Development of a Plate Manufacturing CAD/CAM Program for a Optimal Layout and Distributed Control System

Hunmo Kim*
School of Mechanical Engineering, Sungkyunkwan University

A Problem of relevant interest to some industries is that of obtaining optimum two-dimensional layout. To solve this provlem, one is given a number of rectangular sheets and an order for a specified number of each of certain types of two-dimensional regular and irregular shapes. The aim is to cut the the shapes out of the sheets in such a way as to minimize the amount of waste produced. A DCS (Distributed Control System) is an integrated system which applies the decentralization concept to a control system handling both sequential and analog control. A DCS performs many operations such as data gathering, data processing, data storing and monitoring the operatin conditions for the operator. IN this paper, we propose a genetic algorithm based on rotation parameters from which the best pattern of layout is found as well as a layout method for better performance time. A DCS for the plate cutting process system, which is performed by a virtual system, is also identified.

Key Words : CAD/CAM System, Genetic Algorithms, Optimal 2-D (2-Dimensional) Layout, Rotation Parameter, DCS(Distributed Control System)

1. Introduction

One problem with rectangular shapes or irregular pattern arrangements is to effectively employ algorithms for 2-D shapes that minimize the amount of wasted space in a limited area or maximize the number of shapes. This shape layout problem acts as an important parameter in productivity (Gilmore and Gemory, 1965; Adamowicz and Albano, 1976; Sarin, 1983; Lamousin et al., 1996).

There are two types of pattern layouts : one is a rectangular pattern layout and the other is an irregular pattern layout. The problem with rectangular patterns was noted by Gilmore and others. They solved 1-D, 2-D and 3-D cutting problems using linear problem and knapsack functions (Gilmore and Gemory, 1965; Gilmore and Gemory, 1966). Adamowicz and Albano (1976) proposed a two-stage approach induced by Heuristic algorithms. The first stage is to form rectangles into strips and the second is to lay these strips out in an optimal pattern using a dynamic problem (Albano, 1977; Albano and Sapuppo, 1980). However, since this approach can only be used with simple rectangular shapes, it can not be generalized.

A study of irregular pattern layout was first proposed by Adamowicz and Albano (1976). Using a two-stage approach, they solved the layout problem by forming minimal rectangular modules, which surround an arbitrary pattern or pattern group, and optimally laying out the modules. Albano and Sapuppo (1980), who found a rough layout automatically by using a Heuristic method and a dynamic problem, then proposed a dialogue method for use by all users. However, with this Heuristic method, it is possible to fall into a regional solution, rather then....

In this research, we intend to approach the pattern layout problem using genetic algorithms. This method allots numbers to patterns and the

* **E-mail** : kimhm@me.skku.ac.kr
TEL : +82-31-290-7450; **FAX** : +82-31-290-5849
School of Mechanical Engineering, Sungkyunkwan University(Manuscript **Received** March 27, 2000 ; **Revised** July 8, 2000)

order of these numbers is used to create genetic letter-rows. Because in an established pattern, the genetic algorithms of letter-rows simply lay out the given row-order, this method can result in a lot of wasted space (Watanabe and Ono, 1997; Bounsaythip and Maouche, 1996). Therefore in this research, using order-numbers for the genetic parameter of each shape, and at the same time considering the rotation of each shape to the genetic parameter, we intend to achieve an optimal layout by executing the genetic algorithms. This method has the advantage of solving the problem of overlaps as the genetic parameter is used to coordinate each shape, cutting down on the time needed to solve the problem and reducing wasted space by using a rotating parameter. The CAD data from the optimal layout generates NC (Numerical Control) code throughout the CAM process and this code is used as input data for the PC-Based DCS. The DCS distributes the control system to modules. It also gathers, stores, and manages data, and monitors the situation for the operator. DCS's are mostly used in the field of chemical engineering. In the process of plate manufacturing, the operator obtains the CAD data for the objective plate and generates NC code throughout the CAM process. After these operations, the NC code is used as input data for the plate machine controller. Therefore, to adapt a DCS to the plate manufacturing process, certain things are needed. These include an exclusive CAD/CAM program, the transfer of the NC code in the network, a monitoring program for the situation, hardware which is suitable for multi-tasking, and a controller. However, the above programs and devices are expensive and they have no flexibility because they only perform the inner control algorithm. So, to overcome the above listed weak points, we propose a DCS using a PC.

2. The Development of a CAD/CAM System in the Plate Manufacturing Process

2.1 The plate manufacturing process

The process of plate manufacturing first involves the design of 2-D drawings. Then, the

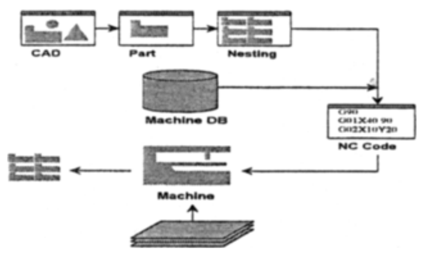


Fig. 1 Procedure of the plate manufacturing procss

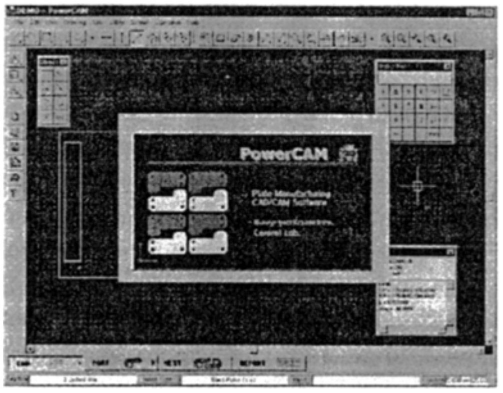


Fig. 2 CAD/CAM program

conditions of manufacture or manufacturing paths are set up, in part from the drawing. The procedure then lays out many shapes in the plate in an optimal pattern. If this procedure is completed, final changes in the overall manufacturing procedure can be made using the NC code. This process is shown in Fig. 1. A GUI (Graphic User Interface) of a CAD/CAM program used in plate manufacturing to execute this overall process is shown in Fig. 2.

