Automatic Tool Selection Algorithm for Sheet Metal Fabication

*Cho Kyung-ho**

판금작업을 위한 공구 자동선정 알고리즘

趙 慶 鎬*

Introduction

Recently, most industrial products tend to have a short life cycle due to the need for quick model change. Therefore, a sheet metal fabrication method using NCT numerically controlled turret punch press draws more attractions than conventional punching methods using a fixed die. With NCT, it is possible to punch any complex shape with the combination of tools which have already been installed on the turret of the machine, while conventional methods require the preparation of a new punching die whenever the shape of the sheet part changes.

To use the NCT machine effectively, it is necessary to provide it with proper tool processing information. Here tool processing information is associated with the identification number of the punching tools together with their locations and orientations to punch out the boundary curves of a given sheet metal part. This information is derived from the unfolded sheet model through a process called tool processing or tool selection. In general, however, punching tool selection has been done either manually or semi-automatically by a skilled person and is obiously a time-consuming and error prone task. As a result, productivity depends mainly on the skill of human experts.

In this paper, a scheme to automate tool processing is proposed. It includes the automatic selection of the tools and the determination of their locations and orientations for a given sheet part.

Related Works

There have been a lot of studies on pattern recognition or curve matching to be applied in

* Dept.of Nuclear and Energy Engineering, College of Engineering

areas such as computer vision and character recognition. In these studies, criteria such as critical point, break point, or land mark were introduced for shape recognition problems (Ansari and Delpt, 1990; Ayache and Faugeras, 1986; Freeman, 1978; Kalvin, 1986; Hong and Wolfson, 1988). Some literature shows that the template concept or a filter has been used (Niemann, 1981; Schmidt, 1990). Since the methods mentioned above basically look for the points on the curves where the curvature changes rapidly, they may not work successfully if the change is not dramatic or the extraction of a desirable criterion is not possible due to occlusion and etc. (Wolfson, 1990).

Pattern recognition methods using moment invariants (Dudani, 1977; Reeves, 1988) or Fourier descriptors (Granlund, 1972; Wallace and Wintz, 1980; Lin and Chellapa, 1987; Gorman, 1988), which are calculated from the points on the shape boundaries have been also suggested. The method using moment invariants usually requires heavy computation. The Fourier descriptors are very useful to describe the whole boundary of a shape, but they are not suitable for describing only the portion of the boundary because of the inherent periodicity of the Fourier expansion (Gormann, 1988). A statistical approach has been also introduced to handle shape recognition or object classification problems (Dubois and Glanz, 1986). A useful concept of least-square-distance was introduced by Schwartz and Sharir (1987) to determine if two curves in 2 or 3 dimensional space match each other, and Wolfson (1990) showed good results by applying this algorithm to object assembly and object recognition.

However, none of the methods above seems to have been applied to the automatic punching tool selection in NCT operation. Kimura et al. (1987) once announced the development of a bending simulator for sheet metal products. But his work didn't consider the automation of punching tool selection at all. There are several commercial software packages (EUCLID-IS, 1987; BRAVO 3, 1988; CIMATRON, 1990; UNIGRAPHICS Ⅱ, 1991) with modules for sheet metal design and fabrication. These packages provide the capability by which the punching tools can be selected interactively but not automatically.

Automatic Tool Selection Algorithm

An automatic tool selection algorithm is developed to meet the requirement that the tool selection should be succeeded in if there exists a tool whose boundary matches that of the sheet part either partially or completely.

1. Data Structure

Figure 1 shows the data structure used to implement the tool selection algorithm effectively. All the curves composing a boundary are represented by NURB (non-uniform rational B-spline (Tiller, 1983) and each curve has a flag indicating the curve types; line, circular arc, and free curve. Since the tool selection algorithm is implemented on the assumption that all the boundaries of tools and sheet parts are assumed to be defined by the minimum number of curves, the adjacent straight lines or circular arcs are merged into one in advance by the system if they lie on the same straight line or circular arc respectively. Then the curves are stored in a doubly-linked-list to

```
typedef       struct curve              CURVE;
typedef       struct control            CONTROL;
typedef       struct point              POINT;
typedef       struct cvs                CVS;
typedef       struct sheet              SHEET;
typedef       struct tool               TOOL;
typedef       struct shape-index-set    SIS;
struct        control
{ double            hx,hy,hz,h;  };
struct        point
{ double            x,y,z;  };
struct        curve
{ int               Order;          * order of NURB */
  int               NumCon;         /* number of control points */
  double            Knot;           /* knot vector */
  CONTORL           *Control;       /* control point list */
  int               *Type;          /* type of curve */
} :
struct        cvs
{ CURVE             *cv;            /* curve equation */
  CVS               *pre, *next;    /* doubly-linked-list */
  float             length;         /* curve length */
  short             punch-flag;     /*punching flag */
};
struct        sheet
{ short             id;             /* id no. of sheet */
  CVS               *cvs;           /* pointer to doubly-linked-list */
  SHEET             *next;          /* pointer to next sheet */
} :
struct        tool
{ short             id;             /* id no. of tool */
  short             index;          /* indexing flag */
  float             area;           /* area of tool */
  POINT             centroid;       /* centroid of tool */
  CURVE             **cv;           /* array of tool curves */
  SIS               *template       /* SIS list of tool */
};
struct        shape-index-set
{ short             thetal, theta2; /* angle index */
  short             length;         /* curve-length index */
  short             rho;            /* curvature index */
};
struct match-inform
{ short             match-gain;     /* match-count or length */
  CURVE             **scv, **tcv;   /* matching curve list of sheet & tool */
  TOOL              *tool;          /* candidate tool */
};
```

