Flexible packing of arbitrary two-dimensional shapes

Paul F. Whelan, MEMBER SPIE
Dublin City University
School of Electronic Engineering
Dublin, Ireland
E-mail: whelanp@dcu.ie

Bruce G. Batchelor, MEMBER SPIE
University of Wales, Cardiff
Department of Computing Mathematics
Wales, United Kingdom
E-mail: bruce@cm.cf.ac.uk

Abstract. A computational framework for the flexible packing of arbitrary planar shapes under visual control is described. Our aim in this work has been to produce an efficient packing strategy that is flexible enough for a wide variety of industrial uses and which can be implemented in fast, moderately priced hardware. We have deliberately adopted a systems approach, versus a purely algorithmic one, since we are concerned with industrial vision problems in which significant problem constraints exist. The packing procedure that we devised consists of two major components. The first is a geometric packing technique that is based on morphological image processing operations. This is used in conjunction with a heuristic packing procedure. Some of the factors considered at the heuristic level include shape ordering and shape orientation, both of which must be carried out prior to applying the geometric packer. The heuristic procedure also deals with problem constraints that are specific to a given application. Various issues arising from this approach, such as the properties and performance of the procedure, are discussed within the background of some sample applications. The ideas outlined are currently being used in the development of a visually controlled intelligent packing work cell.

Subject terms: machine vision; heuristics; material handling; automated nesting; automated assembly; systems engineering.

Optical Engineering 32(12), 3278–3287 (December 1993).

1 Introduction

One of the key areas in the current research on industrial automation is the development of flexible vision systems. Vision systems that are highly adaptable and can cope with a wide range of variable products must be developed for work in an unconstrained environment. Other key elements in these systems include the ability to manipulate arbitrary, previously unseen objects and to cope with unanticipated situations. For example, to achieve a high degree of self-reliance in supposedly automatic assembly machines, it is necessary to find engineering solutions to a wide range of long-standing industrial parts handling problems. Vision will, no doubt, have a key role in reaching this end, even though present-day industrial vision systems have some serious limitations.[1] We and many other research workers are hoping to develop generic parts handling techniques that extend the bounds of vision applications.

When we are faced with a specific application requirement, we find it well worthwhile to analyze the problem from a systems engineering perspective. By adopting a systems approach, the maximum use is made of problem-specific "contextual" information derived, for example, from the nature of the product being handled, the process used to manufacture it, and the special features of the manufacturing environment. By doing so, we hope to reduce the complexity of the application. For example, it may be found that, by mechanically restricting the orientation and the order of arrival of objects considered by the packing system, the problem can be simplified. In fact, by taking heed of such constraints, in a practical packing application, the procedure might well reduce to a standard well-tried technique. We believe that in packing, as happens so often elsewhere, systems considerations are always worth investigating.

The ability to manipulate previously unseen objects under visual control is one of the key tasks in the successful implementation of robotic, automated assembly and adaptive material handling systems. According to Freeman[2] the strongest growth-rate forecast for the use of machine vision in industry is in the area of material handling. It has been estimated that a quarter (approximately) of the parts in industrial artifacts are too large or complex to be handled in automated part feeders. Many products are transferred from one manufacturing work station to another on pallets. To automate palletizing, a flexible packing strategy, by machine vision, needs to be developed.

It is in the context of this framework that our industrial vision packing strategy has been developed. Its two main components are a geometric packer, based on the principles

Paper 25102 received Oct. 26, 1992; revised manuscript received April 19, 1993; accepted for publication April 19, 1993.

of mathematical morphology,[3] and a heuristic packer, based on the application of rule-based procedures implemented in PROLOG. Together, these form a flexible strategy that allows the packing of arbitrary 2-D shapes. While the technique will pack any set of shapes presented to it, the efficiency is critically dependent on the application. Therefore, we need to use any clues we may glean from the context information to ensure that we obtain an efficient packing strategy for that application (see Fig. 1).

Even the simpler packing problems, such as palletizing,[4] have been shown to be NP-Complete.[5] It is clearly impossible to guarantee that we will reach an optimal procedure for the more general problem. Hence, our aim has been to produce an *efficient* packing strategy that is flexible enough for industrial use. To achieve this objective, the systems approach to the packing problem is essential.

As we have already pointed out, our packing scheme consists of two major components:

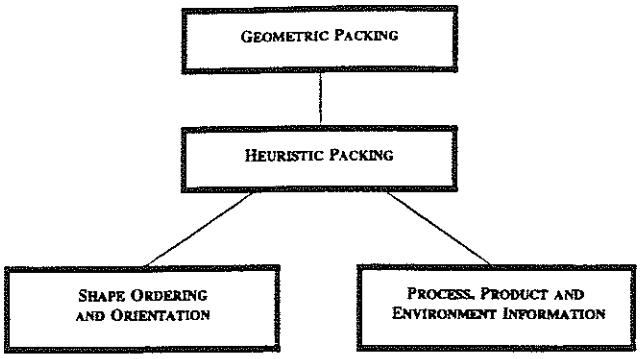
1. A *geometric packer* (GP) based on the principles of mathematical morphology, which takes an arbitrary shape in a given orientation and puts the shape into place in that orientation. This has been discussed elsewhere.[3] Section 6 contains a summary of these operations.
2. A *heuristic packer* (HP), which is concerned with the ordering and alignment of shapes prior to applying them to the geometric packer. This component also deals with other general considerations, such as the conflict in problem constraints and the measurement of packing performance. In addition, it deals with practical constraints, such as the effects of the robot gripper on the packing strategy, packing in the presence of defective regions, anisotropy ("grain" in the material being handled) and pattern matching.

This paper concentrates on the applications constraints that give rise to the rules in the heuristic component of the packing procedure. We shall also discuss several sample packing applications.

By using heuristics in the packing strategy, we hope to produce an efficient but not necessarily optimal solution. However, the main problem with such an approach is that there is a tendency to generate a set of overly complex rules, incorporating a variety of paradoxes and logical conflicts. It is necessary, therefore, to keep all the logic decisions as simple as possible.

The packing procedure was developed using the PROLOG+ environment developed at the University of Wales.[6] This incorporates a rich set of image processing primitives within a conventional PROLOG environment, hosted on a Macintosh computer. It provides a high degree of flexibility, while minimizing the development overhead. This allows the programmer to concentrate on the problem at hand, without becoming unnecessarily entangled in the implementation details.

