ELSEVIER Computers in Industry 29 (1996) 197–208

COMPUTERS IN INDUSTRY

An algorithm for converting the contour of a 2D workpiece into a rectilinear polygon

Muh-Cherng Wu *, J.T. Wang

Department of Industrial Engineering and Management, National Chiao Tung University, Hsin-Chu, Taiwan

Received 5 September 1994; revised 10 January 1996; accepted 10 January 1996

Abstract

This paper presents an algorithm for converting the contour of a 2D workpiece into a rectilinear polygon. The purpose of providing such a conversion is to model the global shape information of the workpiece. Using an existing algorithm, the rectilinear polygon can further be modeled by a tree structure of line segments, known as the simplified skeleton, which can concisely model the global shape information of a workpiece. That is, workpieces with similar global shape can be classified into the same group by considering the similarity among their simplified skeletons. This algorithm is helpful to enhance the modeling capability of the traditional group technology coding schemes.

Keywords: Group technology; Rectilinear polygon; Skeleton; Global shape information modeling

1. Introduction

In a flexible manufacturing environment, workpieces to be produced in general vary widely. Numerous coding schemes, known as group technology (GT) [1], were developed to classify workpieces into groups according to their similarity in design and manufacturing characteristics. Such a grouping would facilitate the tasks of design and manufacturing by referring to the relevant information of similar workpieces.

Significant milestones in the classification of workpieces have been made by existing GT coding schemes. In order to facilitate the implementation of GT, some studies aim to develop methods for automating the tasks of workpiece coding and building a CAD database [2–5]. However, most of them concentrate on modeling local features and are deficient in modeling the global shape information of workpieces.

Recently, a representation scheme known as the simplified skeleton [6,7] was proposed to model the global shape information of a rectilinear polygon, a polygon where any two of its boundary edges are either parallel or perpendicular to each other. The simplified skeleton of a rectilinear polygon, being a tree structure, is composed of two types of line segments: real links and virtual links. A real link models a rectangle and a virtual link describes the spatial relationship between two real links. The union of modeling rectangles described by all real links is equal to the original rectilinear polygon. That is, the rectilinear polygon can be decomposed by a set of rectangles with their spatial relationships explicitly given. The simplified skeleton therefore provides a way to model the global shape information of a rectilinear polygon.

* Corresponding author.

As shown in Fig. 1(a), the simplified skeleton of the I-shaped workpiece (a rectilinear polygon) is composed of five real links (solid line segments) and four virtual links (dashed line segments). Each real link models a rectangle (Fig. 1(b) and Fig. 1(c)), and the union of these five rectangles is equal to the original workpiece (Fig. 1(d)). Ignoring the length of virtual links, which denotes only the spatial relationships, these real links behave as an I-shaped tree. This concise representation is very helpful to the classification of workpieces because another workpiece, similar in global shape, would also give an I-shaped simplified skeleton (Fig. 1(e) and Fig. 1(f)). A neural network approach to classify 2D rectilinear workpieces by utilizing simplified skeletons have been proposed and shown satisfactory results [6].

However, the contours of most workpieces, represented in their 2D projections, are not rectilinear polygons. In order to concisely model the global shape information of workpieces, an algorithm for converting the contour of a 2D drawing into a rectilinear polygon is needed. As shown in Fig. 2, considering only the contour information, the top view of the 3D workpiece may be approximately modeled by a rectilinear polygon, which is similar to the original global shape yet relatively concise in representation.

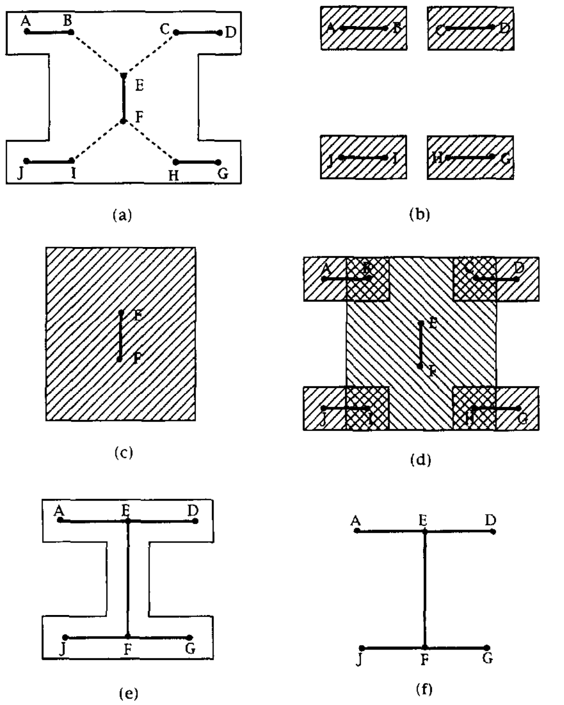


Fig. 1. Rectilinear polygon, simplified skeleton, and the area modeling characteristics of real links.

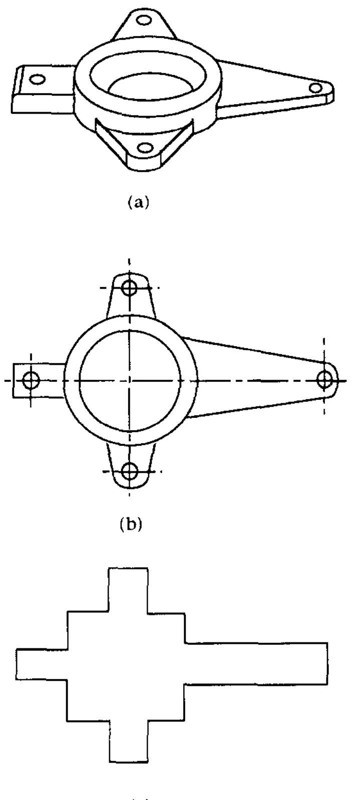


Fig. 2. Contour of a 2D workpiece and its modeling rectilinear polygon.

