n$, the universe set can be written as

$$\mathfrak{U} = \{(i, j) | 0 \leqslant i < m \text{ and } 0 \leqslant j < n\}. \tag{11}$$

Any set of shapes can be denoted as $\mathfrak{M}$,

$$\mathfrak{M} = \{(i, j) | (i, j) \text{ is inside the shapes}\}. \tag{12}$$

Its complement $\overline{\mathfrak{M}}$ is defined as

$$\overline{\mathfrak{M}} = \mathfrak{U} \setminus \mathfrak{M}, \tag{13}$$

where "\" denotes the set subtraction.

*Remark.* The universal set shall be understand as the whole mask region in our problem.

**Definition 2** (Neighbor pixels)**.** As shown in Figure 6, the pixels $x_1, x_3, x_5, x_7$ are the 4-neighbors of the pixel $p$. The pixels $x_1, x_2, \ldots, x_8$ are the 8-neighbors of $p$.

| | | |
|---|---|---|
| $x_4$ | $x_3$ | $x_2$ |
| $x_5$ | $p$ | $x_1$ |
| $x_6$ | $x_7$ | $x_8$ |

Figure 6. Pixels in the neighborhood of $p$. $x_i$ ($i = 1, 3, 5, 7$) are 4-neighbors of $p$. $x_i$ ($i = 1, \ldots, 8$) are 8-neighbors of $p$.

**Definition 3** (Boundary pixels)**.** A 4- (or 8-) boundary pixel is a pixel with at least one 4- (or 8-) neighbor pixel having a different color. We call a boundary pixel an *inner boundary* pixel when it is gray, and a boundary pixel an *outer boundary* pixel when it is white. For a gray pixel set $\mathfrak{M}$, we denote the set of all its outer boundary pixels as $\beta_c^+(\mathfrak{M})$ and the set of all its inner boundary pixels as $\beta_c^-(\mathfrak{M})$, where $c = 4$ or 8. We denote the union of them as $\beta_c(\mathfrak{M}) = \beta_c^+(\mathfrak{M}) \cup \beta_c^-(\mathfrak{M})$, called the 4- (or 8-) boundary of $\mathfrak{M}$.

**Definition 4** (Path)**.** A sequence of pixels $y_1, y_2, \ldots, y_n$ is called a 4- (or 8-) path if $y_{i+1}$ is a 4- (or 8-) neighbor of $y_i$, $i = 1, 2, \ldots, n-1$.

**Definition 5** (Connectivity)**.** A pair of pixels $x, y$ in a subset $\mathfrak{S}$ of a pixel set $\mathfrak{M}$ are 4- (or 8-) connected if there exists a 4- (or 8-) path from $x$ to $y$ consisting of pixels in $\mathfrak{S}$ only. The subset $\mathfrak{S}$ is 4- (or 8-) connected if every pair of pixels $x, y$ in $\mathfrak{S}$ are connected. In this case, $\mathfrak{S}$ is said to be a 4- (or 8-) component of $\mathfrak{M}$. The number of 4- (or 8-) components of $\mathfrak{M}$ is called the degree of 4- (or 8-) connectivity of $\mathfrak{M}$, denoted as $D_4(\mathfrak{M})$ (or $D_8(\mathfrak{M})$).

It has been suggested that connectivities for $\mathfrak{S}$ and its complement $\overline{\mathfrak{S}}$ should be different to avoid the paradoxes of $\mathfrak{S}$ and $\overline{\mathfrak{S}}$ being both connected or both disconnected [27]. For example, in Figure 7, if the "diamond" loop (consists of all the gray pixels) is connected, the white pixels should be dissected into two components, intuitively. Otherwise, all the white pixels shall be connected. However, if we

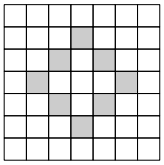


Figure 7. The connectivity paradox (taken from [27]).

choose 4-connectivity for both the gray and white pixels, we end up with that the pixels on the loop are disconnected and its interior region is also disconnected from its exterior region. If we choose 8-connectivity for both the gray and white pixels, then all the pixels on the loop are connected and all white pixels are connected. To avoid this paradox, we choose 4-connectivity for gray pixels and 8-connectivity for white pixels, which makes two intuitively disconnected mask shapes $\mathfrak{M}_1$ and $\mathfrak{M}_2$ in Figure 8 really disconnected. We will define the *topological equivalence* based on the above choice of connectivities.