Fig. 1. Data structure for automatic tool selection.

compose the boundary of the sheet part and each curve has a punch-flag indicating whether the corresponding curve has been already punched out or not. With these flags, the data structure is updated as the punching operation proceeds.

The curve direction is set to form the counter-clockwise contour for the outer boundary (peripheral loop) and clockwise for the inner ones (hole loops). The direction of curves of the tool boundary is also set to form the clockwise contour and the curves are stored in an array. By setting the direction of the curves in this way, the tools to punch out the inner holes and outer boundary can be searched by the same procedure as illustrated in figure 2.

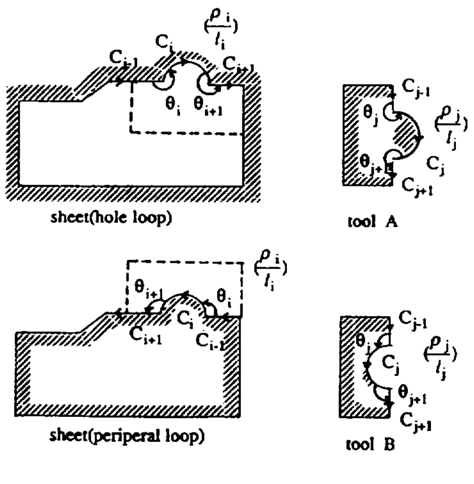


Fig. 2. Curve direction and shape indices in sheet and tool.

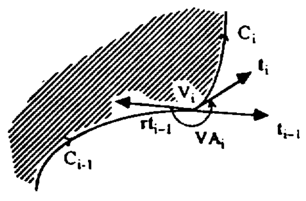


Fig. 3. Definition of vertex angle V_i.

2. Shape Index Set

In order to facilitate the search for the tool which matches the boundary of a given sheet part, every boundary curve C_i is represented by a set named *shape-index-set*, SIS(i), as follows.

$$SIS(i) : (\theta_i, \rho_i, \ell_i, \theta_{i+1})$$

Here, the angle-index, θ_i, is the integer value of the multiplication of the vertex angle and a constant M_θ. The vertex angle illustrated in figure 3 is defined as follows. Let $\mathbf{t}_{i-1}$ and $\mathbf{t}_i$ be the tangent vectors of the curve C_{i-1} and C_i respectively at the vertex V_i, and the vector $\mathbf{rt}_{i-1}$ is the reverse of $\mathbf{t}_{i-1}$ as shown in figure 3. Then the vertex angle at V_i is the angle between $\mathbf{rt}_{i-1}$ and t_i measured from $\mathbf{rt}_{i-1}$ in the counter clockwise direction. The length-index, ℓ_i, is the integer value of the multiplication of the length of curve C_i and a constant M_ℓ. The curvature-index, ρ_i, is the integer value of the multiplication of the radius of curvature of curve C_i and a constant M_ρ. If curve C_i is a straight line, its ρ_i becomes infinite. For a circular arc of radius r, the absolute value of ρ_i equals the integer value of r multiplied by M_ρ. The sign of ρ_i is negative if the curve C_i is a concave arc and positive if convex. If the curve C_i is a free curve, ρ_i is no longer a constant. In this case we merely assign a minimum integer value to ρ_i in order to distinguish it from a line or circular arc. The multiplication constants, M_θ, M_ℓ and M_ρ are the predefined values which will affect the accuracy of the curve matching algorithm, which will be described in section 3.3 later.

Once the *shape-index-set* for each curve is constructed, the shape of any sheet metal part or tool composed of n curves can be described by the SIS list composed of n *shape-index-sets*. Figure 2 shows the typical examples of the members of *shape-index-sets* in sheet metal part and tool. The *shape-index-set* introduced here has the following features :

—It describes the shape uniquely regardless of translational and rotational effects.