Some research relevant to the dealing of rectilinear polygons have been published in previous literature [8–13]. These studies intended to decompose a rectilinear polygon into a set of rectangles that have the fewest number so that the required computation efforts for certain applications would be minimized. The modeling rectangles are allowed to be either overlapped or non-overlapped. The rectilinear polygon decomposition problems can be applied to pattern recognition, VLSI layout, computer graphics, databases, and image processing. Several useful algorithms for the decomposition of rectilinear polygon have been developed. However, the problem of converting the contour of a 2D workpiece drawing

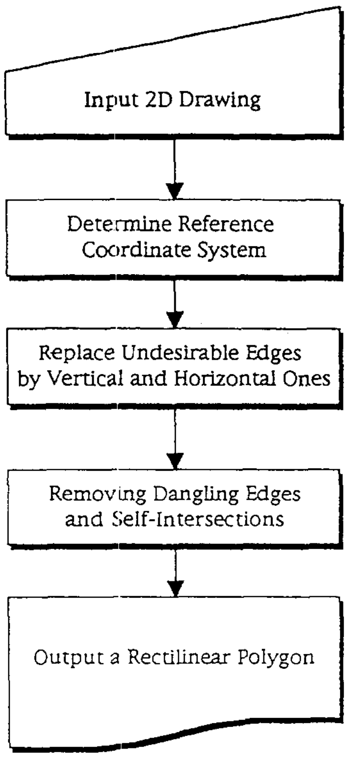

Input 2D Drawing
Determine Reference Coordinate System
Replace Undesirable Edges by Vertical and Horizontal Ones
Removing Dangling Edges and Self-Intersections
Output a Rectilinear Polygon

Fig. 3. System framework of the algorithm.

into a rectilinear polygon is seldom discussed in previous literature.

This paper presents an algorithm for converting the contour of a 2D workpiece drawing into a rectilinear polygon. The contours of concerned 2D drawings are limited to those which are composed of line segments and circular arcs. Other than contours, center lines on a 2D drawing, which are generally used to locate center points, axes of cylindrical parts, and axes of symmetry, are also considered as major input to the algorithm.

The proposed algorithm is composed of three major modules (Fig. 3). The first module is intended to determine a reference coordinate system on which to define the resulting rectilinear polygon. That is, each edge on the rectilinear polygon should be parallel to either X or Y axis of the reference coordinate system. And on the rectilinear polygon, an edge is known as a *horizontal or vertical edge* depending upon it is parallel to X or Y axis of the reference coordinate system.

With respect to the contour of the input drawing, the second module is designed to remove its circular arcs and those line segments which are not parallel to any axis of the reference coordinate system; and replace them by vertical or horizontal edges in order to construct a rectilinear polygon. However, rectilinear polygons constructed in this way may have their edges self-intersected and include dangling edges, which are undesirable and contrary to the definition of polygons. In the third module, procedures are designed to detect and remove these undesired phenomena.

Through a large number of empirical testing, the proposed algorithm has shown satisfactory results in converting the contour of a 2D workpiece drawing into a rectilinear polygon. This algorithm is an intermediate step in implementing a group technology technique based on a novel workpiece classification criteria – the global shape information. The GT technique is briefly explained below. For a rectilinear polygon representing a 2D workpiece drawing, its simplified skeleton is first derived for modeling its global shape information. Secondly, the simplified skeletons are used to classify the 2D drawings into groups. Finally, 3D workpieces can be grouped according to the classification results of its three 2D drawings.

2. Determining reference coordinate system

2.1. Significance of determining the base line

The choice of the reference coordinate system is very important. For an input 2D drawing, an inappropriate choice of the reference coordinate system may create a rectilinear polygon which is far from similar to the original 2D drawing. As shown in Fig. 4, the original workpiece (Fig. 4(a)) can be modeled by a similar rectilinear polygon (Fig. 4(b)) if edge $\overline{AB}$ is taken as the *base line*, which by definition denotes that either X or Y axis of the reference coordinate system should be parallel to the particular edge ($\overline{AB}$). On the other hand, if edge $\overline{CD}$ is chosen as the base line, then the resulting rectilinear polygon would be formed as shown in Fig. 4(c), which comparatively looks not so similar to the original input drawing by considering their global shapes.

2.2. Criteria for choosing the base line

In this algorithm, a composite of two criteria is used to justify the choices of the baseline. The first criterion is known as *the number of inclined edges* which by definition are line segments on the contour of the input drawing which are neither parallel nor perpendicular to the base line. As shown in Fig. 4(d), by choosing edge $\overline{AB}$ as the base line, $\overline{CD}$ is an inclined edge; conversely, by choosing $\overline{CD}$ as the base line, edges $\overline{AB}$, $\overline{BC}$, and $\overline{DE}$ then become inclined edges.

The second criterion is known as the *sum of area differences* of inclined edges. Referring to Fig. 4(d), for each inclined edge (eg., $\overline{CD}$), a value known as the *area difference* can be defined, which denotes the area sum of two triangles (ΔJKC and ΔDLK). The two triangles are formed by first creating either a vertical or a horizontal edge passing through the midpoint (K) of the inclined edge ($\overline{CD}$) and then constructing two perpendicular edges ($\overline{CJ}$ and $\overline{DL}$) at its ending points. Of the two options, the first created edge ($\overline{JL}$) should be chosen in such a way that the formed triangles have smaller area. That is, the chosen one ($\overline{JL}$) should have smaller intersection angle with the inclined edge ($\overline{CD}$). For an inclined edge, the value of its area difference estimates the degree of local similarity between the original drawing and the resulting rectilinear polygon. Summing up the area differences for all inclined edges would give the value of the second criterion.

