

Holo-extraction of information from paper drawings for 3D reconstruction

Ke-Zhang Chen^{a,*}, Xi-Wen Zhang^b, Zong-Ying Ou^c, Xin-An Feng^c

^a*Department of Mechanical Engineering, The University of Hong Kong, Pokfulam Road, Hong Kong*

^b*State Key Laboratory for Novel Software Technology, Nanjing University, Nanjing, 210093, China*

^c*Faculty of Mechanical Engineering, Dalian University of Technology, Dalian, 116024, China*

Received 27 April 2001; accepted 4 June 2001

Abstract

The research on the conversion from 2D paper drawings to 3D computer feature models has been stuck in low-level coding. One of the reasons for it is that the three phases of understanding process has been isolated and people have been doing their research on only one of the phases since the whole conversion is very complicated and more difficult. For instance, the vectorization methods for the first phase were developed only for getting straight lines, arcs, circles, etc. so that much information contained in the drawing has been lost after the vectorization. This paper develops a holo-extraction method of information from paper drawings, the networks of single closed regions (SCRs), which can not only provide a unified base for recognizing both the annotations and the outlines of projections of parts, but also build the holo-relationships among SCRs so that it is convenient to extract lexical, syntactic and semantic information in the subsequent phases for 3D reconstruction. © 2002 Elsevier Science Ltd. All rights reserved.

Keywords: Engineering drawing understanding; 3D reconstruction; Vectorization; Networks of single closed regions

1. Introduction

In order to survive worldwide competition, enterprises have tried to use the computer's huge memory capacity, fast processing speed, and user-friendly interactive graphics capabilities to automate and tie together cumbersome and separate engineering or production tasks, including design, analysis, optimization, prototyping, production planning, tooling, ordering materials, programming for numerically controlled (NC) machines, quality control, robot assembly, and packaging. The technologies for all these tasks rely on three-dimensional (3D) computer feature models of products made during design. Some advanced computer-aided design systems have been developed so that people can use these systems to build 3D computer feature models for new product designs, and the productions without drawings have thus appeared in some enterprises of developed countries. But enterprises have had a great deal of two dimensional (2D) mechanical paper drawings made in the past for their existing products, which need to be converted to 3D computer feature models for applications. The advanced computer-aided design systems can easily convert 3D computer models made with it into their 2D orthogonal projections, but cannot do the other way round. How to

convert 2D mechanical paper drawings into 3D computer feature models has been one of major issues many enterprises are very concerned about.

In order to convert 2D mechanical paper drawings into 3D computer feature models, the 2D paper drawings are first scanned by an optical scanner and inputted to a computer in the form of raster (binary) image. The conversion from the raster image to 3D model needs two processes: understanding and 3D reconstruction. The research on the understanding process has been implemented following three phases [1].

1. Lexical phase: The raster image is converted into the vectorized information, such as straight lines, arcs, and circles.
2. Syntactic phase: The outlines of orthographic projections of a part and the annotations are separated; the dimension sets, including their values of both the nominal dimensions and the tolerances, are aggregated; the crosshatching patterns are identified, and the text is recognized.
3. Semantic phase: Much semantic information needed for 3D reconstruction is obtained by functional analyses of each view, including the analysis of symmetries, the recognition of technologically meaningful entities from symbolic representation (such as bearing and threading), and so on.

* Corresponding author. Tel.: +852-2859-2630; fax: +852-2858-5415.

E-mail address: kzchen@hkucc.hku.hk (K.-Z. Chen).

After the understanding process, a 3D computer model will then be reconstructed by using geometric matching techniques with the recognized syntactic and semantic information.

Up to now, the research on the conversion from 2D paper drawings to 3D computer feature models has been stuck in low-level coding: essential vectorization, basic layer separation and very limited symbol recognition [1]. One of the reasons for it is that the three phases of understanding process have been isolated and people have been doing their research on only one of the phases since the whole conversion is very complicated and more difficult. For instance, the vectorization methods were developed only for getting straight lines, arcs, circles, etc., so that much information contained in the drawing has been lost after the vectorization. Even in some researches, different methods were developed and applied for recognizing the text and the outlines of orthographic projections of parts, respectively. In fact, the 3D reconstruction needs not only the vectors themselves but also their relationships and the information indicated by various symbols, from which the syntactic and semantic information can be extracted later on. This paper develops a holo-extraction method of information from paper drawings, the networks of single closed regions (SCRs), which can not only provide a unified base for recognizing both the annotations and the outlines of projections of parts, but also build the holo-relationships among SCRs so that it is convenient to extract lexical, syntactic, and semantic information in the subsequent phases for 3D reconstruction.

2. Review of current vectorization methods

Vectorization is a process that finds the vectors (such as straight lines, arcs, and circles) from the raster images. Much research work on this area has been done, and many vectorization methods and their software have been developed. Although the vectorization for the lexical phase is maturer than the technologies used for other two higher level phases, it is yet quite far from being perfect.

Current vectorization methods can be categorized into six types: Hough Transform (HT) based methods, thinning based methods, contour based methods, sparse pixel based method, mesh pattern based methods, and black pixel region based methods.

2.1. Hough Transform based method [2]

It visits each pixel of the image in (x, y) plane, peaks in its transform (m, c) space are detected, and each peak is used to form a straight line defined in the following equation:

$$Y = mx + c \quad (1)$$

where m is its slope and c is its intercept. Since the slopes and intercepts are sampled sparsely, they may not be as

precise as the original straight lines. Moreover, this method cannot generate polylines [3].

2.2. Thinning based method [4–7]

Thinning is a process that applies certain algorithms on the input raster image and outputs one-pixel wide skeletons of black pixel regions. Three types of algorithms have been developed for it, i.e. iterative boundary erosion [5], distance transform [6], and adequate skeleton [7]. Although their speeds and accuracies are different, they all have disadvantages: high time complexities, loss of shape information (e.g. line width), distortions at junctions, and false and spurious branches [3].

2.3. Contour based method [8–10]

This is to find the contour of the line object first and then calculate the middle points of the pair of points on two opposite parallel contours or edges. Although it is much faster than thinning based methods and the line width is also much easier to obtain, joining up the lines for a merging junction or a cross intersection is problematic, and it is inappropriate for use in vectorization of curved and multi-crossing lines [3].

2.4. Sparse pixel based method [11]

Its basic idea is to track the course of a one pixel wide ‘beam of light’, which turns orthogonally each time when it hits the edge of the area covered by the black pixels, and to record the midpoint of each run. With some improvement based on the orthogonal zig-zag, sparse pixel vectorization algorithm can record the medial axis points and the width of run lengths [3].

2.5. Mesh pattern based methods [12]

It is to divide the entire image using a certain mesh and to detect characteristic patterns by only checking the distribution of the black pixels on the border of each unit of the mesh. A control map for the image is then prepared using these patterns. Finally, the extraction of long straight-line segments is performed by analyzing the control map. This method not only needs a characteristic pattern database, but also requires much more processing time. Moreover, it is not suitable for detection of more complex line patterns, such as arcs and discontinuous (e.g. dashed or dash-dotted) lines [3].

