

# 마스크 아트웍 처리 및 레이아웃 검증을 위한 다각형 정형 알고리즘

## (Polygon Resizing Algorithm for Mask Artwork Processing and Layout Verification)

鄭子春\*, 李哲東\*, 柳瑛昱\*

(Ja Choon Jeong, Chul Dong Lee and Young Uk Yu)

### 要 約

본 논문에서는 주어진 평면상의 다각형들을 일정한방향으로 확대 또는 축소시키는 다각형 정형 문제에 대해서 논의한다. 먼저, 다각형 정형의 정의 및 그와의 문제점을 고찰한 다음, 향상평면주사법을 사용하여 평면을 한방향으로만 주사하는 것에 의해 전체의 정형을 완료할 수 있는 향상XY법을 제안한다. 이 방법의 시간복잡도는  $O(n \log n)$ , 공간복잡도는  $O(\sqrt{n})$ 으로 된다. 여기서,  $n$ 은 일련 다각형의 정점의 수이다. 그리고 LSI 설계의 마스크 아트웍 처리 및 레이아웃 검증에서의 응용에 대해서 논의한다.

### Abstracts

In this paper, we describe about polygon resizing problem where the given polygons are expanded or shrunk in two dimensional plane. First, the definition of polygon resizing and its problems are given, then the enhanced XY method is proposed: the polygon resizing can be completed in one directional sweep of plane only, using enhanced plane sweep method. The time complexity is  $O(n \log n)$ , and space complexity  $O(\sqrt{n})$ , where  $n$  is the number of vertices of polygons. The applications of polygon resizing to the mask artwork processing and layout verification are discussed.

### 1. 서 론

필적으로 설계의 마스크 아트웍 처리 및 레이아웃 검증분야에서 다각형 정형(polygon resizing)은 중요한 기술로 존재한다. 다각형 정형이란 일정한 평면상에 있는 다각형들을 주어진 일정한방향으로 확대(expand) 또는 축소(shrink)시키는 것을 말한다.

이 문제에 대해서 지금까지 제안된 방법들은 크게 보면, 아래의 세분화에서 관찰할 수 있다. 평면주사법

(plane sweep method)을 적용한 XY법으로 구분할 수 있다. 전체의 정형을 완료함에(5)에 보듯이, 그레그의 주사법을 이용하여 일련의 다각형에 대해서 정형 작업을 수행하는 것이 가능하다. 향상평면주사법(time complexity)이  $O(n^2)$ 으로 추정된다.  $O(n^2)$ 의 알고리즘은 작은 VLSI 설계에서 적용하여도, 대용량 설계에 적용할 때에 정형 작업에 소요되는 시간이 매우 길고, 메모리 사용량도 매우 커진다.

향상평면주사법에 적용한 XY법\*은 X축 및 Y축의 한 방향의 대해서만 sweep을 하여서 일련의 다각형 주사 문제의 시간복잡도를  $O(n \log n)$ 으로 낮출 수 있도록 제안하고 있다. 이 알고리즘은 이 논문에서 다각형 정형에 대해서도 X축 및 Y축의 한 방향으로 sweep을 할 수 있는 알고리즘이다. 스

\*正會員, 電子通信研究所 自動設計技術開發部  
(Electronics & Telecommunications Research Institute)  
原受理字: 1987년 3월 10일

입력으로 되는 다각형을 X축 및 Y축에 평행한 복합형 방형 도형(rectilinear rectangle)에 제한하는 등의 작업이 있다.

입력 데이터를 X좌표와 Y좌표 각각에 대해 sort한 2개의 file에 보관하는 것은 sort(이 경우 file sort에 해당)에 상당한 시간이 소모되는 것뿐만 아니라, 다각형의 데이터가 변경되는 경우에 2개의 file에 모순이 생기지 않도록 관리하지 않으면 안되기 때문에 바람직하지 못하다.

본 논문에서는 향상평면주사법(enhanced plane sweep method)<sup>6,7</sup>을 이용해서 한 방향으로만의 평면 sweep을 행하여도 모든 입력 다각형의 정형조각이 가능한 확장XY법을 제안한다.