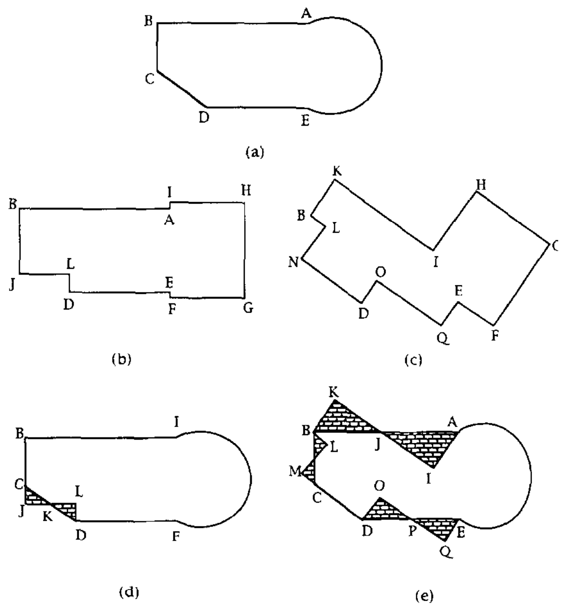


Fig. 4. Two criteria for justifying the choices of the base line, number of inclined edges and the sum of area difference.

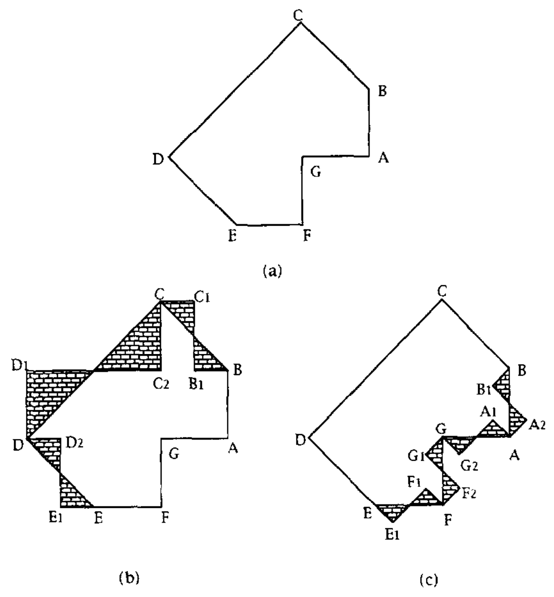


Fig. 5. Conflict of the two criteria for justifying the choices of the base line.

The two criteria may favor different choices of the base line for some 2D drawings. Referring to the workpiece in Fig. 5(a), by choosing edge $\overline{AB}$ as the base line, only three edges ($\overline{BC}$, $\overline{CD}$, and $\overline{DE}$) on the input drawing become inclined edges (Fig. 5(b)); yet, four edges ($\overline{AB}$, $\overline{AG}$, $\overline{GF}$, and $\overline{FE}$) will become inclined ones if edge $\overline{CD}$ is chosen as the base line (Fig. 5(c)). That is, according to the first criterion, choosing edge $\overline{AB}$ as the base line would be favored. However, with respect to the second criterion, choosing edge $\overline{CD}$ as the base line would be better.

In order to solve such a conflict, a composite criterion, which is the multiplication of the two criteria (known as C value), is defined for justifying the choices of base line. The smaller is the C value, the more favored is the base line. According to the composite criterion, in Fig. 5, $\overline{CD}$ should be selected as the base line.

2.3. Candidates of base line

On the input drawing, candidates of the base line involve two types: center lines within the drawing and line segments on the contour. These two types of lines are chosen for their tendency to meet either one of the above two criteria. A center line usually denotes a symmetric feature on the input drawing; choosing it as the base line would tend to reduce the sum of area difference (Fig. 2). On the other hand, choosing a line segment on the contour as the base line would reduce the number of inclined edges (at least one).

For all the candidates of the base line, each one is justified by computing its C value; the one which has the smallest C value is chosen as the base line.

3. Replacing undesirable edges by vertical/horizontal ones

In the second module of the algorithm, circular arcs are first replaced by line segments in order to convert the input drawing into a piecewise linear shape which is further converted into a rectilinear one by replacing all the inclined edges with vertical or horizontal edges.

3.1. Replacing circular arcs by line segments

The procedure for replacing a circular arc involves two steps. First, the circular arc is decomposed into a set of arc segments where the arc angle of each segment is less than $\pi/4$. Second, each arc segment is replaced by two line segments according to the differential characteristics of its two ending vertices.

The way to segmenting a circular arc is by identifying some *breaking points* on the arc, which are points with their tangent lines either parallel or perpendicular to the base line such as points P_1, P_2, P_3, and P_4 in Fig. 6. These breaking points together with the two original ending vertices (P_s and P_e) then are used to chop the original arc into several arc segments, and each subsequent pair forms the two ending vertices of an arc segment.

The ending vertices of all arc segments are classified into two types: smooth and non-smooth junctions. A smooth junction is one which is first order derivative continuous (e.g., vertex V_s in Fig. 7(a)); whereas a non-smooth junction (e.g., vertex V_e in

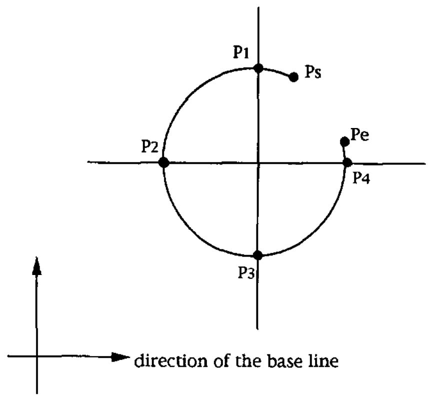


Fig. 6. Determining breaking points of a circular arc.

