A Vision-Based Intelligent System for Packing 2-D Irregular Shapes

Alexandros Bouganis, *Student Member, IEEE*, and Murray Shanahan

***Abstract*—Packing two-dimensional shapes on a surface such that no shapes overlap and the uncovered surface area is minimized is an important problem that arises in a variety of industrial applications. This paper introduces an intelligent system which tackles the most difficult instance of this problem, where two-dimensional irregular shapes have to be packed on a regularly or irregularly shaped surface. The proposed system utilizes techniques not previously applied to packing, drawn from computer vision and artificial intelligence, and achieves high-quality solutions with short computational times. In addition, the system deals with complex shapes and constraints that occur in industrial applications, such as defective regions and irregularly shaped sheets. We evaluate the effectiveness and efficiency of the proposed method using 14 established benchmark problems that are available from the EURO Special Interest Group on Cutting and Packing.**

***Note to Practitioners*—Packing two-dimensional shapes on a surface such that no shapes overlap and the uncovered surface area is minimized is an important problem that arises in a variety of industrial applications, such as shipbuilding, textile, wood, plastic, sheet metal, and leather manufacturing. Although effective and efficient methods have been developed for packing rectangular parts on a rectangular sheet, this is not the case for the most difficult instance of the problem, namely packing irregular parts on an irregularly shaped sheet. This paper presents an automated system that utilizes techniques not previously applied to packing, drawn from computer vision and artificial intelligence, and succeeds in achieving high-quality solutions in short computation times. These techniques enable the system to "look" at the layout during the placement of the parts, and find potential matches of the unplaced ones within the unoccupied regions of the sheet. By integrating a vision system, this new method is: 1) fully automated since no human interference is needed (e.g., the input shapes of the parts and the sheet need not be given manually); 2) highly compatible with robotic applications within manufacturing where the initial positions and orientations of the parts are not precisely known, since the proposed method is not sensitive to the initial configurations of the parts; and 3) flexible in packing parts effectively on a surface with unexpected defective regions. We evaluate the performance of the proposed method using 14 established benchmark problems.**

***Index Terms*—Mathematical morphology, nesting, shape similarity, turning function, two-dimensional irregular packing problem.**

I. Introduction

In the two-dimensional packing problem, a number of two-dimensional shapes must be placed on a regularly or irregularly shaped surface such that no shapes overlap and the uncovered area of the surface is minimized. This problem arises in a variety of applications within shipbuilding, textile, wood, plastic, sheet metal, and leather manufacturing.

In the past, industrial solutions to the packing problem relied on dedicated machinery or manual production techniques. These days, when the market demands systems with high production flexibility, dedicated machinery is not a competent solution, while manual placement results in high labor costs and the repetitive movements can cause serious health problems [1]. Moreover, the quality of a manual solution depends on the effectiveness of the specific operator and on how well he or she will perform at that specific time. In this way, the quality of the solution is not consistent even for a single given instance of the problem. Taking into account the disadvantages of both methods, the present work aims to develop an automated system and endow it with the flexibility of a manual operator, using techniques drawn from computer vision and artificial intelligence.

The system described has the following features.

- It is considerably fast, while producing solutions of high quality.
- The system deals with arbitrarily complex shapes, without approximating the curves as polygons. The representation of an irregular part is independent of its complexity because the method uses a binary image to represent the part's shape rather than a formal mathematical description.
- In contrast to most published algorithms, the system described here can be straightforwardly adapted to a number of variations of the basic packing problem. For example, it can take defective regions and "quality zones" into account, which arise in the leather industry.
- Unlike many existing packing algorithms, the performance of the present system, with respect to both solution quality and computation time, is not sensitive to the initial orientations of the parts. This is especially important in a number of online industrial applications where the parts may not be preoriented with precision.

According to the recent typology of cutting and packing problems proposed by Wäscher *et al.* [2], the present algorithm mainly addresses the two-dimensional irregular open dimension problem. In [2], this problem type is defined as a problem category where a set of two-dimensional irregular small items (i.e., parts) have to be completely accommodated by a two-dimensional rectangular large object, the extension of which in one dimension (i.e., length) is variable and has to be minimized. This problem type has also been referred to as the irregular strip packing problem [3], [4] or the nesting problem [5]. It must be mentioned that although the present algorithm

Manuscript received November 24, 2005; revised April 26, 2006. This paper was recommended for publication by Associate Editor S. Kumar and Editor M. Wang upon evaluation of the reviewers' comments. This work was supported by EPSRC Project EP/C530683/1.

The authors are with the Department of Computing, Imperial College London, London SW7 2AZ, U.K. (e-mail: alexandros.bouganis@imperial.ac.uk; m.shanahan@imperial.ac.uk).

Digital Object Identifier 10.1109/TASE.2006.887158

mainly addresses the two-dimensional irregular open dimension problem, it can also be easily adapted to tackle variations of that problem type, where the large object has irregular shape and/or contains defective regions.

This paper is organized as follows. A brief review of previous work in the field is presented in Section II. In Section III, the proposed system is described in detail. Section IV presents experimental results on benchmark problems from the literature that demonstrate the capabilities of the proposed system.

II. Previous Work

The two-dimensional packing problem arises in several industrial applications and, as a consequence, a lot of research has been conducted in this area. The problem can be categorized into rectangular and irregular packing problems according to the parts' shapes. It has been shown in [6] that finding an optimal solution in the rectangular instance of the problem is NP complete. Thus, finding an optimal solution in irregular packing, which is more complex than the rectangular one, can be considered also as NP complete. Under this perspective, most researchers attempt to find a solution of good quality (but not necessarily the optimal one) in reasonable computational times using heuristic methods [7].

In the past, the majority of researchers were interested in solving the rectangular instance of the problem, where only rectangular shapes were considered, as it was less geometrically complex [8]–[10]. However, most industrial applications demand systems that can deal with irregular shapes.

One of the earliest methods in packing irregular shapes was carried out by Albano and Sappupo [11]. In their work, the problem of finding the optimal placement of a set of irregular parts is transformed into the problem of finding an optimal path in a state space. In the initial state, no part is placed in the sheet while, in the final state, all of them are. The A* algorithm is applied as a search method. The concept of the no-fit polygon (NFP) [12] is used to avoid overlaps during the placement of the parts.

Whelan and Batchelor [1] designed a packing system that consists of two major components: 1) the heuristic packer and 2) the geometric packer. The first component determines the ordering and orientation of parts relying on heuristics that take into account constraints that occur in different industrial applications and the parts' geometry. The second component places the oriented parts in the sheet based on morphological image processing operations. Their method can deal with parts and sheets of arbitrary shapes without formally describing them. A drawback of their method in comparison with ours is the lack of an intelligent mechanism for selecting the ordering and orientation of the parts. Indeed, their method determines the parts' sequence and orientation statically, based on the parts' geometry, and does not take into account the formation of the layout during the placement of the parts.

Jacobs [13] developed a method for rectangular packing in which a genetic algorithm produces a set of possible part sequences in each generation and a bottom-left heuristic is responsible for allocating the parts in the stock sheet accordingly. He extended his method to irregular polygon packing by first embedding the irregular parts to their minimum enclosing rectangles (MERs). A shrinking step that shifts the polygons closer to each other is then applied, as big gaps emerge between the polygons due to the approximation of the polygons with their MERs.

Lamousin and Waggenspack [14] proposed a feature-matching technique. In their method, a preprocessing stage takes place which extracts part features at varying levels of detail. Then, a search algorithm is applied to find possible matches between convex features of parts and concave features from the stock material. The large number of such combinations is a drawback of that method.

A systematic study was produced by Hopper [7], who tested a number of different approaches using benchmark problems from the literature along with nine new ones she introduced herself. In [7], a number of heuristic and metaheuristic approaches were compared for irregular packing. In addition, two commercial nesting packages were also assessed and, most of the time, outperformed her own approaches.

According to a single-pass algorithm proposed by Oliveira *et al.* [5], the final layout is generated by successively adding a new part to the set of parts that have been nested previously. The authors implemented 126 variants of their algorithm, considering various nesting strategies, evaluation functions, and local search techniques. Each version of their algorithm was tested on five data sets, and the best solutions achieved were kept. Given which version of their algorithm performs best on a certain benchmark problem, their algorithm can achieve good solutions in short times. However, since this information is not given before the evaluation procedure, all of the variants of their algorithm have to be tested to achieve the best solution, leading to long computational times.