이 방법의 시간복잡도 및 공간복잡도(space complexity)는 각각  $O(n \log n)$  및  $O(\sqrt{n})$ 이다. 여기서, n은 입력 다각형의 정점수이고, 공간복잡도는 수 메모리안에 보관되는 데이터 양이다. 입력 데이터는 컴퓨터 sequential file에 저장되어 있고, 현재의 차례에 필요한 데이터를 컴퓨터의 수 메모리에 access해서 사용하게 된다.

본 논문은 먼저 다각형 정형의 문제를 시작하고 다음에 본 논문에서 제안된 확장 XY법의 알고리즘을 보였으며, 끝으로 제조기술에 의한 다각형의 수정, DRC 및 notch 제거 그리고, 회로주위에 의한 검증용 마스크 야드 위치리 및 레이아웃 검증에의 응용기술 분야에 대해서 논의한다.

II. 다각형 정형의 문제

본 절에서는 입력 데이터의 표현법 그리고 보통의 다각형 정형의 정의등에 대해서 설명한다.

(1) 입력 데이터의 표현

본 논문에서 취급하는 평면상의 다각형은 그림 1과 같은 형태를 갖는다.

다각형은 그것이 가지는 정점렬에 의해서 표현되며, 다각형의 외부에서는 시계방향으로 열거된 정점들이

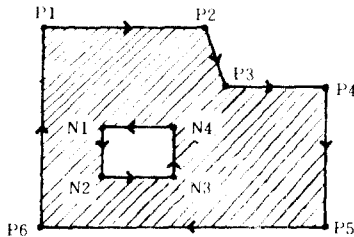


그림 1. 다각형 데이터의 표현 Fig. 1. Representation of polygon data.

고, 내부공(hole)에서는 반시계 방향으로 정점을 열거한다. 예를들어 그림 1의 다각형에 대해서 다음과 같이 표현된다.

P P1 P2 P3 P4 P5 P6 N1 N2 N3 N4

(여기서, P1...P6 : 다각형P의 외부정점, N1...N4 : 다각형P의 내부정점)

다각형의 각 정점은 그것의 평면상의 x, y 좌표값, 그것의 좌, 우, 상, 하의 방향을 나타내는 flag 및 내각에 대한 정보를 가진다. 여기서, 내각이라고 함은 그 점에서 다각형 내부의 각도를 말한다. 내각은 어떠한 특수각(90°, 135° 등)일 필요는 없고 일반 값을 갖는 각이어도 무방하다. 예를들어, 어떤 정점에 대한 record는,

v polygon-identity x y direction angle

로 된다.

이와같은 다각형의 표현에서 선분은 연속한 2개의 정점의 집합으로 표현되며, 어떤 2개의 선분도 그 정점외에서는 겹치지 않게 된다. 그림 2와 같은 것은 본 논문에서 취급하는 다각형의 형태에 어긋난다.

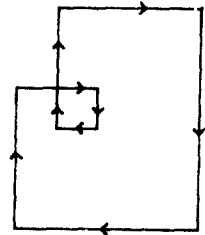


그림 2. 불법적인 다각형 Fig. 2. Illegal polygon.

(2) 다각형 정형의 문제

다각형의 정형이란 다각형의 각 점이나 선분을 일 정복만큼 외부로 확대(expand) 또는 축소 내부로(shrink)시키는 것을 의미한다. 엄밀한 의미에서 그림 3(a) 다각형에 대한 정형의 결과는 그림 3(b)와 같이 된다.

그러나, 이와같은 정형을 실제 마스크 제작에 이용할 경우 구석 부근이 원으로 되기 때문에 다각형의 정점수가 무수히 커질 수가 있을 뿐만 아니라, 레이아웃 데이터의 형식에도 적합하지가 않다. 그림 3(c)의 경우에는 원래의 다각형보다 구석부근만이  $\sqrt{2}$  배 만큼 확대 또는 축소되기 때문에 앞의 예에서 본 문제가 성립되지 않는다. 본 논문에서는 그림 3(c)와 같은 일반적인 정형문제를 취급한다.

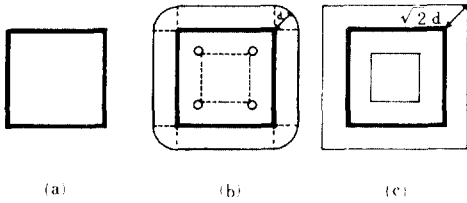


그림 3. 다각형 정형  
 (a) 입력 다각형  
 (b) 엄밀한 의미의 정형  
 (c) 일반적인 의미의 정형

Fig. 3. Polygon resizing.  
 (a) Input polygon.  
 (b) Strict definition of resizing.  
 (c) General definition of resizing.