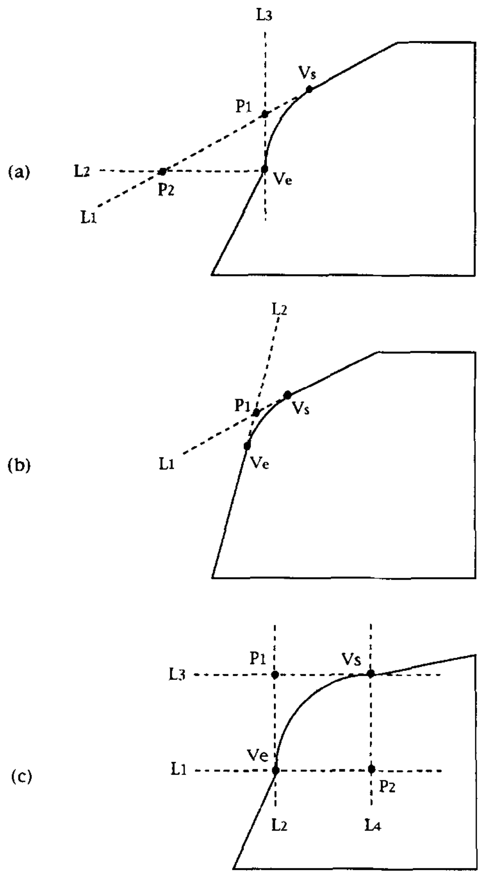


Fig. 7. Determining auxiliary lines for replacing circular arcs.

Fig. 7(a)) is not. To replace an arc segment, one or two auxiliary lines at a junction should be created. Referring to Fig. 7(a), at a smooth junction (V_s), one auxiliary line (L_1) tangent to the arc segment should be constructed. And at a non-smooth junction (V_e), two auxiliary lines (L_2 and L_3), one parallel and the other perpendicular to the base line should be created. For the two junctions of any arc segment, as shown in Fig. 7, their two sets of auxiliary lines at most have two intersection points (P_1 and P_2). The intersection point (P_1) which is outside and closer to the center of the whole circle should be chosen as a new vertex, for creating two new line segments ($\overline{P_1V_s}$ and $\overline{P_1V_e}$) to replace the arc segment (V_sV_e).

3.2. Replacing inclined edges

To create a rectilinear polygon, each inclined edge should be replaced by vertical and horizontal edges. Referring to Fig. 8(a) and Fig. 8(b), the way to replacing an inclined edge ($\overline{AB}$) is first by creating a auxiliary line ($\overleftrightarrow{A_1B_1}$), either parallel or perpendicular to the base line and passing the midpoint (M_1) of the edge, then computing the projection ($\overline{A_1B_1}$) of the inclined edge on the auxiliary line. Finally, the inclined edge $\overline{AB}$ is replaced by three vertical/horizontal edges: its projection on the auxiliary line ($\overline{A_1B_1}$), and the two linear connections of ending vertices between the inclined edge and its projection ($\overline{A_1A}$ and $\overline{B_1B}$). Notice that the auxiliary line may be either parallel or perpendicular to the base line. Of the two options, the one which has smaller intersection angle with the inclined edge should be chosen. For example, the auxiliary line for replacing $\overline{AB}$ is chosen to be parallel to the base line, while that for replacing $\overline{IJ}$ is chosen to be perpendicular.

Notice that, for each edge on the original polygon, a counter-clockwise traversing direction can be defined so that the interior side of the polygon is to the right-side of each edge. The traversing characteristic of each inclined edge should be inherited by its replacing vertical/horizontal edges. As shown in Fig. 8(a) and Fig. 8(b), the traversing direction of edge $\overrightarrow{BA}$ shall be inherited by the three vertical/horizontal edges ($\overrightarrow{BB_1}$, $\overrightarrow{B_1A_1}$, and $\overrightarrow{A_1A}$).

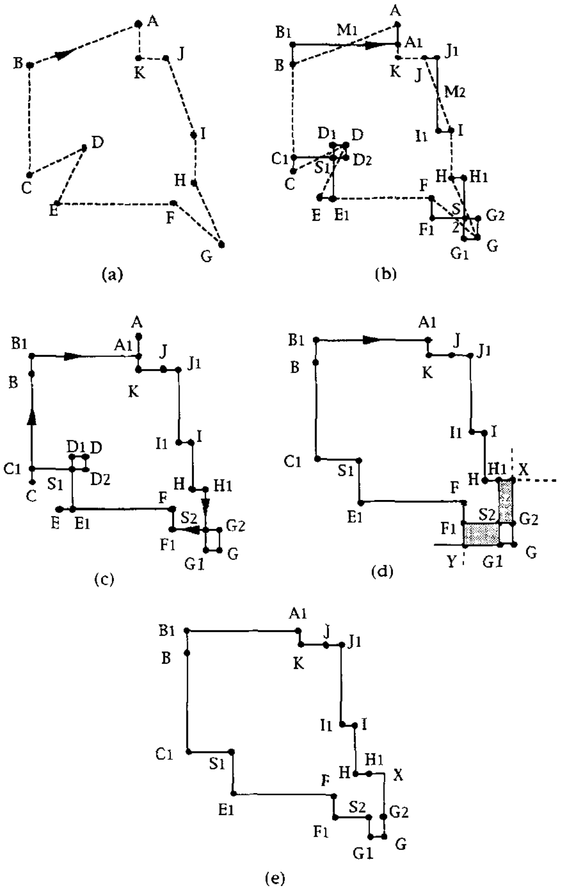


Fig. 8. Dangling edges and self-intersections caused by the replacement of inclined edges.

The inheritance of traversing directions is very helpful to removing some undesirable phenomena such as self-intersections caused by the replacement of inclined edges.