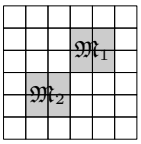



Figure 8. 4-connectivity is chosen for gray pixels to make $\mathfrak{M}_1$ and $\mathfrak{M}_2$ disconnected. Hence, 8-connectivity is chosen for white pixels.

**Definition 6** (Topological Equivalence)**.** Suppose we have two pixel sets $\mathfrak{X}$ and $\mathfrak{Y}$. If we can find a sequence of pixel sets $\mathfrak{M}_1, \ldots, \mathfrak{M}_n$ ($\mathfrak{M}_1 = \mathfrak{X}$ and $\mathfrak{M}_n = \mathfrak{Y}$), which satisfy

- $\mathfrak{M}_i \neq \mathfrak{M}_{i+1}, \quad i = 1, \ldots, n-1$;
- there exists some $x_i$ such that

$$\mathfrak{M}_i \cup x_i = \mathfrak{M}_{i+1} \quad \text{or} \quad \mathfrak{M}_i \setminus x_i = \mathfrak{M}_{i+1}, \quad i = 1, \ldots, n-1;$$

- $D_4(\mathfrak{M}_i) = D_4(\mathfrak{M}_j)$ and $D_8(\overline{\mathfrak{M}_i}) = D_8(\overline{\mathfrak{M}_j})$ for any $i, j$,

we call $\mathfrak{X}$ and $\mathfrak{Y}$ are topological equivalent.

### B. Topological Invariant Mask Operations

**Definition 7** (Removable and insertable)**.** Suppose $\mathfrak{M}$ is a gray pixel set and its complement $\overline{\mathfrak{M}}$ is a white pixel set. A gray pixel $p$ is removable if flipping it off does not change the degree of 4-connectivity of $\mathfrak{M}$, $D_4(\mathfrak{M})$, and the degree of 8-connectivity of $\overline{\mathfrak{M}}$, $D_8(\overline{\mathfrak{M}})$. A white pixel $p$ is insertable if flipping it on does not change $D_4(\mathfrak{M})$ and $D_8(\overline{\mathfrak{M}})$.

*Remark.* Flipping a removable or insertable pixel at one time does not change mask topology. According to Definition 6, we can sequentially apply these operations without changing the mask topology.

We will show below what pixels are removable or insertable pixels. Flipping on or off a pixel could only affect the connectivity of its 8-neighbors. Therefore, we only need to examine all the $3 \times 3$ pixel patterns (totally $2^{3\times 3} = 512$ patterns).

In Figure 9, we list all the removable and insertable cases. In this figure, "x" denote a pixel that could be gray or white and "?" is the pixel under consideration. It is easy to check that flipping the "?" pixel for each cases does not change the degree of connectivity for gray regions and white regions. These cases under rotation by $90^\circ$, $180^\circ$ and $270^\circ$ axes are still removable or insertable. Therefore, there are totally $(1 + 2^2 + 2^3 + 2^4) \times 4 \times 2 = 232$ cases.

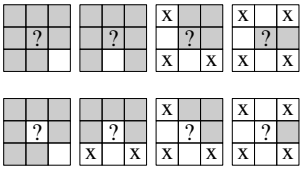


Figure 9. Removable (upper row) and insertable (lower row) pixels (denoted by "?"). There are totally 232 cases.

### C. Lithographic Considerations

Because the optical lens is a low-pass filter, the high frequency components associated with fine features can not pass the lens. Therefore, the sharp corners are not necessary from the lithography imaging point of view. For example, we do not want those patterns in Figure 10.

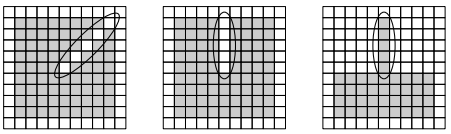


Figure 10. Lithography non-friendly features (marked by ellipses).

