A sampling problem from lithography for chip layout

Eric Cator[*] *Tammo Jan Dijkema*[†] *Michiel Hochstenbach*[‡]
Wouter Mulckhuyse[§] *Mark Peletier*[‡] *Georg Prokert*[‡]
Wemke van der Weij[¶] *Daniël Worm*[||]

Abstract

Given a list with simple polygons and a sampling grid geometry, we calculate the sample (greyscale) value of each grid pixel, such that the residual error in a certain norm is sufficiently small, taking into account that the amount of computational operations should be minimal.

Key words: sampling, Fast Fourier Transform, computational operations.

1 Introduction

1.1 Lithography

ASML is the worldwide leader in *lithographic techniques* for the semiconductor industry. Since the different steps in the lithography process are important for the discussion of this report, we describe them in some detail.

The main function of the lithographic system of ASML is to expose a silicon wafer with a pattern of given light intensity. This exposure step is embedded in the following sequence:

1. A designer creates the design of the desired pattern in a CAD system;
2. The CAD system perturbs the design to pre-correct for deficiencies in the optical system (see below);
3. For each layer of the chip a mask is created;
4. If needed, the material for the current layer of the chip is deposited on the water;
5. A photoresist coating is added on the wafer
6. The lithographic system exposes the photoresist on the wafer with the mask for the current layer;
7. The photoresist is developed;

[*]Technische Universiteit Delft
[†]Universiteit Utrecht
[‡]Technische Universiteit Eindhoven
[§]ASML, Veldhoven, the Netherlands
[¶]CWI, Amsterdam
[||]Universiteit Leiden

8. Those parts of the wafer where the photoresist is gone are further treated (i.e. etched).
9. The remainder of the photoresist is removed.

Steps 4 to 9 may be repeated as many as forty times, thus creating a landscape on the wafer with details at many different heights.

Only step 6 takes place within the lithography machines, and therefore only this step is under control here; the rest of this list is to be treated as a given, and this will be important below.

1.2 Maskless lithography

A new development is the creation of lithography systems that use an array of microscopic mirrors instead of the mask. Step 3 above is then skipped, and the output of step 2 is fed directly into step 6. The mirrors are then positioned in such a way that the resulting illumination pattern on the wafer corresponds to the pattern that a mask-based optical system *would* create.

Maskless lithography will not be competitive with mask-based lithography for production machines, since mask-based systems have higher throughput. Maskless systems will have the advantage, however, of eliminating the costly and time-consuming step of creating the mask. Therefore maskless systems may be beneficial in the pre-production (testing) phase. Since the client should be able to switch from testing on maskless systems to production on mask-based systems without changing the pattern, it is essential that the maskless systems act as a *drop-in replacement* for mask-based systems. In other words,

the maskless system has to imitate a mask-based machine *with all its imperfections*.

Some of the constraints posed to the Study Group stem from this requirement.

1.3 The pattern

The pattern that is created in steps 1 and 2 is defined as a collection of polygons. In mathematical terms we will describe this pattern by the indicator function χ_P of the union of these polygons, i.e.

$$\chi_P(x) := \begin{cases} 1 & x \text{ lies in some polygon} \\ 0 & \text{otherwise.} \end{cases}$$

For reasons of computational efficiency it is important that the pattern is given by polygons as these can be described easily by a sequence of the coordinates of their vertices, in the order that defines their boundary.

1.4 The question

As part of the ‘data path’ that transforms the machine input (the output of step 2 above) into the steering signal for the array of mirrors, a sampling step is necessary. In this step the collection of polygons is transformed into an array of intensity values (in $[0, 1]$) of the same size as the array of mirrors. In this sampling step, *aliasing* is an important problem (see Section 2 and Figures 1 and 2), and the original question as posed to the Study Group was to design an efficient algorithm to perform this sampling step with sufficient control on aliasing. In Figure 1 a part of a chip layout is shown. Clearly visible are the perturbations around corners that are introduced to correct for phenomena occurring in a mask-based machine. Figure 2 shows the desired rasterization of this chip layout. Gray values represent the intensity at which a square should be exposed.