4. Removing dangling edges and self-intersections

The method for removing inclined edges would create some undesirable phenomena: dangling edges, and self-intersections. As shown in Fig. 8(c), a dangling edge, such as edge $\overline{A_1A}$, has a vertex (A) which connects no neighboring edge and cannot form a part of the polygon contour; with this characteristic, a dangling edge can be easily detected and removed. Self-intersection is a phenomenon where two non-neighboring edges (e.g, $\overline{C_1D_2}$ and $\overline{E_1D_1}$) have an undesirable intersection (S_1). Typical examples as shown in Fig. 8(c) involve the intersection point S_1 formed by edges $\overline{C_1D_2}$ and $\overline{E_1D_1}$, and the point S_2 formed by edges $\overline{G_1H_1}$ and $\overline{G_2F_1}$.

The self-intersection phenomena involve two types: *interior and exterior self-intersections*. An interior self-intersection is one such as the self-intersection point S_1 in Fig. 8(c) where a undesirable region ($S_1D_2DD_1$) is formed inside the area of the polygon. Conversely, an exterior self-intersection is one such as the self-intersection point S_2 in Fig. 8(c) where a undesirable region ($S_2G_2GG_1$) is formed outside the polygon. The two types of undesirable regions are respectively known as *interior regions* and *exterior regions*.

The method of distinguishing interior and exterior regions is by checking the patterns of edge traversing directions at each self-intersection point. For example, at the self-intersection point S_2 (Fig. 8(c)), its two neighboring edges can be classified as either moving-in ($\overrightarrow{H_1G_1}$) or moving-out ($\overrightarrow{G_2F_1}$). Notice that the ending point (G_1) of the moving-in edge is to the *outside* (left-side) of the moving-out edge ($\overrightarrow{G_2F_1}$); and the ending point (F_1) of the moving-out edge ($\overrightarrow{G_2F_1}$) is to the *inside* (right-side) of the moving-in edge ($\overrightarrow{H_1G_1}$). The *from-outside-to-inside* traversing pattern denotes that the self-intersection (S_2) belongs to the exterior type. Conversely, at the self-intersection point such as S_1, the traversing pattern is *from-inside-to-outside* and is classified into the interior type.

In the proposed algorithm, an interior region should be removed, and an exterior one should be modified to be included as a part of the resulting rectilinear polygon. Removing an interior region is relatively easy by just deleting the edges ($\overline{S_1D_1}$, $\overline{D_1D}$, $\overline{DD_2}$, and $\overline{D_2S_1}$) which are on the traversing path of the *from-inside-to-outside* pattern.

The method for modifying an exterior region is by creating two virtual rectangles. Referring to Fig. 8(d), one (rectangle $H_1XG_2S_2$) is formed by the self-intersection (S_2) and the starting points (H_1 and G_2) of its two neighboring edges ($\overrightarrow{H_1G_1}$ and $\overrightarrow{G_2F_1}$);

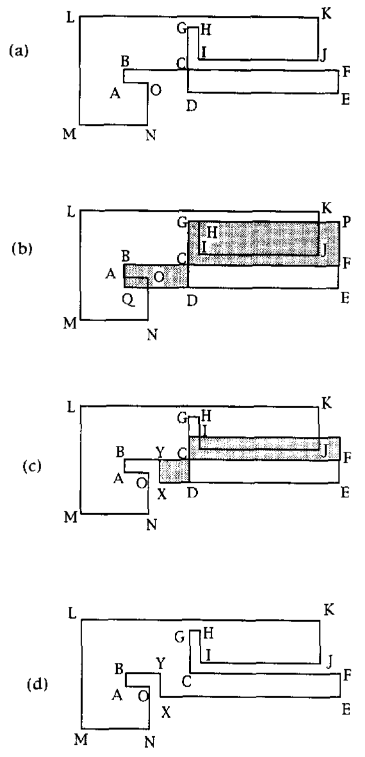


Fig. 9. Creating and adding virtual rectangles to remove the phenomena of self-intersections.

the other (rectangle $S_2F_1YG_1$) is formed by the self-intersection point (S_2) and the ending points (F_1 and G_1) of its two neighboring edges. Of the two virtual rectangles, the one (rectangle $H_1XG_2S_2$) which is smaller in area and has no intersection with other edges is chosen as a part of the resulting rectilinear polygon. The reason for choosing the smaller virtual rectangle is that the resultant polygon would be more close to the original one. By adding this virtual rectangle, the rectilinear polygon would be modified by replacing two existing edges ($\overline{H_1S_2}$ and $\overline{S_2G_2}$) with two newly created edges ($\overline{H_1X}$ and $\overline{XG_2}$). And the resulting rectilinear polygon would be as shown in Fig. 8(e).

As shown in Fig. 9(a) and Fig. 9(b), the two virtual rectangles (BCDQ and GCFP) of a self-intersection point (C) may both have undesirable intersection with neighboring edges. In such cases, the size of each virtual rectangle is reduced by half as shown in Fig. 9(c); the subdivision procedure is repeated until a virtual rectangle which has no intersection

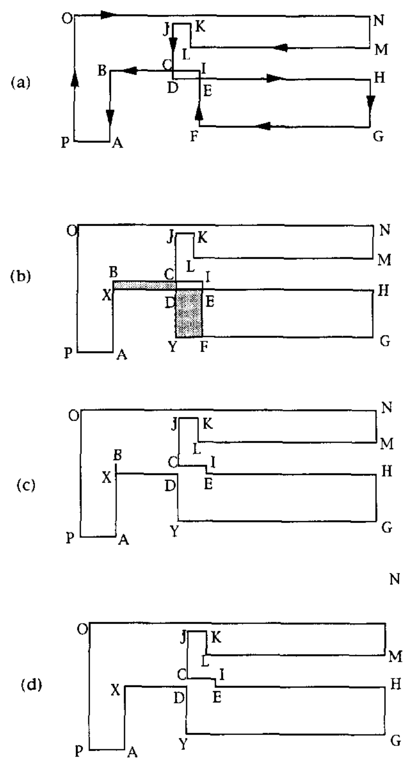


Fig. 10. An exterior region involving two self-intersection points.

with neighboring edges is found. Adding the non-intersecting virtual rectangle to the original region, the resulting rectilinear polygon would be as shown in Fig. 9(d).