To circumvent the creation of these three pattens, we do not allow the following three removable or insertable pixels shown in Figure 11, respectively. We also do not allow the cases under rotation by $90^\circ$, $180^\circ$ and $270^\circ$. Therefore, there are totally $(1 + 1 + 2^2) \times 4 = 24$ cases not allowed.

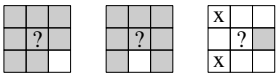


Figure 11. Not allowed removable (left and middle) and insertable pixels (right) (denoted by "?") due to lithographic considerations. There are totally 24 cases.

Deleting the cases in Figure 11 from Figure 9, we end up with removable and insertable pixels that are lithography friendly in Figure 12. These cases under rotation by $90^\circ$, $180^\circ$ and $270^\circ$, or reflection about $x$ or $y$ axes are still in this category. Therefore, there are totally $(((2^2-1)+2^3+2^4)+1+2^2+2^3+(2^4-2^2)) \times 4 = 208$ cases. It is easy to confirm that flipping the pixel marked with "?"

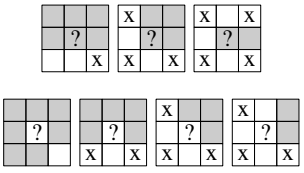


Figure 12. Lithography friendly removable (upper row) and insertable (lower row) pixels, marked by "?". There are totally 208 cases.

does not change the topology, and it will not create any fine features like those in Figure 10.

Since the center pixels in Figure 12 are all boundary pixels, we call them *lithography friendly topological invariant boundary* pixels. We denote the set of inner boundary pixels as $\gamma^-(\mathfrak{M})$ and $\gamma^+(\mathfrak{M})$, respectively. TIP-OPC (Section V) only flips those pixels. Thus, the mask shape topology is maintained. And we guarantee that no lithography unfriendly pattern is generated.

## V. Topological Invariant Pixel Based OPC (TIP-OPC)

In this section, we propose a contour based cost function. Then we show an efficient method to compute the derivative of the cost function with respect to mask changes. TIP-OPC uses this information to flip the mask pixels.

### A. TIP-OPC Cost Function

All the ILT approaches [7, 8, 11, 13] minimize a cost function $F[m]$,

$$\min_m F[m], \tag{14}$$

where $m$ is the mask transmission function, which can take any value between 0 and 1. The cost function $F[m]$ has the following form,

$$F[m] = P_{\text{I}}(I) + P_{\text{c}}(m) + P_{\text{tr}}(m), \tag{15}$$

where $I$ denotes the image intensity distribution.

The first term $P_{\text{I}}(I)$ in (15) penalizes the print image intensity $I$ when it is different from the desired intensity distribution. However, in practice we only care about the printed contour instead of the intensity distribution. Two different contours might have the same $P_{\text{I}}(I)$, and two different intensity distribution (with different $P_{\text{I}}(I)$) might have the same contours. To eliminate this mismatch between contours and intensity distributions, we define the contour based cost function for TIP-OPC as

$$F[M] \equiv \text{area}(R \triangle \widehat{R}), \tag{16}$$

where $M$ is the mask shapes, $R$ is the printed region and $\widehat{R}$ is the target region. Since the contour is explicitly expressed in the above cost function, we can expect its solution is better than the solution from the cost function in (15) in terms of contour fidelity.

The second term $P_{\text{c}}(m)$ in (15) penalizes complexity masks. The third term $P_{\text{tr}}(m)$ in (15) favors masks with certain transmission values (e.g., 0 and 1 for binary masks). Since TIP-OPC maintains the mask shape topology and generates mask pixels with transmission value of 0 and 1 only, it does not need these terms.

The new metric $\text{area}(R \triangle \widehat{R})$ is also better than the Edge Placement Error (EPE) metric, commonly used in MB-OPC algorithm for contour fidelity [17]. EPE which denotes the distance between the target contour and the printed contour at the tagging points. Since EPEs are measured sparsely, making them all zero can not guarantee the printed contour and the target contour the same as shown in Figure 13. But $\text{area}(R \triangle \widehat{R})$ is zero if and only if $R = \widehat{R}$.