2.2 Design of the 2-D drawing

To manufacture plates, we must first design the desired shape in 2-D. Although there are many 2-D design programs, they include 3-D as well as additional functions which are not needed in plate manufacturing. Therefore, the program which we propose includes functions needed for 2-D shape design, enabling users to design shapes efficiently. To be cautious however, we supply a standard exchange format, a program which allows for IGES (Initial Graphics Exchange Specification) and DXF (Drawing Exchange Format). Figure 3 is an example of a 2-D drawing and Fig. 4 is a GUI of a CAD program that

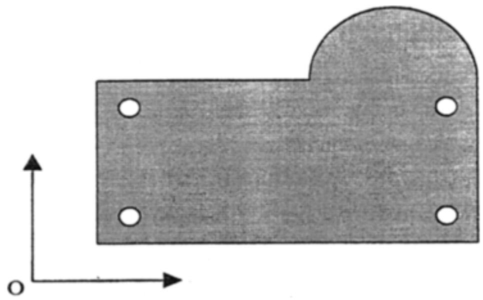


Fig. 3 Example of a 2-D drawing

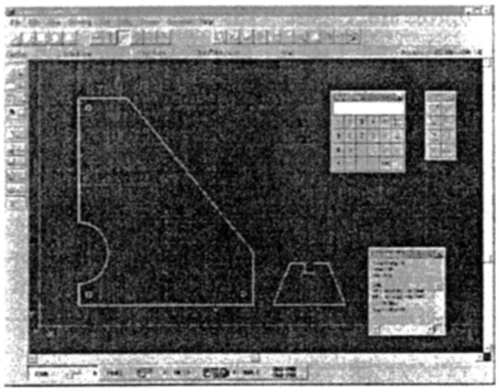


Fig. 4 CAD program for drawing 2-D shapes

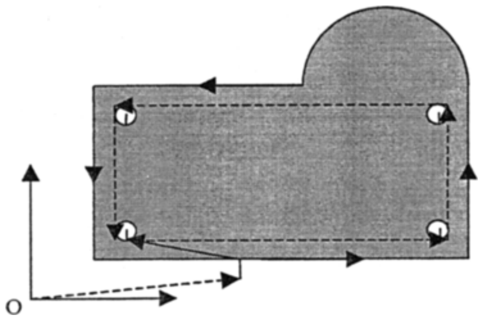


Fig. 5 Set cutting parameter

can draw 2-D shapes.

2.3 Part work

After drawing, we assign materials and paths in stages. A simple example of such a problem is shown in Fig. 5. We must cut the inner hole first and cut the outer part later or decide on the most rapid cutting path. When a laser or plasma is used for cutting, it is necessary to cut the outer line of the designated path first. Otherwise, the material will be harmed. This process is called a piercing process.

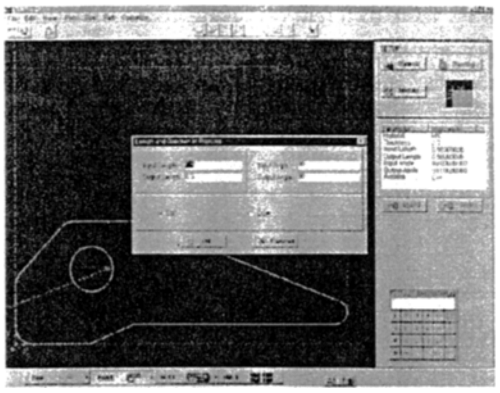


Fig. 6 PART program

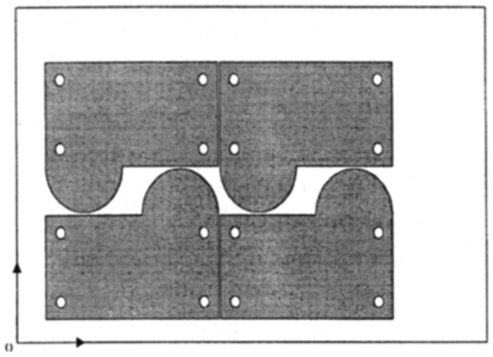


Fig. 7 Example of the nesting

At this stage, we must decide on the class and thickness of the material and set up the method for piercing input or output. A PART program's GUI for PART work is shown in Fig. 6.

2.4 Nesting of a PART

If mass production or a small quantity, of various types are ordered, we should nest the PART in the plate optimally. For example, we properly lay out the ordered number of parts in a rectangular plate and minimize the wasted area as shown in Fig. 7 below. We intend to execute an optimal-layout using genetic algorithms in our paper. Figure 8 shows the GUI of the nesting program.

2.5 Conversion and generation of NC codes

The previous stage is a common technical process, regardless of the particular manufacturing machine. However a somewhat complex process is required to convert the NC code. First, manufacturing machines made by different companies have different NC code systems. Although there are standard NC code systems, individual codes exist and it is necessary to deal with them. Therefore, we should have a database of individual codes and set up each code according to the different machines in use. Table 1 shows the variables and the values that are set up as the database parameters. Through CAD, in part from working and from nesting, we can finally produce a NC code together with the machine's database file as shown in Table 2.

For the same component, the NC becomes the function type. So, we can define the NC code as a subprogram. Figure 9 shows a GUI of the final NC code and Fig. 10 shows the machine file database.

We developed a new CAD/CAM program by

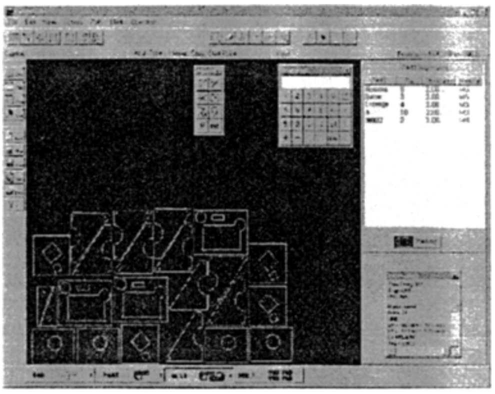


Fig. 8 Nesting program

Table 1 Post file of the machine

[VARIABLES]	[MAINEND]
$1= NCFNAME;	(G91G28Z0.);
$2=THICKNESS;	(G28X0. Y0.);
$3=FEEDRATE;	M30;
$4= KERF;	[MAINEOF]
$5=OFFSET;	%;
$6=DWELLTIME;	[SUBSTART]
...	
...	
$29=DATE;	
$100=DATSIZE;	
[COMMANDS]	
MODE_ABS=G90;	;
MODE_INC=G91;	O⟨$11⟩(SUB OF ⟨$19⟩);
UNITS_MM=G71;	G00, X0, Y0;
UNITS_IN=G70;	[SUBEND]
LINE_NUMBER=N;	M98 P2000;
G_RAPID=G00;	M99;
G_LINEAR=G01;	[CALLSUB]
...	G52, G90, X⟨$13⟩Y⟨$14⟩;
...	G17, G68, X0., Y0., R⟨$27⟩;
REPEAT_STOP=G98;	M98, P⟨$11⟩;
EXACT_STOP=G09;	G69;
[MAINSTART]	
%;	
O⟨$19⟩(PROGRAM NAME =⟨$1⟩. NC);	
(Material Type =⟨$23⟩);	
(Material Thickness = ⟨$2⟩);	
(Sheet X = ⟨$17⟩ Sheet Y = ⟨$18⟩);	
(Nest Size X = ⟨$15⟩ Y = ⟨$16⟩);	
G90G92X0. Y0. ;	