—It allows the reduction of the two-dimensional pattern(shape) matching problem into a one-dimensional comparison of SIS lists.

—It deals with the curve matching problem not in a statistical way but in a definitive one.

—It does not require any polygonal approximation of the shape.

—It requires relatively small memory space to store the shape.

3. Curve Matching by Shape-Index-Set

Suppose that a sheet metal part and a tool are defined by n and m boundary curves respectively. If there exists any curve or list of successive curves of the tool matching the corresponding curve or curves of the sheet metal part after proper Euclidean transformation, they can be easily found by the comparison of SIS lists as follows.

First, construct the list of SIS's, $\{S|S_i,\ i=1,2,\cdots,n\}$ where S_i is the *shape-index-set* corresponding to the i-th curve of the sheet metal part. Similarly, each tool is described by the list of SIS's, $\{T|T_j,\ j=1,2,\cdots,m\}$ where T_j is the SIS corresponding to the j-th curve of tool boundary. Then, the list T is overlapped onto the list S to find out the matching pairs between S_i and T_j, and the matching flag of T_j is turned on if it has the matching pair as illustrated in figure 4. This comparison continues while sliding T to the right step by step. Figure 4 shows a schematic example, where the tool is assumed to be bounded by 4 curves and moved to the right by two steps from the first position. At present situation, the on-off status of the matching flags of T indicates that the 1st and the 2nd curves of the tool successively match the 3rd and the 4th ones of the sheet boundary, at the same time, the 4th curve of the tool matches the 6th of the sheet boundary.

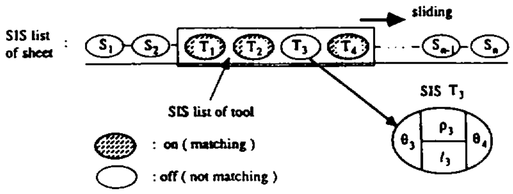


Fig. 4. Curve matching using shape-index-set lists of sheet and tool.

The procedure will be understood better by applying it to the actual examples. In figure, 5, for simplicity, all numbers without any special notes imply the number of the *shape-index-set* on SIS list or the number of the corresponding curve of the sheet and the tool. As a result of the comparison between two SIS lists, figure 5(a) shows that maximum matching occurs between the 4th, 5th, and 6th curves of the tool and the 2nd, 3rd, and 4th of the sheet. Note that SIS for the sheet is cycling for the comparison. In figure 5(b), there are two matching curve groups of the tool i.e one matches the group of the 3rd to the 8th curve and the other does that of the 10th to the 12th of the sheet boundary.

Schwartz and Sharir(1987) also devised an algorithm to find the matching portion of the two curves. In this algorithm, equally spaced points along the arc length are generated on each curve, then curve matching is performed by minimizing the least-square-distance be-

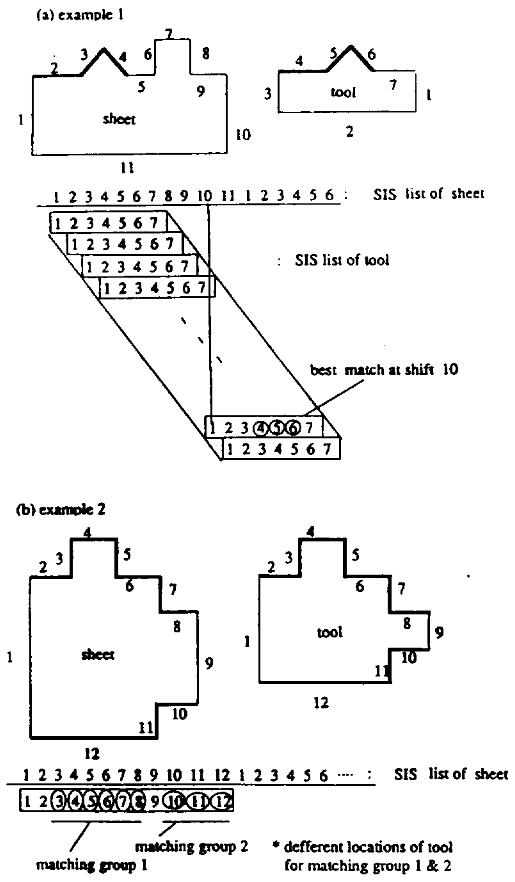

(a) example 1
sheet
tool
1 2 3 4 5 6 7 8 9 10 11 1 2 3 4 5 6 : SIS list of sheet
: SIS list of tool
best match at shift 10
(b) example 2
sheet
tool
1 2 3 4 5 6 7 8 9 10 11 12 1 2 3 4 5 6 ···· : SIS list of sheet
matching group 1
matching group 2
* defferent locations of tool for matching group 1 & 2

Fig. 5. Examples of curve matching using SIS lists.