Another example for illustrating an exterior region involving two self-intersection points is shown in Fig. 10(a). The two vertices C and E are exterior self-intersection points because their traversing paths are both of the pattern *from-outside-to-inside*. That is, with respect to the self-intersection point C, the ending point (D) of moving-in edge ($\overrightarrow{JD}$) is to the left-side of the moving-out edge ($\overrightarrow{IB}$), and the ending point (B) of the moving-out edge ($\overrightarrow{IB}$) is to the right-side of the moving-in edge ($\overrightarrow{JD}$). Likewise, with respect to self-intersection point E, point I is to the left-side of moving-in edge $\overrightarrow{DH}$ and point H is to the right-inside of moving-out edge $\overrightarrow{IF}$.

Similarly, the modification of such an exterior region is by adding a virtual rectangle for each exterior self-intersection. As shown in Fig. 10(b), rectangles BCDX and DEFY are to be included in the resulting rectilinear polygon; therefore, edges $\overline{BC}$ and $\overline{CD}$ should be replaced by edges $\overline{BX}$ and $\overline{DX}$, and edges $\overline{DE}$ and $\overline{EF}$ should be replaced by edges $\overline{DY}$ and $\overline{FY}$. Notice that edge $\overline{BX}$ now can be checked to be a dangling edge and should be removed (Fig. 10(c)), and the resulting rectangle is as shown in Fig. 10(d).

5. Rectilinear polygons and simplified skeletons

The algorithm for converting the contour of a 2D workpiece drawing into a rectilinear polygon has been presented. The purpose for providing such a conversion is to derive the simplified skeleton of the rectilinear polygon in order to model the global shape information of the workpiece, which can be used in certain applications such as fixture design [14] and workpiece grouping [15]. The definition and derivation of simplified skeletons is briefly explained below to highlight potential applications of this algorithm.

For a rectilinear polygon, the definition of its simplified skeleton can be illustrated by the "enhanced firing grassland" paradigm proposed by Wu et al. [6], which is a modification of the "firing grassland" paradigm proposed by Blum [16]. In the modified paradigm, the area enclosed by the rectilinear polygon is similar to a grassland that is simultaneously fired from each of its boundary edges, according to the following three rules.

First, the fire on an edge should advance uniformly in a particular direction – the inward normal of the edge. Second, the fire on each vertex of the polygon should move on the bisector of its internal angle. Third, the firing path of an in-between point on an edge is the linear interpolation of the two firing paths of its two ending vertices. As shown in Fig. 11(a), edge $\overline{BC}$ move to $\overline{A'C'}$; vertices A and B both move to vertex A′; in-between points P and X respectively move to P′ and X′ where their firing paths are the linear interpolation of firing paths $\overline{BA'}$ and $\overline{CC'}$.

Applying the enhanced firing operation on a recti-

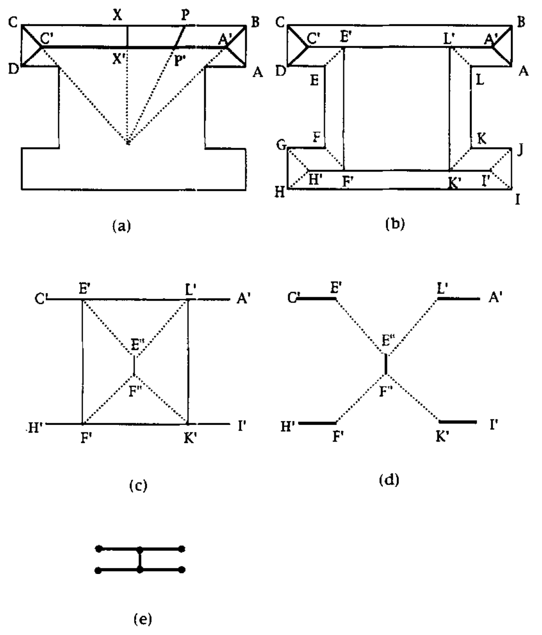


Fig. 11. Procedure for deriving the simplified skeleton of a rectilinear polygon.

linear polygon, the simplified skeleton is defined as *the set of points where the fire of two non-neighboring edges meet in their advancing paths.* When a rectilinear polygon is applied by the enhanced firing operation, its area gradually shrinks and in some cases multiple shrinkage processes may be required to derive its simplified skeleton. An example illustrating the derivation of a simplified skeleton is given below.

Consider a rectilinear polygon consisting of twelve edges (Fig. 11(a)), this region would require two shrinkage processes to derive its simplified skeleton. As shown in Fig. 11(b), in the process of firing, the region gradually shrinks until the fire of non-neighboring edges $\overline{BC}$ and $\overline{AL}$ meet at edge $\overline{A'L'}$. At this instant, a part of simplified skeleton is derived and therefore is called the end of the first shrinkage process. The *shrinkage result* of this process is a rectangular region (K′L′E′F′) connected with four *dangling edges* ($\overline{A'L'}$, $\overline{C'E'}$, $\overline{H'F'}$, $\overline{I'K'}$) as shown in Fig. 11(c).

A dangling edge ($\overline{A'L'}$) is a part of the simplified skeleton as the fire of two non-neighboring edges ($\overline{BC}$ and $\overline{AL}$) meet here, and should be removed from the shrinkage result because it cannot be shrunk any more. By removing the four dangling edges, the rectangular region (K′L′E′F′) then is called the *residual result* of the first shrinkage process and the second shrinkage operation should be applied to the region.