그러나, 예각(내각이 90° 보다 작은 각)을 확대할 경우 새로운 정점이 원래의 거리에 비해서 커지게 된다 (그림 4).

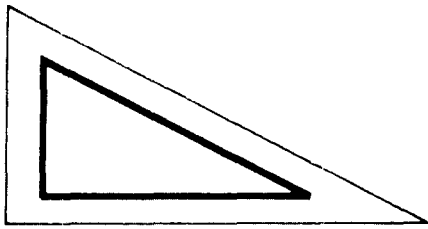


그림 4. 예각의 경우  
 Fig. 4. In case of sharp angle.

이와같이 필요이상으로 확장된 spike는 잘못 해석 될 수 있을 뿐만 아니라, 칩 레이아웃에 악영향을 미칠 수도 있다.

이와같은 것을 피하기 위해서 그림 5와 같은 방법으로 예각을 인식하고 구석 부근을 보상시키기 위해서 다음과 같은 근사적인 방법을 쓴다.

내각이 90° 인 경우 구석부근의 확대거리가  $\sqrt{2}$  배로 커지게 되나, 90° 보다 작은 경우에는 이보다 커지게 된다. 따라서, 구석 정점에서 확대거리 d의 원을 그려서 그림 5(b)와 같이 원과 만나는 부근으로 정형 영역을 한정한다. 이 경우에 새로운 정점 a, b의 생성에 의한 새로이 정형된 다각형으로 된다. 본 논문에서는 이와같은 일반적인 의미의 정형을 행한다. 이와같이 일반적인 의미의 다각형 정형을 정의한 즉, 정형된 결과와 반드시 원래 다각형의 형태를 유지하지 않는 경우도 있다.

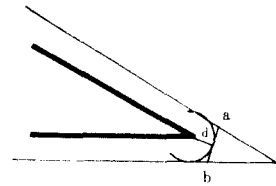
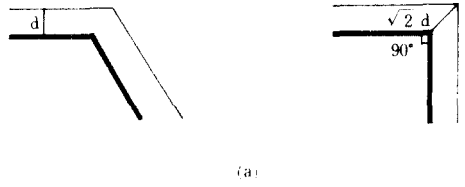


그림 5. 다각형 정형의 예  
 (a) 내각이 90° 또는 90° 보다 큰 경우의 정형  
 (b) 예각의 경우 보상

Fig. 5. Example of polygon resizing.  
 (a) Poly resizing of 90° or greater than 90°.  
 (b) Compensation of sharp angle.

### III. 다각형 정형 알고리즘

#### (1) 기본적인 방법

다각형 정형을 다음과 같은 단계적 방법으로 행한다. 여기서, 다각형의 영역을 R이라 하고, 정형 조작에 의해 새로 생긴 영역을 R' 그리고 정형 거리를 d로 한다. 다음은 이 방법을 단계적으로 보여주고 있다.

- (step1) 다각형R의 정점들을 그 정점이 속한 선분의 방향으로 정형거리d 만큼 평행이동 한다.
- step2) 평행이동후 정점연을 원래도형의 정점연과 1대1 대응관계를 갖는 새로운 폐 loop를 만든다(그림 6(b)).
- step 3) 새로운 폐 loop의 변이 평면상에 교차하는 경우는 그 교차점을 새로운 정점으로 하고 내부의 폐 loop를 제거한 다음 새로운 영역R'를 생성시킨다. (그림 6(c)).