Table 2 Generated NC code

```
%
O0010
(Prg. NO : 67. NC)
(MATERIAL : SPCC 10T)
(SHEET : 288X294)
G90G92X3185. Y1515.
N1G00 X40. 44 Y44. 18
G41 D1
M98 P9900
M98 P9901
G01 X2. 1 Y-2. 1
G01 X0. Y25.
G01 X-25. Y0.
G01 X0. Y-25.
G01 X25. Y0.
G40 G01 X-1. 4 Y1. 4
M98 P2000
N2G00 X6. 91 Y-36. 41
G41 D1
. . .
```

Fig. 9 Generate NC code

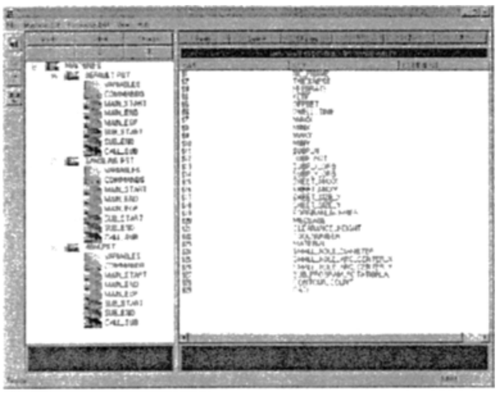


Fig. 10 Post program

strictly choosing functions for 2-D drawings. This new program was able to work with CAM instantly. A Windows-based user interface was used. The process of generating overall NC code is made in stages, enabling workers to gain quick working ability and store databases for different machines to do NC-work independent of the NC-machine.

3. Optimal Layout of Plates

3.1 Layout algorithms

In order to apply genetic algorithms, we must first determine the order of layout after designating a certain number to each form (Gen and Cheng, 1997; Davis, 1996; Michalewicz, 1992). This order is obtained from genetic algorithms. First, the genetic algorithm performs the operations of selection, crossover, and mutation and obtains a string corresponding to the order of layout. The genetic algorithm passes this string to the layout algorithm. After calculating the size of the area according to the order of the string, the layout algorithm obtains a fitness-function. This process repeats for each generation in order to find an optimal value. In this paper, we used two types of genetic parameters to test two kinds of algorithms. The first type applies only strings and the second type applies both strings and the angle of rotation for each form. The second type is divided into two parts as shown in Fig. 11. The first parameter shows the strings of all forms and the second shows the angle of rotation for each form. In the case of the angle of rotation, there are eight conditions. This means that we intend to restrict the angle of rotation to eight conditions over 360 degrees, so as to help improve the speed of execution.

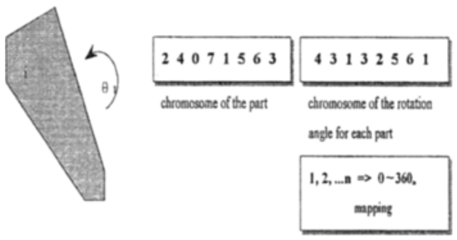

θ
2 4 0 7 1 5 6 3
4 3 1 3 2 5 6 1
chromosome of the part
chromosome of the rotation angle for each part
1, 2, ...n => 0~360, mapping

Fig. 11 Chromosome

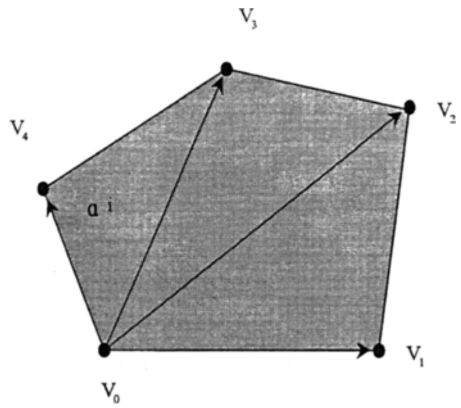


Fig. 12 Calculation of the polygon

The minimal wasted area can be correctly applied to the fitness-function and can be calculated by the following Eq. (1).

$$\min S = W \times L_{req} - \sum_{i=1}^{N} a_i \quad (1)$$

$$\text{subject to } \sum_{i=1}^{N} \sum_{j=i+1}^{N} \beta_{i,j} = 0$$

$$\beta_{i,j} = \begin{cases} 0 & \textit{Non-overlap} \\ 1 & \textit{Overlap} \end{cases}$$

$$\textit{Fitness} = \frac{1}{S} \times a$$

W is the fixed height of the plate, L_{req} is the minimum width that can include all of the nested forms, a_i is an area of arbitrary form I, and $\beta_{i,j}$ is a value to indicate the overlap between i and j. To get the fitness-function, S which is sum of all the forms, is first found and then the reciprocal of S is taken. To obtain a proper scaled value, "the reciprocal of S" is multiplied by a constant a. The area of the polygon type like the line shown in Fig. 12 can be solved by the following Eq. (2).

$$a_i = \frac{1}{2} \sum_{k=1}^{N-2} \left| V_k \times V_{k-1} \right| \quad (2)$$

V_i is a vector from the base point V_0.

In this research, we used Position-Based Crossover as the crossover method. The son's gene is made by crossing over factors in random positions from the first parent's gene with factors from the second parent's gene, which are not equal to the factors from the first parent.

For the operation of mutation, we used an insertion mutation method. This method changes

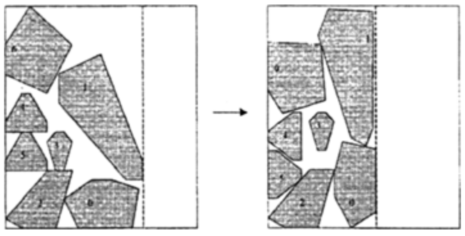


Fig. 13 Example of the layout

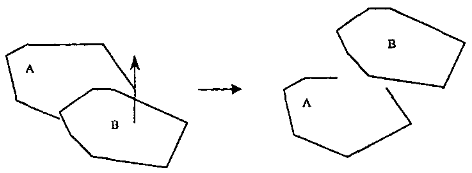


Fig. 14 Move problem for overlap prevention

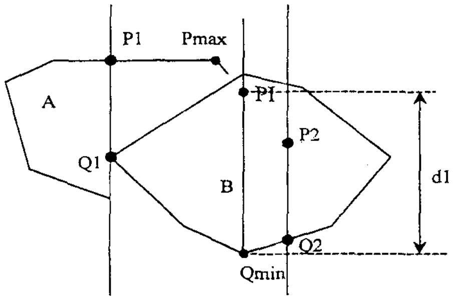
factors in random positions with factors in other positions in the same chromosome.