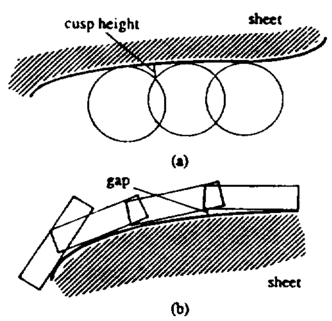


Fig. 6. Example of nibbling operations.

tween these points. This method proved to work well for the matching of free curves including noisy data (Schwartz와 Sharir, 1987; Wolfson, 1990). However, due to its heavy computation, it doesn't seem to be efficient to use this algorithm in searching for the successive matching curves among the curve lists especially when most of the curves are noise-free straight lines or circular arcs. Therefore, the aforementioned algorithm is adopted only for the matching of free curves in this work.

4. Search for Candidate Tools

The tool whose boundary curves match the complete or the partial boundary of the sheet metal part is called a candidate tool. If there exists a candidate tool, it must be searched for successfully and its position and orientation must be also determined to punch out the corresponding boundary of the sheet metal part correctly. For this purpose, following procedure is devised in this work:

Step 1) Construct the SIS list S, $\{S \mid S_i,\ i=1,2,\cdots n\}$ for the list of curves of the sheet which have not been punched yet. Similarly, the SIS list of tools, $T^k(k=1,2,\cdots,ntool)$, is also constructed as $\{T^k \mid T^k_j\ ,\ j=1,2,\cdots,m(k)\}$ where ntool is the total number of tools installed on the current turret and m(k) is the number of the curves bounding the k-th tool.

Step 2) Search for all the candidate tools each of which has at least one SIS matching S constructed in Step 1 above. The candidate tools and their corresponding matching curves compose the matching curve list set, MCLS, as follows;

$$MCLS=\{(C^{k1}_{j1}\ ,C^{k1}_{j1+1}\ ,\cdots),\ (C^{k2}_{j2}\ ,C^{k2}_{j2+1}\ ,\cdots),\cdots\}$$

The superscript, k1 and k2, in the MCLS, denotes the tool identification number, and thus C^{k1}_{j1} denotes the j1-th curve of the tool k1, and so on. The information stored in the matching curve list set will be used to calculate the location and orientation of the tool later. From the set described above, the candidate tools are sorted in the descending order of the number of matching curves, and tested in order against interference until a candidate tool passing through the test is found. If there is no candidate tool or any candidate tool can't pass the interference test, other candidate tools are searched for in the next step.

Step 3) Search for the tools with the shape index, $\{\rho^k_{j-1}, \theta^k_j, \rho^k_j\}$, matching $\{\rho_{i-1}, \theta_i, \rho_i\}$ of S_i within a predefined tolerance. The superscript, k, is the tool id number as mentioned before. Note that the curve length is omitted and thus only the angles at vertices are considered in this comparison. This means that the candidate tools selected in this step can be used to punch out some portion of the sheet metal part around the vertex, where curve C_{i-1} and C_i meet with the angle-index θ_i.

Step 4) Search for the tools whose ρ^k_j equal to ρ_i of S_i within a predefined tolerance so tath the selected candidate tool has the curve of the same curvature with that of C_i of the sheet. If no candidate tool is found and the punching operation is still needed, the searching process continues to the next step.

Step 5) In this step, nibbling tools are selected as candidate tools with which the boundary curves can be punched out approximately within a predefined allowable tolerance. Figure 6 shows some examples of the nibbling operation.

Steps 1 to 5 are repeated until all the boundary curves of the sheet metal part are treated. Whenever a candidate tool succeeds in punching the sheet boundary at any aforementioned step without interference, the punching flags of the corresponding matching curves of the sheet part are updated. The doubly-linked-list of sheet boundary curves must be updated whenever the number of the curves increases due to the punching of only the portion of the curves.

5. Interference Test and Punching Operation

It is necessary to check whether or not the selected candidate tool breaks in the boundary of the sheet to avoid unexpected punching operations. Figure 7 (a) shows the situation where the interference test cannot be passed. Therefore, the punching operation is accomplished in the following procedures. First, the location and orientation of the tool are calculated from the matching pair of curves, and then all curves of the candidate tool are transformed accordingly to be aligned with the corresponding matching curves of the sheet boundary. Second, the interference test is performed between the sheet part and the moved tool. Third, the curves being punched by the tool are searched for from the sheet boundary. In this step, the curves of the sheet boundary may be split so as to handle the situation illustrated in figure 7(b).