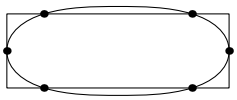


Figure 13. EPEs are zero at the tagging points (dots). But the two contours are different.

### B. Efficient Computation of the Cost Sensitivity

We will use the sensitivity of the cost function $F$ with respect to mask changes to guide TIP-OPC. A naive way to compute the sensitivity is to flip each mask pixel and directly compute the new cost function $F$ and check how much $F$ changes. But this is too slow to be used in practice. In the following, we derive a fast way to compute the sensitivity using convolutions, which can be computed by FFT efficiently.

The change of $F$ with respect to the change of the intensity can be derived as

$$\delta F[M] = \delta \,\text{area}(R \triangle \widehat{R}) = \oint_{\partial R} \frac{1_{\overline{\widehat{R}}} - 1_{\widehat{R}}}{\sqrt{\left(\frac{\partial I}{\partial x}\right)^2 + \left(\frac{\partial I}{\partial y}\right)^2}} \delta I \mathrm{d}s \tag{17}$$

where $\overline{\widehat{R}}$ is the complement of $\widehat{R}$ and $1_A$ is the indicator function of the set $A$ defined as

$$\mathbf{1}_A(x) = \begin{cases} 1 & \text{if } x \in A, \\ 0 & \text{if } x \notin A. \end{cases}.$$

According to (17), the cost $F$ is reduced if the intensity on the in-target printed contour decreases; otherwise, $F$ is increased. The change rate is inversely proportional to the intensity slope.

To enable numerical computation, we write the pixel based version of (17) as

$$\delta F[\mathfrak{M}] = \iint_{\beta_4(\mathfrak{R})} \frac{1_{\overline{\widehat{\mathfrak{R}}} \backslash \beta_4^+(\widehat{\mathfrak{R}})} - 1_{\widehat{\mathfrak{R}} \backslash \beta_4^-(\widehat{\mathfrak{R}})}}{\sqrt{\left(\frac{\partial I}{\partial x}\right)^2 + \left(\frac{\partial I}{\partial y}\right)^2}} \delta I \mathrm{d}x\mathrm{d}y. \tag{18}$$

where the path integral is replace by an integral over the print image boundary pixels $\beta_4(\mathfrak{R})$. We do not use any pixels on the target boundary $\beta_4^+(\widehat{\mathfrak{R}})$ and $\beta_4^-(\widehat{\mathfrak{R}})$ due to the discontinuity of the sensitive when $\mathfrak{R}$ and $\widehat{\mathfrak{R}}$ coincide.

By convoluting the mask transmission function $m$ with the real and imaginary parts of $h_p$ in (2) separately, we can rewrite (2) in the following form

$$I[m] = \sum_{p=0}^{2P-1} \sigma_p (h_p * m)^2, \tag{19}$$

where there are $2P$ terms and $h_p$ is real ($h_p$ here is different from $h_p$ of (2)). Differentiating $I$ with respect to $m$ and plugging it in (18), we can derive the sensitivity

$$\frac{\delta F}{\delta m} = 2 \sum_{p=0}^{2P-1} \sigma_p \left( \frac{1_{\beta_4(\mathfrak{R})}(1_{\overline{\widehat{\mathfrak{R}}} \backslash \beta_4^+(\widehat{\mathfrak{R}})} - 1_{\widehat{\mathfrak{R}} \backslash \beta_4^-(\widehat{\mathfrak{R}})})(h_p * m)}{\sqrt{\left(\frac{\partial I}{\partial x}\right)^2 + \left(\frac{\partial I}{\partial y}\right)^2}} * P h_p \right), \tag{20}$$

where $P$ is an operator defined as

$$Pw(x, y) = w(-x, -y). \tag{21}$$

The convolutions $h_p * m$'s are the intermediate results of the intensity computation through the dense simulation method (3), which do not incur additional runtime since we compute the intensity anyway. Since (20) consists of convolutions, we can still compute it for all the mask pixels using FFT efficiently. TIP-OPC uses this sensitivity information to effectively reduce the cost function $F$.