2.6. Black pixel region based method [13–15]

It is to construct a semi-vector representation of a raster image first and then extract lines based on the semi-vector representation. The semi-vector representations can be a run graph (run graph based methods) [13], rectangular graph [14], trapezoidal graph [15]. A run is a sequence of black

pixels in either horizontal or vertical direction. A rectangle or trapezoid consists of a certain set of runs.

It can be seen, from the current vectorization methods, that, except black pixel region graph based methods, other methods are mainly focused on the speed and accuracy of generating vectors themselves, not on holo-extraction of information. Although black pixel region graph based methods build certain relationships between constructed runs, rectangles, or trapezoids, the regions are so small that it is not appropriate for curve line vectorization and is difficult to construct the relationships among vectors.

In fact, the understanding process for its subsequent 3D reconstruction is an iterative process for searching different level relationships and performing corresponding connections. For instance, linking certain related pixels can form a vector. Connecting a certain set of vectors can form primitives or characters of the text. Combining two projection lines and a dimension line containing two arrowheads with the values for normal dimension and tolerance can produce a dimension set, which can then be used with corresponding primitive for parametric modeling. Aggregating the equal-spaced parallel thin lines and the contour of their area can form a section, which can be then used with certain section symbols for solid modeling. Connecting certain primitives referring to corresponding symbols recognized can form certain features (e.g. bearing and threading), which can be used for feature modeling. Matching primitives in different views according to orthogonal projective relationships can produce 3D model. If the primitives extracted are accurate, their projective relationships can be determined by analyzing the coordinates of end points of these vectors. But the primitives extracted and their projective relationships are inaccurate in paper drawings, so that this method cannot be applied. It needs an expert system that simulates the experienced human designer's thinking mode to transform the inaccurate outlines of part's orthographic projections into 3D objects' image, so that their relationships become more important and crucial. As mentioned in the first section of this paper, vectorization process in the first phase should not loss the information in a drawing or the information needed for 3D reconstruction, which are mainly different level relationships and have been contained in raster image. Accordingly, a holo-extraction of information from the raster image is needed. In order to facilitate the iterative process for searching different level relationships and performing corresponding connections, the method needs a compact representation of raster image as a bridge from raster image to understanding, which should satisfy the following requirements:

1. it can distinguish different types of linking points for different relationships of the related elements (e.g. tangential point, intersecting point, and merging junction) to provide necessary information for extracting lexical, syntactic, and semantic information in the subsequent phases;

2. it can provide a unified base for further recognizing both the outlines of orthogonal projections of parts and the annotations, and facilitate their separation;
3. it can recognize line patterns and arrowheads, and facilitate the aggregation of related elements to form certain syntactic information, such as dimension sets;
4. it can provide holo-graphs of all the elements as a base for the subsequent 3D reconstruction;
5. it can facilitate recognizing vectors quickly and precisely, including straight lines, arcs, circles.

The networks of SCRs reported in this paper is developed for these purposes.

3. Construction of the networks of SCRs

The elements and workflow of constructing the networks of SCRs are shown in Fig. 1, and illustrated in more detail as follows.

3.1. Generating adjacency graphs of runs

Let visiting sequence be from top to bottom for a raster image and from left to right along horizontal direction for its each row. When visiting along a row, the first black pixel converted from white pixel is the starting point of a run and the last black pixel, if next pixel is white, is the end point of the run. The value of its end coordinates minus starting coordinate in horizontal direction is its run length, the unit of which is pixel. If the difference of two runs' vertical coordinates is 1 and the value of their minimal end coordinates minus maximal starting coordinates is larger than or equal to 1, the two runs are adjacent. If A and B are adjacent and A's vertical coordinate is larger than that of B, A is called predecessor of B, and B is called successor of A. There are seven types of runs according to its adjacency relationships with other runs as shown in Fig. 2.

1. Singular run—has no predecessor and no successor.
2. Beginning run—has no predecessor and only one successor.

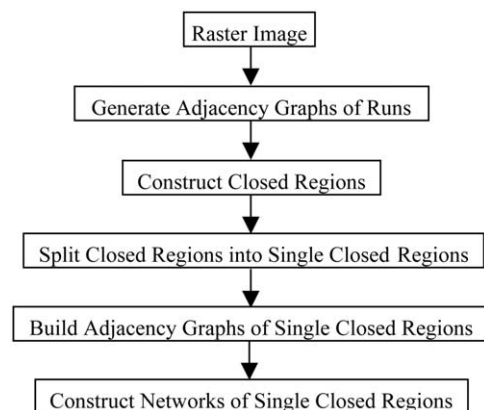


Fig. 1. Elements and workflow of constructing the networks of SCRs.

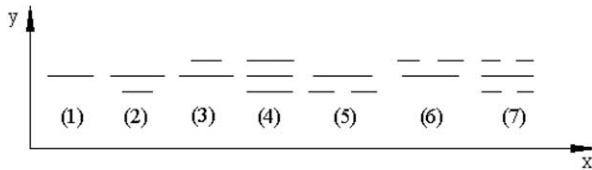


Fig. 2. Seven types of runs.

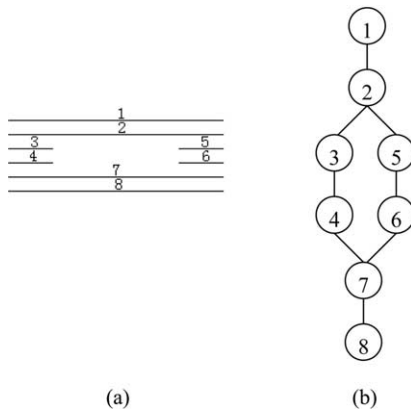


Fig. 3. The adjacency graph of runs for a run graph.

3. End run—has only one predecessor and no successor.
4. Regular run—has only one predecessor and only one successor.
5. Branching run—has one predecessor at most and more than one successors.
6. Merging run—has more than one predecessors and one successor at most.
7. Cross run—has more than one predecessors and successors, respectively.

With recording the adjacency relationships of runs while visiting in the assumed way, the adjacency graphs of runs can be made using a node to represent a run and a line between two nodes to represent an adjacency relationship between the two runs. Different properties of runs (including their starting coordinates, end coordinates, lengths, and types) are stored in the node data file. Fig. 3(b) shows an adjacency graph of runs generated from the run graph in Fig. 3(a). It is obvious that the adjacency graph of runs represents not only runs themselves but also their adjacency relationships.

3.2. Constructing single closed regions (SCRs)

Based on the adjacency graphs of runs constructed, higher level representation of the raster image can be obtained by aggregating related runs into closed regions. These related runs must satisfy the following requirements.

1. They are adjacent to each other.
2. The beginning run is not a branching run or a cross run, and the end run is not merging run or a cross run.

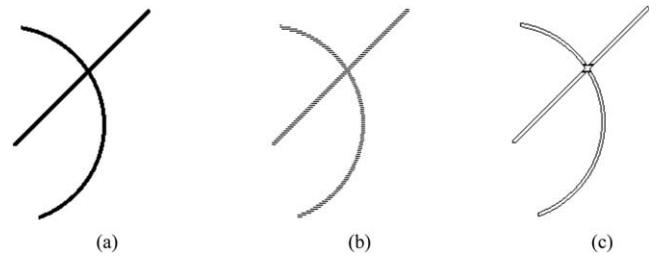


Fig. 4. The closed region graph.