Figure 13 shows an example of the layout results in accordance with the strings moving forward in the y-direction. This is the result which occurs when the algorithm applies both strings and the angle of rotation for each form.

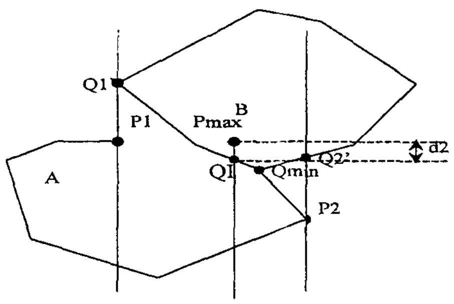
We perform layout algorithms by using strings of the results of the genetic algorithm. In this example, all forms move in the y-direction and are arranged so as to avoid overlapping with one another. Also, if all forms reach the extreme boundary of the plate, these forms must move to a minimum y-point moving in the x-direction simultaneously. The same method then repeats.

The algorithm for overlap prevention is executed as follows. The algorithm first examines and then verifies that all the angular points in one form are enclosed by a different form. When the points are not enclosed, the algorithm examines the line in order to connect all the points that intersected with the line of another form.

All forms are moved in the y-direction with a value to be calculated, from the bottom left of the plate. When a form reaches the boundary of the plate, this method is repeated after moving in the x-direction and the layout is completed. Figure 14 shows an example of this movement. In the following example, to improve the proceeding velocity, we created an algorithm involving the calculation of moving values in the y-direction



(a)



(b)

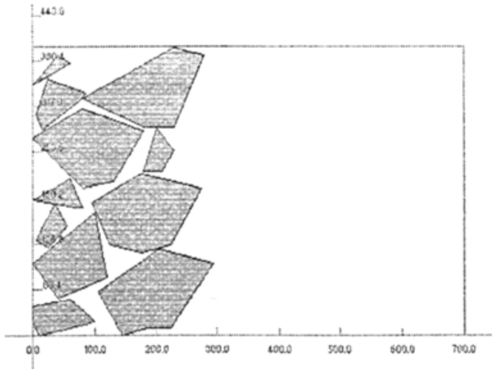
Fig. 15 Calculation of moving distance

like that shown in Fig. 15.

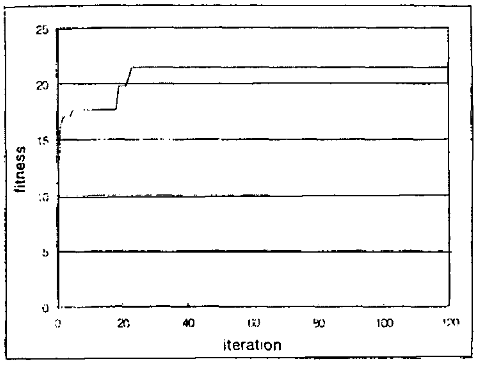
First, find a minimum x-coordinate point and a maximum x-coordinate point on the overlapping area of the forms A, B along the x-direction in Fig. 15. We call these points P_1, P_2, Q_1, Q_2 as shown in Fig. 15. Then find the minimum Q_{min} point of moving form B along the y-direction, and find P_I which is a crossing point between the vertical line from Q_{min} and the fixed form A. At this time, we refer to P_I - Q_{min} as d_1. We find the maximum point P_{max}, which is a value between P_1 and P_2, and find the point Q_I which is a crossing point between the vertical line from P_{max} and the moving form B.

If Q_I is larger than P_{max} , then we do not need to move the moving form B, and there fore, d_2 is 0. If Q_I is smaller than $P_{\max}$, then P_{max} - Q_I is d_2. All moving distances are shown in the following Eq. (3).

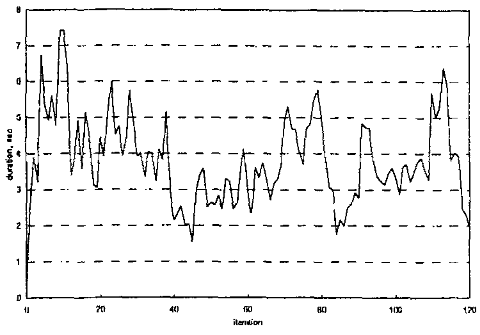
$$d = d_1 + d_2 \tag{3}$$
$$d_1 = P_I - Q_{min}$$
$$d_2 = \begin{cases} 0, & \text{if } Q_I \geq P_{\max} \\ P_{\max} - Q_I, & \text{if } Q_I < P_{\max} \end{cases}$$



(a)



(b)



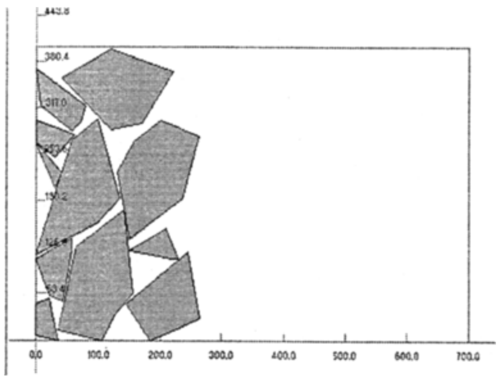
(c)

Fig. 16 (a) The result of the simulation of the genetic algorithm without the rotation parameter(Fitness at 100th generation : 21. 4592), (b) Fitness (c) Average duration(3. 83 sec/gen.)

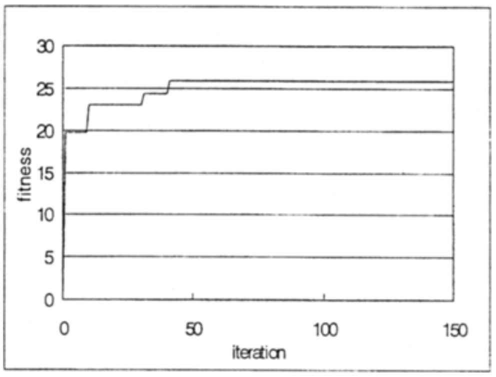
If Q_I is larger than $P_{\max}$, and B is crescent-shaped, overlap with A will occur at point Q_I. In this case, the crescent-shaped B should be moved by the distance dy to prevent overlapping. So the practical moving distance becomes $d_1 + d_2 + dy$.

