Image Processing and Computer Vision

Robert Haralick Editor

Efficient Nesting of Congruent Convex Figures

DOV DORI and MOSHE BEN-BASSAT

ABSTRACT: Optimal nesting is the arrangement of two-dimensional polygons within a rectangular board so that waste is minimized. Our approach follows a top–down, stepwise refinement of the original problem into simpler subproblems; the combined solution of all of the problems permits the solution of the original problem. We first show that the original problem can be decomposed into two subproblems—one consisting of finding all convex paver polygons and the other of optimal (minimal waste) circumscription of the original figure in the most appropriate paver polygon.

1. INTRODUCTION

Optimal nesting is the arrangement of two-dimensional figures of various shapes within a rectangular board so that waste is minimized. The problem frequently arises in industries that cut figures from textile or steel sheets. An actual example of figures nested in a steel board is given in Figure 1.

A special case of the general nesting problem is the *cutting-stock problem*. The cutting-stock problem, in its classical form, is concerned with cutting a rectangular board or roll of material into a specified number of certain types of smaller *rectangular* pieces so that waste is minimized. When the number of pieces of each type to be cut from each board is not restricted, the problem is known as the *template-layout problem*. The cutting-stock problem has been presented with various versions and limitations, such as restricting the cut to a *guillotine* type [1, 2, 4, 11–13, 15, 16] or tilting the rectangles relative to the board [5]. (A guillotine cut is obtained by cutting along a straight line from one edge of the sheet to another.) Another variation is related to restrictions on the sequence of cutting [7, 17].

For small problems, the approach adopted by most researchers is based on linear and/or dynamic programming techniques. For larger problems, where these techniques do not produce a solution within acceptable time, tree search and heuristic algorithms have been proposed that, in some cases, produce suboptimal solutions [2, 4, 16]. (See [1, 2] for a brief survey.)

The above works mostly represent an operations research approach to the problem. Related problems from computational geometry include packing (nesting) circles in a circle and packing squares in a square. Gardner [10] surveys both problems.

In this paper, we deal with congruent replications of a given convex figure in a board so that waste is minimized. This problem has a practical application in mass-production systems where the board area is relatively large compared to the given figure. On one hand, this problem is more general than the above-mentioned problems in that it deals with all kinds of convex figures. On the other hand, it is more restricted in that it handles only one figure with congruent replications. Practically, the algorithms in our paper may also be applied to problems with nonconvex figures by preprocessing the original figures to obtain convex ones with minimal waste. This can be done either by generating the convex hull of individual figures (if it offers effi-

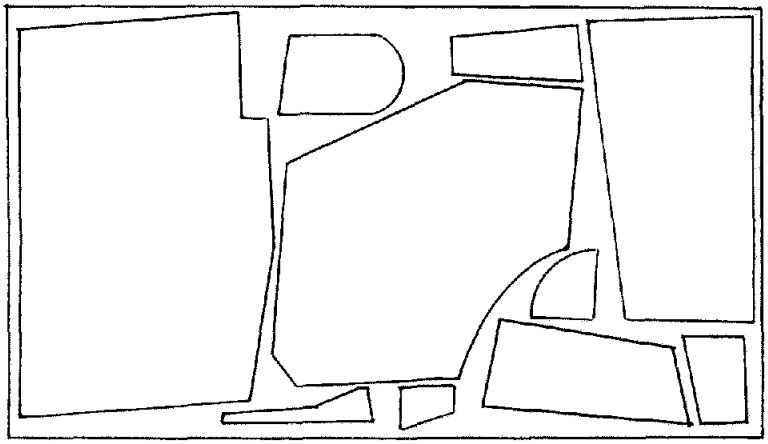

FIGURE 1. A sample of nested figures for flame cutting of steel sheets.

cient circumscription) or by combining several of the original nonconvex figures into one convex figure. Interactive graphic systems are very useful for the later option.

In Section 2, we describe our notation. In Section 3, a new concept is introduced. Section 4 outlines our approach, and, in Sections 5, 6, and 7, the details are given. Numeric illustrations are given in Section 8, and in Section 9, we conclude with a summary and problems for further study.

2. CONVENTIONS FOR A POLYGON NOTATION

The following notation, which is illustrated in Figure 2(a), will be used:

Polygon: n-sided polygon is denoted by P_n.

Vertices: The vertices of a polygon are denoted by $V_1, V_2, \ldots, V_n$. The numbers increase counterclockwise.

Sides: The sides of a polygon are denoted by $S_1, S_2, \ldots, S_n$, where S_n is the line segment connecting vertices V_j and V_{j+1}.

Angles: The interior angles of the polygon are denoted by $A_1, A_2, \ldots, A_n$, according to the numbers of the vertices.

Distance between Points: The distance between a point C and a straight line l is denoted by $D(C, l)$.

Point of Intersection: The point of intersection of two straight lines, l_1 and l_2, is denoted by $C(l_1, l_2)$, provided that l_1 and l_2 are not parallel.

Area of Polygon: The area of a polygon P_n whose vertices are $V_1, V_2, \ldots, V_n$ is denoted by $R(P_n)$ or $R(V_1, V_2, \ldots, V_n)$.