### C. The Overall TIP-OPC algorithm

The TIP-OPC algorithm (Algorithm 1) is an iterative algorithm which successively modify the mask $\mathfrak{M}$ using the mask operations defined in Section IV to reduce the cost function $F[\mathfrak{M}]$. The algorithm stops if no improvement can be made in the cost function.

Since TIP-OPC flips on mask pixels in the topological invariant outer boundary $\gamma^+(\mathfrak{M})$ and flips off mask pixels in the topological invariant outer boundary $\gamma^-(\mathfrak{M})$, we only need $\frac{\delta F}{\delta m}$ on $\gamma^-(\mathfrak{M}) \cup \gamma^+(\mathfrak{M})$. It is easy to see the following facts based on the definition of $\frac{\delta F}{\delta m}$ and that only two mask transmission values 0 and 1 are allowed:

- A gray pixel can be flipped off, if it has positive value of $\frac{\delta F}{\delta m}$.
- A white pixel can be flipped on, if it has negative value of $\frac{\delta F}{\delta m}$.

**Algorithm 1** TIP-OPC algorithm

```
1: function TIPOPC(R̂)
2:     M ← R̂ // initialize mask to target
3:     repeat
4:         compute δF/δm
5:         a ← max( max_{p∈γ⁻(M)} δF/δm, − min_{p∈γ⁺(M)} δF/δm )
6:         if a ⩽ 0 then
7:             break
8:         b ← a/2
9:         failed ← true
10:        for i ← 1, n do
11:            M′ ← AdjustMask(M, a, b)
12:            if cost is reduced then
13:                failed ← false
14:                M ← M′
15:                break
16:            else
17:                b ← b/2
18:    until failed
```

We call the absolute value $\frac{\delta F}{\delta m}$ in the above two cases as the *useful* value of $\frac{\delta F}{\delta m}$. Line 5 in Algorithm 1 computes the maximum useful $\frac{\delta F}{\delta m}$ (denote as $a$ in the algorithm). The following of the algorithm modifies the pixels whose useful $\frac{\delta F}{\delta m}$ are between $(a/2, a]$, $(3a/4, a]$, $(7a/8, a]$, ..., until the cost function gets improved or failed even after $n$ trials. In the later case, the algorithm stops.

Algorithm 2 shows how the mask is adjusted to maintain the topology invariance. Note that $\mathfrak{M}'$ and $\mathfrak{M}$ have the same topology at the beginning. Each change in Line 6 maintains $\mathfrak{M}'$'s topology. Therefore, at the end of the program $\mathfrak{M}'$ is still topologically equivalent to the input $\mathfrak{M}$. Hence, the topology of $\mathfrak{M}$ is maintained.

**Algorithm 2** Mask Adjustment algorithm

```
1: function AdjustMask(M, a, b)
2:     M′ ← M // make a copy
3:     for pixel p ∈ γ⁺(M) ∪ γ⁻(M) do
4:         if p's useful δF/δm is in (a − b, a] then
5:             if flip p on M′ if it does not change M′'s topology
   then
6:                 flip p on M′
7:     return M′
```

## VI. Experimental Results

We implemented both the sparse and dense lithography simulators, the current commonly used MB-OPC algorithm described in [28], and the TIP-OPC algorithm described in this paper using C++. The grid size of $\delta = 5\,\mathrm{nm}$ was used for the following experiments.

Based on the lithography friendly topological invariant mask operations, no SRAFs and excessively small features are created. Therefore, it is guaranteed that TIP-OPC generates less complex masks than ILT by theory, and there is no need to compare them experimentally.

### A. Runtime Comparison of Sparse and Dense Simulations

We used both sparse and dense sparse simulation methods on a $114 \times 114$ via array to simulate the aerial image on all the grid points. Each via was of size $100\,\mathrm{nm} \times 100\,\mathrm{nm}$. The pitches in both $x$ and $y$-directions were $200\,\mathrm{nm}$. We took $K = 240$ and $K' = 20 \times K = 4800$. We employed $P = 6$ kernels in the simulations.