Nielsen and Odgaard [15], based on previous research [16], applied a metaheuristic method called guided local search to the nesting problem. They mainly focused on the following decision problem: Given a set of parts (which they have named stencils), find if it can fit inside a given large object. Their method can easily address the strip-packing problem by following an iterated procedure; they begin with a rectangular large object of fixed width and initial length, and check whether the set of parts can fit inside it (decision problem). In case the answer is positive, the length of the large object is decreased. The authors repeat this loop (define the dimensions of the large object—solve the decision problem—decrease the length of the large object) until the parts cannot fit anymore to the large object. Nielsen and Odgaard [15] have evaluated their algorithm using six benchmark problems. The solutions achieved by their algorithm after a 1-h run were close to the best solutions achieved so far for the specific problems.

Recently, Gomes and Oliveira [3] developed a hybrid algorithm which utilizes simulated annealing on guiding the search, and linear programming models to carry out the local optimization of the resulting layouts during the search process. An interesting technique, called the multistage approach, was introduced in order to decrease their algorithm's computational time. Gomes and Oliveira [3] observed the existence of a high diversity of piece sizes in problems of the garment industry, and noted

TABLE I
NOMENCLATURE FOR THE SYMBOLS USED IN THE TEXT

Symbols	Description
PD	The Packing Density as defined in equation (1)
$F(x,y)$	Denotes the state of the pixel with coordinates (x,y). If a part can be placed in the underlying pixel, the value of $F(x,y)$ is 1. Else, $F(x,y)$ = 0.
$(\bar{x},\bar{y})$	Denotes the coordinates of the scene's center of mass
$\mu_{p,q}$	Denotes the (p+q)th-order central moment
$A \ominus B$	Denotes the erosion of the set A by the structuring element B
$R, R_1, R_2, R_3, R_4, R_5$	Rectangular regions on the layout used to determine existing void regions
M	Denotes the shape similarity measure
$\Theta_A(s)$	The turning function of shape A
PO	The list in which the unplaced parts are sorted according to their area in decreasing order
$PO[x]$	Stands for the part x of the list PO
$PO[x,j]$	Denotes the part $PO[x]$ in orientation j
VR	List of the void regions in the layout, sorted in respect of their position (from bottom to top)
$VR[y]$	The void region y of the list VR
$PD(x,y)$	The Packing Density achieved when the part $PO[x]$ is placed in the void region $VR[y]$ according to the Scene-Driven Approach

that the placement of the biggest ones is the most important in producing good solutions. In this respect, they divided the set of parts into groups, according to their size, and utilized the aforementioned computationally expensive search method only for the placement of the first group of parts (the biggest ones). In the subsequent stages, a faster hybrid algorithm [17], based on pure local search, was used for the placement of the other groups. The multistage approach was followed by their algorithm only for the benchmark problems taken from the garment industry. Their algorithm performs impressively well and achieves the best solutions so far for a number of benchmark problems. The main drawback of their method is its computational cost, which remains high.

In a recent publication of Burke *et al.* [18], a bottom-left fill heuristic has been applied for allocating the parts, and a number of techniques for resolving possible overlaps between the nested parts have been introduced. The part sequence is determined by utilizing local search methods, such as hill climbing and Tabu search. The quality of their solutions is often close to the best solutions achieved for a variety of benchmark problems, but the execution time of their method is high.

III. DESCRIPTION OF THE SYSTEM

The present system aims to tackle the disjunctive placement of N two-dimensional irregular shapes on a rectangular surface (i.e. large object) of fixed width and variable length, wasting as little resource material as possible and deriving a solution in a short time. In order to evaluate the resulting solutions, we consider as a performance measure the ratio of the total area of all the packed shapes to the area of the rectangle that encloses these shapes and has a fixed width equal to the width of the large object (1). This measure is called packing density (PD) and its maximum value is 1, which implies that there is no waste of resource material

$$\mathrm{PD} = \frac{\text{Area of Nested Parts}}{\text{Area of Enclosing Rectangle with Fixed Width}}. \quad (1)$$

To describe the way this performance measure works, let us examine the following cases.

- Two equally sized parts (e.g., A and B) are under consideration for placement, and the potential placement of part A leads to an enclosing rectangle of shorter length than does the potential placement of part B. According to the performance measure (1), the system favors the placement of part A. It is clear that in cases where the system considers parts of the same size, the performance measure defined above provides the same solution with the criterion of minimizing the layout's length.
- Two unequally sized parts are under consideration for placement, and the potential placements of both result in an enclosing rectangle of equal length. The system prefers the placement of the largest part.
- Two parts of different size are candidates for placement, and their potential placements lead to enclosing rectangles of unequal length. If the potential placement of the largest part leads to an enclosing rectangle of shorter length than that of the smallest part, the system favors the placement of the largest part. Otherwise, the placement which is chosen is subject to the following factors: 1) the ratio between the areas of the two candidate parts and 2) the ratio between the lengths of the resulting enclosing rectangles. The potential placement which gives the highest packing density is preferred.

The system has two modes of operation. In the first (off-line) mode, it acquires all of the part shapes and then places them in a sequential manner by deciding which part should be placed next and in which orientation. In the second (online) mode, the system acquires one part at a time and places it with the proper orientation. The offline mode, because of its flexibility, leads to better solutions than the online mode, but demands greater computational time. This paper focuses on the offline mode of operation.

A detailed description of the system follows. The operation of the system is divided into two phases: the prelayout phase and the layout phase. The symbols we use for describing the system are illustrated in Table I.

A. Prelayout Phase

In the prelayout phase, a vision system is used for acquiring and storing the images of the parts and the sheet. It is assumed that the parts do not overlap during the acquisition phase and that

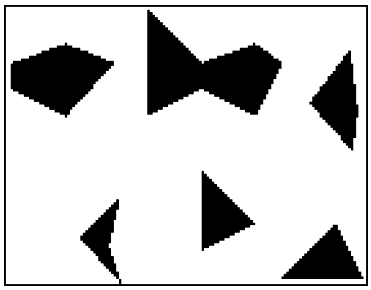

Fig. 1. System acquires the discrete representation of the parts.

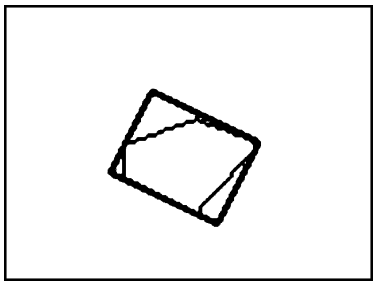

Fig. 2. MER of a part.

the background has a distinct and different color from the parts (Fig. 1). As a result, the system does not need to have *a priori* knowledge of the shape of each part, as it would, for example, if the parts could occlude each other.

The sheet and each part are represented by a set of pixels. For each part, the system determines the dimensions and orientation of its MER (Fig. 2). The part's shape might be irregular but the points of that shape which do not belong to its convex-hull [19] cannot affect the dimensions of its MER. Therefore, the method computes the convex-hull of the part shape and searches for its MER by taking into account only the points that belong to the convex-hull, reducing the required computational time. The algorithm of Toussaint [20] is employed for finding the MER of the shape in $O(n)$ time, where n is the number of vertices in the convex-hull. It is worth mentioning that the system does not replace the part shapes by their MERs. Rather, it uses MERs as additional information for orienting the parts. Replacing irregular shapes with their MERs would cause the method to generate less effective solutions, wasting significant quantities of material.

B. Layout Phase

The placement of the parts follows a single-pass placement strategy and takes place in a sequential manner during the layout phase, which entails that the system considers only one layout.

In the beginning, the system determines the initial ordering and orientation of the parts based on their geometrical features. Thus, at first, the system follows an object-based approach where its decisions rely only on characteristics of the part shapes. However, its decisions are not fixed and can change during the placement procedure. In particular, while the system places parts in the sheet according to the object-based approach, it "looks" at the layout and checks if a part different from the one recommended by this approach would be the best one to place next. In order to accomplish that, the system uses another approach called the scene-driven approach. The scene-driven approach decides which part should be placed next by taking into account not only the geometric characteristics of the parts, but also the potential matches of the remaining parts with the unoccupied, by the already nested parts, regions of the sheet. The system, by interleaving the two approaches, places parts following a dynamic ordering of them. A summary of the algorithm is presented in Fig. 9. More detailed descriptions of the functions provided in Fig. 9 are presented below.