위 방법의 step3)의 조작에서 그림 6(b)의 확장된 경계를 따라 scan해 가면서 교차점을 찾는 방법은, 정형조작 후 모든 다각형이 교차점을 갖는 경우에는  $O(n^2)$ 의 시간이 소모된다. 이와같이 교차점을 찾는 과정을 피하기 위해서 종래의 XY법<sup>[4]</sup>에서는 다각형의 정점에 의해 띠(band)로 분할한 다음, 어느 정점에서 수평한 방향으로 수직변까지의 거리가 정형거리 2d 보다 작다면 영역을 확장시켜 놓은 다음 정형을 수행한다(그림7). 이 방법에서는 위의 step 3)에서 처

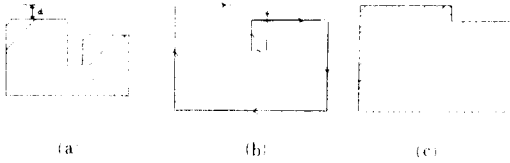


그림 6. 확대의 한가지 방법  
 (a) 입력 다각형  
 (b) d만큼 확대 후 폐 loop 형성  
 (c) 폐 loop의 제거

Fig. 6. One method of expansion.  
 (a) Input polygon.  
 (b) Generate the close loop after expanded.  
 (c) Delete the close loop.



그림 7. XY법의 설명  
 (a) 입력 다각형  
 (b) 띠 형성  
 (c) X축 방향으로 확대 후 모양

Fig. 7. Explanation of XY method.  
 (a) Input polygon.  
 (b) Band generation.  
 (c) The pattern after expanding to x-axis.

럼 폐 loop의 교차점이 생기지 않기 때문에 따로 scan 조사를 필요로 하지는 않는다.

그리고, 평면주사법과 그 work-list를 보관하는 데이터 구조로서 balance tree를 사용함으로써,  $O(n \log n)$ 의 시간복잡도내에 처리할 수가 있었다. 그러나, 이 방법은 x축과 y축에 대해서 평면 sweep을 행할 필요가 있다. 그것은 많은 양의 데이터를 취급하는 VLSI 설계에서는 입력 데이터를 sequential file에 저장하기 때문에 2개의 file이 필요함 뿐만 아니라, 다각형의 데이터가 변경되는 경우에 2개의 file에 모순이 생기지 않도록 관리하여야 되기 때문에 결코 바람직하지 못하다.

(2) 확장XY법의 알고리즘

여기서는 확장평면주사법<sup>67)</sup>을 이용하여 한 방향만에서의 평면 sweep에 의해 모든 입력 데이터의 정형조작이 가능한 확장 XY법을 제안한다.

보통 평면주사법과 확장 평면주사법의 차이점은 평면상을 위에서 아래로 수평한 주사선을 움직여 갈때,

전자는 그 주사선과 교차하는 영역을 보관하는 work-list(W1으로 부른다.)밖에 사용하지 않고 있으나, 후자의 경우는: 주사선의 윗 방향에 수직방향으로 가시한 영역(visible region)을 보관하는 work-list(W2로 부른다.)를 이용하고 있다는 것이다(그림 8). 이렇게 함으로써, 한번의 평면 sweep에 의해 모든 정형조작이 가능하다.

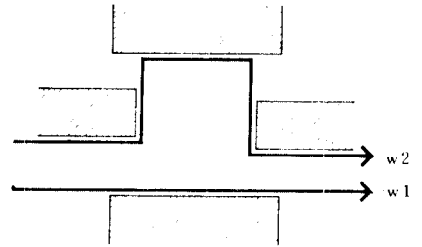


그림 8. 두 개의 주사선에 의한 평면주사법  
 Fig. 8. Plane sweep method with two sweep lines.

다음에는 확장XY법의 알고리즘을 보여주고 있으며 여기서는 확대의 경우에 대해서 설명하고 있으며, 축소인 경우에도 똑같이 적용된다. 그리고, 위의상 XY축에 수평, 수직인 다각형에 한해서 알고리즘을 전개했으나, 입력 데이터의 처리에서 설명한 대로 내각이 임의적인 다각형에 대해서도 확장이 가능하다.

그림 9는 주사선을 상하로 sweep 하면서 다각형 정형 조작을 수행하고 있다. 현재의 주사선W1상의 정점 v에 대해서 그 정점이 속한 다각형이 상단변의 정점이라면 윗방향 및 수평방향으로의 정형 조작을 행하고, 윗 방향의 가시변을 W2의 데이터 구조 상에서 찾아 그 변의 아래방향으로의 정형을 수행한다. 그리고, 정형된 새로운 정점을 결과로서 새로운 정점 file에 보관하고, 상단변의 정점을 W1과 W2의 데이터 구조에 삽입시키고, W2에서는 sweep 완료된 정점의 삭제가 이루어진다. 하단변의 정점일 경우에는 수평방향으로만 정형 조작을 행하며, W1에는 sweep 완료된 다각형 정보가 전부 삭제되고, W2는 그 하단변의 정점을 가시 영역으로서 보관한다.