Section 2 describes the heuristic component of the packing strategy. In particular, we discuss the application of this procedure to the automated packing of 2-D blobs and polygons into arbitrary scenes. (The term *scene* is used in this paper when referring to that 2-D region of space into which we are required to place an arbitrary shape.) Section 3 introduces various performance measures necessary to evaluate the procedure's packing ability. Section 4 discusses the performance of the strategy in the context of a number of different sample applications. Section 5 discusses various practical issues arising from this approach.



GEOMETRIC PACKING
HEURISTIC PACKING
SHAPE ORDERING AND ORIENTATION
PROCESS, PRODUCT AND ENVIRONMENT INFORMATION

Fig. 1 General packing strategy.

2 Heuristic Packing Techniques

The heuristic packer determines the orientation and order in which the shapes are applied to the geometric packer and operates on two classes of shapes: blobs and simple polygons. We must consider both these general shape classes separately, since no single scheme exists for all shape classes. While the geometric packer is independent of the shape class and application context, the heuristic packer is not.

2.1 *Blob Packing Techniques*

This section outlines the heuristics required to deal with 2-D binary images of random shape and size, prior to the application of the geometric packer. The approach outlined was designed specifically for an off-line packing process but the techniques developed could equally well be applied to an on-line packing process. Details of the implementation of this procedure can be found in Sec. 7.1.

All the shapes to be packed are presented simultaneously to the vision system. The shapes are then ranked according to their bay sizes. (A *concavity* is a region that lies within the convex hull of a blob *S*, but not within *S* itself. A *bay* is a concavity that touches the convex hull, in contrast to a lake, which does not.) The shape with the largest bay is the first to be applied to the geometric packer. Once the shape ordering has been decided, each shape must be oriented so that an efficient local packing strategy can be implemented. Four orientation rules are used to align the shape to be packed in the scene, as outlined here:

Orientation rule 1. **If** the current shape has no bay **and if** the circularity measurement of the shape is less than one, **then** the shape is classed as a disk and no realignment need take place.

Orientation rule 2. **If** the size of the shape's largest bay is classified as significant **and if** the circularity value of the shape's largest bay is less than or equal to two, **then** the shape is rotated such that its largest bay lies in the same orientation relative to the centroid as the original scene.

The largest bay is directed toward the section of the scene with the maximum amount of unpacked space; see Fig. 2.

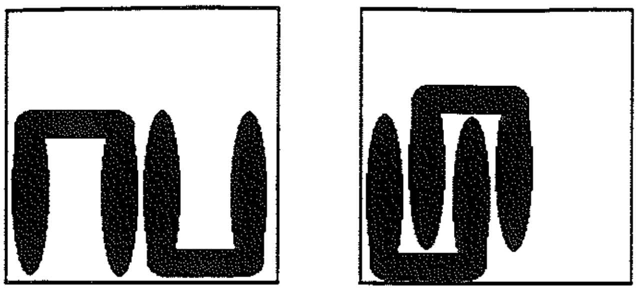


Fig. 2 Orientation rule 2. The packing configuration on the right ensures that the largest bay is directed inward, therefore allowing the second shape to be packed more efficiently. The packing configuration on the left does not implement orientation rule 2.

This last requirement ensures that the shapes can be efficiently packed at a local level.

Orientation rule 3. **If** the size of the shape's largest bay is classified as significant **and if** the circularity value of the shape's largest bay is greater than two, **then** the shape is found to have an elongated bay, so it is oriented such that it lies at the same angle as the scene we wish to pack. (The angle is defined as the axis of least moment of inertia.)

Orientation rule 4. **If** the size of the shape's largest bay is classified as insignificant, **then** the shape is oriented at the same angle as the scene we wish to pack.

Some results. Figure 3(a) shows the results of packing hand tools into a rectangular tray. The shapes were initially presented directly to the geometric packer, without the aid of the heuristic packer. This has the effect of packing each tool at whatever orientation it was in when it was presented to the vision system. Figure 3(b) shows the resultant packing configuration when the heuristic packer precedes the geometric packer; each shape is aligned and ordered before it is applied to the geometric packer. Figure 3(c) shows the packing of the tools into a "random" blob region. The full packing strategy was used again here as in Figure 3(b). Notice that the spacing between the shapes packed in a scene can be controlled during the geometric packing stage. In Fig. 3 the shapes are spaced out so that they can be viewed more clearly.

2.2 *Polygon Packing Techniques*

The approach outlined in Sec. 2.1 is not efficient when packing shapes that do not contain bays of significant area. Hence, a different packing procedure is used to pack simple polygons that do not contain bays of significant area. As before, this procedure was designed to work within an off-line packing system but could also be applied to on-line packing applications. Unlike the previous approach, however, this second procedure has the ability to determine the local packing efficiency for each shape and will reorient it to ensure a more efficient configuration. (This local efficiency check could also be applied to the blob packing strategy outlined in Sec. 2.1.) In our second sample application, we chose to pack nonuniform box shapes (squares and rectangles) into a square scene (Fig. 4). Once all the shapes have been presented to the packing system, they are ordered according to size, with the largest shape being packed first. The shapes must then be oriented prior to the application of the geometric packer.