1) Object-Based Approach: The object-based approach, based on the geometrical characteristics of the parts, makes an initial recommendation of the sequence in which the parts should be placed, and for each part, recommends an orientation and a position. The object-based approach is similar to the approach proposed by Whelan and Batchelor [1].

Optimization methods, such as genetic algorithms ([7], [13]) and simulated annealing [7], have been tested in the literature for finding a near-optimal solution for sequencing the parts. However, the prohibitive computational cost of these algorithms violates one of the main design criteria of the present method, namely that it should be fast. So the system described here avoids these techniques. Instead, the sequence of the parts is determined using a heuristic criterion (i.e., ordering according to their area [21], [22]).

a) Orientation of the parts: The object-based approach finds potential orientations for the parts, based on their MERs. For each part, four orientations are examined. These orientations are determined in such a way, so that one of the MER's sides is aligned with the axis of the scene's least moment of inertia [23].

The angle θ of axis of the scene's least moment of inertia is calculated as follows [23]:

$$\theta = \frac{1}{2} \cdot \arctan\left(\frac{2\mu_{1,1}}{\mu_{2,0} - \mu_{0,2}}\right) \tag{2}$$

where $\mu_{p,q}$ are the central moments. To calculate the central moments, let us consider $x, y \in \mathbb{Z}$ as the coordinates of a pixel, and the following function $F(x,y)$:

$$F(x,y) = \begin{cases} 0, & \text{if the pixel } (x,y) \text{ belongs} \\ & \quad \text{to the background} \\ 1, & \text{otherwise.} \end{cases} \tag{3}$$

Pixels that belong to the background correspond to regions where parts cannot be placed. This could be due to the occurrence of defective regions inside the sheet or because of a potential irregularly shaped sheet. Given the coordinates $(\bar{x}, \bar{y})$ of the center of the foreground region (4), (5), the central moments are calculated by (6)

$$\bar{x} = \frac{\sum_{x\in\mathbb{Z}} \sum_{y\in\mathbb{Z}} x \cdot F(x,y)}{\sum_{x\in\mathbb{Z}} \sum_{y\in\mathbb{Z}} F(x,y)} \tag{4}$$

$$\bar{y} = \frac{\sum_{x\in\mathbb{Z}} \sum_{y\in\mathbb{Z}} y \cdot F(x,y)}{\sum_{x\in\mathbb{Z}} \sum_{y\in\mathbb{Z}} F(x,y)} \tag{5}$$

$$\mu_{p,q} = \sum_{x\in\mathbb{Z}} \sum_{y\in\mathbb{Z}} (x-\bar{x})^p \cdot (y-\bar{y})^q \cdot F(x,y). \tag{6}$$

The method employs a greedy algorithm for deciding which of the possible orientations is the most appropriate. In partic-

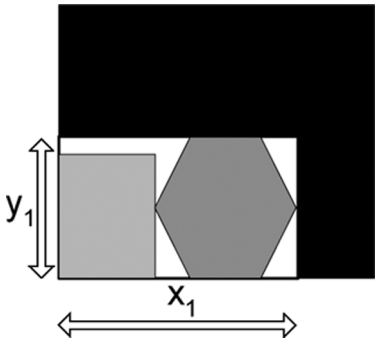

Fig. 3. Minimum enclosing rectangle of the nested parts.

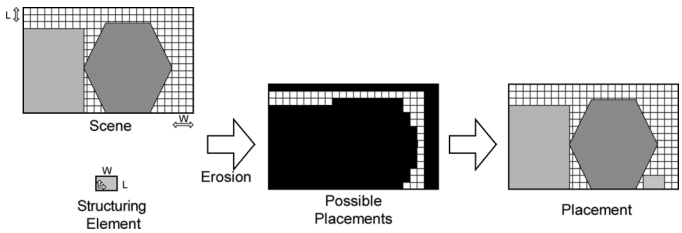

Fig. 4. Placement of a part according to the object-based approach. The image in the middle illustrates the pixels (black pixels) in which the origin of the structuring element cannot be placed.

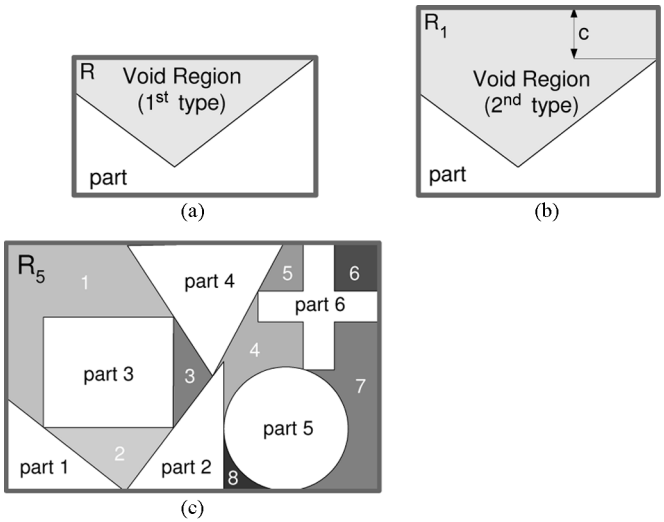

Fig. 5. (a)–(c) depict the three types of void regions. In (c), different shades of gray are used to depict distinct third-type void regions. Moreover, dark gray frames have been utilized to illustrate the rectangles R, R_1, and R_5 in (a), (b), and (c) respectively.

ular, the method orientates the part in each potential orientation, and visualizes its placement for each orientation. The packing density (1) is used for determining the quality of each placement. The object-based approach suggests that the system carry out the placement which gives the best possible packing density (see Steps 3)–5) in the algorithm's summary in Fig. 9).

b) Placement of a part: Given the orientation of the part from the previous step, the system finds feasible locations where the part can be placed. To achieve that, the system uses operations of mathematical morphology [23]–[25]. Let us say that the corner coordinates of the minimum enclosing rectangle of the already placed parts are (Fig. 3) $([0,0],[x_1,0],[x_1,y_1],[0,y_1])$. To add a new part in the layout, the system defines as a scene a rectangular region with the following coordinates for its corners: $([0,0],[x_1+W,0],[x_1+W,y_1+L],[0,y_1+L])$, where L, W are the length and the width of the new part, respectively (Fig. 4). In case the width of the large object is shorter than x_1+W, we set the width of the rectangular scene equal to the width of the large object. As the scene has been stored in the system as an image, we can denote the scene as the image set A. The part constitutes the structuring element B. The origin of the structuring element B is chosen to be the bottom-left corner of the part's MER. It must be noted that the origin does not necessarily have to be inside the part's shape. The operation of erosion[1] is applied to the image set A by the structuring element B, denoted by $A \ominus B$ (7)

$$A \ominus B = \{x \in \mathbb{Z}^2 | B_x \subseteq A\} \tag{7}$$

where B_x denotes the translation of B so that its origin is located at x. By this operation, the set of all elements x of $\mathbb{Z}^2$ such that B translated to x is contained in A are identified [24]. Thus, the result of this operation is another image where feasible locations for the part to be placed have been identified. For each possible placement, the system calculates the packing density (1), and it then chooses the position for which this is maximal. Finally, B is rendered with its origin located in the chosen position (Fig. 4).

2) Scene-Driven Approach: The scene-driven approach is an alternative to the object-based approach for nesting the parts. The description of the scene-driven approach takes place in two subsections. In the first subsection, we provide some definitions and techniques necessary to describe the scene-driven approach. The second subsection presents the scene-driven approach in detail.

a) Definitions and techniques: The scene-driven approach extracts information from the layout in order to determine which part should be placed next and in which orientation. The characteristic of this approach is the utilization of a shape similarity criterion between void regions of the layout and remaining parts. How these void regions are defined will now be explained.