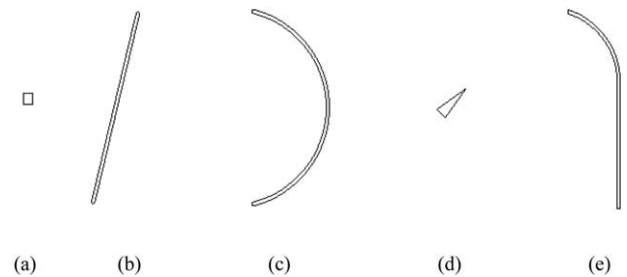


Fig. 5. Closed regions.

3. The difference between their lengths is less than their shorter run length.

From the first run, the related runs can be searched via the adjacency graphs of runs, and their starting points and end points are then recorded to construct closed regions. Fig. 4(c) shows a closed region graph constructed from the run graph in Fig. 4(b) for the outlines in Fig. 4(a).

According to the method, the possible results obtained may be the closed region for a point, a straight line, an arc, an arrowhead, or a combined line, as shown in Fig. 5(a)–(e), respectively, which are identified using Fuzzy Logic method [16] and will be introduced in detail in another paper. A point may be a independent point, branching point, merging point, cross point, and tangential point, as shown in Figs. 5(a), 6(a)–(d), respectively. A straight line can be a sloping, vertical, or horizontal straight line.

Although the closed region for a point, a straight line, an arc, or an arrowhead is a single closed region (SCR), the closed region for a combined line is not a SCR since a combined line may consist of straight lines and/or arcs, which are linked or tangential to each other. In order to construct the adjacency graphs of SCRs, the closed region for a combined line should be decomposed into several SCRs, each of which represents a straight line or an arc. The method for decomposition is to find their optimal split points with least number of segments, each of which can be fitted to a straight line or an arc with least error. This is an optimization problem, which can be solved by using Genetic Algorithm [17] and will be introduced in detail in another paper. For example, with this method, a combined line in Fig. 7(a) can be split up into two straight lines and

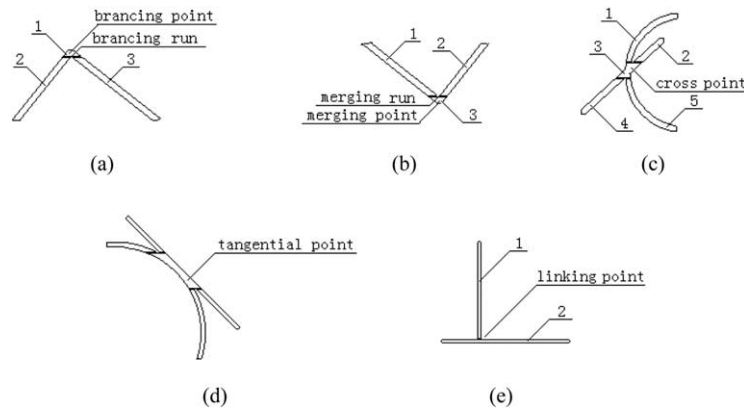


Fig. 6. Single closed regions for points.

two arcs, as shown in Fig. 7(b) where the split points are indicated by small circles.

After its split points are found, the closed region for a combined line can be decomposed into several SCRs. Thus, the original run graph can be transformed into the graph consisting of only SCRs. Each SCR represents a higher level element of primitives or annotations, so that the relationships among vectors can be extracted more easily.

3.3. Building adjacency graphs of SCRs

Based on the adjacency graphs of runs and the single closed regions constructed, the adjacency graphs of SCRs for a raster image can be built using the following rules:

1. use a node to represent a single closed region;
2. add a black node for a split point of combined lines, or a linking point where there appears a branching run, a merging run, or a run between which and its predecessor the length difference is larger than their shorter one as indicated in Fig. 6(e);
3. apply a line between a node and a black node to represent an adjacency relationship between a SCR and a split point or linking point.

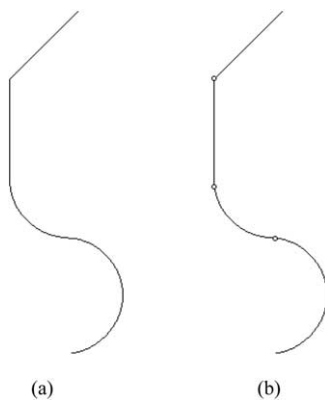


Fig. 7. The split of a combined line.

Fig. 8(a)–(d) shows the adjacency graphs of SCRs built for the single closed region graphs in Fig. 6(a)–(c), and 6(e), respectively.

The single closed regions constructed can also be categorized into seven types according to its adjacency relationships with other single closed regions as shown in Fig. 9.

1. Singular SCR—has no predecessor and no successor.
2. Beginning SCR—has no predecessor and only one successor.
3. End SCR—has only one predecessor and no successor.
4. Regular SCR—has only one predecessor and only one successor.

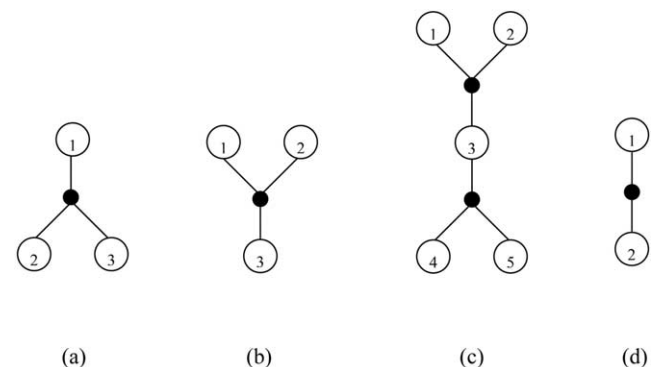


Fig. 8. The adjacency graphs of SCRs for different connections.

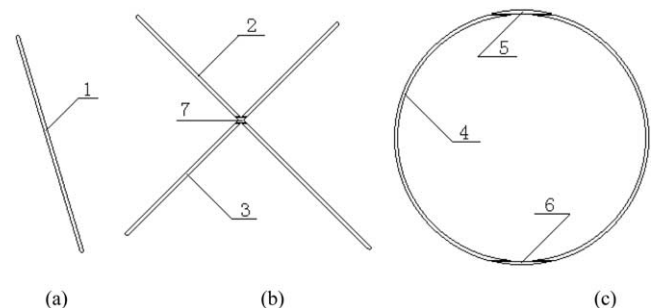


Fig. 9. Seven types of SCRs.

5. Branching SCR—has one predecessor at most and more than one successors.
6. Merging SCR—has more than one predecessors and one successor at most.
7. Cross SCR—has more than one predecessors and successors respectively.