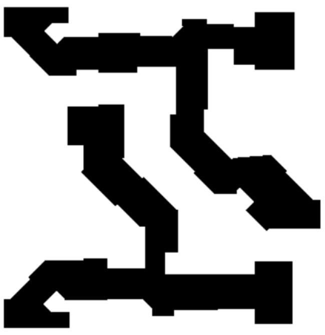

Figure 1: Part of a chip layout.

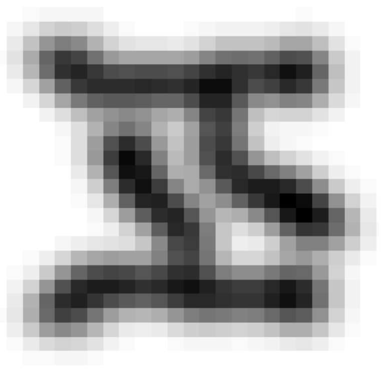

Figure 2: The desired rasterization of this chip layout.

2 Sampling, low-pass filters and aliasing

A well-known error source occurring in the processing of sampled signals is known as *aliasing*. In our context, it is best understood theoretically from the simple observation that sampling and (low-pass) filtering of a signal do not commute. To see this, interpret sampling as multiplication of a given signal in the space domain by a so-called shah distribution or Dirac comb

$$\text{Ш}_a := \sum_{k\in\mathbb{Z}} \delta(\cdot - ak)$$

where $\delta(\cdot - ak)$ denotes a shifted delta distribution and $1/a$ is the sampling rate. It follows from the Poisson summation formula that up to a scaling factor, the Fourier transform (denoted by a hat) of a shah distribution is a shah distribution with inverse sampling rate:

$$\widehat{\text{Ш}}_a = \frac{1}{a}\text{Ш}_{\frac{1}{a}}.$$

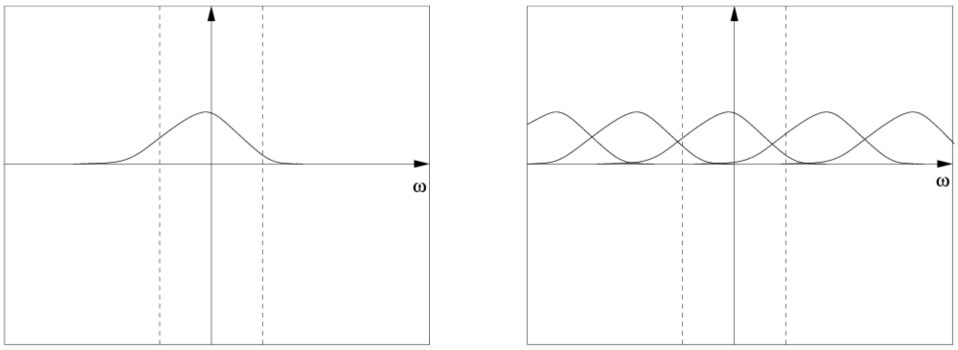

Figure 3: Spatial sampling in frequency domain representation (qualitatively). On the left the original signal is shown, on the right the resulting signal after sampling. This resulting signal consists of an (infinite) sum of scaled and translated copies of the original signal. The dashed lines show the low-pass filter bandwidth.

As pointwise multiplication in space domain corresponds to convolution in frequency domain, spatial sampling is represented in frequency domain by convolution with a shah distribution (Fig. 3). If the sampling rate is taken too small, spurious contributions of higher frequencies will appear in the low frequency band and persist even after applying a low-pass filter. For more details on sampling and the aliasing effect we refer to [1].

There are two possible remedies to avoid this:

1. use of a higher sampling rate
2. preprocessing of the input data to suppress high frequencies before sampling.

The first of these possibilities is impractical because of the prohibitively high computational effort. However, for a smaller model problem this approach can be used to obtain a "golden standard" for the purpose of comparison to results of more efficient algorithms.

The second possibility is known as applying an *anti-aliasing filter* and is used in practice by ASML. Schematically, the signal flow is described as in Fig. 2.