After the placement of the $Part_i$, the system generates a number of void regions. Consider the minimal rectangle R that encloses the nested $Part_i$ and has one of its sides aligned with the axis of the scene's least moment of inertia. There are three types of void regions that are generated with the placement of a new part, describing different features in the layout. The first type corresponds typically to the smallest void regions and describes salient features of the part that has just been placed. It includes a number of regions where each of them is a four-neighbor connected set of uncovered pixels and is restricted by the bounds of the rectangle R (Fig. 5). The union of these regions and of the placed part is the rectangle R. A void region of the second type is usually larger than a first-type void region and aims to describe the most salient features of the placed part and the configuration of its neighborhood. For defining the second type of void region, four other rectangles R_1, R_2, R_3, and R_4 must be defined. Each rectangle emerges from the initial one R

[1]The operation of erosion is known in other knowledge fields as Minkowski sum and no-fit polygon.

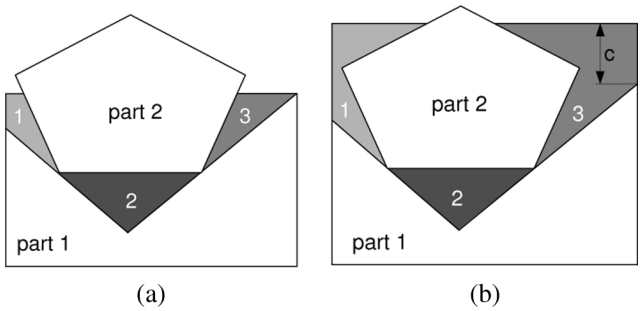

Fig. 6. (a) and (b) illustrate how the void regions of Fig. 5(a) and (b) have been modified after the placement of part 2, respectively. Different shades of gray are used to depict distinct void regions. The void regions of (a) and (b) belong to the first and second type of void regions with respect to part 1, respectively.

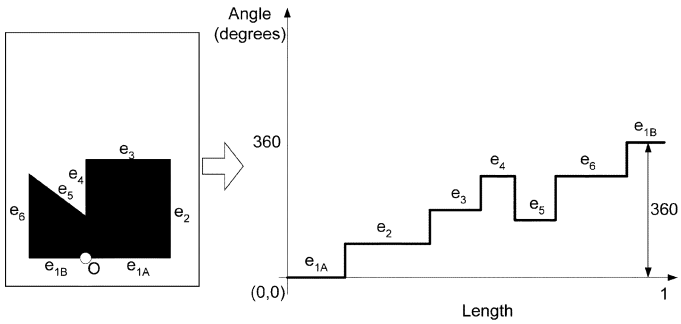

Fig. 7. Turning function.

by increasing one of its sides, respectively, by a constant c. For each rectangle, new void regions are generated following the same rule as before (Fig. 5). The third type of void region, due to its great area, mainly describes holes in the layout rather than features of a specific part. For defining the third type of region, the rectangle R_5 is defined which encloses all of the parts that have been placed, has one of its sides aligned with the axis of the scene's least moment of inertia, and has the minimum possible area. The new void regions that are generated are all of the 4-neighbor connected sets of uncovered pixels that are bounded from the limits of the rectangle R_5 (Fig. 5). As illustrated in Fig. 6, the third type of void region is not the only one that takes into account the neighbors of $Part_i$; the first and the second type also consider them.

There are many possible ways to characterize void regions, and alternatives to the three definitions presented here could be used. However, experiments that have been conducted and are presented at the end of this paper show that this definition of void regions is effective.

An important difference between our approach and the approach of Lamousin and Waggenspack [14] is that our method defines void regions in the layout and compares them with the remaining parts. In contrast, the method in [14] compares only local features (three edges and their interior angles) of the parts and layout. In this way, there are fewer resulting combinations in our method, and the decision made is to be more effective since placements that derive from feature matching are very likely to lead to collisions, as only local information is provided.

Having defined what constitutes a void region in the layout, we can now describe how the scene-driven approach attempts to find effective placements of the remaining parts. In order to decide whether a placement of a part to a void region is likely to produce a solution of good quality, the system considers the similarity of their shapes and the ratio of their areas. The following equation is employed as an evaluation function (EF) for each possible placement

$$\mathrm{EF} = \begin{cases} M, & \text{if } 0.7 \leqslant \frac{\text{Area of Void Region}}{\text{Area of Part}} \leqslant 2.5 \\ 1000, & \text{otherwise} \end{cases} \quad (8)$$

where M is a measure that expresses the similarity between the part's shape and the region's shape and its value decreases as the two shapes become more similar. In the case where the shapes are completely alike, M is zero. A solution with a lower EF value is preferred. In case the area ratio of the void region and the part is not between the thresholds which we have defined (i.e., 0.7 and 2.5), the EF takes a great value (i.e., 1000) in order for the system to prefer other possible solutions. It must be stressed that the evaluation function does not accurately determine the quality of a placement but provides a quick and approximate measure. The actual quality of a potential placement can be determined by first placing the part and then applying the performance measure (i.e., packing density) to the layout. In what follows, the method with which the system derives the measure M and the proper orientation of a part is described.

Shape Similarity—The problem of shape similarity has been well studied in the literature of computer vision, and a number of different methods have been proposed [26]–[30]. Some of the most popular techniques are grid-based methods, moment invariants, Fourier descriptors, and the turning angle. In this paper, the turning angle method [29] is adopted as it has several important properties.

- The similarity measure of two shapes is invariant under translation, rotation, and change of scale.
- It is carried out in time $O(mn\log(mn))$, where m and n are the numbers of vertices of the two polygonal shapes.
- It is insensitive to small variations of the shape, and matches human perceptual judgements.

The turning function $\Theta(s)$ of a shape A measures the angle between the counter-clockwise tangent of A and a reference axis while moving along the boundary of A, as a function of the arc length s, measured from an arbitrary chosen reference point O on A's boundary (Fig. 7). The horizontal axis can be considered as a reference. $\Theta(s)$ increases with left-hand turns and decreases with right-hand turns. The arc length of A is rescaled so that the total perimeter length is 1 ($\Theta[0,1] \rightarrow \mathbb{R}$). The function Θ has the following properties [29]: 1) For a simple closed curve $\Theta(1) = \Theta(0) + 2\pi$. 2) In case that A is a polygon, Θ is piecewise-constant, with discontinuities at the vertices of A. 3) Θ is invariant under translation and change of scale of the shape. 4) Rotation of shape A by an angle w counter-clockwise corresponds to a vertical uppershift of $\Theta(s)$ by w (clockwise rotation of the shape corresponds to a vertical lower-shift of Θ). 5) Changing the location of the origin O by $t \in [0,1]$ along the perimeter of A, results in a new turning function given by $\Theta(s+t)$, which means that the function Θ is horizontally shifted by t. More details about the turning function can be found in [29].

Let us consider two polygons A and B. The resemblance of the two shapes can be measured by taking the minimal distance between the turning functions $\Theta_A(s)$ and $\Theta_B(s)$ (according to the L_2 metric [29]) with respect to vertical and horizontal shifts. Thus, the similarity measure of the two shapes is given by

$$M = \min_{\theta_0 \in \mathbb{R}, t \in [0,1]} \left(\int_0^1 \left(\Theta_A(s+t) - \Theta_B(s) + \theta_0 \right)^2 ds \right)^{\frac{1}{2}}. \tag{9}$$

As proven in [29], the angle θ_0 that minimizes the distance between A, B is given by

$$\theta_0 = \int_0^1 \left(\Theta_B(s) - \Theta_A(s) \right) ds. \tag{10}$$

The system approximates the part shapes and the void regions with polygons using the Douglas–Peucker algorithm [31] in order to employ the method of Arkin *et al.* [29]. It must be stressed that the system uses polygonal approximations of the shapes only temporarily, for measuring their resemblance. Excluding that, the system in the whole procedure considers the shapes as they have been acquired by the vision system, without any approximation (except the one due to digitization) and, therefore, there is no influence on the quality of the solution.

Another aspect of the problem that the system takes into account is the case where there is a partial shape similarity between a void region and a part. That occurs when the part's shape is not similar to the shape of the whole void region but only with a part of this region. In this case, the system proceeds with an iterated elimination of a set of vertices of the void region in order to achieve a closer resemblance between the part shape and the region shape. The elimination of the region's vertices obeys the following constraints: 1) after the elimination of a vertex, the resulting shape must be a simple polygon [19] (i.e., there is no pair of nonconsecutive edges sharing a point). In this way, only meaningfully shaped void regions emerge; 2) the elimination of a vertex must lead to a shape with a smaller area than the original one so that occupied areas are not considered as void; and 3) the vertex elimination of the polygonized void region must lead to a shape that is more similar to the part shape.