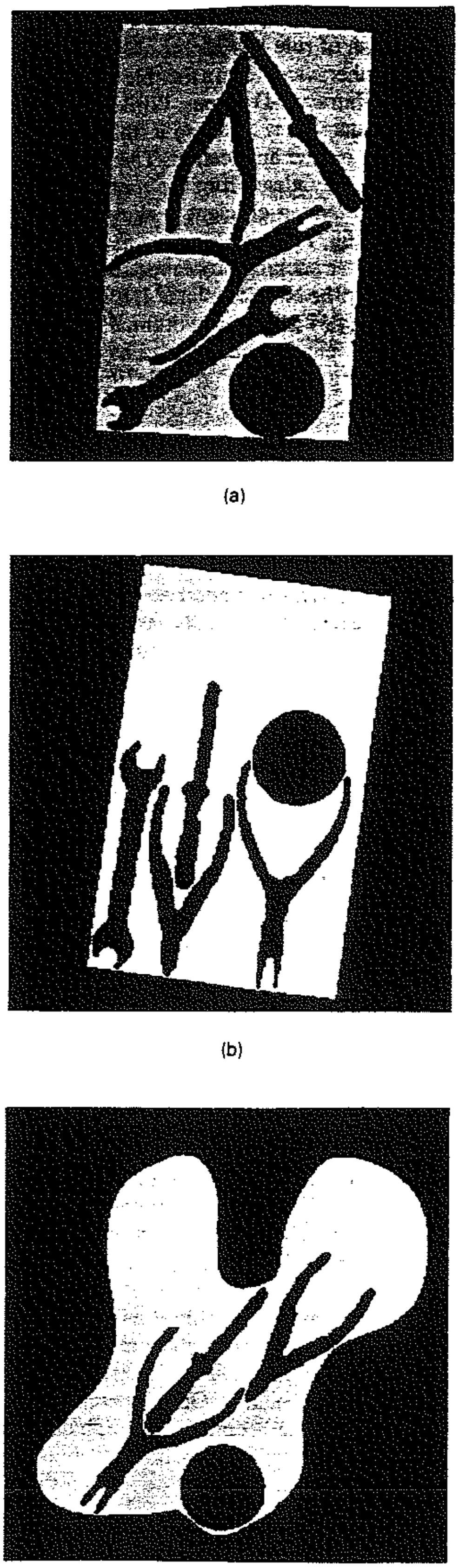


Fig. 3 Packing of tool shapes: (a) tools packed in their current orientation, (b) tools reoriented for better efficiency, and (c) tools packed in an irregular scene.

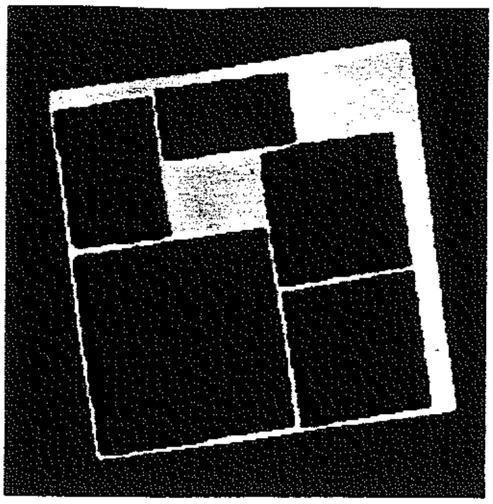


Fig. 4 Packing of nonuniform boxes in a square tray.

Details of the implementation of this procedure can be found in Sec. 7.2.

In the initial versions of this packing procedure, each shape was aligned in such a way that its axis of least moment of inertia was matched to that of the scene under investigation. However, this method proved unreliable for packing squares, because quantization effects produce a digital image with a jagged edge. (A resolution of 256×256 pixels was used.) This can cause errors in the calculation of the moment of inertia. The problem was overcome by aligning the longest straight edge of the shape to be packed with the longest straight edge of the scene. The edge angles for the shape and scene were found by applying an edge detection operator, followed by the Hough transform. The latter was used because it is tolerant of local variations in edge straightness. Once the peaks in the Hough transform image were enhanced and separated from the background, the largest peak was found.[7] This peak corresponds, of course, to the longest straight edge within the image under investigation, whether it be the shape or the scene. Since the position of the peak in Hough space defines the radial and the angular position of the longest straight edge, aligning the shape and the scene is easy.

Once a polygonal shape has been packed, a local packing efficiency check is carried out. This ensures that the number of unpacked regions within the scene is kept to a minimum. The shape to be packed is rotated through a number of predefined angular positions. After each rotation, the number of unpacked regions in the scene is checked. If a single unpacked region is found, then a local optimum has been reached. In this case, the local packing efficiency routine is terminated and the next shape examined. Otherwise, the local packing efficiency check is continued, ensuring that when a shape is packed a minimum number of unpacked regions exist. This reduces the chance of producing large voids in the packed scene and improves its overall efficiency of packing.

Depending on the application, a more restricted subset of the approaches outlined above may be required; otherwise, the application may require a more complex heuristic procedure[8] that classifies the input shapes initially and then handles the packing in the most appropriate way.

3 Performance Measures

To ensure that we have confidence in the global efficiency of any packing strategy, there must be some way of measuring its performance. Traditionally, packing performance has been measured by a single number, called the *packing density*.[9] This is the ratio of the total area of all the packed shapes to that of the total area of the scene. This is referred to as the *worst-case analysis* packing measure. A number of other performance measures have been developed in the field of operational research, particularly for comparing different heuristics for packing rectangular bins by odd-sized boxes. (See Dowsland[4] for a good review of packing procedures used in operational research.) These performance metrics fall into two main categories[10]:

1. *Probabilistic analysis.* It is assumed that a density function for the problem data exists and can be estimated or guessed. This enables probabilistic performance values for the heuristic to be reached. This may be in the form of a bound placed on the likelihood that the heuristic will find a solution within a defined ratio of the optimum packing density.
2. *Statistical.* The heuristic is applied to a large sample of "typical" problems to estimate its likely performance in the statistical sense.

While these two measurements can be quite useful in well-constrained packing problems, they are of little use in dealing with the packing of arbitrary shapes—real data are unlikely to fall neatly into a uniform or any other easily analyzable distribution.

The performance measures used in our strategy are based on the traditional *worst-case analysis.* Details of the implementation of the performance measures can be found in Sec. 7.3. After a packing procedure has been applied to a given scene, the result is assessed by five performance parameters:

- packing density
- performance index
- space usage
- number of shapes presented to the scene
- number of shapes packed in the scene.

The *packing density* is the ratio of the total area of all the shapes packed to the area of their (collective) convex hull after packing (minus the area of the scene defects). This measure has a maximum value of one.

The *performance index* is a modified version of the packing density. A weighting factor is applied. This is referred to as the *count ratio* and is defined as the ratio of the total number of shapes packed to the number of shapes initially presented to the scene. The performance index is equal to the product of the packing density and the count ratio. The performance index also has a maximum value of one. The product of the packing density and the count ratio accounts for any shapes that remain unpacked when the procedure terminates.

The *space usage* is a ratio of the area of the shapes packed to the total area of the scene. Again, this parameter has a maximum value of one. The performance measures for the

Table 1 Comparison of packing configurations by performance measures.