Circumscription Efficiency: Let P' denote a figure circumscribing another figure P. Then the circumscription efficiency is defined as

$$E_c = \frac{\text{Area of } P}{\text{Area of } P'}$$

3. BASIC ORDER OF A CONVEX POLYGON; CIRCULAR AND BASIC POLYGONS

Definition 1

A convex polygon P_n is of *basic order* k—denoted by $BO(P_n) = k$—if it has exactly k sides, $S_{j_1}, S_{j_2}, \ldots, S_{j_k}$, for each of which

$$A_{j_i} + A_{j_i+1} < \Pi; \qquad j_i = 1, 2, \ldots, k$$

A side S_j for which the above expression holds will be

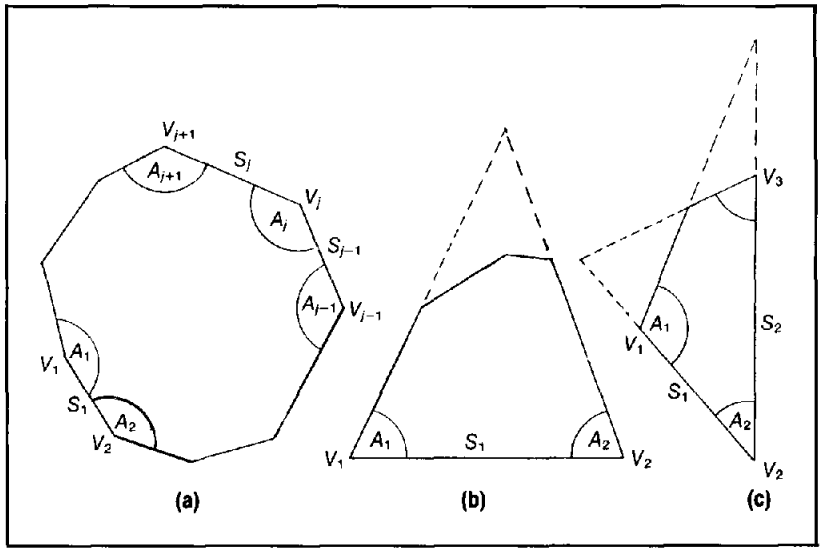

FIGURE 2. The three kinds of convex polygons with more than 3 sides, classified by their basic order.
(a) Circular polygon.
(b) Basic polygon of basic order 1: $\measuredangle A_1 + \measuredangle A_2 < \pi$.
(c) Basic polygon of basic order 2: $\measuredangle A_1 + \measuredangle A_2 < \pi$ and $\measuredangle A_2 + \measuredangle A_3 < \pi$.

referred to as a *basic side*, since intuitively it can serve as a stable basis on which the polygon can stand firm. Geometrically, if we extend the two sides adjacent to a basic side, they will form, together with the basic side, a triangle, that fully contains the original polygon. We have proven in [6] that the number of basic sides in any convex polygon is 2 at most and, in that case, these two sides are adjacent.

Using this definition of basic order, the group of convex polygons may be decomposed into two distinct groups—circular and basic polygons—according to the following definitions. (See Figure 2.)

Definition 2:
A convex polygon is *circular* if and only if it has no basic sides.

Definition 3:
A convex polygon is *basic* if and only if it has at least one basic side.

4. TECHNICAL APPROACH

Our approach to the problem of efficient nesting of congruent convex figures in the infinite plane is based on two main steps. First, we circumscribe the given convex shape by a convex polygon P_n with a sufficiently large number of sides, so that the waste is negligible [9]. Second, we circumscribe the convex polygon P_n by another polygon that can pave the plane. By the term

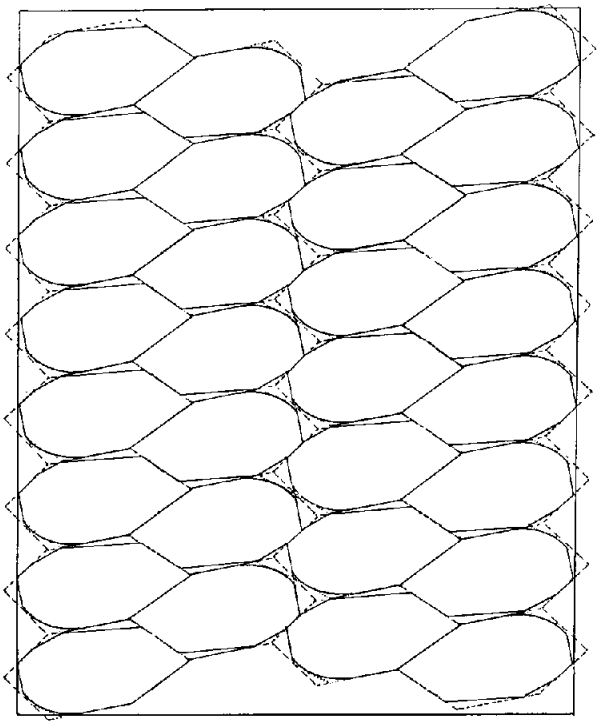

FIGURE 3. Illustration of the approach for the efficient nesting of congruent convex figures.
——— original figure ..... approximation to a polygon
— — — —circumscribing paverhex

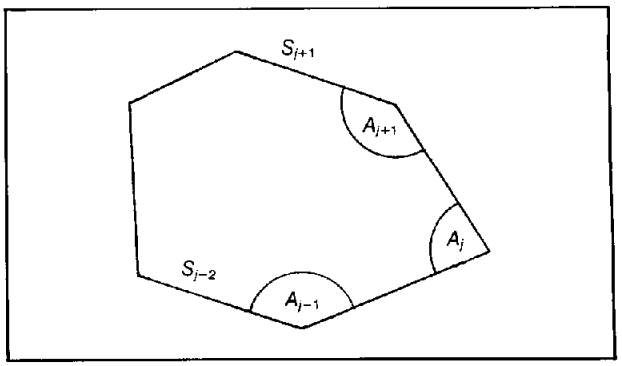

FIGURE 4. Hexagon Paver of Type 1.

pave, we mean covering the plane by replications of the same figure without any gap or overlap [18]. Figure 3 illustrates our approach using a hexagon as the paving figure. Paving is also called *tessalation* or *tiling* in the literature [16].