To summarize, the inputs to the Arkin *et al.* [29] algorithm are the turning functions of a part and a void region. The algorithm returns the resemblance of the two shapes and the angle that the part must be rotated to achieve that degree of resemblance.

b) Scene-driven approach description: Initially, the scene-driven approach places the void regions in a list according to their position in the layout. The most bottom void region is examined first. The system evaluates the placement of each remaining part with respect to the void region under examination according to the evaluation function. If none of the parts gives a value of the evaluation function that reaches a predefined threshold (called shape similarity threshold), the system examines the next region in the list. When a part reaches the threshold, the method orientates it according to the output angle of the Arkin *et al.* algorithm and visualizes its placement in the specific void region. Similar to the object-based approach, the placement of the part occurs by applying the erosion operation of mathematical morphology.

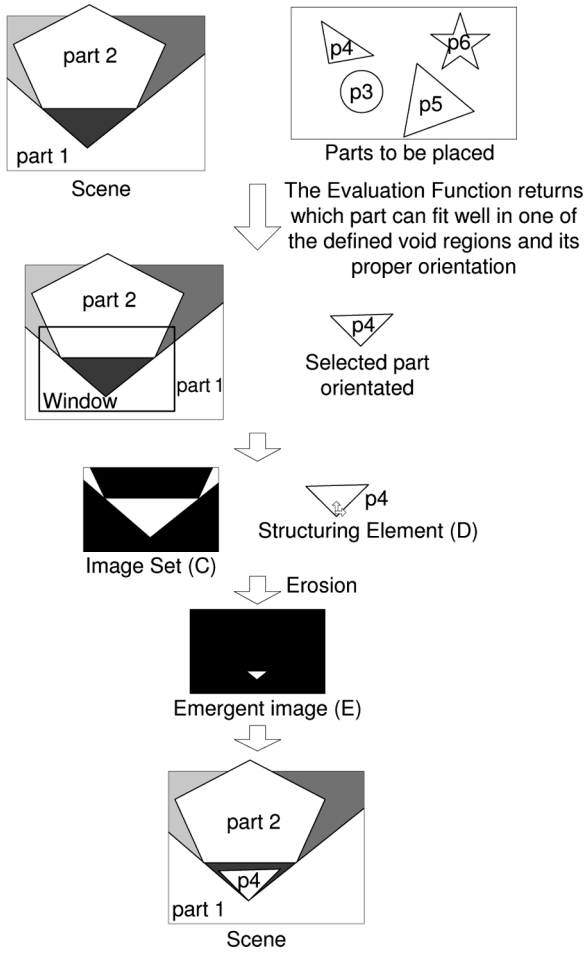

Fig. 8. Procedure of the scene-driven approach.

The image set C is a window of the scene that includes the void region. The structuring element D is the part which matches well with the void region, in the given orientation. The operation of erosion is applied to the image set C by the structuring element D. The result of this operation is another image E where the set of all feasible placements for the part in the image set C have been identified. The system takes into account only the subset of the feasible locations that belong to the void region of interest, and places the part in the bottom-left location of this subset (Fig. 8). The system then quantifies the quality of the layout in terms of packing density and determines whether the potential placement of the part should be accepted or not.

C. Outline of the Overall Algorithm

In summary, the overall algorithm works as follows. First, the prelayout phase takes place where the system uses a vision system to acquire the shapes of the scene and the parts, and stores them as binary images. The system extracts additional geometrical information from each part shape by determining the dimensions and the orientation of its minimum enclosing rectangle. As has been stressed, the shapes are represented by binary images, and their MERs are used only as additional information to the system. Carrying out the prelayout phase, the system is now able to start allocating the parts in the scene.

Algorithm:

Step 1: The system acquires the shapes of the parts to be placed. Set $Number\_of\_Placed\_Parts$ = 0.

Step 2: The parts are sorted according to the size of their areas in decreasing order. Let PO denote the Part Ordering: PO=[Part$_1$, Part$_2$,. . .,Part$_N$], where N is the number of the parts to be nested. Let $PO[x]$ stand for Part$_x$.

Step 3: Find the four potential orientations of $PO[1]$ according to section III-B.1. Let $PO[1, j]$ denote the part $PO[1]$ in orientation j.

Step 4: Repeat for all possible orientations j = 1 to 4:
-Find where the part $PO[1, j]$ would be placed, following the placement strategy of section III-B.1.
-Evaluate the potential placement using the packing density (1).
End Repeat

Step 5: If $Number\_of\_Placed\_Parts$ = 0, then
-Proceed to the actual placement of the part $PO[1]$ in the orientation that maximizes the packing density.
-Delete the part $PO[1]$ from the list PO.
-Set $Number\_of\_Placed\_Parts = 1$.
-If $PO = \oslash$, then
TERMINATE
Else
Go to Step 3. A new part is defined as $PO[1]$.
End If
Else
-The Object-Based Approach favors the placement of $PO[1]$ in the orientation which gives the maximum packing density.
-Set the Packing Density Threshold equal to the maximum packing density achieved.
End If

Step 6: Determine the set of resulting void regions in the layout according to section III-B.2. Put them in a list and sort them according to their position in the layout. Let VR=[Void_Region$_1$,Void_Region$_2$,. . .,Void_Region$_K$] denote the list of the void regions, where K is the number of the void regions. Let the y^{th} void region be denoted by $VR[y]$. Set $t = 0$ and $y = 1$.

Step 7: While $y \leq K$ and $t \neq 1$
Repeat for all the unplaced parts in the list PO, $x = 1$ *to* $Number\_of\_Remaining\_Parts$:
-Evaluate the matching of the void region $VR[y]$ with the part $PO[x]$ according to the Evaluation Function (8), $EF(x, y)$.
-If the value $EF(x, y)$ meets the Shape Similarity Threshold
-Find where the part $PO[x]$ would be placed, following the placement strategy of section III-B.2.
-Evaluate the potential placement using the packing density (1), $PD(x, y)$.
End If
End Repeat
Let x_{chosen} be the value of x that maximizes $PD(x, y)$.
If the value $PD(x_{chosen}, y)$ is greater than the Packing Density Threshold, then
-Set $y_{chosen} = y$.
-Set t=1.
End If
Set $y = y + 1$.
End While
If $t = 0$ (so no correspondence has been found between one of the void regions and remaining parts that gives better packing density than the Packing Density Threshold), then
-Place the part $PO[1]$, as it was suggested by the Object-Based Approach in Step 5.
-Delete the part $PO[1]$ from the list PO.
-Set $Number\_of\_Placed\_Parts = Number\_of\_Placed\_Parts + 1$.
-If $PO = \oslash$, then
TERMINATE
Else
Go to Step 3. A new part is defined as $PO[1]$.
End If
End If

Step 8: Place the part $PO[x_{chosen}]$ in the void region $VR[y_{chosen}]$ according to section III-B.2.
-Delete the part $PO[x_{chosen}]$ from the list PO.
-Delete the void region $VR[y_{chosen}]$ from the list VR.
-Set $Number\_of\_Placed\_Parts = Number\_of\_Placed\_Parts + 1$.
-If $PO = \oslash$, then
TERMINATE
Else
Go to Step 3. A new part is defined as $PO[1]$.
End If

Fig. 9. Summary of the proposed algorithm.

The object-based approach is applied first, and the parts are sorted according to the size of their areas in decreasing order. Each part is oriented such that one side of its MER aligns with the axis of the scene's least moment of inertia. The first part

is placed in the bottommost, leftmost feasible position of the layout using mathematical morphology. After placement of the first part, the scene-driven approach is applied to determine the void regions in the layout. The void regions are sorted according to their position in the layout, with the bottommost region being first. The method, by visualizing the placement of the next part according to the object-based approach, determines a packing density threshold. Doing so, the method will proceed to an alternative placement of the same or another part, only if such placement gives greater packing density than the packing density threshold. In case an alternative placement gives a packing density equal to the packing density threshold, the placement, which fixes the part under consideration in the bottom-most part of the layout is preferred.