3.2 Layout problem

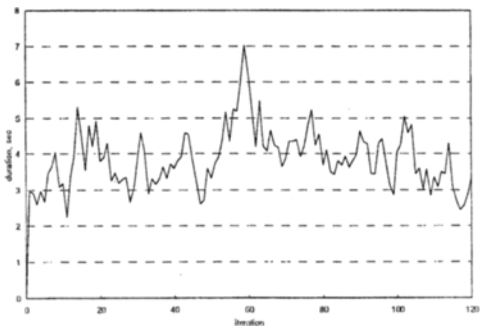
In this paper, we compare two cases: one considers the rotation factor and the other does not. In the former case, we compare the layout performance in accordance with the changes in population. Figures 16 and 17. show the optimum layout of genetic algorithms. In Fig. 16(a) and Fig. 17 (a), we can see that the result in which rotation is considered is better, because the probability that chromosomes include a rotation factor is larger than the probability of a nonrotation case and hence, reduces empty space among forms. We can also verify the same result in Table 3, but there is almost no difference in the layout average duration of the layouts. This shows that the application of the rotation factor does not greatly affect the average duration of the layouts.



(a)



(b)



(c)

Fig. 17 (a) The result of the simulation of the genetic algorithm with the rotation parameter.(Populatioin : 15, fitness at 100th generation : 25.9067), (b) Fitness (c) Average duration(3.89 sec/gen.)

Figure. 18 shows the results due to changes in the population from 6 to 25 to 35. The velocity of fitness that approaches the optimal solution is faster in the case of the largest population. However in Fig. 19, the average duration of the layout increases from 3.83 to 11.66 to 35.87 seconds. Thus the ability to obtain an optimal solution decreases, as the polulation increases.

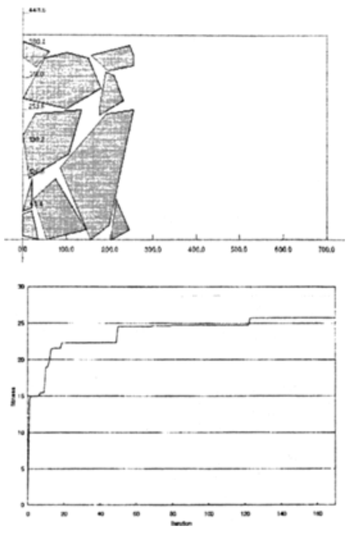
Table 4 shows the fitness-function value and the average duration of the layout.

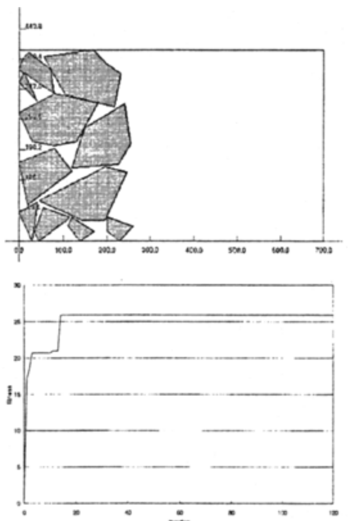
We could tell from the acquired results that the population which satisfies the completion time and the approaching velocity of the optimal solution is about 10~20.

We developed an auto nesting program with genetic algorithms as shown in Fig. 20. This program is able to perform simulations by changing parameters such as the rotation factor and the population.

Table 3 Comparison of the two cases in accordance with the rotation parameter

	Nonrotation	Rotation
Polygon No.	11	11
Population Size.	15	15
Fitness-function	21.4592	25.9067
Cross	4 pair	4 pair
Mutation Rate	0.05	0.05
Average Duration	3.83sec/gen.	3.89sec/gen.


(a) The fitness of the genetic algorithm Population : 6, Fitness at 100th Generation : 24.7200)


(b) The fitness of the genetic algorithm (Population : 25, Fitness at 100th Generation : 25.9067)

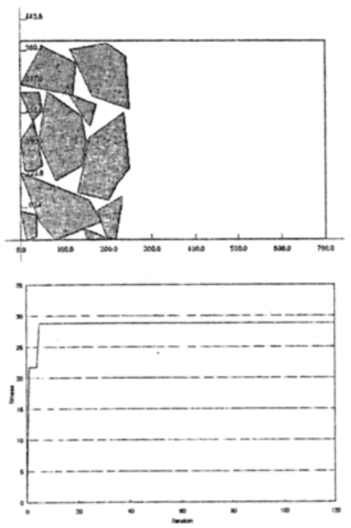

(c) The finess of the genetic algorithm (Population : 35, Fitness at 100th Generation : 28.8324)

Fig. 18 The fitness for various populations

4. PC-Based Distributed Control System

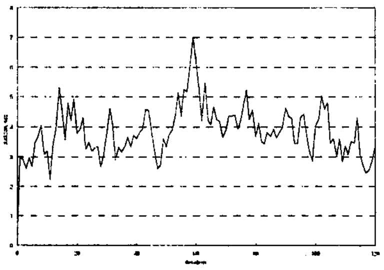
4.1 Structure of DCS

The structure of the PC-Based DCS which is used in plate manufacturing is shown in Fig. 21.

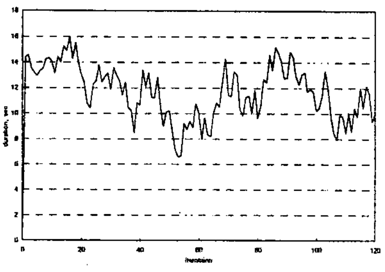
The main server includes a CAD program, a CAM program which generates NC codes, a monitoring program which observes the operation of every machine, a database, and a communication module that can communicate between the main server and the local PC. The local PC carries out the control on each machine and receives the information from the main server. We conducted a simulation to examine the control of a local PC.

Table 4 Comparison of the three cases in accordance with population

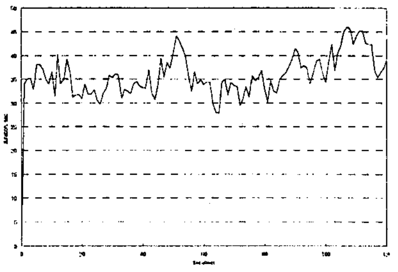
	Population 6	Population 25	Population 35
Polygon No.	10	10	10
Fitness-function	24.7200	25.9067	28.8324
Cross	4 pair	4 pair	4 pair
Mutation Rate	0.05	0.05	0.05
Average Duration	3.83sec/gen.	11.66sec/gen.	35.87sec/gen.



(a) Genetic algorithm duration per generation (Population : 6, Total duratioin (to 120th Generation) : 459.6 sec, Average duration : 3.83 sec/gen.)



(b) Genetic algorithm duration per generation (Population : 25, Total duratioin (to 120th Generation) : 1399.2 sec, Average duration : 11.66 sec/gen.)