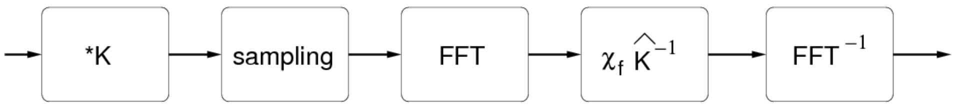

Figure 4: Signal flow scheme. The kernel of the anti-aliasing filter is denoted by K. In the frequency domain, the low-pass filter is applied by pointwise multiplication with the characteristic function χ_f of the set $\{|\omega| < f\}$. The application of the anti-aliasing kernel has to be corrected afterwards by pointwise multiplication by $\hat{K}^{-1}$.

As the main computational effort is in the anti-aliasing filter, it was the task of the study group to assess and, if possible, improve its efficiency. (It has to be remarked here that, of course, the anti-aliasing filter is a low-pass filter in itself, so one might expect either a high computational effort or aliasing effects for this filter as well. However, using the special structure of our data it is possible to achieve a higher efficiency than in a standard FFT-based algorithm.)

3 First approach: numerical convolution

In the case of a sample field of 100μm×50μm with $6{\cdot}10^6$ polygons of 20 vertices each and a grid pitch of 20nm×20nm, the computational effort of the method currently used by ASML is estimated as to be about $1.9{\cdot}10^{13}$ operations, half of them being multiplications.

We propose the following different approach: introduce a second grid which refines the original (coarse) grid by a factor N in both directions. We will denote the gridpoints of the finer grid by P_{ij} with the understanding that P_{ij} is a point of the coarse grid if both i and j are multiples of N.

1. For simplicity we assume that the fine grid has total height and width $2LH$ where H is the distance between two coarse grid points. Let K be the kernel of the anti-aliasing filter as above and set

$$K_{ij} := K\left(\frac{iH}{N}, \frac{jH}{N}\right) \qquad i, j \in \mathbb{Z}, \quad -NL \le i, j \le NL.$$

The calculation of these values has to be performed only once.

2. For all fine grid points P_{ij}, set

$$\chi_{ij} = \begin{cases} 1 & P_{ij} \text{ lies inside some polygon,} \\ 0 & \text{otherwise.} \end{cases}$$

To do this efficiently, one might use a triangulation of all polygons and determine for each of the resulting triangles the range of the indices i, j of points inside the triangle.

3. The convolution in a coarse grid point $P_{Nk,Nl}$ can now be approximated by

$$(K * \chi_P)(P_{Nk,Nl}) \approx \sum_{i,j=-NL}^{NL} K_{ij}\chi_{Nk+i,Nl+j} \tag{1}$$

This approximation is extremely simple to implement and very flexible with respect to the choice of grid sizes and types of kernels used.

The necessary computational effort is dominated by the summations in (1) and is easily seen to be about $4N^2L^2$ additions per coarse grid point. The refinement factor N can be interpreted as an "oversampling rate", and experiments might be needed to determine its optimal value. When the algorithm described here is applied with $N = 100$ to the sample field described above, $3.2{\cdot}10^{13}$ additions are needed. This is in the same order of size as the analytic approach, when additions and multiplications are considered to have the same computational cost.

In Figure 5, the difference is illustrated between an oversampling factor of $N = 50$ and $N = 100$. This figure indicates that the difference between an oversampling factor of 50 and one of 100 introduces only a small error. However, this error is mostly located at the boundary of polygons (where an error causes most damage to the chip)