The system examines whether there is a correspondence between one of the resulting void regions and one of the remaining parts, based on their shape and area. The search starts by evaluating the match between each remaining part with the first region in the list. If the system concludes that a good placement has not been found, it considers the next void region of the list. When there is a correspondence between a part and a void region, the system visualizes the placement of the appropriately oriented part in the corresponding void region, and the packing density achieved is measured. If this packing density is greater than the packing density threshold defined above, the system proceeds to the placement of that part to the underlying void region. There are cases where a void region matches well with more than one part. When this occurs, the method keeps the one which gives the best packing density. After the actual placement of the new part, the list with the remaining parts is updated, and a new set of void regions is generated. The method repeats the procedure of determining a packing density threshold, finding correspondences between void regions and remaining parts, and proceeding to the placement which gives the best packing density, until all of the parts are placed. A summary of the proposed method has been presented in Fig. 9.

Whelan and Batchelor [1] proposed a method that is similar to our object-based approach. The main difference between our method and theirs is that ours also enjoys the advantages of the scene-driven approach. Our algorithm is endowed with an intelligent mechanism for dynamically finding potential matches between remaining parts and unoccupied regions on the sheet during the packing process.

IV. Performance Evaluation

The performance of any nesting algorithm depends significantly on the parts' geometry, especially in the case of irregular packing. Thus, a packing method can only be evaluated properly when it is compared against other methods using benchmark problems from the literature. The proposed method is here evaluated using 14 benchmark problems that are available from the EURO Special Interest Group on Cutting and Packing (ESICUP) website [32], and is compared with state-of-the-art algorithms. The benchmark problems presented below can be categorized as artificial or garment problems. Their names derive from the way they have been produced; in particular, the former ones have been produced artificially, while the latter ones have been taken from the garment industry.

Nine of the benchmark problems under consideration were produced by Hopper [7] and include convex and nonconvex polygonal shapes. Their size varies from 15 to 75 parts. The benchmark problem with the lowest number of parts (i.e., 15) is poly1a. The other problems are divided into two sets, namely A and B. The set A includes four other problems (labeled with "a") that consist of multiples of 2 to 5 of the polygon set in poly1a, depending on their size. The set B also includes four other benchmark problems labelled with "b." In contrast to the problems of set A, these problems contain each polygon only once. It must be noted that Hopper considers a constraint according to which parts can be rotated incrementally by 90°. As stated in her thesis [7], without this constraint the search space is very large and the resulting solutions from her approaches are worse. However, the present method is not affected by this relaxation of the problem and this constraint is not enforced. The proposed method has also been tested on one more artificial benchmark problem, and four real-life instances taken from the garment industry.

In order to compare our method with existing ones, we measure the quality of solutions utilizing the packing density (1). The quality of solutions in many publications has been measured by the required length of the sheet. However, knowing the total area of parts, the width, and the length of the sheet, we can calculate the equivalent packing density (1). The experiments conducted in this paper have been performed on a PC with a 3-GHz Intel Pentium 4 processor and 1-GB RAM. It is also worth mentioning that all of the experiments have been carried out using the same value for the shape similarity threshold (i.e., 0.4). This value was chosen empirically because it gives good results for a wide variety of benchmark problems. Even better results can be achieved by tailoring this threshold for each benchmark problem.

Tables II–IV summarize the performance of this work and other methods from the literature, when they are applied to the aforementioned benchmark problems. Moreover, Fig. 10 depicts the layouts generated by our method. As illustrated in Tables II–IV, our algorithm achieves solutions of comparable quality with the best solutions achieved up to now, in much shorter times. It is easy to observe that the metaheuristic approaches presented in [3], [15], and [18] achieve, most of the time, the best solutions in literature, but their computational cost is high. In particular, Gomes and Oliveira [3] achieve on average 3.7% better packing density than our algorithm, Burke *et al.* [18] 2.6%, and Nielsen and Odgaard [15] 2.5%. As illustrated in Tables II–IV, their execution times are of several minutes to many hours, while our algorithm needs only a few seconds.

Our algorithm outperforms Hopper's approach [7], both in quality and efficiency. The computational time of the proposed method was expected to be much lower since our algorithm is based on single-pass placement and not on a computationally expensive approach such as the one used by Hopper, namely a genetic algorithm. Furthermore, the proposed method leads to solutions of better quality (on average 9.7%) due to the superiority of the proposed placement strategy in comparison with other traditional heuristics. Commercial nesting packages, such as NestLib and SigmaNest, have also been assessed on the same benchmark problems by [7]. Our algorithm outperforms both of

TABLE II
COMPARISON OF THE PROPOSED METHOD WITH EXISTING ALGORITHMS ON THE SET A OF HOPPER'S BENCHMARK PROBLEMS. THE EFFECTIVENESS AND EFFICIENCY OF THE METHODS UNDER COMPARISON ARE PRESENTED. IN SOME CASES, THE EFFICIENCY OF A METHOD IS NOT PRESENTED AS IT WAS NOT AVAILABLE FROM ITS SOURCE. MOREOVER, THE REQUIRED COMPUTATIONAL TIME OF THE BURKE *et al.* ALGORITHM WAS COMPUTED BY TAKING INTO ACCOUNT THE AVERAGE TIME PER ITERATION OF THEIR ALGORITHM, ITS REQUIRED NUMBER OF ITERATIONS FOR ONE RUN (I.E., 100), AND THE NUMBER OF RUNS FOR A VERSION OF THEIR ALGORITHM (I.E., 10). IT MUST BE NOTED THAT THIS IS THE BEST POSSIBLE SCENARIO FOR THEIR METHOD, SINCE IN REALITY THE AUTHORS ACHIEVE THESE SOLUTIONS BY REPEATING THE AFOREMENTIONED PROCEDURE FOUR TIMES FOR EACH PROBLEM, MODIFYING THEIR ALGORITHM IN ORDER TO EXAMINE DIFFERENT SEARCH METHODS AND HEURISTICS, AND KEEPING THE BEST SOLUTIONS. FINALLY, THE GENETIC ALGORITHM IN HOPPER [7] IS CONSIDERED

Benchmark Problem	poly1a	poly2a	poly3a	poly4a	poly5a
Number of Parts	15	30	45	60	75
Fixed Width	40	40	40	40	40
Source	Hopper [7]	Hopper [7]	Hopper [7]	Hopper [7]	Hopper [7]
Method	**Quality of solution (Time)**				
Proposed Method	65.4% (5 sec)	67.3% (12 sec)	68.8% (19 sec)	70.7% (26 sec)	71.2% (38 sec)
Hopper [7]	59.6% (1380 sec)	61.4% (2940 sec)	60.4% (4500 sec)	60.2% (7020 sec)	59.3% (8880 sec)
SigmaNest [7]	59.9%	67.5%	68.3%	69.1%	69.4%
NestLib [7]	69.7%	68.1%	76.1%	72%	71.6%
Burke et al. [18]	73.2% (360 sec)	72.8% (1240 sec)	73.8% (2010 sec)	74.6% (2430 sec)	73.9% (5040 sec)

TABLE III
COMPARISON OF THE PROPOSED METHOD WITH EXISTING ALGORITHMS ON THE SET B OF HOPPER'S BENCHMARK PROBLEMS

Benchmark Problem	poly2b	poly3b	poly4b	poly5b
Number of Parts	30	45	60	75
Fixed Width	40	40	40	40
Source	Hopper [7]	Hopper [7]	Hopper [7]	Hopper [7]
Method	**Quality of solution (Time)**			
Proposed Method	71% (14 sec)	72.5% (21 sec)	74.7% (30 sec)	72.6% (43 sec)
Hopper [7]	57.7%	57.3%	67.9%	65.6%
SigmaNest [7]	68.3%	71.8%	71.7%	71.4%
NestLib [7]	67.3%	73%	73.1%	72.4%
Burke et al. [18]	75.4% (2500 sec)	74.9% (4260 sec)	74.8% (8240 sec)	75.8% (14700 sec)