The runtime for sparse and dense simulation methods were $1.99 \times 10^3\,\mathrm{sec}$ and $70.8\,\mathrm{sec}$. We can see that the sparse simulation method can be much slower than the dense simulation method for dense simulation applications.

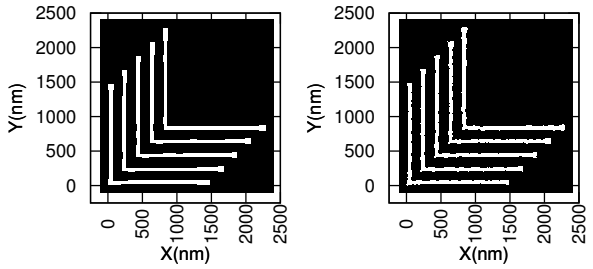
### B. Quality Comparison of MB-OPC and TIP-OPC

We used quadrupole illumination with illumination parameters $\sigma_{\text{center}} = 0.85$ and $\sigma_{\text{radius}} = 0.2$. The numerical aperture was $\mathrm{NA} = 0.8$, the wavelength was $\lambda = 193\,\mathrm{nm}$ and the intensity threshold was $I_{\text{th}} = 0.1$. We employed $P = 8$ kernels in the simulations. We took $n = 10$ in TIP-OPC. For MB-OPC, we used the segment length of $100\,\mathrm{nm}$.

The MB-OPC and TIP-OPC results of 6 test patterns across $130\,\mathrm{nm}$, $90\,\mathrm{nm}$ and $65\,\mathrm{nm}$ technology nodes are shown in Table I. These patterns are typical poly and metal 1 patterns. We used $\mathrm{area}(R \triangle \widehat{R})$ as the metric for the contour fidelity. The "total" columns denote $\mathrm{area}(\widehat{R})$ in the unit of pixel. The errors are in the unit of pixel, as well. The "ratio" columns denote the relative error $\mathrm{area}(R \triangle \widehat{R}) / \mathrm{area}(\widehat{R})$. The "Reduction" columns show the reduction in the relative error between MB-OPC and TIP-OPC. The average reductions are shown for each technology generation. As we can see clearly from the table as technology scales down, TIP-OPC give more improvement in terms of relative error reduction. Thus, it is indeed needed in the future generation when the error budget become more stringent.

Note that as technology scales down, the relative error in TIP-OPC also goes up. This does not mean TIP-OPC does not do a good job. No OPC can make corner printed perfectly since the lithography optical system can not pass the high frequency components associated with the corners. As the technology scales, the percent of the corner region increases. Therefore, we see an increase in the relative error for TIP-OPC, as well. However, TIP-OPC is in fact better than any MB-OPC algorithm. The errors reflect the fact that no other OPC algorithm can correct those residue errors, which means that the layouts shall be modified to be more OPC friendly if we want the error be reduced further.

Due to space limitations, we only show the OPCed mask for the pattern "pat2" (five-jog pattern) in Figure 14. As we can see TIP-OPC maintains topologies and do not create any SRAFs. TIP-OPC also does not create any lithography unfriendly mask features. Higher contour fidelity and low mask complexity demonstrate that TIP-OPC is indeed a promising method.




(a) MB-OPC (b) TIP-OPC

Figure 14. OPCed mask of the pattern "pat2".