(c) Genetic algorithm duration per generation (Population : 35, Total duratioin (to 120th Generation) : 4304.4 sec, Average duration : 35.87 sec/gen.)

Fig. 19 Duration per generation for various populations

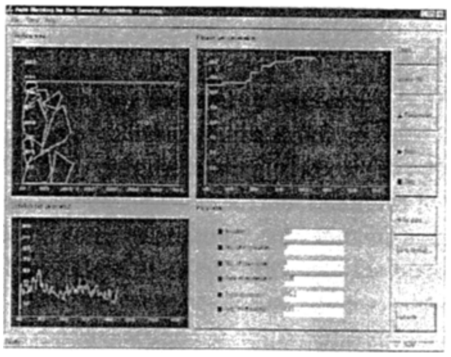


Fig. 20 Auto nesting program with genetic algorithm

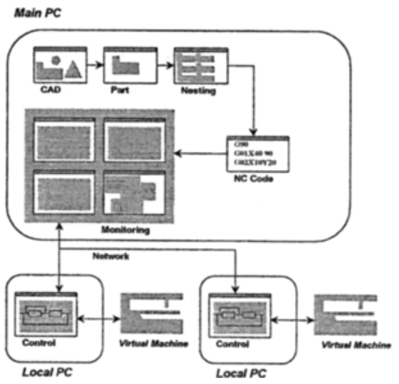


Fig. 21 Structure of the PC-based DCS

The stages of a PC-Based DCS which are used in the plate manufacturing process are shown below.

① Generate the part drawing(CAD)
: generate a 2-D drawing of a closed shape

② Initialize the CAM for each part
: setting up materials, thickness, machine types, manufacturing conditions, and so on.

③ Perform the optimal layout on many parts

④ Generate the NC codes

⑤ Transmit the NC codes from the main server to distributed local PCs.

⑥ The local PC takes charge of the control logic and operates each machine on the basis of codes obtained from the main server. It also transmits the result of the control to the main server. The cycle continues until it finishes all the operations using the codes.

⑦ The main server observes all stages of the process and preserves the required data. Also, it takes suitable measures to correct errors in the local PC.

4.2 PC-based DCS algorithm

The GUI of the total system and its construction are shown in Fig. 22. There is a main server which controls the total system. A large number of local PCs are connected to the main server and communicate bi-directionally using a TCP/IP protocol. The main server can preserve the data as needed.

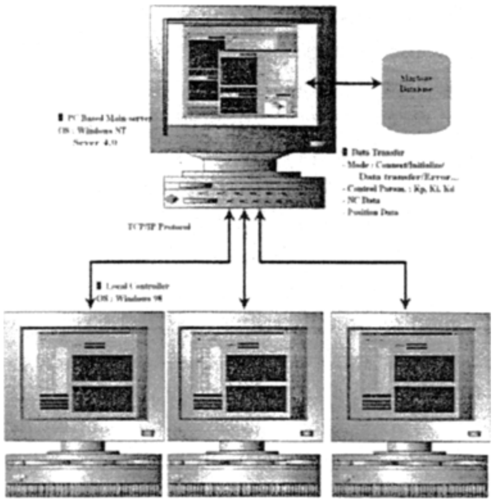


Fig. 22 PC-based DCS

4.2.1 Monitoring program & open type control simulation program

The main server should communicate with the local PC bi-directionally and observe the operation of the individual machines. Also, it should check errors throughout the operation and take suitable measures to correct the error in the local PCs. In this way, it can preserve the data throughout the operation. The GUI of the total system is shown in Fig. 23.

The primary duties of the main server are listed below.

① Setting up files about the machines: The user can make up a database containing information about the various types of machines. Also he can add and change information about each machine within the database. The details of the database are shown in Table 5. It includes the type, capacity, and size of the motor which is used in transferring the axis of a machine.

The GUI connected with the machine database is shown in Fig. 24. The user can change the information in the database with ease.

② Tuning the controller : We made modules for all types of control algorithms. The user can tune the controller of the mechanical system model. If the user sets up the gain value of the control algorithm, the main server transmits it to the controller of the local PC. We applied a PID controller to this program, which was tuned about two axes (x-axis, y-axis). The GUI for the controller tuning is shown in Fig. 25.

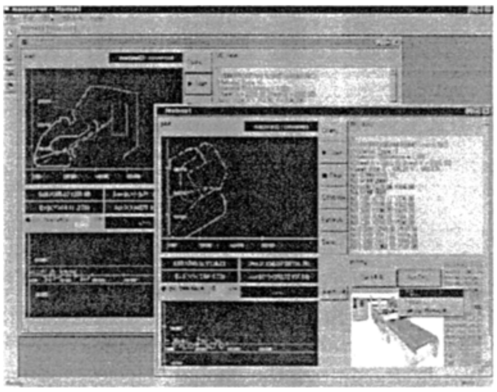


Fig. 23 The GUI of the main-server .

Table 5 Machine database

Detail	Explanation
ID	ID
Name	Machine Name
Motor	Motor type
Move range (X)	Move range (X)
Move range (Y)	Move range (Y)
Moment of Inertia (X)	Moment of Inertia in motor
Moment of Inertia (Y)	Moment of Inertia in motor
Viscous friction coef (X)	Viscous friction coefficient
Viscous friction coef (Y)	Viscous friction coefficient
Kp (X)	Gain value in PID
Ki (X)	Gain value in PID
Kd (X)	Gain value in PID
Kp (Y)	Gain value in PID
Ki (Y)	Gain value in PID
Kd (Y)	Gain value in PID
Total weight	Machine weight
Size W	Machine width
Size L	Machine length
Size H	Machine height
Gas	Laser gas

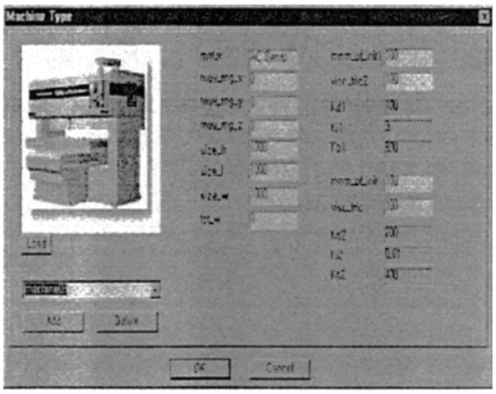


Fig. 24 Setting the machine database

We assumed that the machine was a general rotation system composed of load inertia and viscous friction damper. The result of the modeling process is shown below.