Different properties of SCRs are stored in the node data file and include their types (one of seven types above), the categories of segment represented by the SCR (i.e. point, arrowhead, straight line, or arc) and characteristics. For a point, its characteristics are the coordinates of its center. For an arrowhead, its characteristics cover the coordinates of its top point, its angle to horizontal (x) axis, length, and width. For a straight line, its characteristics are its line width, starting coordinates, end coordinates, length, slope and intercept. For an arc, its characters cover its line width, the coordinates of its curvature center, its curvature radius, starting angle to the x -axis (let the direction of arc be counter-clockwise), and end angle to the x -axis. The coordinates of each linking point and the labels of SCRs adjacent to the linking point are stored in the black node data file as the properties of linking point. It is obvious that the adjacency graphs of SCRs represent all the single closed regions, their linking points and their adjacency relationships.

3.4. Constructing the networks of SCRs

In the adjacency graphs of SCRs, there are some black nodes, which represent the points linking the SCRs of the elements of primitives or annotations, not the SCRs of the elements (i.e. straight lines or arcs) themselves. It is the linking points that contain useful information about the relationships among elements, such as their tangential relationship or intersection relationship at a certain angle. These linking points should be further analyzed to extract different types of relationships among elements for constructing the networks of SCRs.

Whether the elements adjacent to a linking point are intersected or tangential to each other can be determined by calculating the angles among them. Accordingly, the networks of SCRs can be further obtained by using the following procedure.

1. Delete each black node and all the lines between the black node and its related nodes; and, instead, connect, with a line, each pair of all the nodes having adjacency relationship with the black node.
2. Eliminate the following nodes and the lines between the node and its related nodes:
 - the node representing a branching SCR if the extensions of its successors pass the center point of the branching SCR;
 - the node representing a merging SCR if the extensions of its predecessors pass the center point of the merging SCR.
3. Record the properties of relationship for each pair of the related nodes
 - the node representing a cross SCR if the extensions of its predecessors and successors pass the center point of the cross SCR. For this case, every two of the nodes for its all predecessors and successors will be connected with a line if there is no line between them.
4. Find out the node representing a SCR of the straight line the slope of which is very close or equal to zero (i.e. horizontal straight line), and check whether the intersecting point between it and one of its predecessors is very close to or at the intersecting point between it and one of its successors. If yes, apply a line to connect the related predecessor and successor.

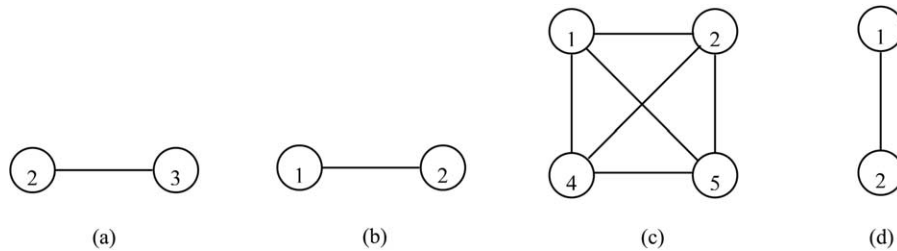


Fig. 10. The networks of SCRs for different connections.

Fig. 10(a)–(d) show the networks of SCRs generated for the adjacency graphs of SCRs in Fig. 8(a)–(d), respectively.

For a view, like the view shown in Fig. 11(a), its SCR graph, adjacency graphs of SCRs, and networks of SCRs can be in turn obtained using this method, as shown in Fig. 11(b)–(d), respectively. In its networks of SCRs, a node represents the properties of a SCR for a related element (a straight line or an arc) and a line between two nodes represents the properties of relationship between the two elements. Therefore, the networks of SCRs satisfy the first requirement of a compact representation of raster image as a bridge from raster image to understanding mentioned in the second section of this paper. That is, the networks of SCRs distinguish different types of linking points for different relationships of the related elements to provide necessary information for extracting lexical, syntactic, and semantic information in the subsequent phases.

4. A bridge from raster image to understanding and 3D reconstruction

The networks of SCRs are holo-extraction of information from a raster image, and can be used for subsequent differ-

ent level understanding and 3D reconstruction. The main elements and workflow of further understanding and 3D reconstruction is shown in Fig. 12, and illustrated in more detail as follows.

4.1. Separation between the annotations and the outlines of projections of parts

In engineering drawing, there are the outlines of orthogonal projections of parts and the annotations. They should be separated first so that the former can be used for reconstructing 3D model of a part and the later can be recognized to provide necessary information for parameters, features, etc. The former consists of straight lines, arcs, and/or points. The later includes non-outlines (e.g. center lines), symbols (e.g. machining and surface texture symbols) and text. The character of text may be Arabic numbers, English characters, Chinese characters, or other language's characters. They all consist of straight lines, arcs, and/or points as well. These straight lines or arcs may be split up into segments (i.e. shorter straight lines or arcs), by branching, merging, cross, tangential, or split points, which have been represented by their corresponding SCRs. Fig. 13(b) shows three graphs of SCRs generated for an English character,

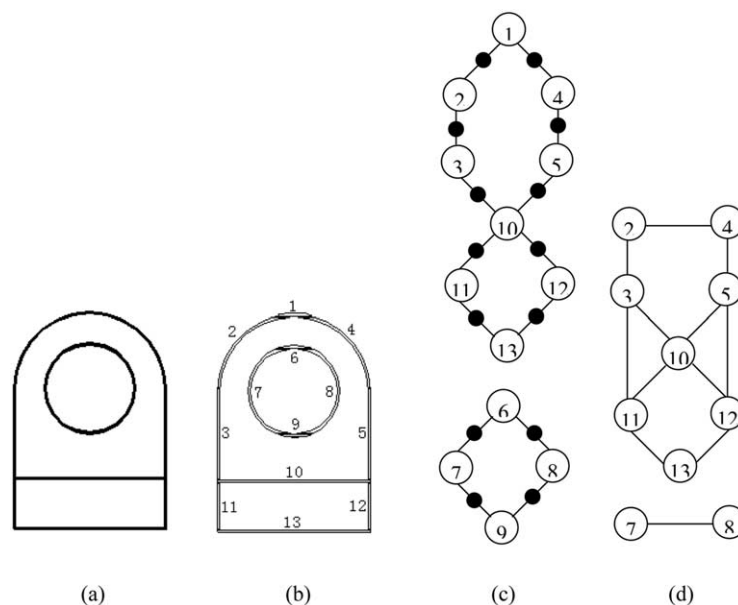


Fig. 11. The SCR graph, adjacency graphs of SCRs, and networks of SCRs for a view.

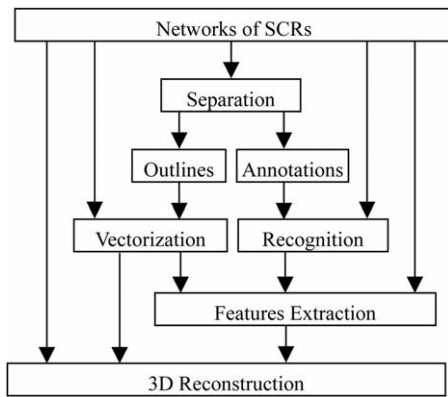


Fig. 12. The main elements and workflow of further understanding for 3D reconstruction.