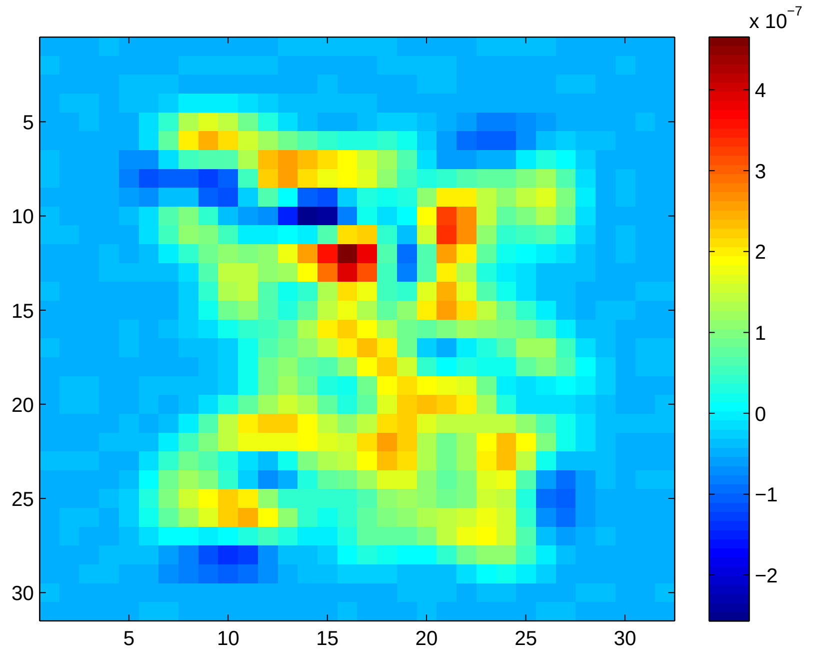

Figure 5: Oversampling effects.

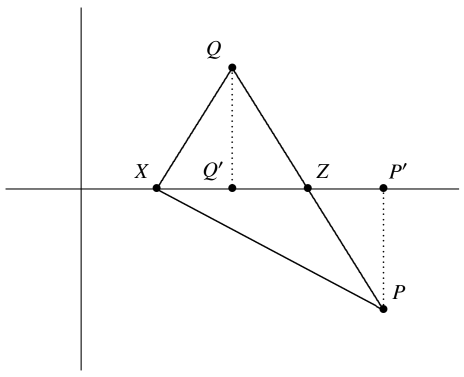

Figure 6: The triangle XPQ

4 Second approach: triangulation and look-up tables

As a second approach to the problem, we start with a triangulation of the data, so we are given the coordinates of the vertices of triangles $T_1, \ldots, T_N$. Furthermore, we will use a two-dimensional normal kernel K with fixed bandwidth σ, so

$$K(x,y) = \frac{1}{2\pi\sigma^2} e^{-\frac{1}{2}\frac{x^2+y^2}{\sigma^2}}.$$

The bandwidth σ is determined in such a way that the aliasing effect on the final coarse grid is negligible. Given a point $P \in \mathbb{R}^2$ of the coarse grid, we need to calculate the integral

$$I = \int_{T_i - P} K(x,y)dxdy$$

for each $1 \leq i \leq N$ such that T_i is close to P, in some precise sense.

The first step in approximating I is to make sure by applying a rotation that the triangle $T_i - P$ has at least one point on the (positive) x-axis. Because of the rotational symmetry of K, this will not change I. We denote this triangle by XPQ, where X is the point on the x-axis, and P and Q are the two other points.

4.1 Decomposing into right-angle triangles

We wish to decompose our triangle XPQ into four right-angle triangles. To this end, we define Z as the intersection of the line through PQ with the x-axis, P' as the projection of P onto the x-axis and Q' as the projection of Q onto the x-axis. If Z is not defined (or extremely far away), we will use a separate approach. Figure 6 depicts the situation.

Now consider XPQ as an oriented curve in $\mathbb{R}^2$, where the order of the letters determines the orientation. We know that we can rewrite I as an integral over this curve. However, we can also add up curves, and it is easy to check that

$$XPQ = XPP' + ZP'P + XQ'Q + ZQQ'.$$

Clearly, these four curves all correspond to right-angle triangles. To calculate I, we can therefore determine the integral of K over the four right-angle triangles, and determine whether these integrals should have a positive or a negative sign: if the orientation of XPP' is equal to the orientation of XPQ, the contribution of XPP' gets a positive sign, otherwise it gets a negative sign. The same holds for the other three right-angle triangles. Determining these orientations is a simple algebraic calculation.