TABLE IV
COMPARISON OF THE PROPOSED METHOD WITH EXISTING ALGORITHMS ON ADDITIONAL ARTIFICIAL AND GARMENT BENCHMARK PROBLEMS. THE TIMES STATED IN THIS TABLE FOR THE METHOD OF GOMES AND OLIVEIRA [3] ARE THE AVERAGE TIMES REQUIRED FROM THEIR ALGORITHM FOR ONE RUN ON EACH BENCHMARK PROBLEM. IN REALITY, GOMES AND OLIVEIRA [3] RUN THEIR ALGORITHM 20 TIMES FOR EACH BENCHMARK PROBLEM AND THEN KEEP THE BEST SOLUTION AMONG THESE RUNS. THE ACTUAL TIMES OF THEIR ALGORITHM CAN BE COMPUTED BY MULTIPLYING THE TIMES MENTIONED IN THIS TABLE WITH 20. AS IT CONCERNS OLIVEIRA *et al.* [5], THE TIMES MENTIONED IN THIS TABLE REFER ONLY TO THE EXECUTION TIMES OF THE BEST VARIANT OF THEIR ALGORITHM. THE AUTHORS IN [5] CONCLUDE WHICH IS THE BEST ONE FOR EACH BENCHMARK PROBLEM, AFTER TESTING 126 VERSIONS OF THEIR ALGORITHM. HENCE, THE ACTUAL TIMES REQUIRED BY THEIR METHOD TO ACHIEVE THESE RESULTS ARE MUCH LONGER

Benchmark Problem	Shirts	Trousers	Swim	Shapes1	Dagli
Number of Parts	99	64	48	43	30
Fixed Width	40	79	5752	40	60
Source	Oliveira et al. [5]	Oliveira et al. [5]	Oliveira et al. [5]	Oliveira et al. [5]	Hopper [7]
Method	**Quality of solution (Time)**				
Proposed Method	85.3% (76 sec)	87.2% (42 sec)	69.5% (24 sec)	67.8% (17 sec)	81% (8 sec)
Hopper [7]	79.6%	76.7%	56.8%	54.7%	72.5%
SigmaNest [7]	72.2%	79.1%	58.3%	57.2%	75.6%
NestLib [7]	78.1%	76.8%	67.3%	62.3%	77.3%
Burke et al. [18]	84.6% (4990 sec)	88.5% (7890 sec)	68.4% (12390 sec)	66.5% (820 sec)	83.7% (2040 sec)
Gomes et al. [3]	86.79% (10391 sec)	89.96% (8588 sec)	74.37% (6937 sec)	71.25% (10314 sec)	87.15% (5110 sec)
Nielsen et al. [15]	86.5% (3600 sec)	89.8% (3600 sec)	71% (3600 sec)	72.5% (3600 sec)	-
Oliveira et al. [5]	81.3% (210.5 sec)	82.8% (37.2 sec)	-	65.4% (23.3 sec)	-

these packages; it gives on average 1.4% better packing density than NestLib, and 4.6% than SigmaNest. Although Hopper mentions that these packages are fast, she does not state their execution times.

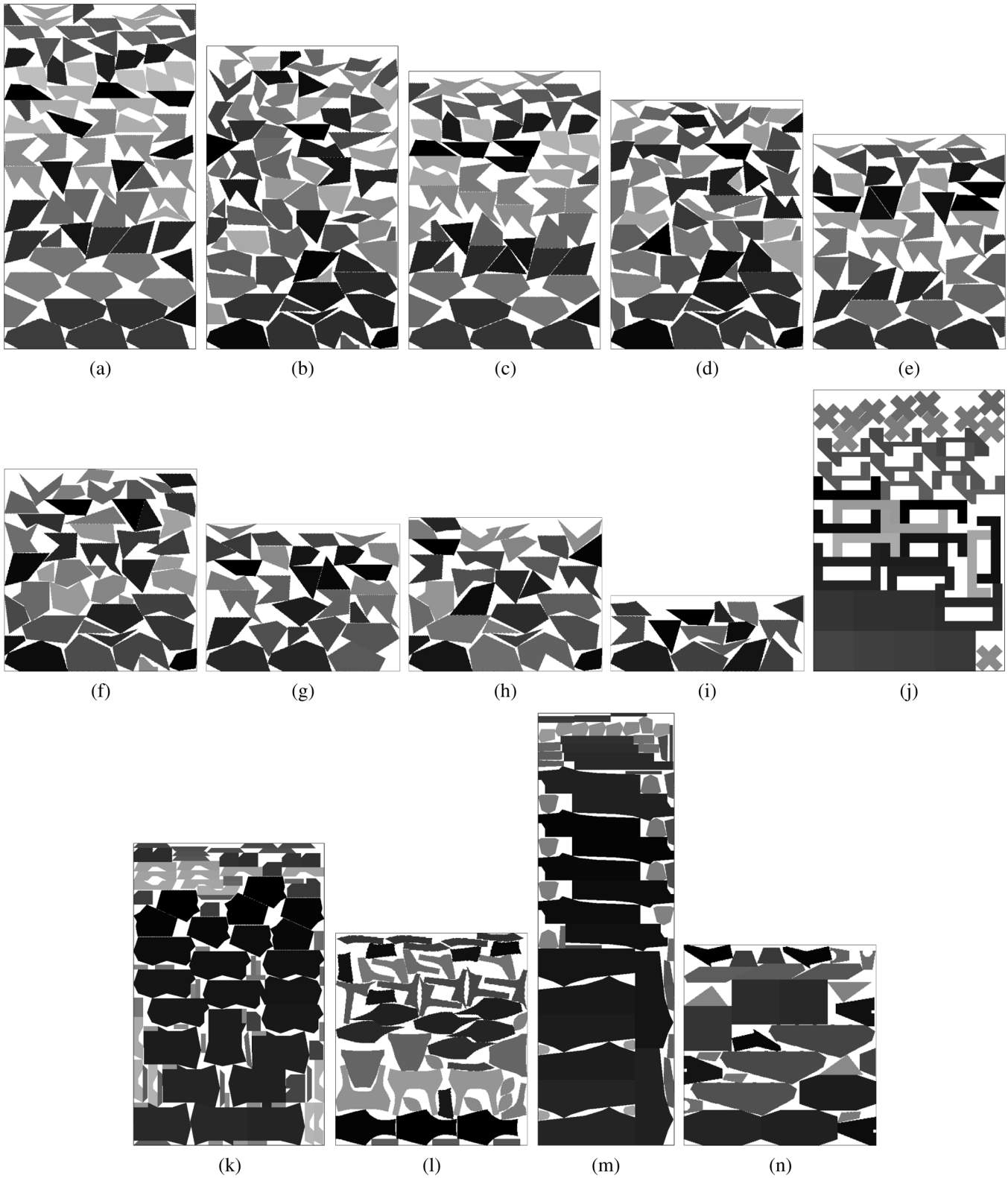

Fig. 10. Evaluation of the proposed algorithm on artificial benchmark problems (a)–(j) and real life instances (k)–(n). (a) Poly5a; (b) poly5b; (c) poly4a; (d) poly4b; (e) poly3a; (f) poly3b; (g) poly2a; (h) poly2b; (i) poly1a; (j) shapes1; (k) shirts; (l) swim; (m) trousers; and (n) dagli.

The method is compared also with a single-pass algorithm proposed by Oliveira *et al.* [5]. The authors conducted their experiments using a Pentium Pro processor at 200 MHz. Our algorithm achieves better packing densities than their algorithm (on average, 3.6%) in similar times. It must be emphasized that our algorithm is actually much faster, since the authors of [5] run their algorithm 126 times for each benchmark problem, using various nesting strategies, evaluation functions, and local search

techniques, and then keep the best solution among them. The times presented for their algorithm in Table IV refer to the execution times required by the best variant of their algorithm on each benchmark problem. An important comment made in [5] is that the best solution for each benchmark problem is not produced by the same variant of their algorithm. Hence, a fairer comparison between the efficiency of our algorithm and theirs should take into account the fact that the authors in [5] can achieve the given solutions only if they conduct experiments with all 126 versions of their algorithm for each benchmark problem. Doing so, their times are of several minutes or even hours.

As illustrated by the results, our method is able to examine many potential orderings and orientations of the parts in short times, achieving high-quality solutions. This is due to the procedure it follows; it "looks" at first at the existing void regions in the layout, narrows its search (about which part should be placed next, in which position and orientation) based on the shape similarity criterion of the void regions and the parts, and successively places the parts. One important feature of the method is its independence from the complexity of the parts under placement. In particular, the method can deal with any arbitrary shape, and its computational time is primarily independent from the number of shapes' vertices. The only part of the algorithm which is affected from the number of vertices is the method used for calculating the similarity of shapes. However, the execution time of that method is short and does not result in a noticeable increase of the computational time of the overall algorithm.