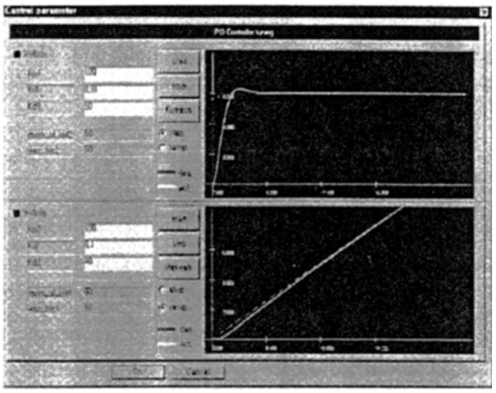


Fig. 25 Controller tuning

$$J\omega' = -b\omega + T \tag{4}$$

where, J : load inertia, [Kg-m^2]
b : viscous friction coefficient, [N-m/rad/sec]
ω : angle deceleration, [rad/sec]
T : torque, [N-m]

After obtaining suitable gain values by tuning, the main server preserves these gain values in the database and transmits them to the local PC. Therefore, if the system parameters for other machines are changed, the user can obtain suitable gain values. This is a characteristic of an open type system.

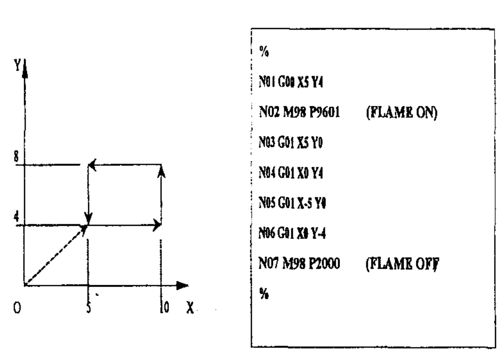
③ Monitoring the operation : The main server observes the operation in the local PC. The main server observes the process by which the laser cutting machine manufactures shapes, and in the case of an error, the main server displays an error message as a kind of alarm.

④ Managing the operation data : The main server preserves data or error information as files. The data or error information may be used later.

The main server communicates with the local PCs bi-directionally using a TCP/IP protocol.

4.2.2 Local PC program

The local PC includes a communication module for communicating with the main server, an interpreter for the NC codes, and a location control loop module.



```
%
N01 G00 X5 Y4
N02 M98 P9601     (FLAME ON)
N03 G01 X5 Y0
N04 G01 X0 Y4
N05 G01 X-5 Y0
N06 G01 X0 Y-4
N07 M98 P2000     (FLAME OFF
%
```

Fig. 26 Cutting path and Nc code

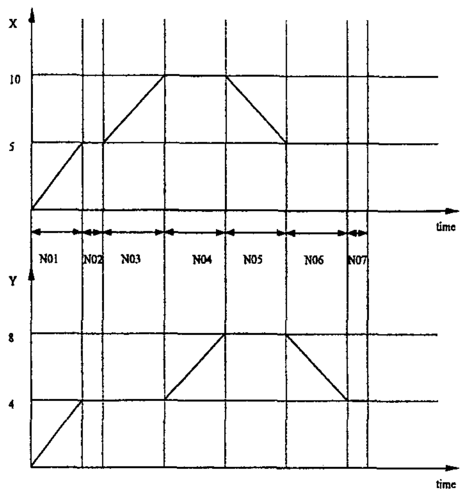


Fig. 27 Reference value

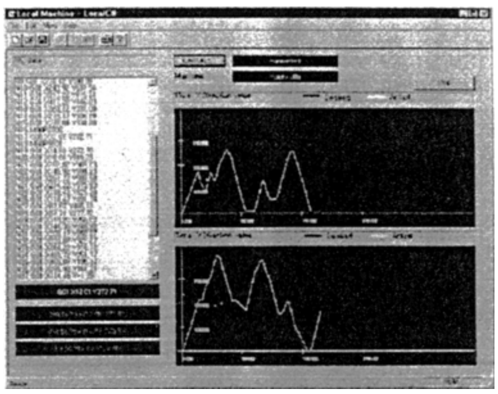


Fig. 28 Local controller

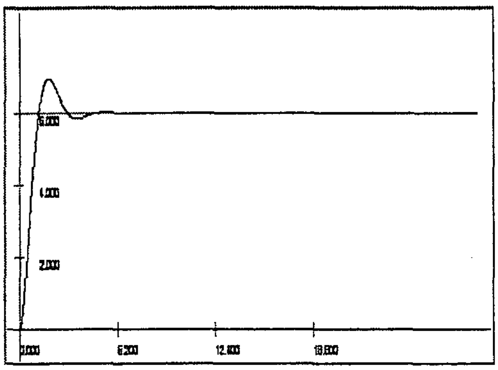


Fig. 29 Step response of the motor system (PID Gain : Kp=200, Ki=0.01, Kd=40)

In Fig. 26, with regard to flame setting, moving and cutting, the reference input changes continuously until the operation is over. The control reference input is computed as follows.

Let us suppose that the moving velocity and the moving distance from one point to another point are v and d, respectively, and that the moving time is $t_m = d/v$. After Δt time, the location toward x will be $x_m = x_{m(old)} + \Delta x / t_m \times \Delta t$ and the location toward y will be $y_m = y_{m(old)} + \Delta y / t_m \times \Delta t$. The computation result of the reference input is shown in Fig. 27.

We made a communication module to facilitate communication between the main server and the local PC. We used the TCP/IP which is a standard network protocol.

The NC codes are used as reference points for a particular machine. The NC codes must be changed into suitable reference input. So we developed an interpreter which changes the NC codes into suitable numerical values.

We determined that the machine was based on two axes that made a right-angle, and we made a simulation module that controls the location toward each axis. The controller is of the PID control type.

The GUI of the program that performs the control simulation on the virtual machine is shown in Fig. 28.

We can see the total course of the DCS manufacturing operation from the shape shown in Fig. 28 to Fig. 33. At the beginning of the total course, the main server gets information needed in the control from the database of machine files, and it tunes the control gain values. This is shown in Figs. 29 and 30. After getting suitable gain values, it transmits the gain values with the NC codes for each shape to the local PC as shown in Fig. 31.