Arabic number, and Chinese character in Fig. 13(a) respectively. The networks of SCRs for each character or symbol can also be constructed using the same method for the outlines of projections of parts introduced above. The networks of SCRs thus provide a unified base for further recognizing the primitives and the annotations.

Normally, the extraction of text is implemented based on the areas of the region containing a set of connected segments and the densities of black pixels in the region since the area for the character of text is smaller and has higher density of black pixels than the primitives do. If the character string is not arranged horizontally, it should be identified and rotated to horizontal position, and then recognized using further methods. The extraction of outlines of projections of parts is done according to their line widths since their line width is thicker than others. The separation between non-outlines and symbols is carried out using pattern recognition. Since one network of SCRs represents a set of connected segments and the networks of SCRs contain all the necessary information, including line widths and those necessary for calculating their areas, densities, sloping angles to the x -axis, and pattern, these extractions or separations can be easily implemented and the further recognizing methods can then be applied, all based on their networks of SCRs. Therefore, the networks of SCRs meet the second requirement of a compact representation of raster image as a bridge from raster image to understanding mentioned in the second section of this paper. That is, the networks of SCRs provide a unified base for recognizing both the outlines of orthogonal projections of parts and the annotations, and facilitate their separation.

4.2. Vectorization

4.2.1. Recognition of horizontal thick lines and arrowheads

Since the line widths can be calculated and have been stored in the node data file, thick lines or thin lines can be detected by comparing the line widths of all the nodes in the networks of SCRs for a drawing. The arrowhead can also be extracted according to the change of its SCR's width. But, it

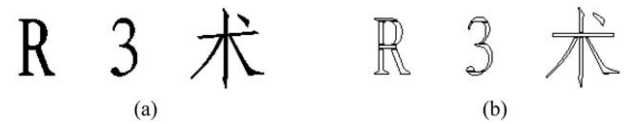


Fig. 13. The SCR graphs for characters of the text.

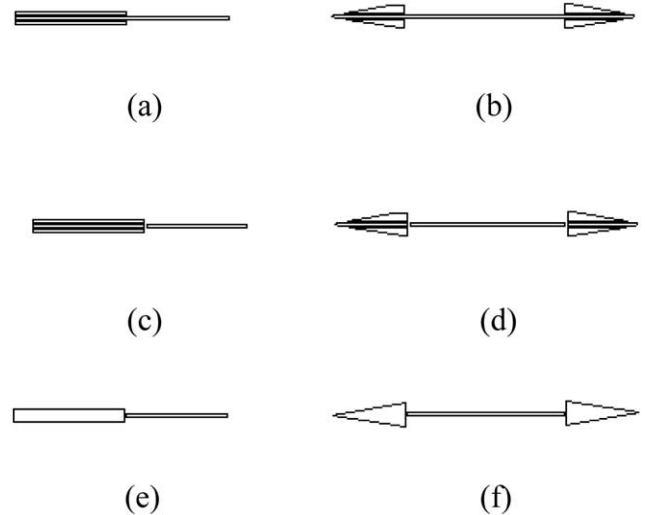


Fig. 14. Identification of horizontal thick lines and arrowheads.

is difficult to identify a horizontal thick line or horizontal arrowhead connected with a horizontal thin line, since a SCR for a thin line will contain the part of SCR for a thick line or an arrowhead connected with the thin line as shown in Fig. 14(a) and (b), respectively.

In these cases, the SCR for a thin line will be divided into several parts according to its adjacency relationships with other related SCRs, as shown in Fig. 14(c) and (d), respectively. That part of SCR for a thin line, which has adjacency relationships with other related SCRs, will then be aggregated into a horizontal thick line or a horizontal arrowhead, as shown in Fig. 14(e) and (f), respectively. Since all the information needed for the operations is contained in their networks of SCRs, the extraction of horizontal thick lines or arrowheads connected with horizontal thin lines can be implemented conveniently based on their networks of SCRs. Therefore, the networks of SCRs also satisfy the third requirement of a compact representation of raster image as a bridge from raster image to understanding mentioned in the second section of this paper. That is, the networks of SCRs facilitate recognizing line patterns and arrowheads.

4.2.2. Recognition of complete arcs and circles

In 2D drawing, there may be many arcs or circles, which are divided into several segments by other straight lines, arcs, or circles. These segments have their own nodes in the networks of SCRs. A circle may also be divided into two SCRs for two halves of a circle, which have been represented in the network of SCRs. In order to extract the

vectors (i.e. the complete arcs or circles), the SCRs for these segments should be combined together. The method is to search for the first SCR representing the arc in the networks of SCRs as a seed SCR; then to find the SCRs, which are adjacent to the seed SCR and belong to the same arc, via the networks of SCRs; and finally to combine the related SCRs to get a complete arc or circle. If there is no more related SCR for arc, the second seed SCR for arc will be found in the networks, and the above process will be repeated until no seed SCR for arc can be found. Fig. 11(a) can be used as an example for recognizing complete arc and circle. As shown in Fig. 11(b), the arcs represented by SCRs 2 and 4 belong to the same arc, and are connected by point represented by SCR 1. The arcs represented by SCR 7 and 8 belong to the same circle, and connected by linking points represented by SCR 6 and 9. While searching for the SCR for arc in the networks of SCRs for the view, the SCR for arc 2 will be found as a seed SCR first, and then the SCR for arc 4 will be found adjacent to it and belonging to the same arc via the network of SCRs and will be combined with the seed SCR to obtain an extended arc. Since there is no other related arc, the complete arc is extracted. Then the SCR for arc 7 will be found in the networks of SCRs as the second seed arc, and the same process will be repeated to connect the SCR for arc 8 with it into a complete circle. Since there is no more SCR for arc left, the process will be stopped and the another process for recognizing complete straight lines will be started.

4.2.3. Recognition of complete straight lines

The complete straight lines can also be obtained using the same method. Fig. 11(b) can be still used as an example of recognizing complete straight lines. The straight lines represented by SCRs 3 and 11 belong to the same straight line, and the straight line represented by SCRs 5 and 12 belong to another straight line. While searching for the SCR for a straight line in the networks of SCRs for this view, SCR 3 will be found as a seed SCR first, then SCR 11 will be found adjacent to it and belonging to the same line via the network of SCRs since the type of their relationship is stored in line data file, and SCRs 3 and 11 will be combined together as an extended straight line. Since there is no SCR for straight line found belonging to the same straight line, the complete straight line is extracted. Then, SCR 5 will be found as the second seed SCR, from which SCR 12 will be found adjacent to it and belonging to the same straight line via its network of SCRs and combined with the seed line to form an extended straight line. Since there is no other SCR for straight line, which are collinear, the vectorization process will be stopped.