Table I
OPC QUALITY COMPARISON

| 130nm | | | | | | |
|---|---|---|---|---|---|---|
| pattern | total | MB-OPC | | TIP-OPC | | Reduction |
| | | error | ratio | error | ratio | |
| pat1 | 48216 | 3780 | 7.84% | 1053 | 2.18% | 5.66% |
| pat2 | 227200 | 15615 | 6.87% | 1914 | 0.84% | 6.03% |
| pat3 | 38130 | 2833 | 7.43% | 720 | 1.89% | 5.54% |
| pat4 | 53360 | 4021 | 7.54% | 1177 | 2.21% | 5.33% |
| pat5 | 42255 | 3172 | 7.51% | 799 | 1.89% | 5.62% |
| pat6 | 58860 | 4493 | 7.64% | 1225 | 2.08% | 5.55% |
| average | | | | | | 5.62% |

| 90nm | | | | | | |
|---|---|---|---|---|---|---|
| pattern | total | MB-OPC | | TIP-OPC | | Reduction |
| | | error | ratio | error | ratio | |
| pat1 | 24360 | 2126 | 8.73% | 797 | 3.27% | 5.46% |
| pat2 | 113680 | 7113 | 6.26% | 1806 | 1.59% | 4.67% |
| pat3 | 19533 | 2796 | 14.31% | 1176 | 6.02% | 8.29% |
| pat4 | 27294 | 3647 | 13.36% | 1136 | 4.16% | 9.20% |
| pat5 | 21714 | 2934 | 13.51% | 1161 | 5.35% | 8.17% |
| pat6 | 30252 | 4088 | 13.51% | 2023 | 6.69% | 6.83% |
| average | | | | | | 7.10% |

| 65nm | | | | | | |
|---|---|---|---|---|---|---|
| pattern | total | MB-OPC | | TIP-OPC | | Reduction |
| | | error | ratio | error | ratio | |
| pat1 | 12600 | 1609 | 12.77% | 788 | 6.25% | 6.52% |
| pat2 | 58000 | 5176 | 8.92% | 2091 | 3.61% | 5.32% |
| pat3 | 10026 | 3618 | 36.09% | 2466 | 24.60% | 11.49% |
| pat4 | 13968 | 5314 | 38.04% | 2642 | 18.91% | 19.13% |
| pat5 | 11118 | 3813 | 34.30% | 2480 | 22.31% | 11.99% |
| pat6 | 15424 | 5703 | 36.97% | 3322 | 21.54% | 15.44% |
| average | | | | | | 11.65% |

## VII. CONCLUSIONS

In this paper, we proposed the topological invariant pixel based OPC (TIP-OPC) paradigm. To reduce mask complexity that could result from ILT, the current pixel based approach, we maintain the mask shape topology and do not allow lithography unfriendly mask patterns. Our TIP-OPC algorithm has efficient lithography friendly topologically invariant mask operations and FFT based cost-to-mask sensitivity computation. Our experimental results show that TIP-OPC is much better than MB-OPC in terms of contour fidelity. Since the mask shapes are topologically invariant, the resulting mask has relatively low mask complexity compared with ILT.

## ACKNOWLEDGMENTS

This work is partially supported by SRC, IBM Faculty Award, Fujitsu, Qualcomm, Sun, and Intel equipment and KLA-Tencor software donations. The authors would like to thank Mr. Wing-Chi Poon from UT Austin Computer Science Department for helpful discussions.

## REFERENCES

[1] N. B. Cobb and A. Zakhor, "Fast, low-complexity mask design," in *Proc. SPIE 2440*, May 1995, pp. 313–327.

[2] N. B. Cobb, A. Zakhor, and E. Miloslavsky, "Mathematical and CAD framework for proximity correction," in *Proc. SPIE 2726*, Jun. 1996, pp. 208–222.

[3] N. B. Cobb, "Fast Optical and Process Proximity Correction Algorithms for Integrated Circuit Manufacturing," Ph.D. dissertation, University of California at Berkeley, 1998.

[4] N. Cobb and Y. Granik, "New concepts in OPC," in *Proc. SPIE 5377*, 2004, pp. 680–690.

[5] C.-Y. Hung, B. Zhang, D. Tang, E. Guo, L. Pang, Y. Liu, A. Moore, and K. Wang, "First 65nm tape-out using inverse lithography technology (ILT)," in *Proc. SPIE 5992*, Nov. 2005, pp. 596–604.

[6] Y. Liu, D. Abrams, L. Pang, and A. Moore, "Inverse lithography technology principles in practice: unintuitive patterns," in *Proc. SPIE 5992*, Nov. 2005, pp. 886–893.