For the interference test, an efficient and practical method has been developed. Basically, it compares the angles between the curves at all the points where the curves of the tool and sheet intersect. Following paragraph briefly explains how the interference test works.

Let's denote the curve segments of the tool incoming to and out-going from the intersection point by t_1 and t_2, similarly, those of the sheet boundary by g_1 and g_2 respectively as illustrated in figure 8. Note that the curves are directed as described earlier. If the curve segment is not a straight line, that segment will be replaced by a tangent line at the intersection point. Let θ_t, θ_{g1} and θ_{g2} be the angles measured in the counter clockwise direction from t_1 to t_2, g_1, and g_2 respectively. Then following condition should be satisfied for the tool not to break in the sheet boundary (i.e. to pass the interference test);

$$0 < \theta_t \le \theta_{g2} < \theta_{g1} \le 2\pi$$

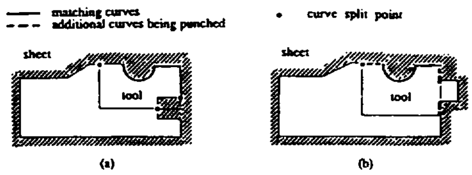


Fig. 7. Interference test and punching.

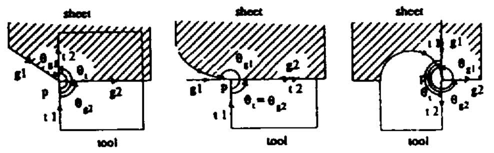


Fig. 8. Interference test between sheet and tool.

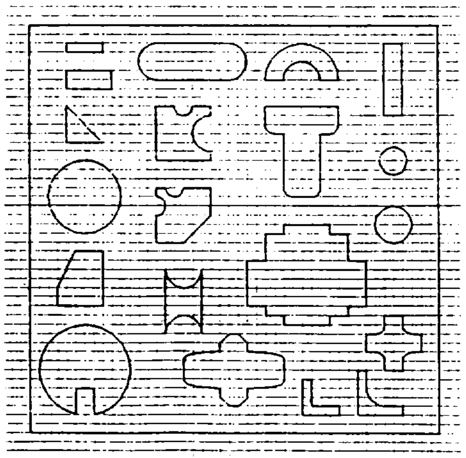
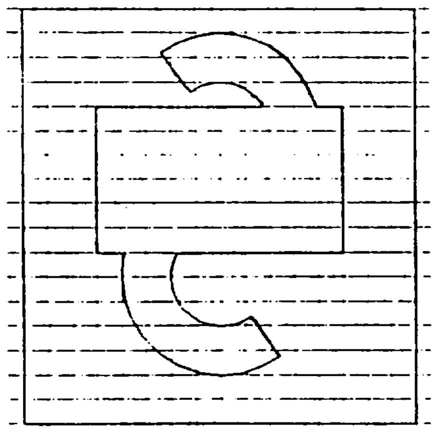


Fig. 9. Example of tools.

Computational Experiments and Discussion

The algorithm has been implemented in the C language on an engineering workstation. Several computational experiments have been performed on the assumption that all the tools in a turret can rotate to arbitrary orientations.

Figure 9 shows some of the tools used in the examples to test the algorithm. Figure 10-15 show several results of automatic tool selection. For the examples, the number of tool types used, total number of hits, and total CPU time of the program on a 15 MIPS workstation are summarized in Table 1. The result of these experiments proves the success of the tool selections with few redundant



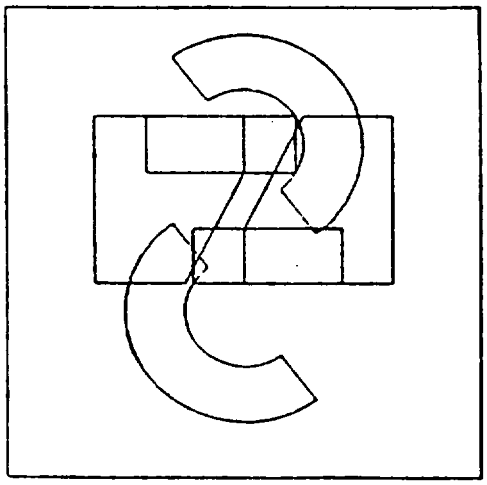


Fig. 10. Tool processing result-example 1.