4.2.4. Software prototype

Based on the method introduced above, a software prototype has been made in C++. Many paper drawings have been vectorized successfully by using this software prototype. As an example for it, Fig. 15(c) shows the results of

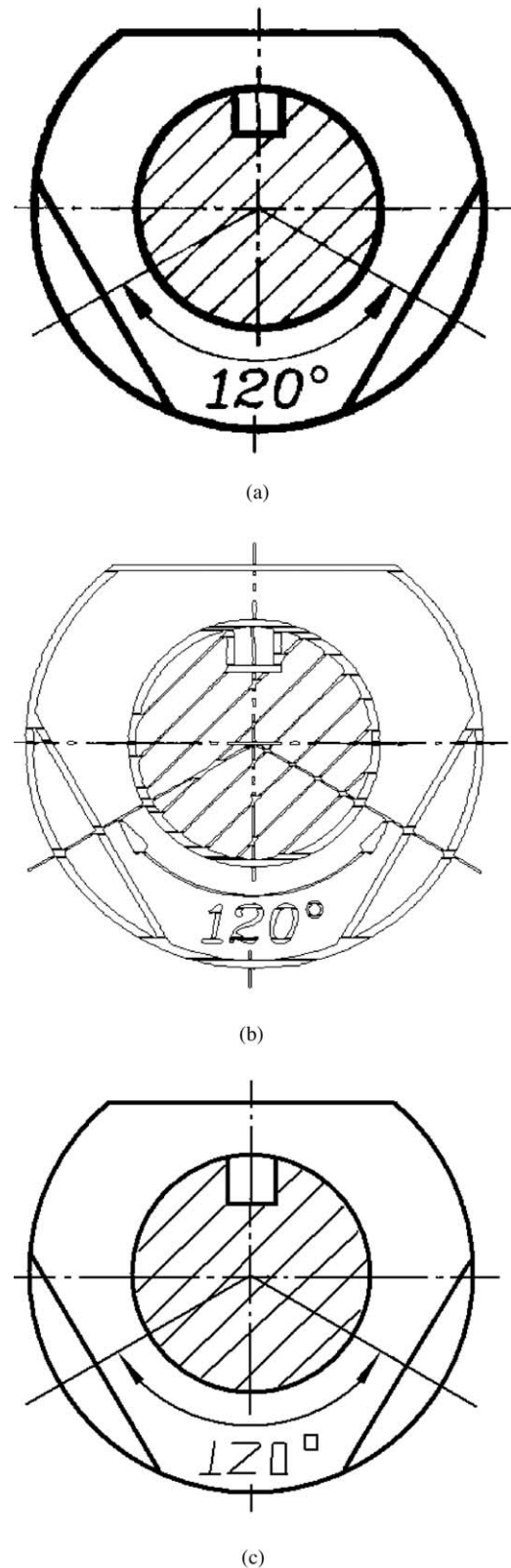


Fig. 15. The results of vectorization using the software prototype based on the networks of SCRs.

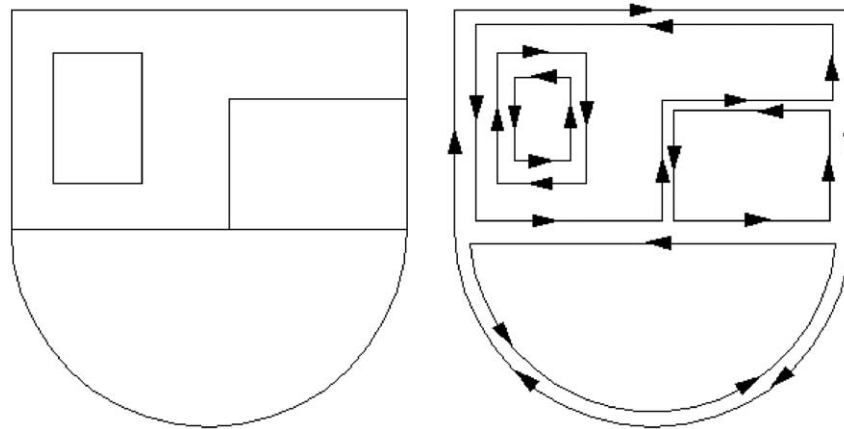


Fig. 16. Search for circuits to create faces of 3D models.

running the software prototype for a view of a drawing shown in Fig. 15(a). Fig. 15(b) shows its SCR graph for illustration. It is obvious that the vectorization method based on the networks of SCRs is feasible and that the networks of SCRs meet the fourth requirement of a compact representation of raster image as a bridge from raster image to understanding mentioned in the second section of this paper. That is, the networks of SCRs facilitate recognizing vectors quickly and precisely.

4.3. 3D reconstruction

For 3D reconstruction, two families of methods have been developed. The first family is known as the ‘fleshing out projections’ concept [18,19]. Another family is volume-oriented instead of privileging wire-frames [20].

The former is based on the reconstruction of a wire-frame model by matching vertices and edges; the real faces of the object are then found by propagating constraints on this wire-frame. Its idea is to search for the circuits on each view, by sorting the edges in trigonometric order with respect to the normal to the surface, at each vertex, and by traversing the resulting graph of oriented edges; to create a face for each finite circuit and a hole for each infinite circuit; and to make up a real solid with a group of faces without violating the Moebius rule [21,22]. The Moebius rule says that when a real edge is on the border of two real faces, the traversals along the circuits of these two faces follow the edge in opposite directions. Fig. 16(b) [21] shows the circuits for the view in Fig. 16(a). Therefore, the circuits are crucial for this method of 3D reconstruction, and, sometimes, a vector-based representation yielded is not very useful [21].

The later is to decompose each view into several predefined types of sub-views [23] or exterior boundary and interior boundary [24]; to generate their 3D subparts; and then to combined them to build the complete object [20]. Each view and its predefined types of sub-views or exterior boundary and interior boundaries are all circuits. Therefore, the

circuits are also crucial for this method, and, sometimes, a vector-based representation yielded is not very useful as well.

Since a circuit consists of elements, which are represented by SCRs, and the networks of SCRs provide holorelationships of all the elements, the circuits can be searched easily via the networks of SCRs. Moreover, 3D reconstruction also needs various features, which are also extracted from the networks of SCRs as indicated in Fig. 12. It is obvious that the networks of SCRs are more useful and can be used as a base for the subsequent 3D reconstruction. Therefore, the networks of SCRs meet the fifth requirement of a compact representation of raster image as a bridge from raster image to understanding mentioned in the second section of this paper.

Based on the networks of SCRs, a 3D reconstruction method have been developed. As an example for it, Fig. 17 shows the results of running the software prototype for a drawing shown in Fig. 18, where each SCR is represented by its corresponding symbol (e.g. F_4 is the symbol for the bottom SCR in front view). Fig. 19 shows its networks of SCRs generated according to the drawing, where V

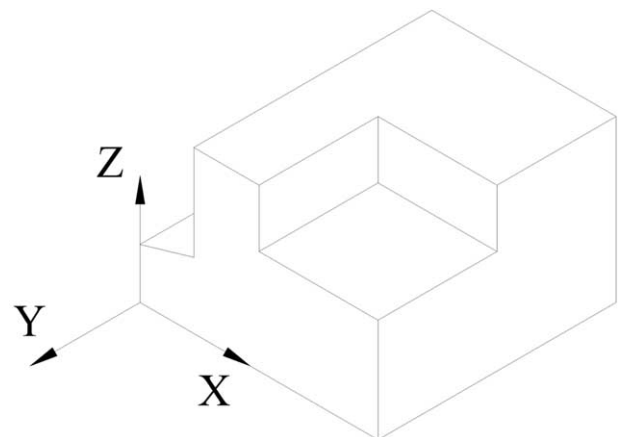


Fig. 17. An example of 3D reconstruction.