Definition 4:
A convex figure by which the plane may be paved will be termed a *paver*.

Following the above approach, our problem may be decomposed into two subproblems:

Problem a: Find all the possible pavers.
Problem b: Find the paver that efficiently circumscribes the convex polygon P_n that circumscribes the original figure.

Problem *a* is an old geometrical problem. It involves the determination of all types of two-dimensional convex figures that share the ability to pave the two-dimensional plane with the triangle, the square, and the regular hexagon. Kershner [18] characterizes three types of hexagon pavers and eight types of pentagon pavers (of which three are redundant cases of the three types of hexagons). Kershner also states that these are the only pavers. This statement, however, is incorrect, as we know of many other pavers today [14].

Problem *b* deals with the efficient circumscription of a convex polygon by a paver. Since there are many types of pavers, we elect to restrict our search for a hexagon of Type 1 (Figure 4), which is defined [18] as a hexagon for which there exists j, $j \in \{1, 2, \ldots, 6\}$, such that

$$A_{j-1} + A_j + A_{j+1} = 2\pi,$$
$$\text{length } (S_{j-2}) = \text{length } (S_{j+1}),$$

and

$$S_{j-2} \text{ is parallel to } S_{j+1}$$

This decision was motivated by the following reasons:

1. The circumscription efficiency of P_n by a polygon of fewer sides P_m reduces as m decreases [6]. Therefore, hexagons are likely to yield better efficiency than pentagons, quadrilaterals, or triangles.
2. Type 1 hexagons are characterized by the simultaneous existence of two equalities only—one concerning the parallelism of two opposite sides (i.e.,

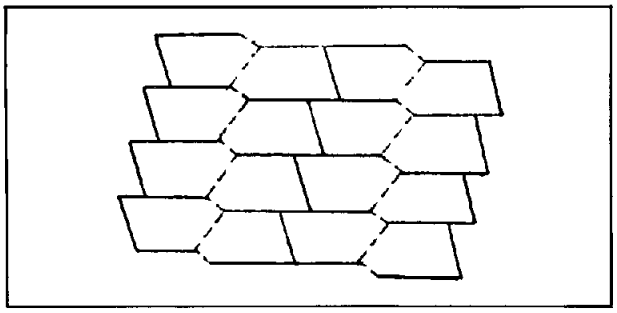

FIGURE 5. Paver Pentagons of Type 1.

S_j and S_{j+3}) of the hexagon and the other concerning the equality of length of these two sides. The requirements for the other types of paver hexagons are much more complex and involve the simultaneous existence of 3 or 6 equations. (See [18].) Type 1 hexagons seem to be more promising because the probability of meeting two simultaneous requirements for the price of adding a certain amount of area to P_n is likely to be higher than the probability of meeting 3 or 6 similar simultaneous requirements with the same area addition or less.

Figure 5 illustrates paving by a pentagon that is a redundant case of a Type 1 hexagon, and is defined by the existence of j for which

$$A_{j-1} + A_j + A_{j+1} = 2\pi, \qquad j \in \{1, 2, \ldots, 6\}$$

These pentagons, to be named Type 1 pentagons, are included in our search.

Thus, the efficient circumscription of P_n by a paver is restricted to paver hexagons of Type 1 or the special case of paver pentagons of Type 1. Of course, the final results might have been somewhat better (for some cases) had we tried optimization over many other possible paver types. However, this would require duplicating the kind of research presented in this paper for all these types.

Since a paver hexagon of Type 1 will be frequently mentioned in the sequel, it will, from now on, be termed a *paverhex* and denoted by P_6^*. Paver pentagons of Type 1 will be denoted P_5^*, and both pavers will be termed *parallel pavers* and denoted by P^*.

The circumscription of P_n by a parallel paver proceeds in two stages:

Stage 1: Circumscribe P_n by a hexagon P_6 with minimal area addition.

Stage 2: Circumscribe P_n by a parallel paver P^* with minimal area addition, using P_6 as a mediator.

The problem of Stage 1 is a special case and is solved in [6]. A related problem, solved by Freeman and Shapira [8], is concerned with determining the minimum area *rectangle* encasing an *arbitrary* closed curve. (In [6], the encasing figure is a general convex polygon; however, the encased figure is restricted to a convex, closed curve.)

The problem of fitting one polygon into another has also been recently treated by Chazelle [3] in the context of pattern recognition, where matching a mobile object against a static one is required. This problem is related to the encasing problem in the sense that in both cases some figure has to fit into the other. However, whereas in the encasing problem we form the circumscribing figure, in the later problem we compare two given polygons and decide whether they fit or not. In any case, the algorithm presented in [6] will solve the problem of Stage 1 and, therefore, our only concern here is the solution of the problem presented in Stage 2.

Like any convex polygon, P_6 achieved at the end of Stage 1 may be either circular or basic. (See Section 3.) The following two sections address the circumscription of circular and basic hexagons by a parallel paver.