Figure Number	Packing Density	Shapes Presented	Shapes Packed	Performance Index	Space Usage
3 (a)	.36	5	5	.36	.25
3 (b)	.435	5	5	.435	.25
3 (c)	.381	5	4	.305	.191
4	.86	6	5	.72	.735
5 (a)	.38	6	6	.38	.261
5 (b)	.44	6	6	.44	.317
6 (a)	.81	5	4	.64	.566
6 (b)	.73	7	5	.52	.61
7	.71	6	5	.59	.48
9	.41	5	3	.246	.23
10 (a)	.36	5	5	.36	.249
10 (b)	.511	8	5	.319	.381

packing examples discussed in this paper are shown in Table 1.

4 Experimental Results

In this section the performance of the packing strategy adopted is discussed in light of our experience with a number of sample applications.

Figure 5 shows the results of packing some standard household items, such as scissors, keys, and pens, into a rectangular tray [Fig. 5(a)] and into an irregular scene [Fig. 5(b)].

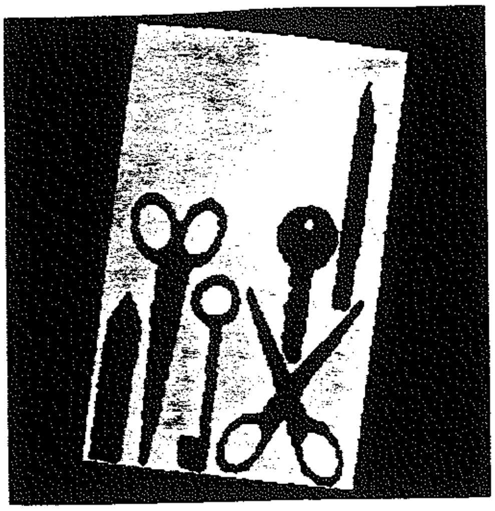
Figure 6 shows the automated packing of simple polygon shapes by the approach outlined in Sec. 2.2. Figure 6(a) shows the result of assembling a simple block jigsaw, while Fig. 6(b) shows the result of applying this procedure to a 2-D "apictorial" Chinese tangram puzzle.[11] Whereas other researchers[12] have developed specific routines to solve this puzzle, the solution shown here is based on the application of the "general-purpose" polygon packing procedure. [The reader should note that better results than those illustrated in Fig. 6(b) would be obtained by the addition of further application-specific heuristics to the packing procedure.]

4.1 *Pallet Packing*

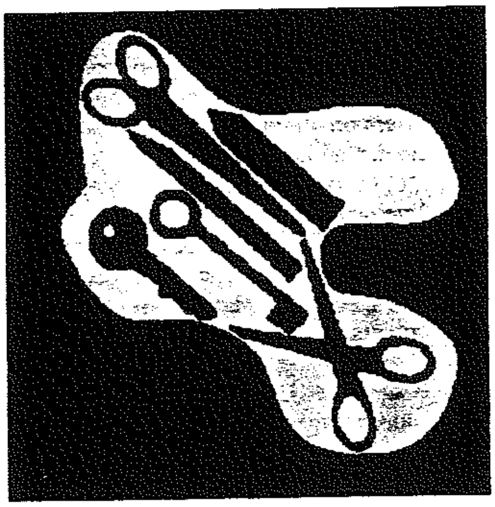
In Fig. 7, the general-purpose polygon packing procedure is applied to the problem of pallet packing, which has been extensively studied by operational research workers.[13] Here, a number of empty pallets that must be packed comprise the scene in our terminology. We studied the packing of three pallets with a number of boxes of various sizes and aspect ratios. Our task is, of course, to maximize the number of boxes packed, while at the same time to ensure efficient packing of each pallet. Again, the unmodified polygon packing procedure of Sec. 2.2 was applied. The result is shown in Fig. 7. In this instance, the only system constraint imposed was that all of the pallets be aligned in the same orientation.

4.2 *Robot Gripper Considerations*

Any supposedly general-purpose strategy for packing must be robust enough to cope with a range of material handling systems. For all of the applications considered previously, we have tacitly assumed that some form of suction or magnetic gripper could be used to lift and place the objects during



(a)

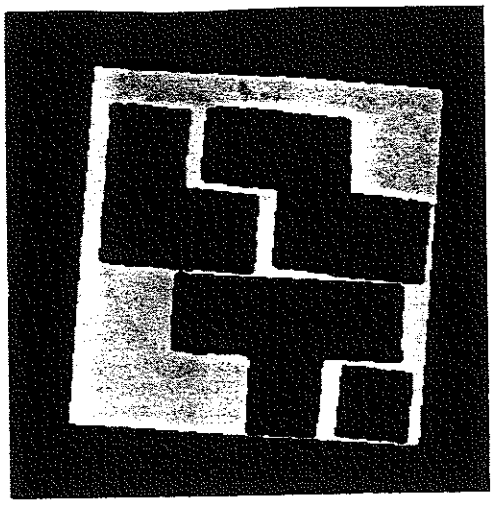


(b)

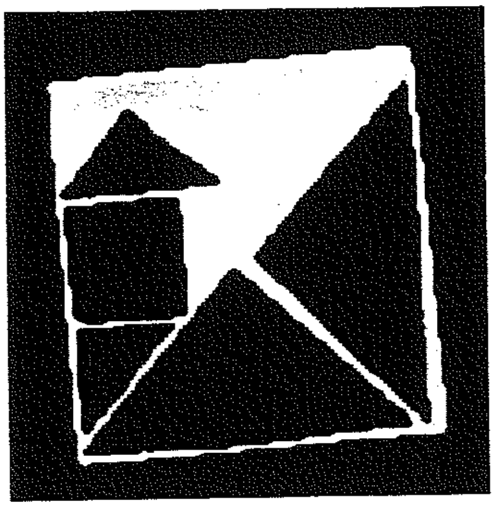
Fig. 5 Packing items into (a) a rectangular tray and (b) an irregular scene.

packing. In this case, the "footprint" of the gripper is assumed to lie within the outer edge of the shapes being manipulated.

Automated material handling systems frequently make use or robotic grippers that have two, three, or more "fingers." This complicates the problem of packing, since the gripper requires access to objects within a partially packed scene. Therefore, any packing strategy must make allowances for the gripper. The worst-case position usually (but not always) occurs when the gripper is fully open, just after placing an object in position. The problem of gripper access can be dealt with very effectively by the simple expedient of overlaying a gripper template on the shape to be packed prior to the application of the geometric packer (Fig. 8). The gripper footprint is based on the positions of the fingers in both the open and closed positions. In fact, the convex hull of each of the fingertips in the open and closed positions is formed when computing the composite footprint; this convex hull is indicated by the shaded region in Fig. 8.