4.2 Calculating the integral over right-angled triangles

We have reduced the problem of determining I to calculating the integral of K over a right-angle triangle with two vertices on the x-axis, among which is the vertex with the right angle. We can make sure by reflection in the y-axis that the x-coordinate x of the right angle is positive, so $x \geq 0$. Note that x either corresponds to P' or to Q'. Furthermore, by reflection in the x-axis, we can ensure that the point outside the x-axis has a positive y-coordinate, which we call $h \geq 0$. The x coordinate of the third point is given by $x + b$, where $b \in \mathbb{R}$. See Figure 7.

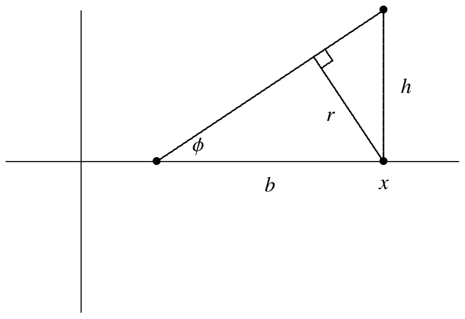

Figure 7: The right angle-triangle

We will use a different parametrization of the triangle (x, b, h), as is shown in Figure 7. We define

$$r = \frac{|b|h}{\sqrt{b^2 + h^2}} \quad \text{and} \quad \phi = \arctan\left(\frac{h}{b}\right) \in (-\pi/2, \pi/2).$$

Note that in Figure 7, $b < 0$ and $\phi < 0$. The advantage of this parametrization lies in the fact we only need to consider a bounded subset of the parameter space, since if (x, r, ϕ) lies outside this set, $I(x, r, \phi)$ will either be very small, or almost equal to $I(x', r', \phi')$, where $(x', r'\phi')$ is an element of this bounded set. To see this, choose $R > 0$ such that

$$\int_{\{u^2+v^2>R^2\}} K(u, v)\, dudv < \varepsilon_0,$$

where ε_0 is a small, fixed constant, corresponding to the accuracy with which we need to evaluate I. First suppose $x > R$. Then if $b > R - x$, we have $I(x, r, \phi) < \varepsilon_0$. Otherwise, we get

$$I\left(R, \frac{|b| - (x - R)}{|b|} r, \phi\right) < I(x, r, \phi) < I\left(R\frac{|b| - (x - R)}{|b|} r, \phi\right) + \varepsilon_0.$$

Now suppose $0 \leq x \leq R$ and fix $\phi \in (-\pi/2, \pi/2)$. We can define r_0 such that the hypotenuse of the triangle $I(x, r_0, \phi)$ is tangent to the circle with radius R. Clearly, if $r > r_0$, the set $I(x, r, \phi) \setminus I(x, r_0, \phi)$ falls outside of this circle. It is not hard to check that

$$r_0 = x - R\sin(\phi) \leq 2R.$$

So if $r > 2R$, we can choose $r = 2R$, and we will have

$$I(x, 2R, \phi) < I(x, r, \phi) < I(x, 2R, \phi) + \varepsilon_0.$$

4.3 Creating the "lookup table"

As we saw in the previous section, we only need to calculate $I(x, r, \phi)$ for $(x, r, \phi) \in [0, R] \times [0, 2R] \times (-\pi/2, \pi/2) := \Theta$. To do this, we define a grid on Θ, by dividing each of the three intervals in M parts, creating M^3 grid points. For all these points, we calculate I and also $\partial I/\partial x$, $\partial I/\partial r$ and $\partial I/\partial \phi$. This would allow us to calculate $I(x, r, \phi)$ by interpolation from the nearest grid point. Another possibility would be to just calculate I in all grid points, and use a weighted average over all nearby grid points as an approximation for $I(x, r\phi)$.

To calculate $I(x, r, \phi)$, it is easier to work in the parametrization (x, b, h), so

$$b = \frac{r}{\sin\phi} \quad \text{and} \quad h = \frac{r}{\cos\phi}.$$

Note that $\phi \approx 0$ is handled as an exception, since this means that the point Z is not well defined. We get, supposing that $b > 0$,

$$I = \frac{1}{2\pi\sigma^2} \int_x^{x+b} \int_0^{\frac{b+x-u}{b} h} e^{-\frac{1}{2}\frac{u^2+v^2}{\sigma^2}}\, dv du.$$