5. CIRCUMSCRIBING A CIRCULAR HEXAGON BY A PARALLEL PAVER

The optimal circumscription of a circular hexagon by a parallel paver consists of performing six different constrained optimal circumscriptions, where each circumscription is constrained to one side of P_6 and includes two stages:

Stage 2.1: Optimal circumscription of P_6 by $\bar{P}_6$, where $\bar{P}_6$ denotes a hexagon that has a pair of parallel opposite sides, one of which is an original side of P_6.

Stage 2.2: Optimal circumscription of $\bar{P}_6$ by P^*.

For each of these stages, we first give a brief description of the algorithm and then proceed to a more rigorous presentation.

Algorithm 1. Optimal Circumscription of P_6 by $\bar{P}_6$. Beginning with a certain side S_j of P_6 (see Figure 6), since P_6 is circular, the vertex farthest from S_j is either V_{j-2} or V_{j+3}. This vertex will be denoted by V_f. V_f can either be a vertex of the original polygon P_n or a vertex that was created during the iterative circumscription of

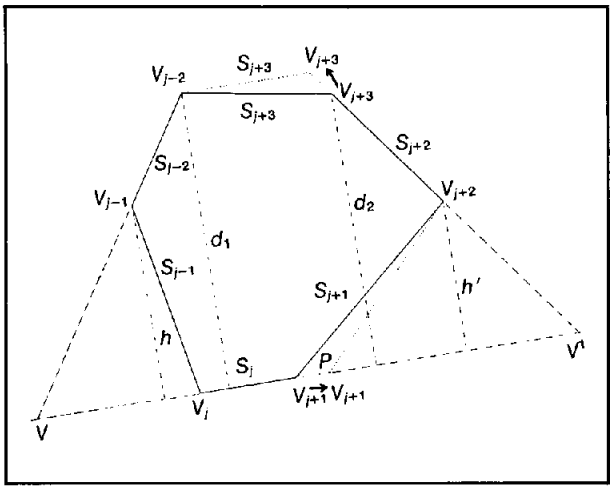

FIGURE 6. Circumscribing a circular hexagon by a parallel paver with minimal area addition.

P_n by P_6. In the latter case, a new V_f is determined as the vertex belonging to P_n that is farthest from S_j. This operation ascertains that the solution is optimal with respect to P_n rather than P_6. Once V_f is determined, S_{j+3}, which is the side opposite S_j, is replaced by S'_{j+3}, which passes through V_f, and is parallel to S_j.

Step 1: Find $d_1 = d(V_{j-2}, S_j)$ and $d_2 = d(V_{j+3}, S_j)$.
Step 2: If $d_1 > d_2$, then $V_f = V_{j-2}$, else $V_f = V_{j+3}$.
Step 3: If V_f is an original vertex of P_n, go to Step 5, else continue.
Step 4: Set $V_f = V_m$ such that $d_m = \max_{i=1}^{n} \{d_i = d(V_i, S_j)$; V_i an original vertex of $P_n\}$. V_f is selected as the farthest vertex from S_j of all the original vertices of P_n.
Step 5: Build the new side S'_{j+3} such that it passes through V_f and is parallel to S_j.
Step 6: Replace the two vertices V_{j-2} and V_{j+3}, that are the endpoints of S_{j+3}, by V'_{j-2} and V'_{j+3}, respectively, as follows:

$$V'_{j-2} = C(S'_{j+3}, S_{j-2}) \quad \text{and} \quad V'_{j+3} = C(S'_{j+3}, S_{j+2}).$$

Step 7: Set $V_{j-2} = V'_{j-2}$, $V_{j+3} = V'_{j+3}$ and $S_{j+3} = S'_{j+3}$.
Step 8: End.

The optimality of this algorithm is due to the fact that the parallel side is passed as close as possible to S_j while still satisfying the requirements of being parallel to S_j and completely outside of P_n. The algorithm could have begun at Step 4 and yielded the same result. However, for a large n this would require greater computational effort.

Algorithm 2. Optimal Circumscription of $\bar{P}_6$ by P^*

To convert $\bar{P}_6$, which was obtained by Algorithm 1, into a parallel paver P^*, we have to lengthen the short parallel side of $\bar{P}_6$. Lengthening S_j either clockwise or counterclockwise ceases when either P_5^* or P_6^* is obtained. P_6^* is obtained when S_j is lengthened by an amount p such that it equalizes S_{j+3}, but if this operation yields a concave hexagon, S_j is lengthened by less than p such that P_5^* is obtained.

The proper direction for lengthening S_j is determined as the one that adds as little area as possible to $\bar{P}_6$. To do this, we compute the areas Q_5 and Q_6 which are added to $\bar{P}_6$ when S_j is lengthened clockwise to yield P_5^* and P_6^*, respectively, and the areas Q'_5 and Q'_6 which are added to $\bar{P}_6$ when S_j is lengthened counterclockwise to yield P_5^* and P_6^*, respectively.

S_j is lengthened clockwise if either Q_5 or Q_6 is smaller than both Q'_5 and Q'_6; otherwise, it is lengthened counterclockwise. P_5^* is obtained if either Q_5 or Q'_5 is smaller than both Q_6 and Q'_6; otherwise, P_6^* is obtained. If P_5^* is obtained, it is converted to a pseudo-P_6^* to facilitate further treatment.

In the following algorithm, whenever some subscript exceeds 6, that subscript should be decreased by 6. Step 2 assures that the short parallel side is always denoted S_j, and Step 9 assures that the original notation of all subscripts at the end is the same as at the beginning of the algorithm.