(a)



(b)

Fig. 6 (a) Automated packing of simple polygons in a rectangular scene and (b) partial assembly of Chinese tangram puzzle.

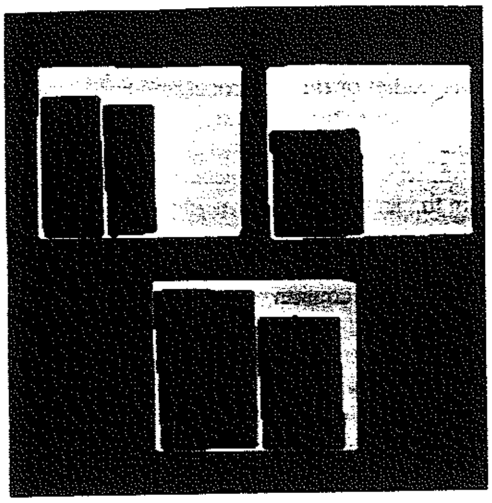


Fig. 7 Pallet packing.

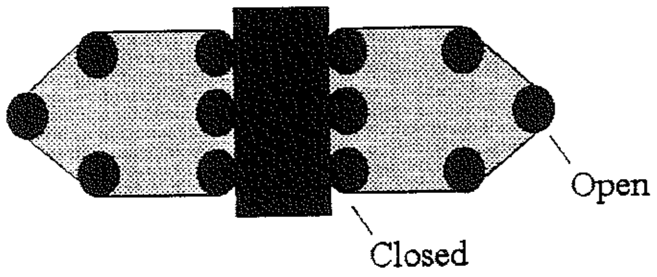


Fig. 8 Generation of the gripper "footprint" based on the fully open and closed positions of a multifingered robot gripper.

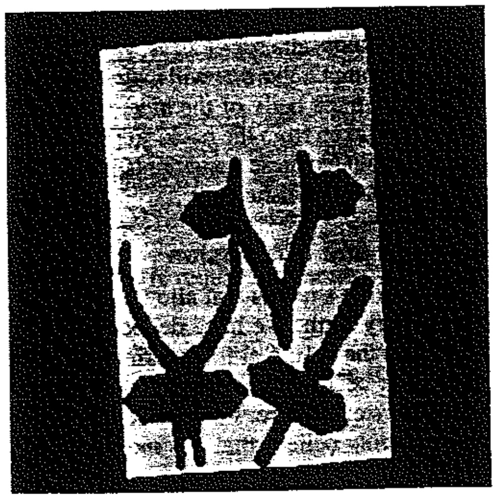


Fig. 9 Tool packing with worst-case gripper footprint.

Figure 9 shows the result of packing tools into a rectangular tray, taking the gripper footprint into account. Although the general blob packing strategy outlined in Sec. 2.1 remains the same, the procedure's performance is inevitably weakened when allowance is made for the robot gripper. For example, compare Figs. 3(b) and 9 (also see Table 1 for a comparison of the packing performance measures). Clearly, the gripper footprints indicated in Fig. 9 are not those for the ideal gripping positions for these objects. They are used merely to indicate how our packing strategy can cope with multifinger grippers.

In a practical situation, care must be taken to ensure that any change in the shapes of the objects to be packed, due to squeezing by the robot gripper, does not adversely affect the packing. The same is true of articulated and other hinged objects, such as scissors or pliers, which can change their shape during handling. Again, this type of application constraint could also be dealt with by the introduction of suitable heuristic packing rules, and may also be used as a factor when calculating the gripping position.

The strategy outlined here for working with multifinger grippers does have the advantage of allowing the shapes to be unpacked from the scene in any order. One possible modification to the approach outlined previously results in a denser configuration that, in general, can only be packed safely in reverse order. This modification consists of packing each shape, taking the robot footprint into account, but removing the footprint from the scene prior to the application of the next shape.

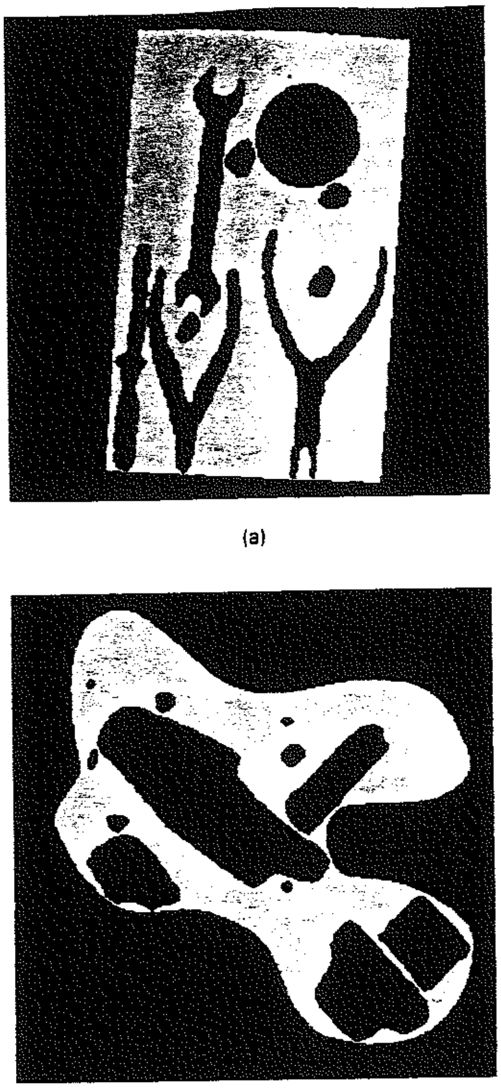
4.3 *Packing Scenes with Defective Regions*

Any practical automated packing system for use in such industries as leather or timber processing must be able to pack "objects" into a scene that may contain defective regions. (Recall that depletion is effectively the process of packing "holes" into a space.) The importance of good packing procedures in the leather industry is obvious, since the raw material is both expensive and nonrecyclable. Our packer can readily accommodate defects like these; we simply define the initial scene to contain a number of holes. Figure 10(a) illustrates the effect of packing tools into a rectangular tray that contains four small bloblike "defects." By comparing the packing configuration shown in Figs. 10(a) and 3(b) (also see Table 1), it is clear that the packing is not as tight when defects are taken into account. Obviously, our intuition was undisturbed by this result! Figure 10(b) shows the packing of leather templates onto a hide. The small bloblike regions indicate the defective areas of the hide. These defective regions are not to be included in the leather pieces to be cut. Both of the results shown in Fig. 10 indicate the flexibility of the packing strategies described in Sec. 2.