The proposed system is well suited to robotic applications as it can take into account imprecisions of robotic movements, due to the integration of a vision system. In online industrial applications, where parts are acquired by the system and need to be placed one at a time, the system predetermines their position and orientation at the time of acquisition. However, inaccuracies in robotic movements may occur and the part may be placed in a different position from the one the system defined. As the system does not know the actual position of the nested part, occupied regions will be considered as void, and future collisions may occur between placed and remaining parts. In order to avoid this, the vision system can send feedback to the placement system informing it of the actual position of the placed part. Then, the system proceeds to the packing algorithm, taking into account the actual placement of the part. Most packing algorithms are developed only for offline applications and such limitations have not been considered.

V. Conclusion

In this paper, an intelligent system for packing two-dimensional irregular shapes was presented. Drawing on techniques from computer vision and artificial intelligence, it was shown that a high-quality ordering and orientation of the shapes can be found efficiently.

The proposed system was assessed on 14 established benchmark problems and performed very well. In particular, the presented algorithm performs better than approaches from recent literature [5], [7], both in effectiveness and efficiency, while producing solutions of better quality than commercial nesting packages. It should be mentioned that recently published methods [3], [15], [18] achieve better solutions than our algorithm, but with much longer computational times.

Unlike most other published methods, the present method can deal with complex shapes and a variety of industrial constraints, such as defective regions and irregularly shaped sheets. This flexibility is due to the pictorial representation that the system uses, where parts and sheets are represented by their binary images.

Acknowledgment

The authors would like to thank Dr. C. Bouganis and M. Angelopoulou for their valuable help.

References

[1] P. Whelan and B. Batchelor, "Automated packing systems—a systems engineering approach," *IEEE Trans. Syst., Man, Cybern. A, Syst., Humans*, vol. 26, no. 5, pp. 533–544, Sep. 1996.

[2] G. Wäscher, H. Haußner, and H. Schumann, "An improved typology of cutting and packing problems," Working Paper 24, Faculty Econ. Manag., Otto von Guericke Univ. Magdeburg, 2006.

[3] A. Gomes and J. Oliveira, "Solving irregular strip packing problems by hybridising simulated annealing and linear programming," *Eur. J. Oper. Res.*, vol. 171, pp. 811–829, 2006.

[4] E. Hopper and B. Turton, "A review of the application of meta-heuristic algorithms to 2D strip packing problems," *Artif. Intell. Rev.*, vol. 16, pp. 257–300, 2001.

[5] J. Oliveira, A. Gomes, and J. Ferreira, "TOPOS—a new constructive algorithm for nesting problems," *OR Spectr.*, vol. 22, pp. 263–284, 2000.

[6] R. Fowler, M. Paterson, and S. Tanimoto, "Optimal packing and covering in the plane are NP-complete," *Inf. Process. Lett.*, vol. 12, pp. 133–137, 1981.

[7] E. Hopper, "Two-dimensional packing utilizing evolutionary algorithms and other meta-heuristic methods," Ph.D. dissertation, School Eng., Cardiff Univ., Cardiff, U.K., 2000.

[8] C. Dagli and P. Poshyanonda, "New approaches to nesting rectangular patterns," *J. Intell. Manuf.*, vol. 8, pp. 177–190, 1997.

[9] D. Liu and H. Teng, "An improved BL-algorithm for genetic algorithm of the orthogonal packing of rectangles," *Eur. J. Oper. Res.*, vol. 112, pp. 413–420, 1999.

[10] E. Hopper and B. Turton, "A genetic algorithm for a 2D industrial packing problem," *Comput. Eng.*, vol. 37, pp. 375–378, 1999.

[11] A. Albano and G. Sappupo, "Optimal allocation of two-dimensional irregular shapes using heuristic search methods," *IEEE Trans. Syst., Man, Cybern.*, vol. 10, no. 5, pp. 242–248, May 1980.

[12] M. Adamowicz and A. Albano, "Nesting two-dimensional shapes in rectangular modules," *Comput. Aided Des.*, vol. 8, pp. 27–33, 1976.

[13] S. Jacobs, "On genetic algorithms for the packing of polygons," *Eur. J. Oper. Res.*, vol. 88, pp. 165–181, 1996.

[14] H. Lamousin and W. Waggenspack, "Nesting of two-dimensional irregular parts using a shape reasoning heuristic," *Comput. Aided Des.*, vol. 29, pp. 221–238, 1997.

[15] B. Nielsen and A. Odgaard, "Fast neighborhood search for the nesting problem," Univ. Copenhagen, Copenhagen, Denmark, 2003, Tech. Rep.

[16] O. Faroe, D. Pisinger, and M. Zachariasen, "Guided local search for the three-dimensional bin packing problem," *INFORMS J. Comput.*, vol. 15, pp. 267–283, 2003.

[17] A. Gomes and J. Oliveira, "A 2-exchange heuristic for nesting problems," *Eur. J. Oper. Res.*, vol. 141, pp. 359–370, 2002.

[18] E. Burke, R. Hellier, G. Kendall, and G. Whitwell, "A new bottom-left-fill heuristic algorithm for the two-dimensional irregular packing problem," *Oper. Res.*, 2006, accepted for publication.

[19] F. Preparata and M. Shamos, *Computational Geometry: An Introduction*. Berlin, Germany: Springer-Verlag, 1985.

[20] G. Toussaint, "Solving geometric problems with the 'rotating calipers'," presented at the MELECON, Athens, Greece, 1983.

[21] K. Dowsland, S. Vaid, and W. Dowsland, "An algorithm for polygon placement using a bottom-left strategy," *Eur. J. Oper. Res.*, vol. 141, pp. 371–381, 2002.

[22] P. Whelan and B. Batchelor, "Flexible packing of arbitrary two-dimensional shapes," *Opt. Eng.*, vol. 32, pp. 3278–3287, 1993.

[23] A. Jain, *Fundamentals of Digital Image Processing*. Englewood Cliffs, NJ: Prentice-Hall, 1989.
[24] R. Haralick and L. Shapiro, *Computer and Robot Vision*. Reading, MA: Addison-Wesley, 1992, vol. I.
[25] J. Serra, *Image Analysis and Mathematical Morphology*. New York: Academic, 1982.
[26] A. Sajjanhar and G. Lu, "A grid based shape indexing and retrieval method," *Austral. Comput. J.*, vol. 29, pp. 131–140, 1997.
[27] ——, "A comparison of techniques for shape retrieval," in *Proc. Int. Conf. Computational Intelligence and Multimedia Applications*, 1998, pp. 854–859.
[28] D. Zhang and G. Lu, "Content-based shape retrieval using different shape descriptors: a comparative study," in *Proc. IEEE Int. Conf. Multimedia and Expo*, 2001, pp. 317–320.
[29] E. Arkin, L. Chew, D. Huttenlocher, K. Kedem, and J. Mitchell, "An efficiently computable metric for comparing polygonal shapes," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 13, no. 3, pp. 209–216, Mar. 1991.
[30] B. Mehtre, M. Kankanhalli, and W. Lee, "Shape measures for content based image retrieval: a comparison," *Inf. Process. Manag.*, vol. 33, pp. 319–337, 1997.
[31] D. Douglas and T. Peucker, "Algorithms for the reduction of the number of points required to represent a digitised line or its caricature," *Can. Cartographer*, vol. 10, pp. 112–122, 1973.
[32] [Online]. Available: http://www.apdio.pt/sicup/.

Alexandros Bouganis (S'05) received the M.Eng. (Hons.) degree from the Mechanical Engineering and Aeronautics Department, University of Patras, Patras, Greece. He is currently pursuing the Ph.D. degree in the Department of Computing, Imperial College London, U.K.

His research interests include robotics, industrial automation, and computer vision.

Mr. Bouganis received several awards from the Greek State Scholarship's Foundation and the Technical Chamber of Greece during his undergraduate studies.

Murray Shanahan received the Ph.D. degree in computer science from the University of Cambridge, Cambridge, U.K., in 1988.

He carried out postdoctoral work in artificial intelligence in the Department of Computing at Imperial College, London, U.K., in the 1990s, and joined the faculty of Imperial College's Department of Electrical Engineering in 1998. Currently, he is Professor of Cognitive Robotics in the Department of Computing at Imperial College.