Step 1: Find l_j, the length of S_j, and l_{j+3}, the length of S_{j+3}, as follows:

$$l_j = l(V_j, V_{j+1}) \quad \text{and} \quad l_{j+3} = l(V_{j-2}, V_{j+3}).$$

Step 2: If $l_j \le l_{j+3}$ continue, else replace each subscript k by $k + 3$. (This step assures that the side to be lengthened is always S_j).
Step 3: Compute $p = l_{j+3} - l_j$.
Step 4: Compute

$$h = d(V_{j-1}, S_j),$$
$$h' = d(V_{j+2}, S_j),$$
$$V = C(S_j, S_{j-2}),$$
$$V' = C(S_j, S_{j+2}),$$
$$q = l(V, V_j),$$
$$q' = l(V', V_{j+1}),$$
$$Q_5 = q \cdot h/2,$$
$$Q'_5 = q' \cdot h'/2,$$
$$Q_6 = p \cdot h/2,$$

and

$$Q'_6 = p \cdot h'/2.$$

Step 5: If $Q_5 = \min(Q_5, Q'_5, Q_6, Q'_6)$, then set $V_j = V$, set $V_{j-1} = V_{j-2}$, and set V_{j-2} as a new redundant vertex along S_{j+3} such that $d(V_{j-2}, V_{j+3}) = l_j$.
Step 6: If $Q'_5 = \min(Q_5, Q'_5, Q_6, Q'_6)$, then set $V_{j+1} = V'$, set $V_{j+2} = V_{j+3}$, and set V_{j+3} as a new redundant vertex along S_{j+3} such that $d(V_{j+3}, V_{j-2}) = l_j$.
Step 7: If $Q_6 = \min(Q_5, Q'_5, Q_6, Q'_6)$, then move V_j clockwise along S_j by p.
Step 8: If $Q'_6 = \min(Q_5, Q'_5, Q_6, Q'_6)$, then move V_{j+1} counterclockwise along S_j by p.
Step 9: If in Step 2 each subscript k was replaced by $k + 3$, replace each subscript k by $k + 3$.
Step 10: End.

The resulting P_6^* is optimal under the constraint that S_j and S_{j+3} are the parallel and equal sides. It is obtained by a sequential application of Algorithms 1 and 2 on P_6, and it is optimal because the area added to P_6 to get P_6^* is as small as possible under the above constraint. This constraint may be removed by checking each $j = 1, 2, \ldots, 6$, and selecting the best possible P_6^*. This is done in the following algorithm.

Algorithm 3. Circumscribing a Circular Hexagon by P^* with Minimal Area Addition.

Step 1: Set $j = 0$.
Step 2: Set $j = j + 1$.
Step 3: Perform Algorithm 1 on P_6 to obtain $\bar{P}_6(j)$.
Step 4: Perform Algorithm 2 on $\bar{P}_6(j)$ to obtain $P^*(j)$.
Step 5: If $j = 6$ continue, else go to Step 2.
Step 6: Of the six possible pavers $P^*(j)$, $j = 1, 2, \ldots, 6$ select the one whose area is minimal, and set $P^* = P^*(j)$.
Step 7: End.

This algorithm generates six different circumscribing parallel pavers, each of which is optimal under the

constraint that a certain side S_j of the circular hexagon must be parallel to its opposite side and is achieved by applying Algorithms 1 and 2 sequentially for $j = 1, 2, \ldots, 6$. Selection of the circumscribing parallel paver that has the smallest area of all six possibilities yields the optimal solution for circumscribing the hexagon, which, in turn, optimally circumscribes the original P_n by a parallel paver.

6. CIRCUMSCRIBING A BASIC HEXAGON BY A PAVERHEX

Two congruent basic hexagons may be optimally circumscribed by a parallel paver by joining them such that their equal basic sides overlap and a circular decagon (10-sided polygon) that has five pairs of equal and parallel opposite sides is obtained. (See Figure 7.) Applying four single-side reductions to the decagon following the iterative circumscription algorithm described in [6] yields an optimal circumscribing paverhex. The optimality is due to the fact that the pair of hexagons is embedded in the decagon with no waste of area and to the optimality of the iterative circumscription algorithm.

Algorithm 4. Circumscribing Two Congruent Basic Hexagons by a Paverhex with Minimal Area Addition.

Step 1: Put the two basic hexagons next to one another such that their two corresponding basic sides overlap to get a decagon P_{10}.

Step 2: Perform on P_{10} four single-side reductions using the iterative circumscription algorithm (Algorithm 0 in [6]) and compute the area of the resulting paverhex P_6.

Step 3: If $BO(P_6) = 1$, then P_6 is the optimal solution, else repeat Steps 1 and 2 using the second pair of congruent basic sides and select the paverhex whose area is minimal as the optimal solution.

Step 4: End.

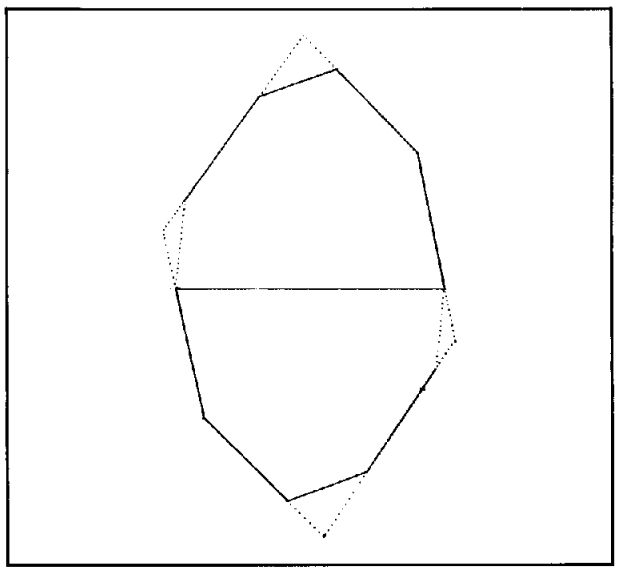

FIGURE 7. Circumscribing two congruent basic hexagons by a paverhex with minimal area addition.