4.4 *Additional Points on Packing in the Leather Industry*

The design of packing systems for the shoe manufacturing industry is made easier by the fact that leather, like fabric, wood, marble, and many other natural materials, has a pronounced grain. This means that only two orientations of a given shoe component may be permissible, a fact that can greatly enhance the speed of the packing procedure. Again, our packer can easily take this type of application constraint into account.

Packing shoe component templates onto a hide is not quite as simple as we have suggested, because the leather is not uniform in its thickness and suppleness. When making shoes, for example, the components that will make up the soft leather uppers are cut from the belly of the hide, while the tougher, more rigid sole is taken from the back. Adding heuristic rules to assist packing under these constraints is not difficult but has not yet been accomplished.


(a)

(b)

Fig. 10 Packing items into defective regions: (a) tools into a defective tray and (b) leather template pieces into a leather hide that contains defects.

5 Discussion and Conclusion

This paper outlines a strategy that allows the packing of 2-D shapes of arbitrary form. We have deliberately taken a systems approach to the design of the packing strategy, because this has so often been of invaluable help in designing industrial vision systems. Here, we have seen this in evidence yet again. We firmly believe that it is most unwise to ignore the constraints imposed by the nature of the application. As demonstrated, by taking them into account, we may well obtain a faster, cheaper solution. To summarize, we assert that packing systems are best designed with regard to such issues as gripper shape, grain, material defects, etc. We do not feel that there is a strong justification for seeking a unified algorithmic solution that is capable of tackling the totality of packing applications without the intervention of an intelligent designer.

The results presented here show the power of the heuristic approach when presented with a wide range of problems, including the packing of shapes into materials with defective regions. We have attempted to maximize the use of application-specific information to produce an efficient packing strategy. The paper also outlines a technique that will allow a range of performance measures to be computed so that different packing procedures can be compared. This is necessary because the heuristic approach we have taken does not guarantee an optimal result.

One of the problems with our approach to packing is that we are trying to produce an efficient global packing strategy based on local optimization. There is always a danger in this type of situation that the (packing) procedure may quickly become trapped into a grossly suboptimal solution. To date, we have not encountered this difficulty. The concept of nonlocal assembly is similar in nature to the "intelligent groping" mentioned by Penrose.[14] He suggests that it may occasionally be necessary to overlook the fact that an individual shape is not in a locally optimal position to ensure that an efficient global packing strategy is implemented. In fact, some researchers[15] observed that the more complex the set of input shapes, the more likely a random packing strategy, such as the one outlined by Uhry,[16] is to succeed.

This paper has shown that the combination of a heuristic packing strategy with the use of the application domain information is likely to yield good results, although we cannot be sure that they are anywhere near optimal. Further research in this area is needed to develop even more effective heuristics and to evaluate and compare them with nonlocal and random packing techniques.

6 Appendix A: Summary of the Morphological Operations

This summary outlines the intermediate steps involved in the morphological packing of an arbitrary shape. This shape is denoted by the structuring element B. The image scene is denoted by the image set A. The morphological operations are also summarized in the image flowchart of Fig. 11. A detailed description of this procedure can be found in Ref. 3. (Morphological notation: A, B = digital image sets; $\oplus$, $\ominus$ = dilation and erosion transformations; $\bigcirc$, $\bullet$ = opening and closing transformations; $\veebar$ = exclusive-OR Boolean operation.)

Step 1. Erode the image scene A by the structuring element B to produce the erosion residue image $C = A \ominus B$. Every white pixel in this residue represents a valid packing location.

Step 2. Scan the erosion residue image for the location of the first object (white) pixel. This location is denoted by (*fitx*, *fity*). This coordinate corresponds to the first possible packing location of B in the scene A.

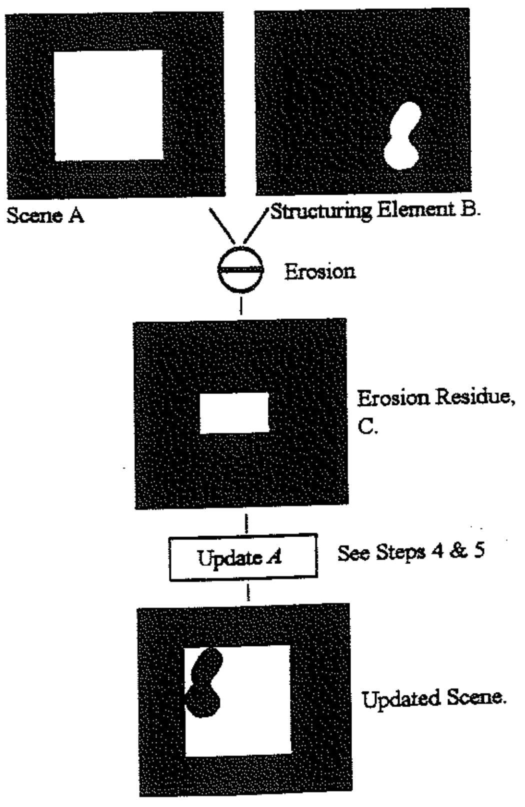


Fig. 11 Morphological operations image flowchart.

Step 3. Replace the erosion residue image C by a single object pixel placed at (*fitx*, *fity*). This is the new residue image and is denoted by C_{XY}.

Step 4. The new residue image C_{XY} is dilated by the structuring element B to produce a modified opening transform of A by B, denoted by $M(A \bigcirc B) = C_{XY} \oplus B$. This effectively places the shape to be packed B at the coordinate of the first possible packing location (that is, the image translation operation).

Step 5. The resultant image of step 4 is combined, by means of the exclusive-OR operation ($\veebar$), with the original image set A to produce a new value for the image set A, therefore packing B into the scene. This can be represented algebraically as $A = A \veebar M(A \bigcirc B)$.

7 Appendix B: PROLOG+ Packing Predicates

The PROLOG+ predicates outlined detail the techniques discussed in Secs. 2.1 and 2.2. (The symbol $ preceding a PROLOG+ predicate indicates that the operation is carried out in the image processor.)