확장XY법의 알고리즘은 다음 단계를 거쳐서 행한다.

입력: 평면상의 다각형의 집합, 정형거리 d.

출력: d만큼 확대 또는 축소된 다각형 집합

방법:

step 1) 주어진 다각형들의 정점을 y좌표치가 큰 순서대로 sort 한다(이 경우는 file sort에 해

당). 이때, 각 정점은 입력 데이터의 처리에서 설명한 정보를 가진다.

step 2) Sequential file에서 y 좌표값이 큰 순서(y 좌표값이 같은 경우는 x 좌표값이 작은 순서)대로 정점v를 추출한다. 만약 모든 다각형들의 정점이 탐색되었다면 종료한다.

step 3) v의 종료에 의해 다음의 어떤 조사를 행한다.

A. v가 어떤 다각형의 상단변의 정점인 경우

이 경우는 v의 내각이 90° (불룩한 점)인 경우와 270° (오목한 점)인 경우가 있으나, 두 경우가 거의 같은 방법으로 이루어지기 때문에 여기서는 90°인 경우에 대해서만 설명한다. 그리고 윗 방향 및 수평 방향으로 정형조작이 수행된다.

- v에서 수평한 방향으로 인접한 변w를 찾아서 그 변과의 거리 l을 측정한다.

i)  $l < d$ 인 경우 (그림 9(a))

이 경우 v에서 변형거리 d만큼 이동시킨다면, w가 속한 다각형상에 커 loop가 생길 것이다. 이를 방지하기 위해서 변w상에서 y좌표값이 큰 쪽으로 d만큼 증가시킨 다음, 새로운 정점v'를 생성시킨다.

ii)  $l > d$ 인 경우

수직방향으로 인접한 변h를 W2의 데이터 구조소에서 찾아서 그 거리( $l_y$ )를 측정한다.

(a)  $l_y < 2d$ 인 경우

(그림 9(b)와 같이 정점v에서는 x좌표치에 d만큼 증가시키거나 수평방향으로 이동한 점v'를 생성시키고, 변h에서 수평방향의 점v\_x를 찾아서 그 y좌표에 아래방향으로 d만큼 증가시킨 새로운 정점v\_x를 생성한다. (정점v\_x의 수평방향으로의 정형조작은 W1이 v\_x 상에 있을 때 이미 수행되었다.)

(b)  $l_y > 2d$ 인 경우

그림 9(c)와 같이 정점v와 v\_x의 x, y좌표에 각각 d만큼 증가시켜서 새로운 정점v'와 v'\_x를 생성한다.

- W1에 v'를 삽입한다.
- W2에 v'를 삽입한다.
- v, v\_x를 각각 v', v'\_x로 변형시킨 새로운 정점 file을 만든다.

B. v가 어떤 다각형의 하단변의 정점인 경우

이 경우도 v의 내각이 90°인 경우와 270°인 경우가 있으나, 거의 같은 조작이기때문에 90°인 경우에 대해서만 설명한다. 그리고 수평방향만으로 정형조작을 행한다.

- 수평한 방향으로의 변w를 인식하고 그 변까지의 거리 l을 측정한다.

i)  $l < d$ 인 경우

변w상에 새로운 정점v\_x를 생성시킨다(그림 9(d)).

ii)  $l > d$ 인 경우

에 x방향으로 d만큼 증가시킨 새로운 정점을 생성시킨다(그림 9(e)).

- W1에 정점v가 속한 도형의 윗쪽의 점v\_u를 삭제한다.

- W2에서 점v\_u를 삭제시킨 다음, v\_x를 삽입한다.

step 4) step 2)로

부록에는 위의 알고리즘을 pseudo 프로그램으로 나타내었다.