7. SOLUTION OF THE NESTING PROBLEM

Algorithms 3 and 4 provide the procedures for an optimal circumscription of circular and basic hexagons by a parallel paver, respectively. The solution of the original problem of nesting congruent figures is obtained by paving the plane with the resulting parallel paver.

Algorithm 5. Efficient Nesting of Congruent Convex Figures.

Step 1: Circumscribe the figure by a convex n-sided polygon P_n with the desired degree of approximation. (See [9].)

Step 2: If $n = 3$ or $n = 4$, then set $P^* = P_n^*$, and go to Step 6, else continue.

Step 3: If $n = 5$, convert P_5 to P_6 by inserting a redundant vertex along the circumference of P_5.

Step 4: Circumscribe P_n by P_6 using Algorithm 0 of [6].

Step 5: Circumscribe P_6 by P^* using Algorithm 3 if $BO(P_6) = 0$ or Algorithm 4 otherwise.

Step 6: Use P^* to pave the plane.

Step 7: End.

Figure 3 illustrates the result of nesting a figure (solid line) in a plane using Algorithm 5. The dotted line is the approximation of the original figure by a polygon P_8 and the dashed line is the circumscribing paverhex. Since [6] yields an $O(n)$ algorithm, Algorithm 5 is also $O(n)$ because all of the steps in Algorithm 5, except Step 4, do not depend on n.

8. NUMERIC ILLUSTRATIONS

To be able to get numeric information about circumscription efficiencies, a random convex-polygon generator was developed. It is based on selecting random vertices within a unit circle and correcting their position until convexity is obtained. The detailed algorithm is described in [6].

To test whether the fact that the random polygons are confined within a unit circle causes any biased result, we also "stretched" them so that they were confined within ellipsoids of various ovalities (that is, various ratios of the major axis to the minor axis of the ellipsoid). The results are summarized in Table I; the left part has already appeared in [6]. The number of sides of the input polygon n ranges from 7 to 50. Each n is represented in the table by two lines. The first line for each n contains data about the average relevant quantities of 40 random polygons that are confined within a unit circle (ovality = 1), and the second line contains data about the average quantities of 120 polygons confined within ellipsoids of three different ovalities: 3, 9, and 27, 40 of each kind. Examination of each pair of lines shows nearly no difference between the results in each line. This indicates that the algorithms are not sensitive to the shape of the original figure.

As was discussed in [6], the circumscription efficiency of the polygon P_n by the hexagon P_6, $R(P_n)/P(P_6)$ decreases from 99.8% for $n = 7$ to 91.8 for $n = 50$ (fourth column). The coefficient of variation of the average efficiency is fairly stable at about 0.02 with no

obvious trend with respect to n, and it never exceeds 0.07 (sixth column).

The average efficiency of circumscribing P_6 by a parallel paver P^*, $R(P_6)/R(P^*)$, is about 95.6% (tenth column). The coefficient of variation tends to decrease with n from about 0.13 for $n = 7$ to 0.02 for $n = 50$, and with a mode at about 0.04 for n ranging from 13 to 20. The final average paving efficiency $R(P_n)/R(P^*)$ that is the product of the two efficiencies discussed above, varies from 96.8% (for $n = 8$) to 87.2% (for $n = 50$) with a mode at about 93% (for $n = 12$ until $n = 18$).

9. SUMMARY

An $O(n)$ algorithm for efficient nesting of congruent convex figures is presented and illustrated. The algorithm is based on two main steps. First, we circumscribe the given convex figure by a convex polygon P_n with a sufficiently large number of sides so that the waste is negligible. Second, we circumscribe the convex polygon P_n by a parallel paver hexagon or pentagon P^* that is then duplicated.

The limitations of this algorithm are as follows. First, the algorithm is efficient only with respect to parallel hexagon or pentagon pavers. Improved results might be obtained if we consider circumscribing the original figure by all of the 13 possible paver types. Since the average efficiency we now obtain is 93%, any significant improvement is unlikely.

A second limitation of the algorithm stems from the assumption that the plane is infinite, while in practice the board in which the figures are nested is of finite dimensions; hence, waste in the margin should also be considered. (See Figure 3.) If the area of the figure is small relative to that of the board, then this marginal waste is negligible.