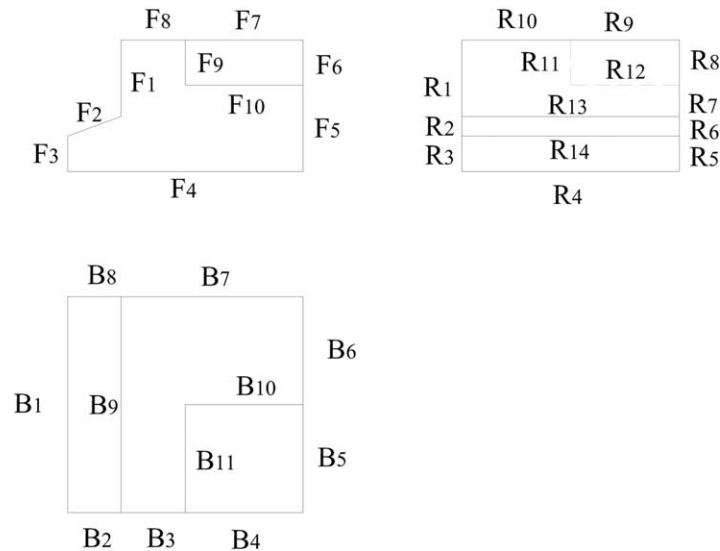


Fig. 18. A 2D drawing of an object.

indicates a linking point by two SCRs and C by more than two SCRs. Fig. 20 shows circuits obtained via the networks of SCRs, which are indicated by closed chain lines and/or symbols. For example, L_{F2} is the symbol for the circuit represented by a closed chain line in front view, and L_{F1} is the symbol for the circuit covering L_{F2} and L_{F3} , i.e. the outside outline of the front view.

5. Conclusions

The research on the conversion from 2D paper drawing to 3D computer feature model has been stuck in low-level coding. One of the reasons for it is that the three phases of understanding process has been isolated and people have been doing their research on only one of the phases since the whole conversion is very complicated and more difficult.

For instance, the vectorization methods for the first phase were developed only for getting straight lines, arcs, circles, etc. so that much information contained in the drawing has been lost after the vectorization. In fact, the understanding processes for its subsequent 3D reconstruction is an iterative process for searching different level relationships and performing corresponding connections. In order to facilitate the iterative processes, a holo-extraction of information from raster image is needed, and a compact representation of raster image should be generated as a bridge from raster image to understanding, which should satisfy five requirements introduced in the second section of this paper. This paper develops a holo-extraction method of information from paper drawings, i.e. the networks of single closed regions (SCRs), for these purposes; describes the algorithms or methods to generate the networks of SCRs for a given view; analyzes their functions which indeed meet the five

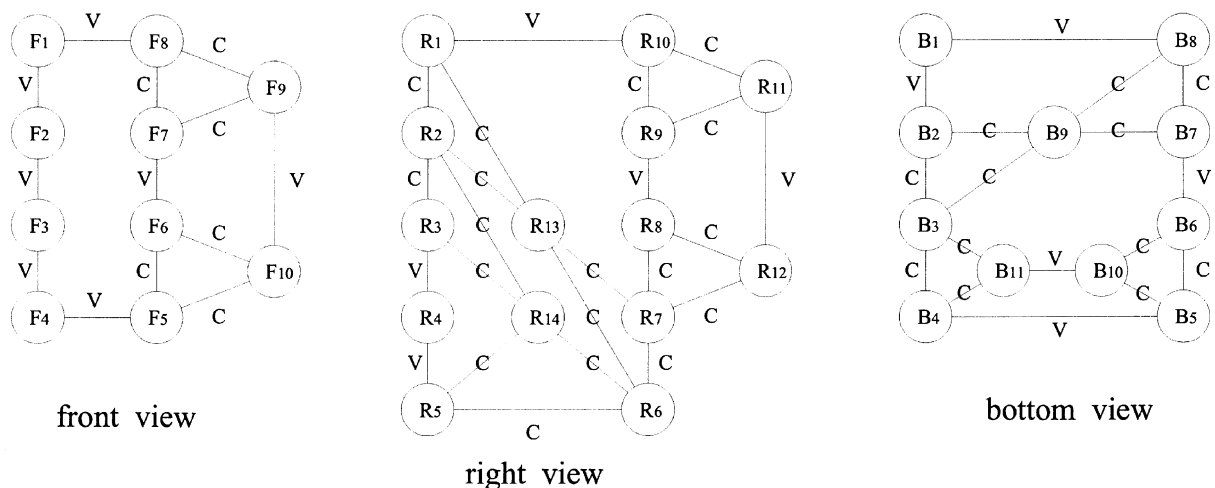


Fig. 19. The networks of SCRs generated from a 2D drawing.

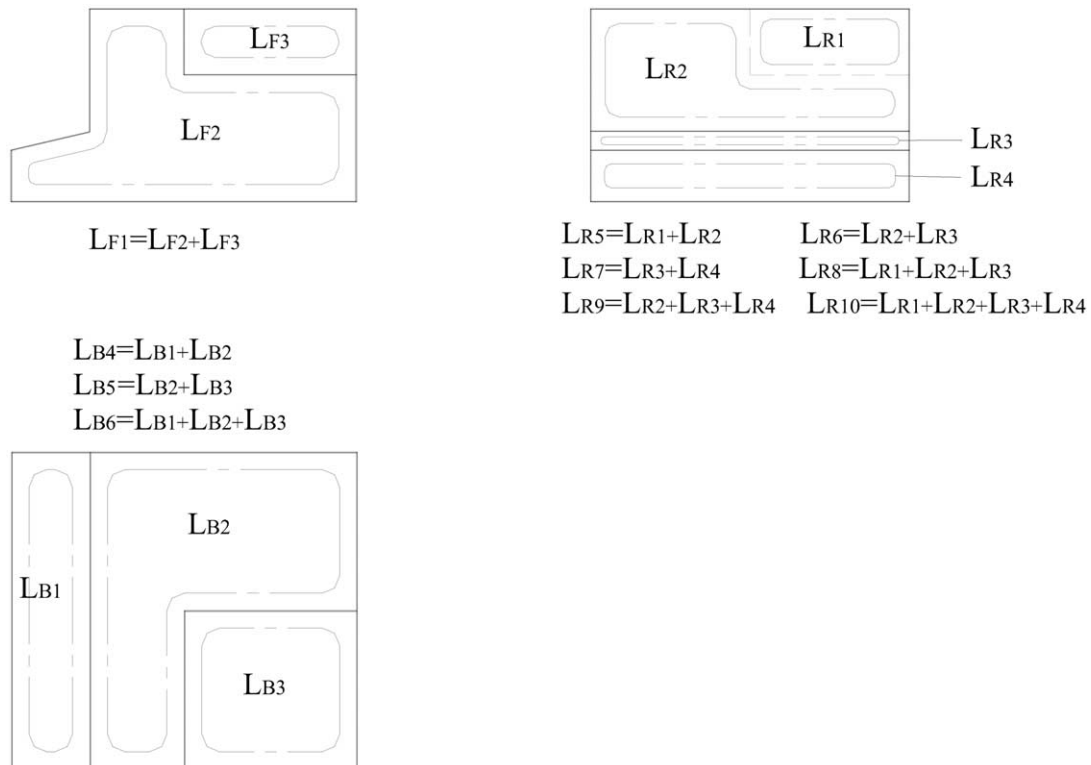


Fig. 20. The circuits obtained via the networks of SCRs.

requirements. A software prototype for it has also been developed in C++, and verified by its applications. This holo-extraction method of information from paper drawings and its software prototype can be used as a base for further research on subsequent understanding and 3D reconstruction. Based on this software prototype, a more sophisticated system with more functions for 3D reconstruction is being developed favorably.