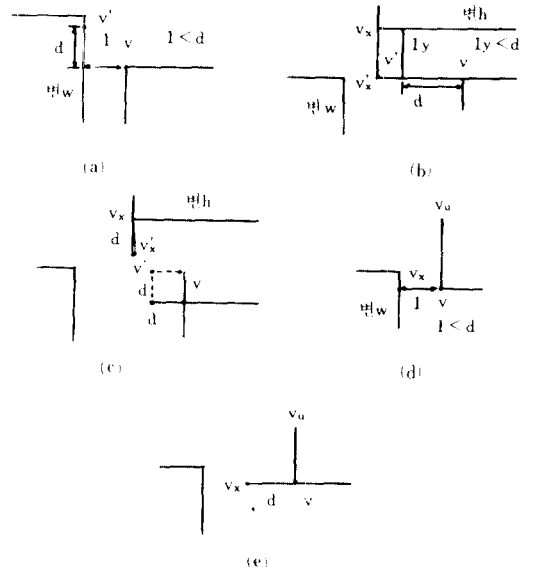


그림 9. 확장XY법의 설명

- (a)  $l < d$
- (b)  $l > d, l_y < d$
- (c)  $l > d, l_y > d$
- (d)  $l < d$
- (e)  $l > d$

Fig. 9. Explanation of enhanced XY method.

- (a)  $l < d$ .
- (b)  $l > d, l_y < d$ .
- (c)  $l > d, l_y > d$ .
- (d)  $l < d$ .
- (e)  $l > d$ .

(3) 확장XY법의 해석

step 1)은 file sort와 선형시간에 실행되는 부가적인 조작이기 때문에  $O(n \log n)$ 의 시간안에 가능하다.

여기서  $n$ 은 입력으로 되는 다각형의 정점수이다. step 2)와 step4)에서는 step3)의 조작을  $O(n)$ 회 반복시키고 있다. step3)에서는 2개의 work list  $W1, W2$ 에 대해서 다음의 기본조작을 행할 필요가 있다.

- a. insert( $v$ ) : work-list에 정점 $v$ 를 삽입한다.
- b. delete( $v$ ) : work-list에서 정점 $v$ 와 같은  $x$ 좌표값을 가진 정점을 삭제한다.
- c. left( $v$ ) : 정점 $v$ 의  $x$ 좌표값보다 작은  $x$ 좌표값을 가진 work-list 내의 정점중에 최대  $x$ 좌표값을 가진 것을 구한다.
- d. right( $v$ ) : 정점 $v$ 의  $x$ 좌표값보다 큰  $x$ 좌표값을 가진 work-list 내의 정점중에 최소  $x$ 좌표값을 가진 것을 구한다.
- e. member( $h$ ) : work-list 내의 정점중에 수평한 변 $h$ 의 양단점 $x$ 좌표에 둘러싸인  $x$ 좌표를 갖는 것들을 일거한다.

$W1$ 에 대해서는 a~d의 조작이,  $W2$ 에 대해서는 a~e의 조작이 필요하다. 이상의 기본조작을 효율 좋게 처리하기 위해 work-list를 표현하는 데이터 구조에 평형 2분목\*<sup>1)</sup>을 이용하면, a~d의 조작은  $O(\log m)$ 의 시간에, e의 조작은  $O(\log m + k)$ 의 시간내에 수행할 수 있다. 여기서,  $m(m \ll n)$ 은 work-list 내의 정점수이고,  $k(k \ll m)$ 은 일거된 정점의 수이다. 이상의 조작이  $O(n)$ 회 수행되고, member( $h$ )에 의해 일거된 정점은 바로 후에 삭제되는 것을 고려하면 step2)~step4)의 시간복잡도는  $O(n \log m)$ 으로 된다. 이상에 의해 확장 XY법의 시간복잡도는  $O(n \log n)$ 이다.

수 메모리 내의 공간복잡도는  $O(m)$ 이 된다. VLSI의 마스크 패턴설계에 있어서는  $m$ 이  $\sqrt{n}$ 에 비례하는 것이 알려졌다<sup>2)</sup> 때문에, 확장 XY법의 공간복잡도는  $O(\sqrt{n})$ 으로 된다.

IV. 다각형 정형 알고리즘의 응용

이성 설명한 다각형 정형 알고리즘은 레이아웃 검증 및 마스크 아트웍 생성부분에서 많이 사용된다. 그 예로서는 PG(pattern generator)의 입력을 만드는 과정에서 제조기술에 의한 오차의 수정, DRC(design rule check) 및 notch 제거 그리고 회로주물에 의한 검증등이다.