TABLE I. Efficiency of Circumscribing n-sided Polygons by Paver Hexagons

n	Sample Size	Ovality	Average Efficiency $E(n, 6)$	$SD[E(n, 6)]$	SD/E	Average Polygon Area	SD of Area	SD/Area	Average E_P	$SD(E_P)$	SD/E_P	Final Efficiency
7	40	1.	99.79%	0.66%	0.007	1.12	0.75	0.67	95.05%	8.61%	0.091	94.9%
7	120	3, 9, 27	99.78%	0.75%	0.008	12.96	11.43	0.82	96.03%	13.15%	0.137	95.8%
8	40	1.	99.42%	1.36%	0.014	1.18	0.66	0.56	97.40%	14.36%	0.147	96.8%
8	120	3, 9, 27	99.46%	1.36%	0.014	14.11	10.12	0.65	97.24%	11.49%	0.118	94.9%
9	40	1.	98.92%	1.97%	0.020	1.27	0.50	0.39	95.91%	8.63%	0.090	95.4%
9	120	3, 9, 27	98.97%	1.98%	0.020	16.60	8.27	0.47	96.39%	9.25%	0.096	94.6%
10	40	1.	98.71%	1.81%	0.018	1.55	0.63	0.40	95.89%	5.77%	0.060	94.7%
10	120	3, 9, 27	98.79%	1.73%	0.018	18.28	8.51	0.45	95.88%	5.76%	0.060	93.6%
11	40	1.	98.35%	1.59%	0.016	1.60	0.64	0.40	95.18%	6.85%	0.072	93.7%
11	120	3, 9, 27	98.34%	1.64%	0.017	20.74	6.64	0.36	95.32%	6.07%	0.064	93.7%
12	40	1.	98.34%	4.32%	0.044	1.87	0.60	0.32	95.24%	4.79%	0.050	93.7%
12	120	3, 9, 27	97.98%	2.86%	0.029	22.33	6.99	0.32	95.37%	4.78%	0.050	93.4%
13	40	1.	97.45%	2.09%	0.021	1.93	0.64	0.33	94.97%	4.04%	0.042	92.6%
13	120	3, 9, 27	97.45%	2.37%	0.024	23.89	6.50	0.30	95.38%	4.36%	0.046	92.9%
14	40	1.	97.07%	2.29%	0.024	2.00	0.51	0.25	95.82%	3.61%	0.038	93.0%
14	120	3, 9, 27	97.15%	2.34%	0.024	25.10	5.39	0.24	96.49%	4.64%	0.049	92.8%
15	40	1.	97.10%	6.79%	0.070	2.15	0.51	0.23	95.53%	4.16%	0.044	92.8%
15	120	3, 9, 27	96.94%	3.92%	0.040	26.65	5.19	0.21	96.80%	3.39%	0.035	92.9%
16	40	1.	96.88%	5.52%	0.057	2.27	0.40	0.18	95.41%	3.69%	0.039	92.4%
16	120	3, 9, 27	96.64%	3.35%	0.035	27.55	4.98	0.19	95.46%	3.75%	0.039	92.3%
17	40	1.	96.61%	1.97%	0.020	2.24	0.44	0.20	95.23%	3.21%	0.034	92.0%
17	120	3, 9, 27	96.40%	2.10%	0.022	27.85	4.84	0.17	95.35%	3.21%	0.034	91.9%
18	40	1.	96.20%	2.60%	0.027	2.37	0.36	0.15	96.40%	6.02%	0.062	92.8%
18	120	3, 9, 27	96.15%	2.23%	0.023	29.37	3.95	0.15	96.05%	4.27%	0.044	92.4%
19	40	1.	95.78%	2.04%	0.021	2.43	0.40	0.17	95.43%	2.67%	0.028	91.4%
19	120	3, 9, 27	95.63%	2.03%	0.021	30.32	3.60	0.13	95.48%	2.72%	0.029	91.3%
20	40	1.	96.13%	2.52%	0.026	2.44	0.30	0.12	95.36%	3.57%	0.037	91.7%
20	120	3, 9, 27	96.13%	4.66%	0.048	30.48	3.04	0.11	95.57%	3.07%	0.032	91.9%
25	40	1.	94.87%	1.51%	0.016	2.66	0.17	0.06	95.56%	2.52%	0.026	90.7%
25	120	3, 9, 27	94.66%	2.28%	0.024	32.99	2.10	0.06	95.20%	2.55%	0.027	90.1%
30	40	1.	93.47%	1.63%	0.017	2.81	0.12	0.04	95.70%	2.63%	0.028	89.5%
30	120	3, 9, 27	93.66%	1.57%	0.017	34.35	1.60	0.04	95.45%	2.45%	0.026	89.4%
35	40	1.	93.98%	3.67%	0.039	2.87	0.09	0.03	97.33%	11.20%	0.115	91.8%
35	120	3, 9, 27	93.29%	2.05%	0.022	35.72	0.74	0.03	96.10%	5.20%	0.054	89.8%
40	40	1.	92.72%	1.71%	0.018	2.94	0.06	0.02	95.13%	2.28%	0.024	88.2%
40	120	3, 9, 27	92.49%	1.32%	0.014	36.22	0.68	0.02	94.95%	2.08%	0.022	87.8%
45	40	1.	91.60%	0.94%	0.010	2.99	0.04	0.01	95.83%	4.96%	0.052	87.8%
45	120	3, 9, 27	92.00%	1.18%	0.013	36.73	0.53	0.01	95.20%	3.09%	0.032	87.6%
50	40	1.	92.01%	1.10%	0.012	2.99	0.04	0.01	95.51%	1.86%	0.020	87.9%
50	120	3, 9, 27	91.80%	2.10%	0.023	37.09	0.45	0.01	94.94%	1.90%	0.020	87.2%

The algorithm is applicable only for the nesting of congruent convex figures. This disadvantage may be partly overcome by an efficient manual nesting of a few different figures, some of which may be nonconvex and/or with holes, such that the outer contour of the resulting bulk of manually nested figures is the same and convex.