Acknowledgements

The reported research is supported by University Research Committee Grants (CRCG) in the University of Hong Kong. The financial contribution is gratefully acknowledged.

References

- [1] Dori D, Tombre K. From engineering drawings to 3D CAD models: are we ready now? *Computer-Aided Design* 1995;27(4):243–54.
- [2] Hough PVC. A method and means for recognizing complex patterns. USA Patent 3,096,654, 1962.
- [3] Liu W, Dori D. From raster to vectors: extracting visual information from line drawings. *Pattern Analysis and Applications* 1999;2(2):10–21.
- [4] Lam L, Lee SW, Suen CY. Thinning methodologies—A comprehensive survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 1992;14(9):869–87.
- [5] Naccache NJ, Shinghal R. SPTA: a proposed algorithm for thinning binary patterns. *IEEE Transactions on System, Man, and Cybernetics* 1984;14:409–18.
- [6] Peleg S, Rosenfeld A. A min-max medial axis transformation. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 1981;3:208–10.
- [7] Davies ER, Plummer APN. Thinning algorithms: a critique and a new methodology. *Pattern Recognition* 1981;14:53–53.
- [8] Han CC, Fan KC. Skeleton generation of engineering drawings via contour matching. *Pattern Recognition* 1994;27(2):261–75.
- [9] Jimenez J, Navalon JL. Some experiments in image vectorization. *IBM Journal of research and Development* 1982;26:724–34.
- [10] Shapiro B, Pisa J, Sklansky J. Skeleton generation from x-y boundary sequences. *Computer Graphics and Image Processing* 1981;15:136–53.
- [11] Liu W, Dori D. Sparse pixel tracking: a first vectorization algorithm applied to engineering drawings. *Proceedings of the 13th International Conference on Pattern Recognition, Vienna, Austria, 1996. Vol. III (Robotics and Applications)*, p. 808–11.
- [12] Lin X, Shimotsuji S, Minoh M, Sakai T. Efficient diagram understanding with characteristic pattern detection. *Computer Vision, Graphics and Images Processing* 1985;30:84–106.
- [13] Monagan G, Roosli M. Appropriate base representation using a run graph. *Proceedings of the Second International Conference on Document Analysis and Recognition, Tsukuba, Japan, 1993. p. 623–6.*
- [14] Bley H. Segmentation and preprocessing of electrical schematics using picture graphs. *Computer Vision, Graphics, and Image Processing* 1984;28:271–88.
- [15] Zenzo SD, Cinque L, Levialdi S. Run-based algorithms for binary image analysis and processing. *IEEE Transaction on Pattern Analysis and Machine Intelligence* 1996;18(1):83–9.
- [16] Tsoukalas LH, Uhrig RE. *Fuzzy and neural approaches in engineering*. John Wiley & Sons Inc, 1997.
- [17] Gen M, Cheng R. *Genetic Algorithms and Engineering Design*. New York: John Wiley & Sons Inc, 1997.

- [18] Wesley MA, Markowsky G. Fleshing out projections. IBM Journal of Research and Development 1981;25(6):934–54.
- [19] Preiss K. Constructing the solid representation from engineering projections. Computers and Graphics 1984;8(4):381–9.
- [20] Yoshiura H, Fujimura K, Kumii L. Top-down construction of 3-D mechanical object shapes from engineering drawings. IEEE COMPUTER Magazine 1984;17(12):32–40.
- [21] Ah-Soon C., Tombre K. A step towards reconstruction of 3-D CAD models from engineering drawings. Proceedings of the Third International Conference on Document Analysis and Recognition, Montreal, Canada, 1995. p. 331–4.
- [22] Hoffmann CM, Hopcroft JEH. Geometric ambiguities in boundary representations. Computer-Aided Design 1987;19(3):141–7.
- [23] Chen Z, Perng DB. Automatic reconstruction of 3D solid objects from 2D orthographic views. Pattern Recognition 1988;21(5):439–49.
- [24] Shum SSP, Lau WS, Yuen MMF, Yu KM. Solid reconstruction from orthographic views using 2-stage extrusion. Computer-Aided Design. 2001;33:91–102.



Ke-Zhang Chen received his PhD degree from the University of Manchester (UMIST), UK in 1984. After he had worked at the University of Manchester (UMIST) (UK), East China University (China), MIT (USA), and the University of California at Davis (USA), he joined the Department of Mechanical Engineering in the University of Hong Kong in 1996. His current research interests include design theory and methodology, concurrent engineering, intelligent CAD, and design for heterogeneous components.



Zong-ying Ou received his BS degree from the Department of Mechanical Engineering, Dalian University of Technology, China, in 1956. He has then been working in the university, and is now a Professor, Head of Mechanical Design Division and Director of CAD/CG lab in the Faculty. From 1981 to 1983, he worked as a visiting scholar in University of Wisconsin-Madison, USA. His research interests include intelligent CAD/CAM, solid modeling, geometric reasoning, image processing and understanding, computer vision and 3D reconstruction.



Xi-Wen Zhang received his BS degree from the Department of Chemical Engineering, Fushun Institute of Petrol, China, in 1995, and PhD degree from the Department of Mechanical Engineering, Dalian University of Technology, China, in 2000. He is now a Postdoctor at State Key Laboratory of Novel Software Technology, the Department of Computer Science and Technology, Nanjing University, China. His research interests include computer vision, artificial intelligence, and understanding of scanned engineering drawings.



Xin-An Feng received his BS degree from the Department of Mechanical Engineering, Dalian University of Technology, China, in 1956. He has then been working in the university, and is now a Professor, Deputy director of CIMS research center, standing council member of China Association for Mechatronics Technology and Applications, and council member of China Institution for Production Engineering. From 1980 to 1982, he worked as a visiting scholar in the University of Manchester (UMIST), UK. His current research interests include CAD/CAM, intelligent CAD, concurrent engineering, and CIMS.

CIMS Papers' World

Here scientific papers on Computer Integrated Manufacturing System(CIMS) and other fields!for you FREELY!

1.<http://www.cimspaper.com>
2.<http://xcchen.yeah.net>

3.<http://cimspaper.icpcn.com>
4.<http://cimspaper.51.net>