(1) 제조기술에 의한 오차의 수정

제조 공정기술의 특성으로 인하여 레이아웃 데이터를 그대로 PG(pattern generator) machine의 입력으로 쓰는 경우, 실제의 소자는 원하지 않는 모양이 생길 수 있게된다. 이와같은 제조 공정기술상의 오차를 최소한 적게하기 위해서, 미리 확대 또는 축소된 도형

을 입력 데이터로서 제공하여 마스크에 노광시킬 필요가 있다. 이 조작에 본 다각형 정형 알고리즘을 적용시킬 수 있다.

(2) DRC 및 Notch 제거에 응용

마스크 데이터는 주로 설계자의 실수에 의해서 최소폭, 최소간격등의 설계규칙의 위반을 포함하게 된다. 이러한 설계규칙의 위반을 검출하고 그 장소를 보고하는데 이 알고리즘을 적용할 수 있다.(그림10)의 a에서처럼 최소간격 $d$ 의 반만큼을 먼저 확대시킨 다음, 겹치는 부분을 조사하고 다시 본래의 위치로 환원시키면 error가 발생한 부분을 검출할 수가 있다.

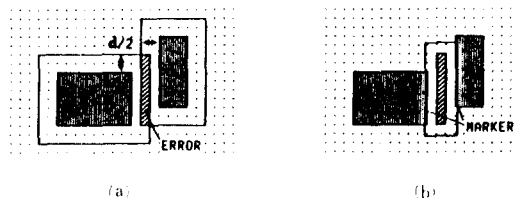


그림10. 최소간격 위반의 검출  
 (a)  $d/2$ 만큼 확대  
 (b)  $d/2$ 만큼 축소(본래의 위치로 환원)  
 Fig. 10. Detection of minimum space violation.  
 (a) Expand equal to  $d/2$ .  
 (b) Shrink equal to  $d/2$  (return to the original position).

또한, 설계규칙을 위반하고 있는 부분(notch)라 한다)을 제거하는 데에도 적용된다(그림11)과 같이 설계규칙만큼 확대한 후에 같은 변경폭으로 축소하면 notch가 제거된다. 역으로, 축소한 후 확대를 한다면, 미소돌출 부분이 제거된다. 이때 notch 제거에 의해 호의 윤리적 접속상태가 변하지 않도록 해야한다.

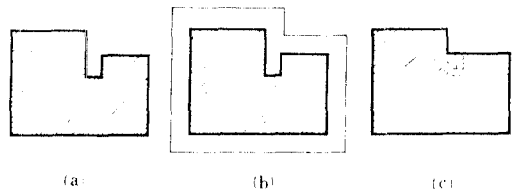


그림11. Notch 제거  
 (a) 입력영역  
 (b) 확대  
 (c) 축소

Fig. 11. Notch deletion.  
 (a) Input region.  
 (b) Expand.  
 (c) Shrink.

## (3) 회로추출에 의한 검증에 응용

레이아웃 자체의 기생성분(저항, capacitor 등)에 의한 논리 지연상태를 검증하기 위해서 회로추출 방법이 많이 사용된다. 이때, 레이아웃 설계단계의 도형 데이터와 실제 마스크 상에서의 그것과는 차이가 있게 된다. 이러한 차를 충분히 반영하여 회로추출을 행한다면 보다 정밀한 시뮬레이션을 할 수가 있다. 이 경우에 다각형 정형 알고리즘을 적용할 수 있다.

## V. 결 론

본 논문에서는 평면상에 주어진 다각형들을 일정복만큼 확대 또는 축소시키는 다각형 정형 문제들, 평면을 한 방향으로만 sweep하는 것에 의해  $O(n \log n)$ 의 시간복잡도에 가능한 알고리즘을 제안했다. 여기서,  $n$ 은 입력다각형의 정점수이다. 제안된 알고리즘의 입력 데이터는 sequential file에서 현재 주사선상에 있는 데이터만을 주 메모리에 보관해서 처리하기 때문에 공간복잡도는  $O(\sqrt{n})$ 이 되며, 특히 처리하는 데이터양이 엄청나게 커지고 있는 최근의 VLSI 설계에 유용하게 쓰일 수 있다.