Table 1. Results of automatic tool processing

Example No.	No. of Tools	No. of Hits	Time* (sec)
1	3	6	12
2	6	11	27
3	6	10	29
4(a)	8	17	65
4(b)	8	18	67
5(a)	5	34	76
5(b)	6	41	62
6	10	55	351

* total CPU time on 15 MIPS Unix workstation

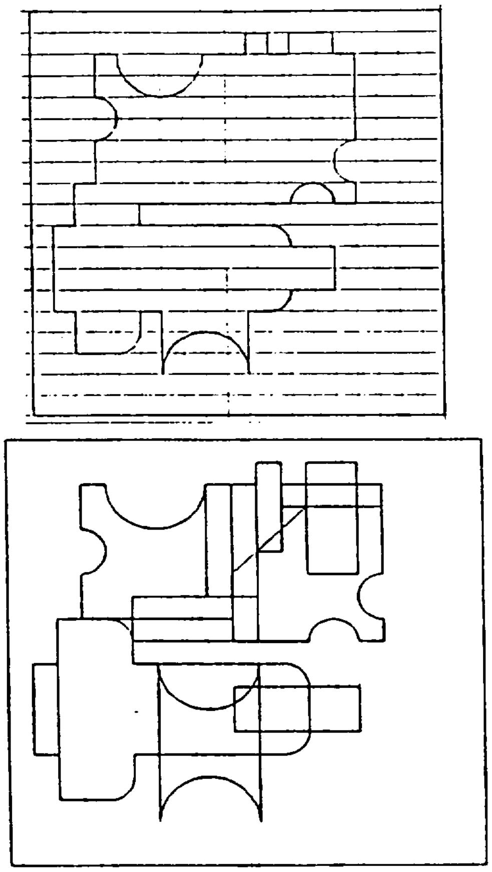


Fig. 11. Tool processing result-example 2.

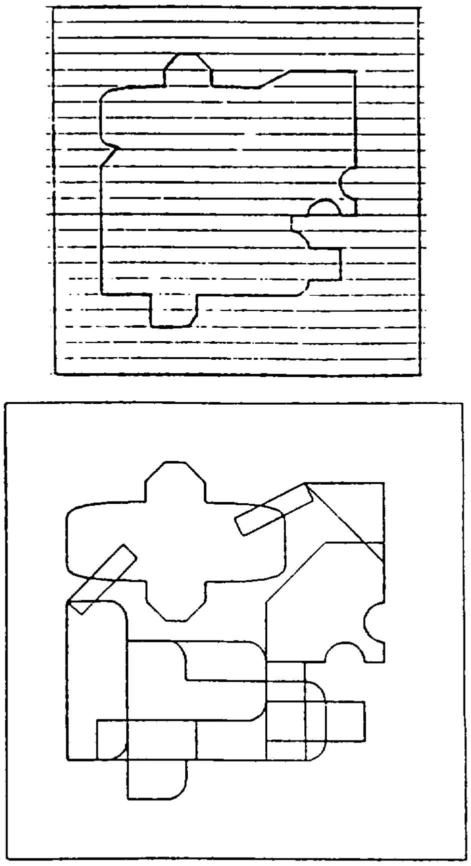
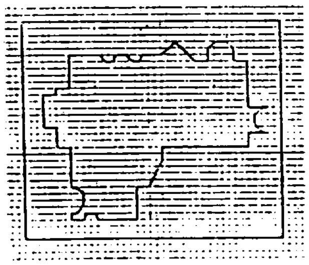


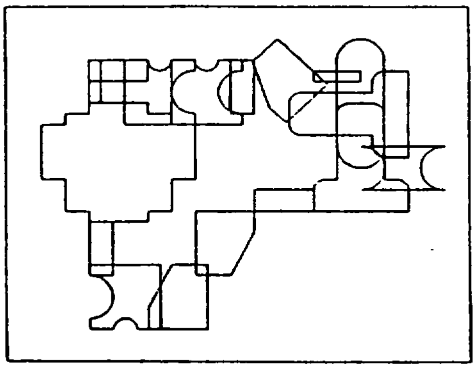
Fig. 12. Tool processing result-example 3.

hits.

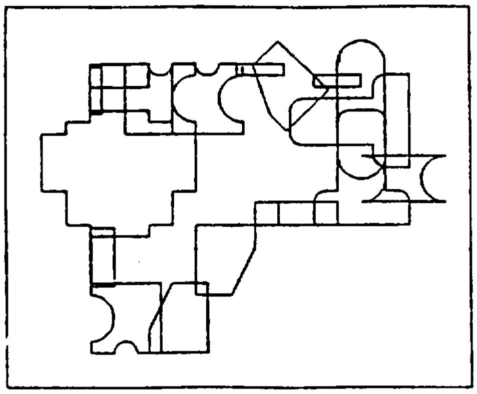
Several tool processing results are presented including free curve matching in figure 12 and nibbling operations in figure 14-15. In general, a few tools such as circular or rectangular shaped tools are defined as indexing tools which can rotate to any angle and thus can be used as nibbling tools. In nibbling operations, tools are selected automatically so that the cusp height or gap is within the predefined value.