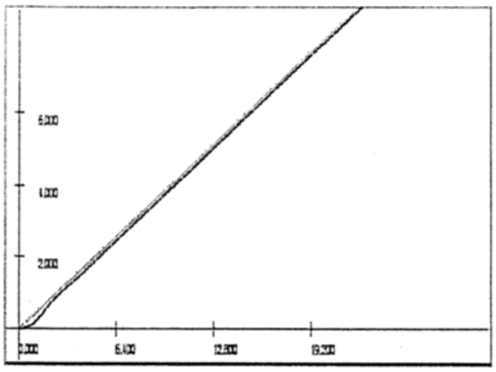


Fig. 30 Ramp response of the motor system (PID Gain : Kp=200, Ki=0.01, Kd=40)

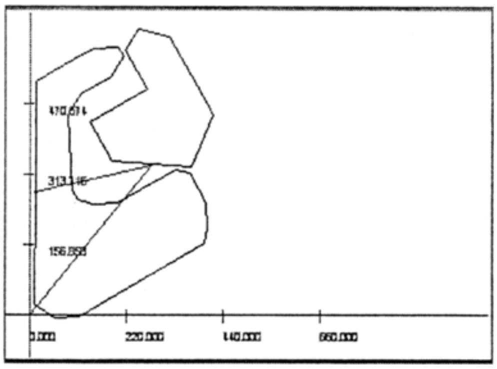


Fig. 31 Cutting shape and torch path

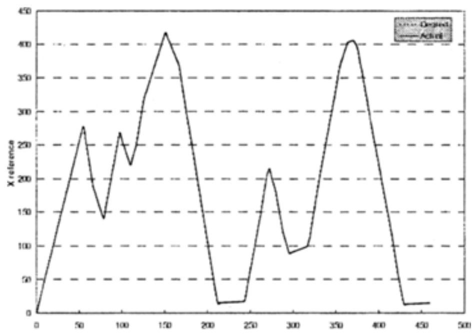


Fig. 32 Time response of the X-axis

Figures 32 and 33 show the results of a simulation for a virtual machine using NC codes and gain values from the main server. We can see that the actual values fit the demand values and that the error range between the actual values and the demand values is 5mm as shown in Fig. 34.

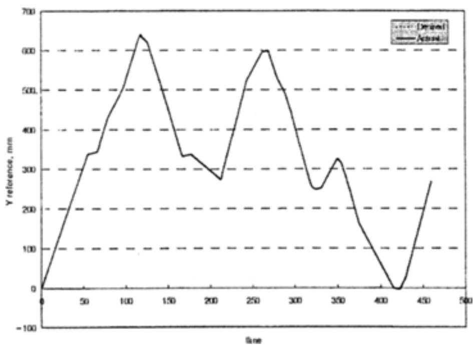


Fig. 33 Time response of the Y-axis

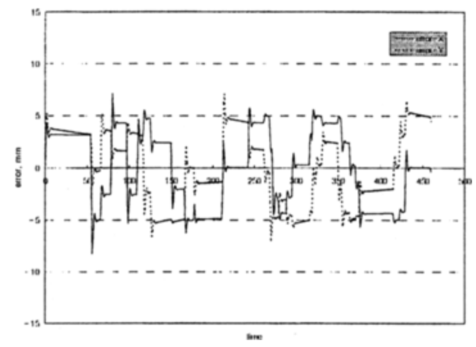


Fig. 34 Response error of the X, Y-axis

5. Result

We discussed three main topics in this paper. The first is the development of plate manufacturing CAD/CAM programs. The second is the optimal layout of two dimensional shapes. The third is the development of a PC-Based DCS for plate manufacturing.

The results of our research are explained below.

(1) We developed a new plate manufacturing CAD/CAM program for overcoming the problems of existing plate manufacturing CAD/CAM programs. This can reduce the cost of manufacturing. Additionally, this program can perform CAM operations directly.

(2) We made an optimal layout algorithm using genetic algorithms. We presented two cases; one case considered the rotation in genes, while the other case did not. From the results of simulations for both cases, we found that the former is a better situation. Also we found that there was a

time delay in the layout algorithm when the pattern order was used as the input. This result was caused by a moving algorithm that was applied toward the y direction. If we add a moving algorithm toward the x direction, we can obtain better performance.

(3) We made a virtual simulation program for a small sized, PC-Based DCS which is suitable for plate manufacturing. The existing DCS's are very expensive and are suitable for just large scale projects. Additionally, the existing NC systems are of the closed type and therefore, they can not be used for independent user interface or to perform functions. When the large scale DCS is applied to plate manufacturing, there are some moderate problems. Consequently, we developed a PC-Based DCS that was suitable for medium-scale systems and also performed simulations. For this system, we made a monitoring program using a TCP/IP protocol, built a machine database, and developed an open type control simulation program with which we could apply various control algorithms. We used a PID algorithm as the control algorithm in this study, but it will be possible for users to add fuzzy or neural network modules in the future.

Finally, this PC-Based DCS which is based on Windows and Windows NT environments, are suitable for various control algorithms. It can serve as a substitute for the more expensive CNC systems. This will serve to reduce costs in the plate manufacturing industry.

References

Adamowicz, M. and Albano, A. , 1976, "Nesting Two-Dimensional Shapes in Rectangular Modules," *Computer-Aided Design*, Vol. 8, No. 2, pp. 27~33.

Albano, A., 1977, "A Method to Improve Two- dimensional Layout," *Computer-Aided Design*, Vol. 9, No. 1, pp. 48~52.

Albano, A. and Sapuppo, G., 1980, "Optimal Allocation of Two-Dimensional Irregular Shapes Using Heuristic Search Method," *IEEE Transactions on System, Man and Cybernetics*, pp. 242 ~ 248.

Bounsaythip, C. and Maouche, S., 1996, "A Genetic Approach to a Nesting Problem," *Proceedings of the Second Nordic Workshop on Genetic Algorithms and their Applications (2NWGA)*, pp. 89~104.

Davis, L., 1996, *Handbook of Genetic Algorithms*, Van Nostrand Reinhold, New York, pp. 1 ~53.

Gen, M. and Cheng, R., 1997, *Genetic Algorithms & Engineering Design*, Wiley, New York, pp. 1~41.

Gilmore, P. C. and Gemory, R. E., 1965, "Multistage Cutting Stock Problems of Two and More Dimensions," *Oprn. Res*. Vol. 13, pp. 94 ~120.

Gilmore, P. C. and Gemory, R. E., 1966, " The Theory and Computation of Knapsack Functions," *Oprn. Res*. Vol. 14, pp. 1045~1074.

Lamousin, Waggenspack, and Dobson, 1996, "Nesting of Complex 2-D Parts Within Irregular Boundaries," *ASME, Journal of Manufacturing Science and Engineering*, Vol. 118, pp. 615~622.

Michalewicz, M., 1992, *Genetic Algorithms + Data Structures = Evolution Programs, 2nd Edition*, Springer-Verlag, New York, pp. 93 ~184.

Sarin, S. C., 1983, "Two-Dimensional Stock Cutting Problems and Solution Methodologies," ASME, *Journal of Engineering for Industry*, Vol. 105, pp. 155~160.

Watanabe, G. and Ono, T., 1997, "Determination of Cutting Layout of Two-dimensional Pattern by Genetic Algorithms," *IEEE of Japan Transactions*, Vol. 117, No. 3. pp. 356~363.