In the second shrinkage process, the rectangular region (K′L′E′F′) could be reduced to a single line segment $\overline{E''F''}$, which is known as the *residual edge*, a part of the simplified skeleton. Notice that points E′, L′, K′, and F′ are called *special points* because at these points the fire of two non-neighboring edges have been met and these points remain on the residual result to be shrunk further. The firing trace of special points always fulfill the definition of simplified skeleton and therefore have to be recorded as a part of simplified skeleton. In a simplified skeleton, the firing traces of special points (i.e., $\overline{E'E''}$, $\overline{L'E''}$, $\overline{K'F''}$, and $\overline{F'F''}$) are known as *virtual links*; the other line segments (dangling or residual edges) are called *real links* (i.e., $\overline{A'L'}$, $\overline{C'E'}$, $\overline{H'F'}$, $\overline{I'K'}$ and $\overline{E''F''}$).

As stated previously (Fig. 1), each real link models a rectangular region and a virtual link models the spatial relationship between two real links. That is, a simplified skeleton can be seen as a tree structure consisting of only real links, and the information embedded in virtual links can be modeled as the attributes of real links. By such a modeling, the simplified skeleton of the rectilinear polygon (Fig. 11(a)) can be concisely represented as an I-shaped tree structure as shown in Fig. 11(d), which essentially models the global shape information of the workpiece. With the availability of simplified skeletons, the global shape similarity between two workpieces can be measured by determining the similarity between two tree structures [14]. Such a similarity measure would be helpful to enhance the capability of the traditional GT coding schemes.

6. Implementation

The algorithm has been implemented on a PC486 machine equipped with Turbo C 2.0 and AutoCAD softwares. The input 2D drawings are drawn in the AutoCAD environment and the IGES format [17] of the input drawing is generated. The program will decode the IGES file and generate the rectilinear polygons in IGES format, which will subsequently

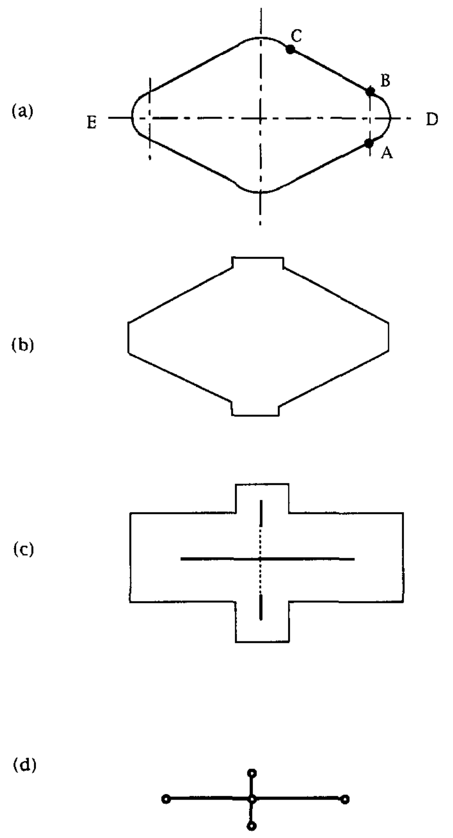


Fig. 12. Demonstrated workpiece 1 and relevant results of the algorithm.

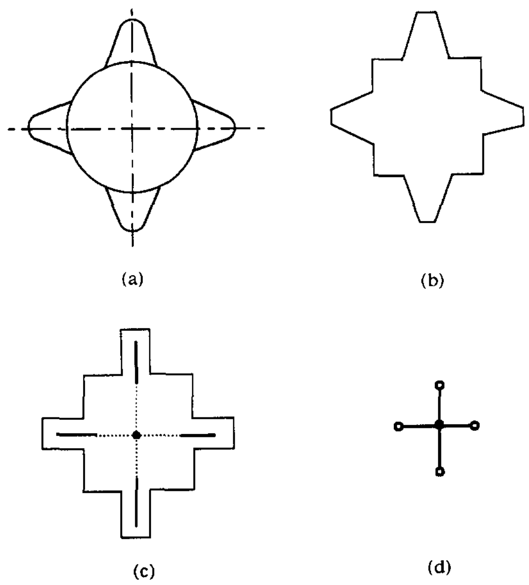


Fig. 13. Demonstrated workpiece 2 and relevant results of the algorithm.

be used in extracting their simplified skeletons. Some examples of the algorithm are demonstrated in this section.

Three typical 2D workpieces are used to illustrate the intermediate states and final results in the execution of the proposed algorithm. Fig. 12(a) gives the contour of the first workpiece; Fig. 12(b) shows the intermediate state after removing circular arcs; and the resulting rectilinear polygon is as shown in Fig. 12(c). Likewise, Figs. 13 and 14 show the intermediate states and final results of the other two workpieces. Note that there are two self-intersections in Fig. 14(c), and they are removed by the addition of two virtual rectangles to generate a rectilinear polygon as shown in Fig. 14(d).

It is notable that the simplified skeletons of the three demonstrated workpieces are quite concise in their representation. The two simplified skeletons

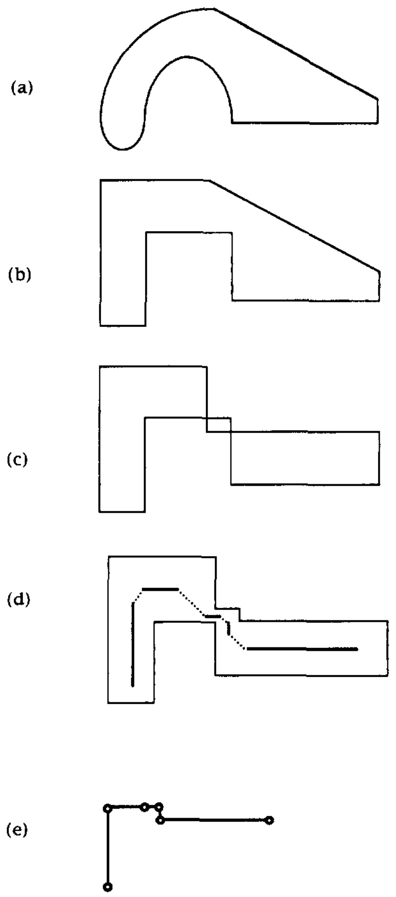


Fig. 14. Demonstrated workpiece 3 and relevant results of the algorithm.

shown in Fig. 12(d) and Fig. 13(d) both appear like a cross, and the one in Fig. 14(d) is like a distorted L-shape. That is, among the three workpieces, the two in Figs. 12 and 13 are more similar in their global shapes. The two similar ones, if needed, can further be distinguished by considering the attribute of each branch on their tree structures (simplified skeletons).