[7] Y. Granik, "Solving inverse problems of optical microlithography," in *Proc. SPIE 5754*, May 2005, pp. 506–526.

[8] Y. Granik, "Fast pixel-based mask optimization for inverse lithography," *Journal of Microlithography, Microfabrication and Microsystems*, vol. 5, no. 4, p. 043002, Dec. 2006.

[9] D. S. Abrams and L. Pang, "Fast inverse lithography technology," in *Proc. SPIE 6154*, Apr. 2006, pp. 534–542.

[10] D. Van Den Broeke, M. Hsu, J. F. Chen, S. Hsu, U. Hollerbach, and T. Laidig, "Manufacturing implementation of IML™ technology for 45nm node contact masks," in *Proc. SPIE 6283*, Jun. 2006.

[11] A. Poonawala and P. Milanfar, "Mask Design For Optical Microlithography — An Inverse Imaging Problem," *IEEE Trans. on Image Processing*, vol. 16, no. 3, pp. 774–788, Mar. 2007.

[12] L. Pang, Y. Liu, and D. Abrams, "Inverse Lithography Technology (ILT): what is the impact to the photomask industry?" in *Proc. SPIE 6283*, Jun. 2006.

[13] A. Poonawala and P. Milanfar, "OPC and PSM design using inverse lithography: a nonlinear optimization approach," in *Proc. SPIE 6154*, Apr. 2006, pp. 1159–1172.

[14] M. Born and E. Wolf, *Principles of Optics : Electromagnetic Theory of Propagation, Interference and Diffraction of Light*, 7th ed. Cambridge University Press, Oct. 1999.

[15] J. Mitra, P. Yu, and D. Z. Pan, "RADAR: RET-aware detailed routing using fast lithography simulations," in *Proc. Design Automation Conf.*, 2005, pp. 369–372.

[16] P. Yu, D. Z. Pan, and C. A. Mack, "Fast lithography simulation under focus variations for OPC and layout optimizations," in *Proc. SPIE 6156*, Apr. 2006, pp. 397–406.

[17] P. Yu, S. X. Shi, and D. Z. Pan, "Process Variation Aware OPC with Variational Lithography Modeling," in *Proc. Design Automation Conf.*, 2006, pp. 785–790.

[18] ——, "True Process Variation Aware Optical Proximity Correction with Variational Lithography Modeling and Model Calibration," *Journal of Microlithography, Microfabrication and Microsystems*, vol. 6, no. 3, Jul.–Sep. 2007.

[19] N. Cobb, "Flexible sparse and dense OPC algorithms," in *Proc. SPIE 5853*, Jun. 2005, pp. 693–702.

[20] N. B. Cobb and Y. Granik, "Dense OPC for 65nm and below," in *Proc. SPIE 5992*, Nov. 2005, pp. 1521–1532.

[21] N. Cobb and D. Dudau, "Dense OPC and verification for 45nm," in *Proc. SPIE 6154*, Apr. 2006, pp. 191–196.

[22] Y. Granik, D. M. Medvedev, and N. Cobb, "Toward standard process models for OPC," in *Proc. SPIE 6520*, Apr. 2007.

[23] S. W. Smith, *The Scientist & Engineer's Guide to Digital Signal Processing*, 1st ed. California Technical Pub., 1997, ch. 18.

[24] T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein, *Introduction to Algorithms*. MIT Press, 2001.

[25] Y. Granik, "Dry etch proximity modeling in mask fabrication," in *Proc. SPIE 5130*, Aug. 2003, pp. 86–91.

[26] L. Lam, S.-W. Lee, and C. Y. Suen, "Thinning methodologies-a comprehensive survey," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 14, no. 9, pp. 869–885, 1992.

[27] A. Rosenfeld and J. L. Pfaltz, "Sequential operations in digital picture processing," *J. ACM*, vol. 13, no. 4, pp. 471–494, 1966.

[28] N. B. Cobb and Y. Granik, "Model-based OPC using the MEEF matrix," in *Proc. SPIE 4889*, Dec. 2002, pp. 1281–1292.