When two punching tools meet together at a point on the boundary curve without any overlap, an undesirable burr usually occurs around that point. A common practice to





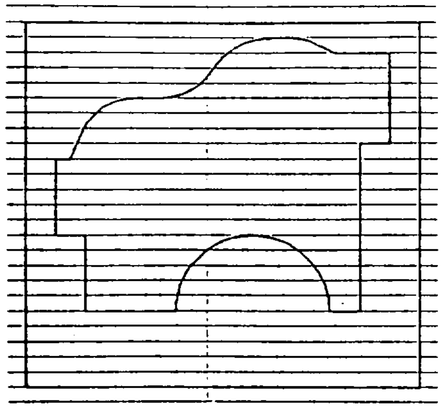
(a)

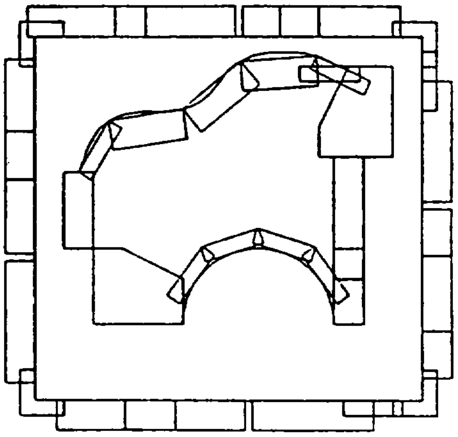


(b)

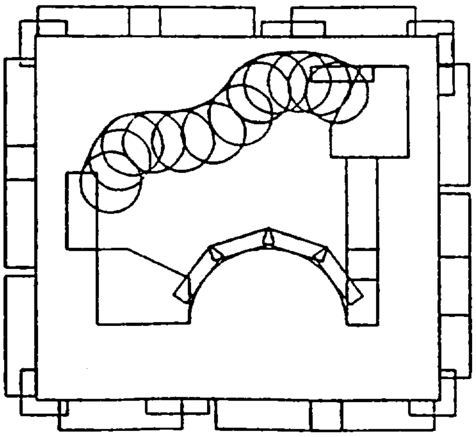
Fig. 13. Tool processing result-example 4.

minimize the burring effects is to provide the proper locations with overlap-hits, as has been done by hand in real practice. Figure 13 (a) shows the tool processing results without the overlap-hit, while figure 13(b) shows





(a)



(b)

Fig. 14. Tool processing result-example 5.

several overlap-hits done by the system automatically.

The outer boundary of the sheet metal part must not be punched out completely because the separated parts may move in an uncontrolled manner on the NCT bed and cause the malfunction of the machine. For the purpose of this, a small connection, named 'bridge', should be prepared between the sheet part and the raw sheet material. Figure 14-15 show the 'bridges' generated at the locations specified interactively through the graphic terminal.

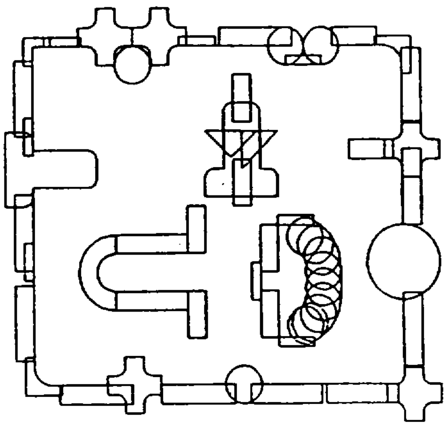


Fig. 15. Tool processing result-example 6.

Conclusions

A concept of the *shape-index-set* is introduced to handle the shape of sheet metal parts and tools, and an algorithm is developed using this concept a so that the successive matching curves between two curve lists, one from the punching tool and the other from the sheet metal part are easily identified. Based on this algorithm, a procedure that can automatically select the tools to punch out the boundary of sheet metal parts is also developed. Through several experiments, the efficiency and the robustness of the automatic tool selection procedure was verified.

With the procedure for the automatic tool selection, many problems due to manual tool processing are to be reduced. The productivity of the manual processing has been mainly dependent on the capability of the human expert and the efficient usage of the sheet metal through an optimal nesting has often been sacrificed for easy manual processing. The following works are left for further study.

—In this work, it was assumed that all tools could rotate to any orientation. In general, however, only a few of them are allowed to rotate because of the limited capacity of the NCT hardware. Therefore, the information on the real indexing tools must be integrated with the tool selection algorithm.