또한, 제조 공정상의 특성에 의한 오차의 수정, DRC 및 notch제거 그리고 회로추출에 의한 검증등 레이아웃 검증 및 마스크 아트웍 생성분야에의 응용가능한 분야에 대해서 논했다.

## 부 록

```
PROGRAM RESIZING
  file1 = file-sort(infile);
  d = read(resizing-distance);
  procedure expand(file1, d) {
    while (not EOF(file1)) {
      v = extract(file1);
      if (v is upper-side-vertex) {
        if (v is convex and oriented-right) {
          vline = RIGHT(W1, v);
          dx = dist(vline);
          if (dx <= d) vbar = gen(vline, v);
        } else {
          hline = MEMBER(W2, v);
          dx = dist(hline);
          vx = left-vertex(hline);
          if (dx <= 2*d) vbar = add-y(vx);
        } else {
          vxbar = add-xy(vx);
          vbar = add-xy(v);
        }
      }
    }
  }
  else if (v is convex and oriented-left) {
    almost the same operation is done as
    v is convex and oriented-right. }
```

```
else if (v is concave vertex) {
  almost the same operation is done as
  v is convex vertex. }
W1 = INSERT(W1, vbar);
W2 = INSERT(W2, vbar);
mod(file1, v, vbar); mod(file1, vx, vxbar);
}
else if (v is lower-side-vertex) {
  if (v is convex and oriented-left) {
    vline = RIGHT(W1, v);
    dx = dist(vline);
    if (dx <= d) then vbar = gen(vline, v);
    else vbar = add-x(v);
  }
  else if (v is convex and oriented-right) {
    almost the same operation is done as
    v is convex and oriented-left. }
  else if (v is concave vertex) {
    almost the same operation is done as
    v is convex vertex. }
  W1 = DELETE(W1, v-upper);
  W2 = DELETE(W2, v-upper);
  W2 = INSERT(W2, vbar);
}
}
```

## 參 考 文 獻

- [1] K. Yoshida, "Layout Verification", Chap. 7, "Layout Design and Verification" (T. Ohtsuki, ed.), North Holland, pp. 237-265, 1986.
- [2] K. Kozawa, A. Tsukizoe, J. Sakemi, C. Miura and T. Ishii, "A Concurrent Pattern Operation Algorithm for VLSI Mask Data", *IEEE Proc. of the 18th DAC*, pp. 563-570, 1981.
- [3] B.W. Lindsay and B.T. Preas, "Design Rule Checking and Analysis of IC Mask Design", *IEEE Proc. of the 13th DAC*, pp. 301-308, 1976.
- [4] M. Sato, M. Tachibana and T. Ohtsuki, "An Algorithm for Resizing Polygon Regions and Its Applications to LSI Mask Pattern Design", *Electronics and Communications in Japan*, vol. 67-c, no. 4, pp. 93-101, 1984.
- [5] Y.S. Huang and S.P. Chen, "A Graph Theoretic Approach to the IC Layout Resizing Problem", *IEEE Trans. on CAS*, CAS-27, pp. 380-391, 1980.
- [6] M. Sato and I. Ohtsuki, "Enhanced Plane Sweep Methods for LSI Pattern Design Problems", Institute of Electronics and

- Communications Engineering of Japan, Report CAS, 86-169, pp. 87-94, 1987.
- [7] J.C. Jeong, J.B. Kim, M. Sato and T. Ohtsuki, "A Fast Minimum Width/Space Verification Algorithm", Institute of Electronics, Information and Communications Engineering of Japan, Report CAS 87-16, pp. 39-46, 1987.
- [8] T. Asano, M. Sato and T. Ohtsuki, "Computational Geometry Algorithms", "Layout Design and Verification" (T. Ohtsuki, ed.), North Holland, pp. 295-347, 1986.
- [9] A.V. Aho, J.E. Hopcroft and J.D. Ullman, *The Design and Analysis of Computer Algorithms*, Addison Wesley, Reading Mass., 1974.
- [10] N. Wirth, Algorithms + Data Structures = Programs, *Prentice Hall*, Englewood Cliffs, 1976.
- [11] E.M. McCreight, "Priority Search Trees", *SIAM J. Comput.*, vol. 14, no. 2, pp. 274-276, 1985.
-