7. Concluding remarks

An algorithm for converting the contour of a 2D workpiece into a rectilinear polygon is proposed. The purpose of developing such an algorithm is to provide a preparatory stage for modeling the global shape information of the workpiece. That is, the rectilinear polygon would further be processed to derive its simplified skeleton, a tree structure consisting of line segments, which is quite concise in representation and can effectively model the global shape of the original workpiece.

The algorithm is notable in providing orientation independent results; that is, in whatever orientation the workpiece is placed, the derived rectilinear polygon is always the same. Likewise, the procedure for deriving simplified skeletons from rectilinear polygons is also orientation independent. These two characteristics provide a justification for adopting the simplified skeleton as a representation scheme for modeling the global shape information of a workpiece.

Acknowledgements

This work is partially supported by NSC83-0415-E009-001, National Science Council, Taiwan, R.O.C.

References

[1] M.P. Groover and E.W. Zimmers, *CAD / CAM: Computer-Aided Design and Manufacturing*, Prentice Hall, Englewood Cliffs, NJ, 1984.

[2] J. Grum, B. Logar, G. Hlebanja and J. Peklenik, "Design of the database for CAD based on group technology", *Robotics and Computer Integrated Manufacturing*, Vol. 4, No. 1/2, 1988, pp. 49–62.

[3] B. Logar and J. Peklenik, "Feature-based part database design and automatic forming of part families for GT", *Annals of the CIRP*, Vol. 40, No. 1, 1991, pp. 153–156.

[4] Y. Kakazu and N. Okino, "Pattern recognition approaches to GT code generation on CSG", *16th CIRP International Seminar on Manufacturing Systems*, Tokyo, July 13–14, 1984.

[5] N. Tokuoka, T. Maeda and G.T. Sato, "Development of GT-CAD for definition of views of machined parts", *Proceedings of the 1987 International Conference on Engineering Design*, Boston, Augustus 17–20, 1987.

[6] M.C. Wu, J.R. Chen and S.R. Jen, "Global shape information modeling and classification of 2D workpieces", *International Journal of Computer Integrated Manufacturing*, Vol. 7, No. 5, 1994, pp. 261–275.

[7] M.C. Wu and J.R. Chen, "A skeleton approach to modelling 2D workpieces", *Journal of Design and Manufacturing*, Vol. 4, 1994, pp. 229–243.

[8] D.S. Franzblat and D.J. Klettman, "An algorithm for covering polygon with rectangles", *Information and Control*, Vol. 63, 1984, pp. 164–189.

[9] K.D. Gourley and D.M. Green, "A polygon-to-rectangle conversion algorithm", *IEEE Computer Graphics and Applications*, Vol. 3, No. 1, 1983, pp. 31–36.

[10] A. Hegedüs, "Algorithms for covering polygons by rectangles", *Computer Aided Design*, Vol. 14, No. 5, 1982, pp. 257–260.

[11] W.T. Liou and R.C.T. Lee, "Minimum rectangular partition problem for simple rectilinear polygons", *IEEE Transactions on Computer-Aided Design*, Vol. 9, No. 7, 1990, pp. 720–732.

[12] J. O'Rourke and K. Supowit, "Some NP-hard polygon decomposition problems", *IEEE Transactions on Information Theory*, Vol. IT-29, No. 2, 1983, pp. 181–183.

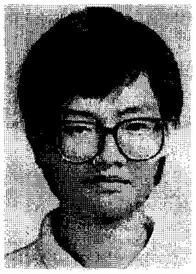
[13] S.Y. Wu, "Covering rectilinear polygon by rectangles", *IEEE Transaction on Computer-Aided Design*, Vol. 9, No. 4, 1990, pp. 377–387.

[14] M.C. Wu, and J.Y. Yen, "A skeleton-retrieving technique for aiding modular fixtures design", *International Journal of Advanced Manufacturing Technology*, Vol. 8, No. 3, 1993, pp. 123–128..

[15] M.C. Wu and S.R. Jen, "A neural network approach to the classification of 3D prismatic parts", *International Journal of Advanced Manufacturing Technology* (accepted for publication).

[16] H. Blum, "A transformation for extracting new descriptors of shape", in: W. Whaten-Dunn (ed.), *Models for the Perception of Speech and Visual Form*, MIT Press, 1967, pp. 362–380.

[17] I. Zeid, *CAD / CAM Theory and Practice*, McGraw-Hill, New York, 1989.



M.C. Wu is currently a Professor at the Department of Industrial Engineering and Management, National Chiao Tung University, Taiwan, R.O.C. He received the B.S. degree in Electronic Engineering from National Chiao Tung University in 1977, and the M.S. and Ph.D. degrees in Industrial Engineering from Purdue University in 1984 and 1988 respectively. His research interests include CAD/CAM and production management.

J.T. Wang is currently an administrator at a steel company. He received the B.S. degree in Computer Science from Feng-Chia University in 1991, and the M.S. degree in Industrial Engineering from National Chiao Tung University in 1993. This research is a part of his M.S. thesis work.