7.1 *Blob Packing Predicates*

The order in which the shapes are placed by the packer is determined by the *sort_by_bay* predicate. If the area of the largest bay is significant compared to the area of the current shape, then the shape is sorted by its largest bay size (largest first); otherwise, the shapes are sorted by their size (largest first). The *bay_rot* predicate rotates a shape such that the largest bay is at the scene's angle of least moment of inertia. This predicate also ensures that the biggest bay is facing into the scene (that is, facing to the right and upward). This operation is summarized as follows:

- **If** object_Y_coordinate>bay_Y_coordinate, **then** rotate shape by 180 deg.
- **If** object_Y_coordinate = bay_Y_coordinate **and** object_X_coordinate>bay_X_coordinate, **then** rotate shape by 180 deg.
- **If** object_Y_coordinate = bay_Y_coordinate **and** object_X_coordinate≤bay_X_coordinate, **then** no action is required because it is in correct orientation.
- **If** object_Y_coordinate<bay_Y_coordinate, **then** no action is required because it is in correct orientation.

The predicate *main_pack* finds the valid packing location in the erosion residue. The structuring element is then placed at this location.

```
packbay:-
   get_all_shapes,          % Get all shapes to be packed.
   sort_by_bay,             % Sort shapes by bay size.
                            % The sorted data is stored in the new_blob_db database.
   !,
   pack_bay_1.              % Pack shapes - largest bay first.

/*
   Main packing coordination predicates.
*/
pack_bay_1:-
   big_bay_first,           % Pack the shapes by bay size.
   read_shapes,             % View remaining shapes.
```

```
  count_white_pixels(N).     % Count image pixels.
  N > 0,                     % If no pixels to process - fail.
  centre_screen_se,          % Place the shape in the centre of the FOV (128,128) prior to
                             % rotation, to ensure the image does not go outside the FOV.
  count_white_pixels(SHAPESIZE),
                             % Current shape area.
  get_shape_parameters(ROUNDNESS,BAYSIZE,BAY_ROUNDNESS),
                             % Current shape parameters.
  bay_rotate_options(SHAPESIZE,ROUNDNESS,BAYSIZE,BAY_ROUNDNESS).
                             % Choose rotation procedure.

  !,
  pack_bay_1.

pack_bay_1:-
  performance_meas.          % Calculate performance values.

/*
  Shape alignment and rotation after sorting.

  Orientation rule 1: Use this sorting option if the bay size is zero and the shape has a
  roundness <= 1. If the above conditions occur then the object is classified as 'round' (ie
  a disk shape) and therefore it need be rotated or aligned.
*/
bay_rotate_options(_,ROUNDNESS,BAYSIZE,_):-
  BAYSIZE = 0,
  ROUNDNESS < 1,
  pack_bay_main.             % Morphological packing.
/*
  Orientation rule 2: Use this sorting option if the largest bay size > a quarter of the shape
  area.Therefore if the bay is large the shape is rotated such that the largest bay is at the
  angle of the least MOI of the shape to be packed, ensuring that the bay region always
  points into the main body of the shape to be packed. By checking the bay roundness we
  ensure that we do not rotate the image by its bay if the bay is elongated.
*/
bay_rotate_options(SHAPESIZE,_,BAYSIZE,BAY_ROUNDNESS):-
  BAYSIZE > SHAPESIZE/4,
  BAY_ROUNDNESS =< 2,
  bay_rot,
  pack_bay_main.             % Morphological packing.

/*
  Orientation rule 3: Use this sorting option if the bay size > a quarter of the shape area.
  By checking the bay roundness we ensure that we do not rotate the image by its bay if the
  bay is elongated. We align the shape with respect to the least MOI of the scene.
*/
bay_rotate_options(SHAPESIZE,_,BAYSIZE,BAY_ROUNDNESS):-
  BAYSIZE > SHAPESIZE/4,
  BAY_ROUNDNESS > 2,
  shape_rot,                 % Rotate shape such that it is at scenes angle of the least MOI.
  pack_bay_main.             % Morphological packing.

/*
  Orientation rule 4: Use this sorting option if the bay size <= a quarter of the shape area.
  Therefore if the bay is considered small we align the shape with respect to the scenes least
  MOI.
*/
bay_rotate_options(SHAPESIZE,_,BAYSIZE,_):-
  shape_rot,
  pack_bay_main.             % Morphological packing.

/*
  Morphological packing predicate.
*/
pack_bay_main:-
  read_structuring_element,
  $'PACK',                   % Erosion residue using system macro call.
  space_out,                 % Dilate image by single pixel to
                             % make packing clearer.
  $'TRES',                   % Pack original SE.
  main_pack.                 % Place SE onto erosion residue.
```

7.2 *Polygon Packing Predicates*

The packing order is determined by the shape size (largest first). The rotation of the shapes by the packer is based on the angle of the largest face (longest straight side of the polygon) of the unpacked region. The predicate *shape_face_angles* finds the largest face angle and stores it in the face angle database. This is implemented with techniques based on the Hough transform. This database also contains a selection of rotational variations for the current shape. The face angles are sorted such that the angle of the largest face appears at the top of the database. The other entries are modified (by a fixed angle rotation factor) versions of this value. The predicate *blob_cnt* counts the number of "free space blobs," that is, the number of blocks of free space available to the packer:

- **If** blob count is 1, **then** best fit has occurred so exit **and** view the next shape.
- **If** blob count is 0, **then** read the new angle from face angles database **and** retry.
- **If** blob count<local optimum, **then** update blob count **and** update the local optima storage buffer before trying the next angle in the face angles database.
- **If** blob count≥local optimum, **then** try the next angle in database.

```
polypack:-
  scene_capture,             % Capture scene to be packed.
  get_all_shapes,            % Find all the input shapes.
  poly_pack_1.               % Pack shapes - largest first sorted.

/*
  Main packing coordination predicate.
*/
poly_pack_1:-
  big_shape_first,           % Pack the shapes by SE size.
  centre_screen_se,          % Centre SE in the field of view.
  shape_face_angles.         % Find the angle of the largest face.
  face_angles_db(LONGEST),
  retract(face_angles_db(LONGEST)),
                             % Recover a face angle database value.
  turn(LONGEST,A,B),         % Rotates the structuring element about its centre of gravity, by
                             % the angle recovered from the face angle database.

  poly_pack_main,
  !,
  poly_pack_1.               % Get the next shape to pack.

poly_pack_1:-
  performance_meas.          % Calculate performance values.

poly_pack_main:-
  morph_pack,                % Morphological packing.
  read_updated_image,
  blob_cnt.

/*
  Morphological packing predicate.
*/
morph_pack:-
  updated_image,
  $'PACK',                   % Find the erosion residue.
  space_out,                 % Dilate image by single pixel to make packing clearer.
  residue_check,             % No residue exists then try next angle. No residue indicates
                             % that the shape cannot be packed in current orientation so quit.
  $'TRES',                   % Pack original SE.
  main_pack.                 % Place SE onto erosion residue.
```