This means that

$$\begin{aligned}
\frac{\partial I}{\partial x} &= -\frac{1}{2\pi\sigma^2} \int_0^h e^{-\frac{1}{2}\frac{x^2+v^2}{\sigma^2}}\, dv + \frac{1}{2\pi\sigma^2}\frac{h}{b} \int_x^{x+b} e^{-\frac{1}{2}\frac{b^2u^2+(b+x-u)^2h^2}{b^2\sigma^2}}\, du, \\
\frac{\partial I}{\partial b} &= \frac{1}{2\pi\sigma^2} \int_x^{x+b} \frac{(u-x)h}{b^2} e^{-\frac{1}{2}\frac{b^2u^2+(b+x-u)^2h^2}{b^2\sigma^2}}\, du, \\
\frac{\partial I}{\partial h} &= \frac{1}{2\pi\sigma^2} \int_x^{x+b} \frac{b+x-u}{b} e^{-\frac{1}{2}\frac{b^2u^2+(b+x-u)^2h^2}{b^2\sigma^2}}\, du.
\end{aligned}$$

Using the formula

$$\frac{\partial I}{\partial r} = \frac{\partial I}{\partial b}\frac{\partial b}{\partial r} + \frac{\partial I}{\partial h}\frac{\partial h}{\partial r}$$

and its equivalent for $\partial I/\partial \phi$, we can create the entire lookup table.

If we have that Z is not well defined, meaning that PQ is parallel to the x-axis, we need to calculate the integral of K over the rectangle $P'PQQ'$ (and then subtracting the integral over $XP'P$ and XQQ'). Calling x the location of a point of the rectangle on the x-axis, $b \in \mathbb{R}$ the width and $h \geq 0$ the height, we can assume that $0 \leq x \leq R$, $b \in [-R, R]$ and $h \in [0, R]$, by neglecting anything outside the rectangle $[-R, R] \times [0, R]$. For the rectangle, we need to calculate (assuming again $b > 0$)

$$\begin{aligned}
I &= \frac{1}{2\pi\sigma^2} \int_x^{x+b} \int_0^h e^{-\frac{1}{2}\frac{u^2+v^2}{\sigma^2}}\, dv du \\
&= \frac{1}{2\pi\sigma^2} \int_x^{x+b} e^{-\frac{1}{2}\frac{u^2}{\sigma^2}}\, du \int_0^h e^{-\frac{1}{2}\frac{v^2}{\sigma^2}}\, dv.
\end{aligned}$$

This can be done by making a table of

$$\frac{1}{\sqrt{2\pi}\sigma} \int_x^{x+b} e^{-\frac{1}{2}\frac{u^2}{\sigma^2}}\, du$$

for a grid of M^2 points with $(x, b) \in [0, R] \times [-R, R]$. Of course also the derivative can be easily calculated and put in the table.

5 Conclusion

We have given two alternative approaches for the anti-aliasing filter that ASML currently uses, These approaches, however, do not clearly improve upon the current method. Our two approaches are both adjustments to this method. In our first approach, we replace the analytical approach used by ASML by numerical calculations on a finer grid, where the values of the kernel at those grid points can be calculated and stored beforehand. We get the same order of number of operations as the analytic approach used by ASML, if we choose $N = 100$, where N is the factor with which the original (coarse) grid has been refined in both directions.

Some numerical experiments have been done on this approach. For the masks in the test set it seems that N can be chosen smaller than 100 and still deliver the required accuracy. However, this is not necessarily true in general because patterns with worse aliasing behavior may exist. Further analysis / experiments will need to be done in order to determine the correct N. A disadvantage of this approach is that it requires a lot of memory to store the values of the kernel at the fine grid points, and this will cause the process to be slower.

In the second approach, we first need to triangulate the polygons, which can be done quickly, and many algorithms are available for this. Then we need to find for every triangle which grid points are close to it, and for each of those points find the right decomposition in rectangular triangles, which also should not take long, and find the required values of the integral in the look-up table. Here too, experiments need to be done to find out how large the look-up table should be to get the required accuracy, without needing too much memory.

References

[1] Bracewell, R.: The Fourier Transform and its Applications, McGraw-Hill 1965