The algorithm described in this paper for efficient nesting of congruent convex figures may serve as a basis for a computerized tool for industries in which congruent nesting is required, e.g., a numeric control (NC) system with which modern flame-cutting machines are equipped. The efficiency currently achieved by manual nesting ranges in the ballpark of 85%. The average efficiency of 93% achieved using our procedure together with the potential saving of man-time involved in nesting provide the economic justification for further research and development in the area of efficient nesting of various figures in a two-dimensional board.

REFERENCES

1. Adamowicz, M. and Albano, A. A solution of the rectangular cutting stock problem. *IEEE Trans. Syst. Man Cybern. SMC-6*, (1976), 302–310.
2. Albano, A. and Orisini, R. An heuristic solution of the rectangular cutting stock problem. *Comput. J. 23* (1979), 338–343.
3. Chazelle, B. The polygon containment problem. Dept. of Computer Science. Carnegie-Mellon University, November 1981.
4. Christophides, N. and Whitlock, C. An algorithm for two dimensional cutting stock problems. *Oper. Res. 25* (1977), 30–44.
5. DeCani, P. A note on the two dimensional rectangular cutting stock problem. *J. Oper. Res. Soc. 29* (1978), 703–706.
6. Dori, D. and Ben-Bassat, M. Circumscribing a convex polygon by a polygon of fewer sides with minimal area addition. *Comput. Gr. Image Process.* (in press).
7. Dyson, R.G. and Gregory, A.S. The cutting stock problem in the flat glass industry. *Oper. Res. Quart. 25* (1974), 41–53.
8. Freeman, H. and Shapira, R. Determining the minimum-area encasing rectangle for an arbitrary closed curve. *Commun. ACM 18* (1979), 409–413.
9. Sklansky, J. and Gonzalez, V. Fast polygonal approximation of digitized curves. *Pattern Recog. 11* (1979), 604–609.
10. Gardner, M. Some packing problems that cannot be solved by sitting on the suitcase. *Scientific American*, (Oct. 1979), 22–26.
11. Gilmore, P.C. and Gomory, R.E. A linear programming approach to the cutting stock problem. *Oper. Res. 9* (1961), 849–859.
12. Gilmore, P.C. and Gomory, R.E. Multistage cutting problems of two and more dimensions. *Oper. Res. 13* (1965), 94–120.
13. Gilmore, P.C. and Gomory, R.E. The theory and computation of knap-sack functions. *Oper. Res. 15* (1967), 1045–1075.
14. Grinbaum, B. and Shepher, G.C. The 81 types of isohedral tilings in the plane. *Math. Proc. Cambridge Philosophical Soc. 82* (1977), 122–146.
15. Haims, M.J. and Freeman, H. A multistage solution of the template layout problem. *IEEE Trans. Syst. Sci. Cybern., SSC-6* (1970), 145–151.
16. Herz, J.C. A recursive computing procedure for two dimensional stock cutting. *IBM J. Res. Dev. 16* (1972), 462–469.
17. Hinxman, A.I. A two dimensional trim-loss problem with sequencing contraints. *Proc. 5th Int. Joint Conf. Artif. Intell.* (1977), 859–864.
18. Kershner, R.B. On paving the plane. *Am. Math. Monthly, 75* (1968) 839–844.

CR Categories and Subject Descriptors: F.2.2 [**Analysis of Algorithms and Problem Complexity**]: Nonnumerical Algorithms and Problems—*geometrical problems and computations*; I.3.5 [**Computer Graphics**]: Computational Geometry and Object Modeling—*geometric algorithms*

General Terms: Algorithms, Theory

Additional Key Words and Phrases: optimal nesting

Received 7/82; revised 3/83; accepted 9/83

Author's Present Address: Dov Dori and Moshe Ben-Bassat, Faculty of Management, Tel Aviv University, Tel Aviv 69978 Israel

CORRIGENDUM. *George E. Forsythe Award.*

Jeffrey L. Eppinger. An empirical study of insertion and deletion in binary search trees. *Commun. ACM 26*, 9 (September 1983), 663–669.

Page 664: Column 2: The caption "**Figure 3. Examples of Insertion and Asymmetric Deletion.**" should read "**Figure 4. The Simulation Outer Loop.**"

Page 665: Column 1: The caption "**Figure 4. The Simulation Outer Loop.**" should read "**Figure 3. Examples of Insertion and Asymmetric Deletion.**"

Page 665: Column 1: In line 23, "... the approximation $\approx 1.386n\ I_n \lg n - 2.846n$." should read "... the approximation $I_n \approx 1.386n \lg n - 2.846n$."

Page 665: Column 2: In line 6, "In this paper, *T* empirically examine ..." should read "In this paper, I empirically examine ..."

Page 665: Column 2: In line 25, "... an *n*-node binary tree ..." should read "... an *n*-node binary tree ..."

Page 666: Column 2: In line 1, "... $IPL_{n,i}$ ..." should read "... $\overline{IPL}_{n,i}$ ..."

Page 668: Column 1: The caption "**Figure 10. Comparison Chart for Symmetric (Alternating) Deletions.**" should read "**Figure 11. Comparison Chart of the Asymptotic Values of IPL(n,i).**"

Page 668: Column 1: The caption "**Figure 11. Comparison Chart of the Asymptotic Values of IPL(n, i).**" should read "**Figure 10. Comparison Chart for Symmetric (Alternating) Deletions.**"

Page 669: Column 1 and Column 2: The titles above the third columns in Tables 1 and 2 should be written as "$\overline{IPL}_{n,>n^2}/I_n$".

Page 669: Column 1: In line 27, "$I_n = \Theta(n, \log n)$" should read "$I_n = \Theta(n \log n)$".