—Tool path optimization, though it is not directly concerned with automatic tool selection, must be accomplished to improve the production performance in NCT operation.

References

Ansari, N. and E. J. Delpt, 1990, Partial Shape Recognition : A Landmark Based Approach, *IEEE Trans. on Pattern Analysis & Machine Intelligence*, Vol.12, No.5, May, 470~483.

Ayache, N. and O. D. Faugeras, 1986, HYPER : A new approach for recognition & positioning of 2D objects, *IEEE Trans. on Pattern Analysis & Machine Intelligence*, Vol. PAMI-8, Jan., 44~54.

BRAVO 3, 1988, Sheet Metal Design/

Fabrication User's Guide, U.S.A.

CIMATRON '90, 1990. Cimatron Co., Israel.

Dubois, S. R. and F. H. Glanz, 1986, An Autoregressive Model Approach to 2D Shape Classification, *IEEE Trans. on Pattern Analysis & Machine Intelligence* Vol. PAM 1-8, No.1, Jan., 55~66.

Dudani, S. A. *et al.*, 1977, Aircraft identification by moment invariants, *IEEE Trans. Comput.* Vol. C-26, Jan., 39~46.

EUCLID-IS, 1987, Sheet Metal Module-Reference Manual, Matra Datavision, France

Freeman, H., 1978, Shape description via the use of critical points, *Partiern Recognition,* Vol.10, 159~166.

Gorman, J. W. *et al.*, 1988, Partial Shape Recognition Using Dynamic Programming, *IEEE Trans. on Pattern Analysis & Machine Intelligence,* Vol.10, No.2, March, 257~266.

Granlund, G. H., 1972, Fourier preprocessing for hand printed character recognition, *IEEE Trans. Comput.* Vol. C-21, Feb., 195~201.

Hong, J. and H. J. Wolfson, 1988, An improved model-based matching method using footprints, *Proc. Int. Conf. Pattern Recognition,* Rome, Italy, Nov., 72~78.

Kalvin, A., *Sharir, M. et al.*, 1986, 2D model based boundary matching using footprints, *Int. J. Robotics Res.*, Vol.5, No.4, 38~55.

Kimura, F. *et al.*, 1987, Automatic Process Planning for Sheet Metal Parts with Bending Simulation, *Intelligent & Integrated Manufacturing Analysis & Synthesis,* ASME PED-Vol.25, 245~258.

Lin, C. C. and R. Chellapa, 1987, Classification of Partial 2-D Shapes Using Fourier Descriptors, *IEEE Trans. on Pattern Analysis & Machine Intelligence,* Vol. PAM 1-9, No.5, Sep, 686~690.

Niemann, H., 1981, *Pattern Analysis,* Springer-Verlag Berlin, Germany

Reeves, A. P. *et al.*, 1988, 3D-Shape Analysis Using Moments & Fourier Descriptors, *IEEE Trans. on Pattern Analysis & Machine Intelligence* Vol.10, No.6, Nov., 937~943.

Schmidt, W., 1990, Modified Matched Filter for Cloud Clutter Suppression, *IEEE Trans. on Pattern Analysis & Machine Intelligence,* Vol.12, No.6, June, 594~600.

Schwartz, J. T. and M. Sharir, 1987, Identification of partially obscured objects in two & three dimensions by matching of noisy characteristic curves, *Int. J. Robotics Res.* Vol. 6, No.2, 29~44.

Tiller, W., 1983, Rational B-Spline Curve and Surface Representation, *IEEE,* 61~69.

Unigraphics Ⅱ, 1991, McDonnel Douglas, U.S.A.

Wallace, T. P. and P. A. Wintz, P. A., 1980, An efficient 3D aircraft recognition algorithm using Fourier Descriptors, *Comput. Graphics Image Processing,* Vol.13, 99~126.

Wolfson, H. J., 1990, On curve matching, *IEEE Trans. on Pattern Analysis & Machine Intelligence* Vol.12, No.5, May, 483~489.

〈國文抄錄〉

판금작업을 위한 공구 자동선정 알고리즘

판재부품 가공분야에서 사용하고 있는 수치제어 펀칭기계의 효율적인 구동을 위해 시급히 해결되어야 할

사항중의 하나는 펀칭공구선정의 자동화문제이다.

이를 위해 본 연구에선 형상지표집합이라는 개념을 도입하여 임의의 두 곡선군에서 서로 일치하는 곡선들의 부분집합을 쉽게 탐색할 수 있는 알고리즘을 개발하였고, 이를 이용하여 판재부품 가공을 위한 펀칭 공구를 자동으로 선정해주는 모듀울을 완성하였다.