7.3 *Performance Measurement Predicates*

The following predicates evaluate the goodness of fit of a given packing procedure. The packing density is defined as the ratio of the optimal packing area (which is the sum of the area of the individual shapes to be packed) to that of the area of the convex hull of the packed shapes (minus the area of any scene defects). This is a standard packing measurement that will reach a maximum as it approaches one. The performance index is a modified version of the packing density that accounts for unpacked shapes. This equals packing density × count ratio, and has a maximum of one. The count ratio is defined as the ratio of the total number of shapes packed to that of the number of original shapes presented to the packing procedure. Space usage is defined as the ratio of the area of the shapes packed to that of the unpacked original shape. This gives us an idea of the amount of space unpacked in the original scene. This ratio has a maximum value of one, which occurs when no space remains unpacked.

```
performance_meas:-
   read_unpacked_image,
   count_white_pixels(TOTAL_AREA),
                              % Unpacked image area.
   read_packed_image,
   count_shapes(PACK_SHAPE_COUNT),
                              % Number of shapes actually packed.
   read_input_shapes,
   count_shapes(ORG_SHAPE_COUNT),
                              % Number of original shapes presented to the packing system.
   find_areas(OPT_AREA,PACK_AREA),
                              % Optimal and packed area, accounting for defective regions.
   ORG_SHAPE_COUNT > 0, PACK_AREA > 0, TOTAL_AREA > 0,
                              % Prevent division by zero.
   COUNT_RATIO is PACK_SHAPE_COUNT / ORG_SHAPE_COUNT,
   PACKING_DENSITY is OPT_AREA / PACK_AREA,
   PERFORMANCE_INDEX is PACKING_DENSITY * COUNT_RATIO,
   SPACE_USAGE is OPT_AREA / TOTAL_AREA,
   print_performance_measures.

performance_meas:-
   writenl(''),
   writenl('Divide by zero error ').          % Error checking.
```

Acknowledgments

This research work was supported by the School of Electronic Engineering at Dublin City University and by a travel grant from Eolas and the British Council.

References

1. H. L. Dreyfus and S. E. Dreyfus, *Mind Over Machine*, pp. 184–188, The Free Press, New York (1986).
2. H. Freeman, "Is industry ready for machine vision—A panel discussion," in *Machine Vision for Inspection and Measurement*, pp. 223–236, Academic Press, New York (1989).
3. P. F. Whelan and B. G. Batchelor, "Automated packing of arbitrary shapes," *Machine Vision Architectures, Integration, and Applications, Proc. SPIE* **1615**, 77–86 (1991).
4. W. B. Dowsland, "Two and three dimensional packing problems and solution methods," *New Zealand Operational Res.* 13(1), 1–18 (1985).
5. M. R. Garey and D. S. Johnson, *Computers and Intractability—A Guide to the Theory of NP-Completeness*, W. H. Freeman and Co., San Francisco (1979).
6. B. G. Batchelor, *Intelligent Image Processing in Prolog*, Springer-Verlag, London (1991).
7. B. G. Batchelor, "Interpreting the radon transform using Prolog," *Machine Vision Architectures, Integration, and Applications, Proc. SPIE* **1615**, 87–97 (1991).
8. E. E. Bischoff and M. B. Marriott, "A comparative evaluation of heuristics for container loading," *European J. Operational Res.* **44**, 267–276 (1990).
9. I. Stewart, "How to succeed in stacking," *New Scientist*, pp. 29–32 (July 1991).
10. H. L. Ong, M. J. Magazine, and T. S. Wee, "Probabilistic analysis of bin packing heuristics," *Operations Res.* **32**(5), 983–998 (1984).
11. H. Wolfson, E. Schonenberg, A. Kalvin, and Y. Lamdan, "Solving jigsaw puzzles by computer," *Annals of Operations Res.* **12**, 51–64 (1988).
12. N. Hudson, C. F. Kelly, E. McQuade, and M. A. Rahman, "Application of artificial intelligence to two dimensional assembly tasks," in *AI and Cognitive Science '89*, pp. 3–19, British Computer Society (1989).
13. C. S. Chen, S. Sarin, and B. Ram, "The pallet packing problem for non-uniform box sizes," *Intl. J. Prod. Res.* **29**(10), 1963–1968 (1991).
14. R. Penrose, *The Emperor's New Mind*, pp. 562–566, Oxford University Press, New York (1989).
15. J. L. Schneiter and T. B. Sheridan, "An automated tactile sensing strategy for planar object recognition and localization," *IEEE Trans. PAMI* **12**(8), 775–786 (1990).
16. J. Uhry, "Applications of simulated annealing in operation research," in *New Methods in Optimization and Their Industrial Uses*, pp. 191–203, Birkhuaser-Verlag, Basel, Switzerland (1989).

Paul F. Whelan received his BEng (Hons) degree from the National Institute for Higher Education, Dublin, Ireland, and his MEng degree in machine vision from the University of Limerick in 1985 and 1990, respectively. He was employed by Industrial and Scientific Imaging Ltd. from 1985, where he was involved in the research and development of machine vision inspection systems. He was later employed by Westinghouse as a senior design engineer working on their range of machine vision inspection products. In August 1992 he was appointed to a lectureship in electronic engineering on the Faculty of Engineering and Design at Dublin City University. He is a member of SPIE, IEEE, and a senior member of SME(MVA).

Bruce G. Batchelor was awarded the BSc and PhD degrees from the University of Southampton, United Kingdom, in 1965 and 1969, respectively. He was employed in the Electronics Research Laboratory of the Plessey Company Ltd. from 1968 to 1970. In January 1971, he was appointed to a lectureship in electronics at the University of Southampton. From 1980 to 1992, he was a professor of electronic engineering at the University of Wales, Cardiff, and currently he is a professor of computing mathematics there. His research interests are in the application of intelligent image processing techniques to industrial inspection and control. He has published five books and more than 180 technical articles on automated visual inspection, machine vision systems, and pattern recognition.